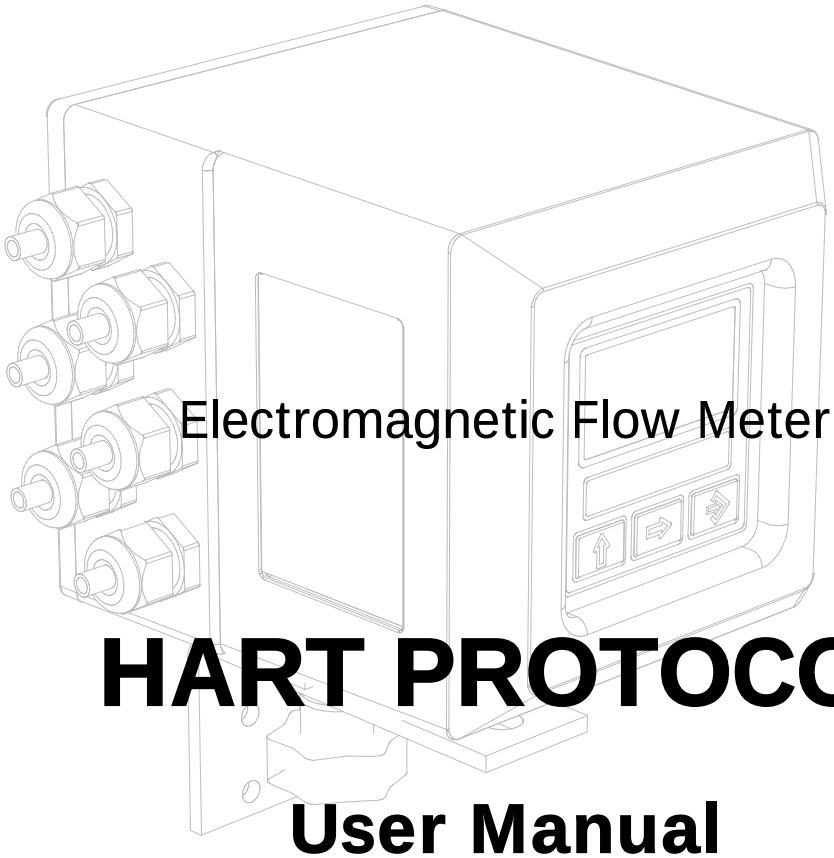
ISOMAG™
THE FRIENDLY MAG METER



Electromagnetic Flow Meter

HART PROTOCOL

User Manual

CONVERTER ML 210

TERMINALS DESIGNATIONS AND CABLE CONNECTION ____ 3

LED INDICATION ____ 3

DESCRIPTION OF THE HART COMMUNICATION MODULE ____ 4

HART UNIVERSAL COMMANDS DESCRIPTION ____ 5

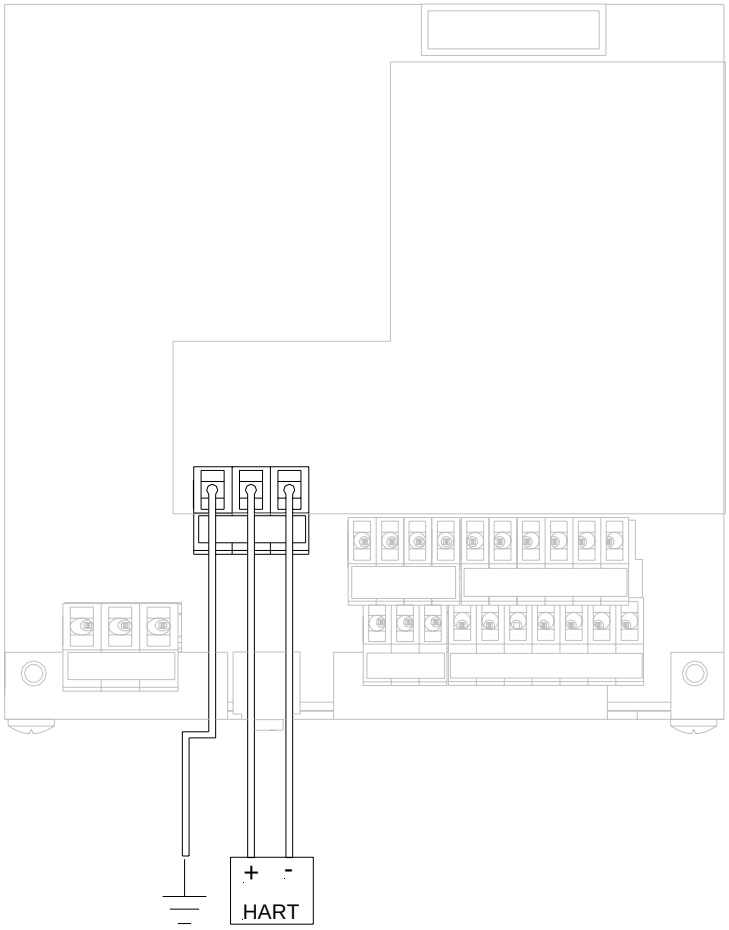
HART COMMON PRACTICE COMMANDS DESCRIPTION ____ 10

HART DEVICE SPECIFIC COMMANDS DESCRIPTION ____ 12

HART PROTOCOL ERROR CODES ____ 22

ETP COMMAND ____ 23

Electrical connection of HART module



Terminals designations and cable connection

LED indication

The HART communication module has a LED that can display the following status:

At start when power supply goes on the LED remain fixed on.

After five second, if the module is OK the LED blinking three times.

At this point the LED flash on-off whith comunication in the current loop.

Wiring
Signal wiring need not be shielded, but twisted pair should be used for best results.
In order to ensure communication, wiring should be 24 AWG or larger and should not exceed 1 mile in length.
Do not run signal wiring in conduit or open trays with power wiring, or near heavy electrical equipment or motors or generators.
Signal wiring may be grounded at any one point in the signal loop, or it may be left ungrounded. The negative terminal of the power supply is a recommended grounding point.

Description of the HART communication module

The HART module mounted in the Electromagnetic Flow Meter is used for connect the converter of the Flow meter to the HART fieldbus.

HART (Highway Addressable Remote Transducer) was developed by Rosemount in the mid-1980s, as microprocessor-based digital 'smart' instruments were first appearing.

HART allowed digital data to be superimposed on to conventional 4-20mA cable loops without disturbing analogue signals.
It is used to communicate digitally with the outside world for remote configuration, diagnostics and management purposes.
HART can provide many *'fieldbus-like'* functions, yet it retains full backwards-compatibility with existing control strategies.

When this option is installed it is possible to read the process data from the Flow meter and set the main functionality of the converter.

The module supports all the available HART commands including Universal, Common Practice and Device Specific Command.

The Hart protocol implemented in the module is conforms to Hart revision 5.8.

The HART protocol uses the Bell 202 standard frequency shift keying (FSK) signal to communicate at 1200 baud.

The HART signal is superimposed at a low level on the 4 to 20 mA analog measurement signal.

The HART signal have an average value of zero and causes no interference with the analog value.

HART is a master-slave protocol which means that a field device only replies when it is spoken to.

The HART slave may be addressed by two Masters for example a permanent master and a temporary Master like a hand held communicator.

When the HART fieldbus is in digital mode configuration (multi-drop) there can be two masters (e.g. a control system and a hand-held communicator) and up to 15 devices connected to the same bus.

The HART slave can operate in the so-called “burst mode”. In “bust mode” the slave will send data cyclically without requiring polling from the Master. It is normally the response on HART command 1 or 2. The command 3 is not implemented in “burst mode”.
The response is selected through the COMMON HART command 108.

HART Universal commands description

In the following section there is a description of the HART Universal commands implemented in the module.

COMMAND 00
Read Unique Identifier

Description:
This command gets the long address of the HART device plus other manufacturer information like Manufacturer Id, Device Type Code, Software Revision, Hardware Revision, etc.

Write Parameters:
NONE

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	Constant "254"	Manufacturer Identification Code
2	Manufacturer Device Type Code	Number of Preambles
3	Universal Command Revision	Transmitter Specific Command Revision
4	Software Revision	Hardware Revision
5	Device Function Flags	Device ID Number 1
6	Device ID Number 2	Device ID Number 3

COMMAND 01
Read Primary Variable

Description:
This command gets the device Primary Variable and the Primary Variable Units

Write Parameters:
NONE

Floating Point Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	Primary Variable Value (Flow rate in tecnicel unit: 32 bit single precision IEEE floating point)	
1		

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	0	0

NOTE: The values in the "32 bits single precision IEEE floating point" format are floating point numbers which can be represented during writing by any decimal digits. To keep the same numerical format visible on the instrument display however, it is necessary to calculate the decimal figures with a rather complex algorithm which takes account of instrument precision, flow rate measurement unit, etc. For this purpose and to avoid useless calculations, the number of decimals to use to represent the flow rate values is supplied separately. See the relative Device Specific command.

NOTE: For the measure unit see the relative commands in the section Device Specific commands.

NOTE: The byte and the words can be swapped. See the relative commands in the section Device Specific commands.

COMMAND 02
Read Current And Percent Of Range

Description:
This command gets the current of the loop that is forced by the HART device and the Percent of Range of the Current.

Write Parameters:
NONE

Floating Point Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	Current (mA)	
1		
2	Percent of Range (Flow rate in %: 32 bit single precision IEEE floating point)	
3		

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	

NOTE: The values in the "32 bits single precision IEEE floating point" format are floating point numbers which can be represented during writing by any decimal digits. To keep the same numerical format visible on the instrument display however, it is necessary to calculate the decimal figures with a rather complex algorithm which takes account of instrument precision, flow rate measurement unit, etc. For this purpose and to avoid useless calculations, the number of decimals to use to represent the flow rate values is supplied separately. See the relative Device Specific command.

NOTE: The byte and the words can be swapped. See the relative commands in the section Device Specific commands.

COMMAND 03
Read Dynamic Variables

Description:
This command gets the current and four (predefined) dynamic Variables.

Write Parameters:
NONE

Floating Point Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	Current (mA)	
1		
2	Primary Variable (Flow rate: 32 bit single precision IEEE floating point)	
3		
4	Second Variable (Totalizer positive: 32 bit long integer)	
5		
6	Third Variable (Totalizer negative: 32 bit long integer)	
7		
8	Fourth Variable(Totalizer partial: 32 bit long integer)	
9		

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	0	0
2	0	0

NOTE: The type of the flow rate can be in % or in technical unit.
For the selection of the type of the flow rate see the Device Specific commands.

NOTE: For the measure unit see the relative commands in the section Device Specific commands.

NOTE: The values in the "32 bits single precision IEEE floating point" format are floating point numbers which can be represented during writing by any decimal digits. To keep the same numerical format visible on the instrument display however, it is necessary to calculate the decimal figures with a rather complex algorithm which takes account of instrument precision, flow rate measurement unit, etc. For this purpose and to avoid useless calculations, the number of decimals to use to represent the flow rate values is supplied separately. See the relative Device Specific command.

NOTE: The counters are expressed with a 32 bit integer. The «counter decimal figures» parameter, indicates the point position starting from the right: 0 = no decimal, 1=1 decimal figure, and so on.

NOTE: The byte and the words can be swapped. See the relative commands in the section Device Specific commands.

NOTE: For the ASCII version of the counters see ETP commands.

COMMAND 06
Write Polling Address

Description:
This command writes the polling address of the HART device.
Extreme care should be taken when you use this command because you can loose the comunication with the device.

The polling address can be readed and changed with the menu Comunication → Address of the Converter.
See the menu Comunication in the Converter.

When modify the polling address with the HART command 06, the menu Address in the menu Comunication of the Converter is updated.

Write Parameters:

WORD	HIGH BYTE	LOW BYTE
0	Polling address	0

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	Polling address	0

COMMAND 11
Read Identifier Devices with TAG

Description:
This command read the Identifier Device information of HART command 00 by inserting the TAG parameter.

Write Parameters:

WORD	HIGH BYTE	LOW BYTE
0	Packed ASCII TAG Byte 0	Packed ASCII TAG Byte 1
1	Packed ASCII TAG Byte 2	Packed ASCII TAG Byte 3
2	Packed ASCII TAG Byte 4	Packed ASCII TAG Byte 5

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	Constant "254"	Manufacturer Identification Code
2	Manufacturer Device Type Code	Number of Preambles
3	Universal Command Revision	Transmitter Specific Command Revision

4	Software Revision	Hardware Revision
5	Device Function Flags	Device ID Number 1
6	Device ID Number 2	Device ID Number 3

COMMAND 12
Read Message

Description:
This command reads an ASCII message contained in the HART Device and written by the Write Message command 17.

Write Parameters:
NONE

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	Message ASCII Character 0	Message ASCII Character 1
2	Message ASCII Character 2	Message ASCII Character 3
. . . . .	. . . . .	. . . . .
11	Message ASCII Character 20	Message ASCII Character 21
12	Message ASCII Character 22	Message ASCII Character 23

COMMAND 13
Read Tag, Descriptor and Date

Description:
This command reads an ASCII Tag which identifies the device, an ASCII descriptor of the device and the last Date it has been configured.
Data can be write with Command 18.

Write Parameters:
NONE

Floating Point Data Returned:
NONE
Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	TAG ASCII Character 0	TAG ASCII Character 1
2	TAG ASCII Character 2	TAG ASCII Character 3
3	TAG ASCII Character 4	TAG ASCII Character 5
4	Descriptor ASCII Character 0	Message ASCII Character 1
. . . . .	. . . . .	. . . . .
9	Descriptor ASCII Character 10	Descriptor ASCII Character 11
10	Date	Date
11	Date	**0**

COMMAND 16
Read Final Assembly Number

Description:

This command reads the final assembly number of the HART device.
Data can be write with Command 19.

Write Parameters:
NONE

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	Final Assembly Number 0	Final Assembly Number 1
2	Final Assembly Number 2	0

COMMAND 17
Write Message

Description:
This command writes an ASCII message contained in the HART Device and that can be read with command 12.

Write Parameters:

WORD	HIGH BYTE	LOW BYTE
0	Packed ASCII Message Byte 0	Packed ASCII Message Byte 1
1	Packed ASCII Message Byte 2	Packed ASCII Message Byte 3
. **.** **.** **.** **.**	**.** **.** **.** **.** **.**	**.** **.** **.** **.** **.**
11	Packed ASCII Message Byte 22	Packed ASCII Message Byte 23

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	

COMMAND 18
Write Tag, Descriptor and Date

Description:
This command writes an ASCII Tag which identifies the device, an ASCII descriptor of the device and the last Date it has been configured.
Data can be read with Command 13.

Write Parameters:

WORD	HIGH BYTE	LOW BYTE
0	Packed ASCII TAG Byte 0	Packed ASCII TAG Byte 1
1	Packed ASCII TAG Byte 2	Packed ASCII TAG Byte 3
2	Packed ASCII TAG Byte 4	Packed ASCII TAG Byte 5
3	Packed ASCII Descriptor Byte 0	Packed ASCII Descriptor Byte 1
. **.** **.** **.** **.**	**.** **.** **.** **.** **.**	**.** **.** **.** **.** **.**
8	Packed ASCII Descriptor Byte 10	Packed ASCII Descriptor Byte 11

9	Date	Date
10	Date	**0**

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	TAG ASCII Character 0	TAG ASCII Character 1
2	TAG ASCII Character 2	TAG ASCII Character 3
3	TAG ASCII Character 4	TAG ASCII Character 5
4	Descriptor ASCII Character 0	Descriptor ASCII Character 1
. . . . .	. . . . .	. . . . .
9	Descriptor ASCII Character 10	Descriptor ASCII Character 11
10	Date	Date
11	Date	0

COMMAND 19
Write Final Assembly Number

Description:
This command writes the final assembly number of the HART device.
Data can be read with Command 16.

Write Parameters:

WORD	HIGH BYTE	LOW BYTE
0	Final Assembly Number 0	Final Assembly Number 1
1	Final Assembly Number 2	

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	Final Assembly Number 0	Final Assembly Number 1
2	Final Assembly Number 2	0

HART Common Practice commands description

In the following section there is a description of the Common Practice commands implemented in the module

COMMAND 40
Enter Exit Fixed Current Mode

Description:
The device is placed in the Fixed Primary Variable Current Mode with the Primary Variable Current set to the value received.
Only two floatig point value are accepted for this function:

4.0 mA = 00 00 80 40 (hex format) for setting the 4 mA point
20.0 mA = 00 00 A0 41 (hex format) for setting the 20 mA point

The value are returned in the Response Bytes.

With this function the device enter in the Fixed Primary Variable Current Mode.
Adjusting the set points with commands 220,221,222 and 223.

A level of "0" exits the Fixed Primary Variable Current Mode. Fixed Primary Variable Current Mode is also exited when power is removed from the device.

Write Parameters:

WORD	HIGH BYTE	LOW BYTE
0	Floating Point Current (mA)	
1		

Floating Point Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	Floating Point Current (mA)	
1		

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	

COMMAND 59
Write Number of response preambles

Description:
This command writes the number of response preambles that the HART device will use in every command response.

The number of response preambles can be changed with the menu Preambles in the menu Comunication of the converter.
See the menu Comunication in the Converter.

When modify the number of response preambles with the HART command 59, the menu preambles in the menu Comunication of the Converter is updated.

Write Parameters:

WORD	HIGH BYTE	LOW BYTE
0	Number of response preambles	0

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	Number of response preambles	0

COMMAND 108
Write Burst Mode Command Number

Description:
This command sets the command number that the HART device will use in Burst Mode.

Write Parameters:

WORD	HIGH BYTE	LOW BYTE
0	**Burst Mode Command Number**	

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	**STATUS WORD**	
1	**Burst Mode Command Number**	**0**

COMMAND 109
Write Burst Mode Control

Description:
This command sets the HART Burst Mode control.

00 → Burst Mode OFF
01 → Burst Mode ON

Write Parameters:

WORD	HIGH BYTE	LOW BYTE
0	**Burst Mode Control**	

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	**STATUS WORD**	
1	**Burst Mode Control**	**0**

HART Device Specific commands description

In the following section there is a description of the Device Specific commands implemented in the module.

COMMAND 170
Write Manufacturer Device Type Code

Description:
This command writes the Manufacturer Device Type Code of the HART device.
Data can be read with HART Command 00.

Write Parameters:

WORD	HIGH BYTE	LOW BYTE
0	Manufacturer Device Type Code	0

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	Manufacturer Device Type Code	0

COMMAND 171
Write Device ID Number

Description:
This command writes the device ID number of the HART device.
Data can be read with HART Command 00.

Write Parameters:

WORD	HIGH BYTE	LOW BYTE
0	Device ID Number 0	Device ID Number 1
1	Device ID Number 2	

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	Device ID Number 0	Device ID Number 1
2	Device ID Number 2	0

COMMAND 172
This command read the swap code of the bytes of the process data when returned

Description:
This command read the swap code of the bytes of the process data when returned:
0 = no swap
1 = swap

Write Parameters:
NONE

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	Swap bytes process data	0

COMMAND 173
This command write the swap of the bytes of the process data when returned

Description:
This command write the swap of the bytes of the process data when returned:
0 = no swap
1 = swap

Write Parameters:

WORD	HIGH BYTE	LOW BYTE
0	Swap bytes process data	0

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	

1	Swap bytes process data	0

COMMAND 174
This command read the swap code of the words of the process data when returned

Description:
This command read the swap code of the words of the process data when returned:
0 = no swap
1 = swap

Write Parameters:
NONE

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	Swap words process data	0

COMMAND 175
This command write the swap code of the words of the process data when returned

Description:
This command write the swap code of the words of the process data when returned:
0 = no swap
1 = swap

Write Parameters:

WORD	HIGH BYTE	LOW BYTE
0	Swap words process data	0

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	Swap words process data	0

COMMAND 176
This command read the type of process data returned with HART command 03

Description:
This command read the type of process data returned:
0 --> flow rate, totalizer positive, totalizer negative, parartial totalizer positive
1 --> flow rate, totalizer posositive, totalizer negative, partial toalizer negative

Write Parameters:
NONE

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	Type of process data returned	0

COMMAND 177
This command write the type of process data returned with HART command 03

Description:
This command select the type of process data returned with HART Command 03:
0 --> flow rate, totalizer positive, totalizer negative, parartial totalizer positive
1 --> flow rate, totalizer posositive, totalizer negative, partial toalizer negative

Write Parameters:

WORD	HIGH BYTE	LOW BYTE
0	Type of process data returned	0

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	Type of process data returned	0

COMMAND 178
This command read the type of flow rate returned with HART command 03

Description:
This command read the type of flow rate returned with HART Command 03:
0 = flow rate in %
1 = flow rate in tecnical unit

Write Parameters:
NONE

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	Type of flow rate	0

COMMAND 179
This command write the type of flow rate returned with HART command 03

Description:
This command write the type of flow rate returned with HART Command 03:
0 = flow rate in %
1 = flow rate in tecnical unit

Write Parameters:

WORD	HIGH BYTE	LOW BYTE
0	Type of flow rate	0

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	Type of flow rate	0

COMMAND 180
Read the Flow rate scale range in t.u.

Description:
This command reads the Flow rate scale range in t.u.

Write Parameters:
NONE

Floating Point Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	**STATUS WORD**	
1	Flow rate scale range in t.u.	
2	(32 bit single precision IEEE floating point)	

Integer Data Returned:
NONE

COMMAND 181
Read the ASCII version of the measure unit for the flow rate

Description:
This command reads the ASCII version of the measure unit for the flow rate.

Write Parameters:
NONE

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	ASCII Character 0	ASCII Character 1
2	ASCII Character 2	ASCII Character 3
3	ASCII Character 4	0

COMMAND 182
This command reads the ASCII version of the measure unit for the totalizers

Description:
This command reads the ASCII version of the measure unit for the totalizers.

Write Parameters:
NONE

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	ASCII Character 0	ASCII Character 1
2	ASCII Character 2	0

COMMAND 183
Read the number of decimals after the point for flow rate display

Description:
This command reads the number of decimals after the point for flow rate display.

Write Parameters:
NONE

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	(8 bit integer) number of decimals after the point for flow rate display	0

COMMAND 184
This command reads number of decimals after the point for totalizers display

Description:
This command reads number of decimals after the point for totalizers display.

Write Parameters:
NONE

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	(8 bit integer) number of decimals after the point for totalizers display	0

COMMAND 185
This command reads process flags

Description:
This command reads process flags:

bit 0 =1 if the excitation is too fast for the sensor connected
bit 1 =1 if the maximum alarm is active
bit 2 =1 if the minimum alarm is active
bit 3 =1 if the Flow rate exceeds the scale range value (overflow)
bit 4 =1 if one or more output impulses are saturated (too many impulses to emit)
bit 5 =1 if the measurement signal is highly disturbed or if the sensor is disconnected
bit 6 =1 if the measurement tube is empty
bit 7 =1 if the circuit powering the coils is not working or the sensor is disconnected
bit 8 =1 if the second measurement scale is active
bit 9 =1 if the Flow rate is lower than the cut-off threshold
bit10=1 if the Flow rate is negative
bit11=1 if a new measurement value calculated for the display is available
bit12=1 if the counter block signal is active
bit13=1 if dosing is in progress
bit14=1 if a calibration cycle is in progress
bit15=1 if a Flow rate simulation is in progress

Write Parameters:
NONE

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	(16 bit unsigned integer, MSB first) process flags	

COMMAND 186
This command reads the measurement samples per second (Hz)

Description:

This command reads the measurement samples per second (Hz):

10(dec) measurement frequency = 10 Hz
20(dec) measurement frequency = 20 Hz
50(dec) measurement frequency = 50 Hz
80(dec) measurement frequency = 80 Hz
150(dec)measurement frequency = 150 Hz
44(dec) measurement frequency = 300 Hz
144(dec)measurement frequency = 400 Hz

Write Parameters:
NONE

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	(8 bits integer) measurement samples per second (Hz)	0

COMMAND 187
This command reads the measurement dynamic variation as a %

Description:
This command reads the measurement dynamic variation as a %.

Write Parameters:
NONE

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	(8 bits integer) measurement dynamic variation as a %	0

COMMAND 189
This command reads the the digital value of the current set in the DAC

Description:
This command reads the digital value of the current set in the DAC.

Write Parameters:
NONE

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	16 bit Unsigned integer	

COMMAND 190
This command reset of HART module

Description:
This command reset of HART module.

Write Parameters:
NONE

Floating Point Data Returned:
NONE

Integer Data Returned:
NONE

COMMAND 200
This command writes an etp command (See section ETP command).

Description:
This command sends an etp command to the converter.
For a description of the ETP (embedded protocol) and a list of the possible etp command see the relative section.
The command return the string sended.
For reading the result of the etp command see the next HART command below.

Write Parameters:

WORD	HIGH BYTE	LOW BYTE
0	ASCII Character 0	ASCII Character 1
1	ASCII Character 2	ASCII Character 3
.	.	.
.	.	.
.	.	.
.	.	.
30	ASCII Character 28	ASCII Character 29
31	ASCII Character 30	ASCII Character 31

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	ASCII Character 0	ASCII Character 1
2	ASCII Character 2	ASCII Character 3
.	.	.
.	.	.
.	.	.
.	.	.
59	ASCII Character 28	ASCII Character 29
50	ASCII Character 30	ASCII Character 31

COMMAND 201
This command reads the results of an etp command (See section ETP command).

Description:
This command read the result of an etp command.
The max number of byte that you can return with the HART command 201 is 24 bytes.

Some etp command can have answer with more of 24 bytes.
If you send the HART command 201 with the code 0xFF, you return the number of bytes of the answer.
If the leght of the etp command is greater than 24 byte you can read the complete answer with an offset.

Example:

You have send an etp commad with HART command 200.
With the HART command 201 and the code 0xFF, you read that the answer have 55 bytes for examples.

With the HART command 201 and the offset 0x00 (0 dec) you read the first block of 24 byte.
With a second HART command 201 and the offset 0x18 (24 dec) you read the second block of 24 byte.
With the third HART command 201 and the offset 0x30 (48 dec) you read the last block (7 bytes)

At the end you have returned: 24 + 24 + 7 = 55 bytes
and you have used three time the HART command 201

NOTE: if you send the HART command 201 without the write parameter, the command return max 24 bytes
You can use the HART command 201 without the write parameter if you have answer with max 24 bytes.

Write Parameters:

WORD	HIGH BYTE	LOW BYTE
0	Len of the answer Or offset	

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	ASCII Character 0	ASCII Character 1
2	ASCII Character 2	ASCII Character 3
. . . .	. . . .	. . . .
11	ASCII Character 20	ASCII Character 21
12	ASCII Character 22	ASCII Character 23

COMMAND 212
This command reads the polling address

Description:
This command reads the polling address of the HART device.
The address can be readed and changed in the Converter by the menu Comunication → Address
See the menu Comunication in the Converter.

Write Parameters:
NONE

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	Polling address	0

COMMAND 220
This command reads the set point for the 20 mA

Description:
This command reads the value of the set point for the 20 mA.
The value can be read in the menu Internal Data → DAC2 20 mA of the Converter.

Write Parameters:
NONE

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	16 bit unsigned integer	

COMMAND 221
This command writes the set point for the 20 mA

Description:
This command writes the value of the set point for the 20 mA.
The value can be reads and changed in the menu Internal Data → DAC2 20 mA of the Converter.

When modify the value with the HART command 221, the menu DAC 20 mA in the menu Internal data of the Converter is updated.
See the menu Internal Data → DAC 20 mA of the Converter in the user manual for changing the value with keypad.

Write Parameters:

WORD	HIGH BYTE	LOW BYTE
0	16 bit unsigned integer	

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	16 bit unsigned integer	

COMMAND 222
This command reads the set point for the 4 mA

Description:
This command reads the value of the set point for the 4 mA.
The value can be read in the menu Internal Data → DAC2 4 mA of the Converter.

Write Parameters:
NONE

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	16 bit unsigned integer	

COMMAND 223
This command writes the set point for the 4 mA

Description:
This command writes the value of the set point for the 4 mA.
The value can be reads and changed in the menu Internal Data → DAC2 4 mA of the Converter.

When modify the value with the HART command 221, the menu DAC 4 mA in the menu Internal data of the Converter is updated.

See the menu Internal Data → DAC 4 mA of the Converter in the user manual for changing the value with keypad.

Write Parameters:

WORD	HIGH BYTE	LOW BYTE
0	16 bit unsigned integer	

Floating Point Data Returned:
NONE

Integer Data Returned:

WORD	HIGH BYTE	LOW BYTE
0	STATUS WORD	
1	16 bit unsigned integer	

HART Protocol Error Codes

These are error codes that are part of the HART protocol. The standard HART error codes are shown in the tables below:
ERROR CODE WORD

FIRST BYTE								**SECOND BYTE**							
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0

When bit 7 of First Byte = 1 → COMUNICATION ERROR then the type of communication error are listed in the table below:

FIRST BYTE:

Bit 7	**1**
Remaining bytes of First Byte	
Bit 6	Parity Error
Bit 5	Overrun Error
Bit 4	Framing Error
Bit 3	Checksum Error
Bit 2	Reserved
Bit 1	RX Buffer Overflow
Bit 0	Undefined

SECOND BYTE:

Bit 7	0
Bit 6	0
Bit 5	0
Bit 4	0
Bit 3	0
Bit 2	0
Bit 1	0
Bit 0	0

When bit 7 of First Byte = 0 → COMUNICATION ERROR then the type of communication error are listed in the table below:

FIRST BYTE:

Bit 7	**0**
Remaining bytes of First Byte: Bits 6 to 0 (not bit-mapped):	
0	No Error
1	Undefined
2	Invalid Selection
3	Passed Parameter Too Large
4	Passed Parameter Too Small
5	Too Few Data Bytes Received
6	Transmitter-Specific Command Error

7	In Write-Protect Mode
8-15	**Command Specific Errors** (see Table Below)
16	Access Restricted
32	Device is Busy
64	Command not Implemented

SECOND BYTE:

Bit 7	Device Malfunction
Bit 6	Configuration Changed
Bit 5	Cold Start
Bit 4	Unused
Bit 3	Output Current Fixed
Bit 2	Analog Output Saturated
Bit 1	Variable (Not Primary) Out of Limits
Bit 0	Primary Variable Out of Limits

8-15 Command Specific Errors

8	Update Failed Update In Progress Set to Nearest Possible Value
9	Applied Process Too High Lower Range Value Too High Not in Fixed Current Mode
10	Applied Process Too Low Lower Range Value Too Low MultiDrop Not Supported
11	In MultiDrop Mode Invalid Transmitter Variable Code Upper Range Value Too High
12	Invalid Unit Code Upper Range Value Too Low
13	Both Range Values Out of Limits
14	Pushed Upper Range Value Over Limit Span Too Small
15	Unused

ETP Command

The ETP commands is a list of formatted strings used for the comunication with the converter.
With ETP commands is possible read process data and change the setting of the parameters of the converter.

The HART command number 200 is for sending an ETP command to the converter.

The HART command number 201 is for return the answer from the converter.

Note:
The max length of the string for the HART command 200 is 24 bytes.
The max length of the string returned by HART command 201 is 250 bytes.

Example of implementation for some ETP commands with HART protocol:

Example 1----------

Send the ETP command FRFS1 with HART command 200:
Insert the string: **FRFS1? + Chr 13** with the HART command number 200.

In hex format there are the following 7 bytes:
46 52 46 53 31 3f 0d

In the answer of the HART command 200 there is the string send:

FRFS1?

For reading the answer of the HART command 200 you must send the command 201:
An example of answer returned with HART command 201 is: 3600

End example 1----------

Example 2----------

Send the ETP command FRFS1=? with HART command 200:
Insert the string: **FRFS1=? + Chr 13** with the HART command number 200.

In hex format there are the following 8 bytes:
46 52 46 53 31 3d 3f 0d

In the answer of the HART command 200 there is the string send:
FRFS1=?

For reading the answer of the HART command 200 you must send the command 201:
An example of answer returned with HART command 201 is: **461 <> 11520 (dm3/h)**

End example 2----------

Example 3----------

Send the ETP command FRFS1=461? with HART command 200:
Insert the string: **FRFS1=461? + Chr 13** with the HART command number 200.

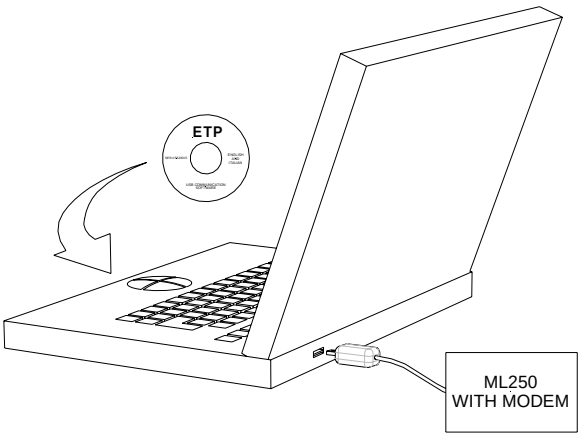
In hex format there are the following 8 bytes:
46 52 46 53 31 3d 34 36 31 0d

In the answer of the HART command 200 there is the string send:
FRFS1=461

For reading the answer of the HART command 200 you must send the command 201:
Below there is the list of the answers present in HART command 201:

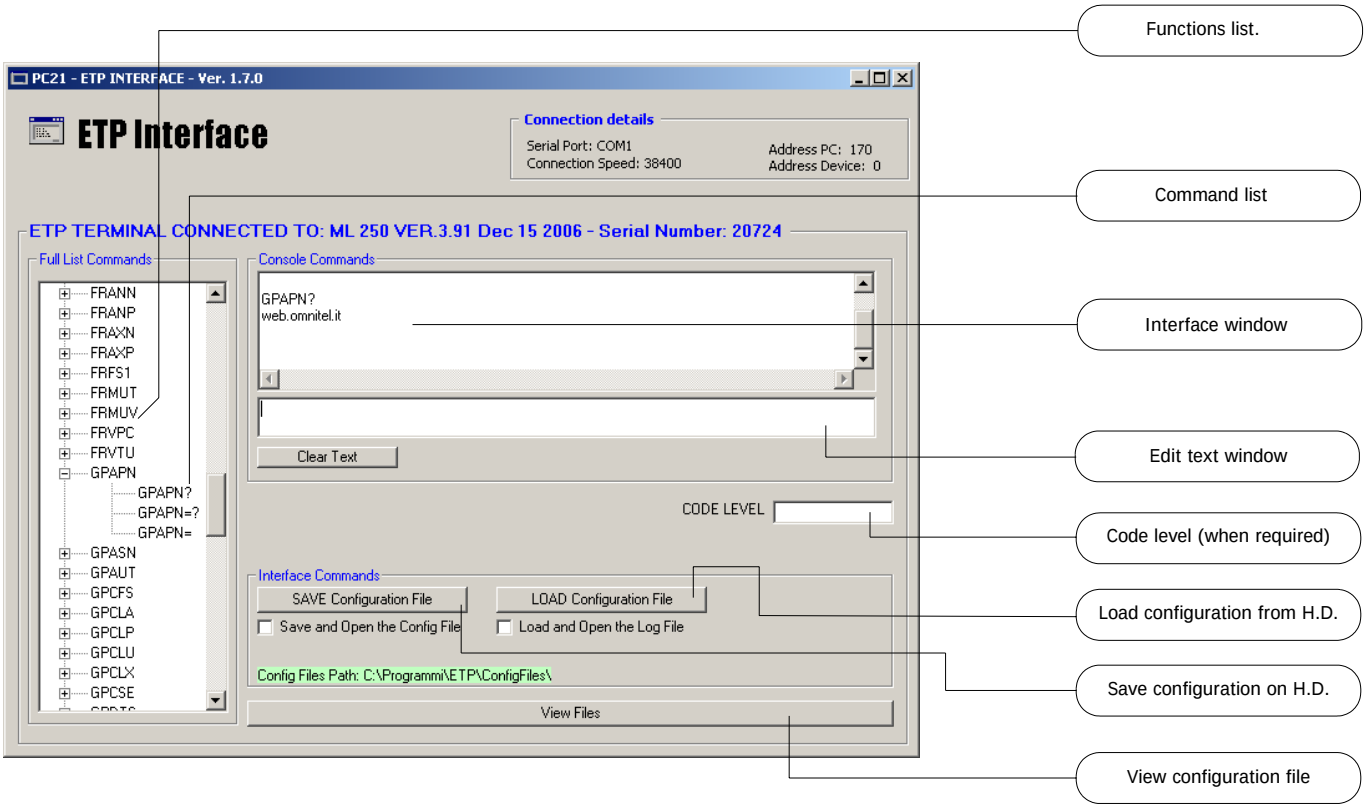
0:OK	the execution was correct
1:CMD ERR	wrong context, execution was not possible due to a configuration limit or wrong working conditions
2:PARAM ERR	the expected parameter was out of the allowed range
3:EXEC ERR	the execution of the command was not successful due to an internal error condition
4:RANGE ADJ	the entered parameter caused an internal automatic adjustment on other ranges
5:ACCESS ERR	the execution of the command was not possible due to an insufficient privilege level
6:BUFFER FULL	the input or the output strings exceed the maximum allowable space

ETP COMMADS DESCRIPTIONS FOR HART MODULE



THE ETP SOFTWARE ALLOW TO PROGRAMMING THE CONVERTER TROUGH A VERY SIMPLE "MNEMONIC" (5 DIGITS) COMMAND . SOME PARAMETERS (SEE THE FOLLOWING PAGES) CAN BE CHANGE/SET ONLY WITH ETP INTERFACE , THAT BECAUSE IT'S MORE EASY THAN TO INTRODUCE THEM BY KEYBOARD.

INSTALL THE ETP SOFTWARE ON YOUR PC AND CONNECT THE IF21 INTERFACE (USB INTERFACE BETWEEN PC AND CONVERTER) TO THE CONVERTER ; LAUNCH THE PROGRAM AND FOLLOW THE INSTRUCTIONS AS INDICATE BELOW:



- **Functions list:** In this window are listed all function of
- **Command list:** Clik on "+" symbol aside of function for open command list. Each function have its commands list
- **Edit text window:** In this area appear the selected command **i.e. GPAPN? or GPAPN=? or GPAPN=** (**type directly by keyboard or press over the COMMAND with the mouse) .**
- **Interface window:** In this area are visualized the result of command
- **Save configuration file:** When the setup of instrument it's finished, it's possible save the configuration on P.C. Hard disk in the directory visualized on green underlined area
- **Load configuration file:** It's possible load in the instrument (for example a new instrument) the configuration list saved .
- **View files:** It's possible to open the configuration file saved for view or edit
- **Code level window:** If required it's possible set the code level in this window

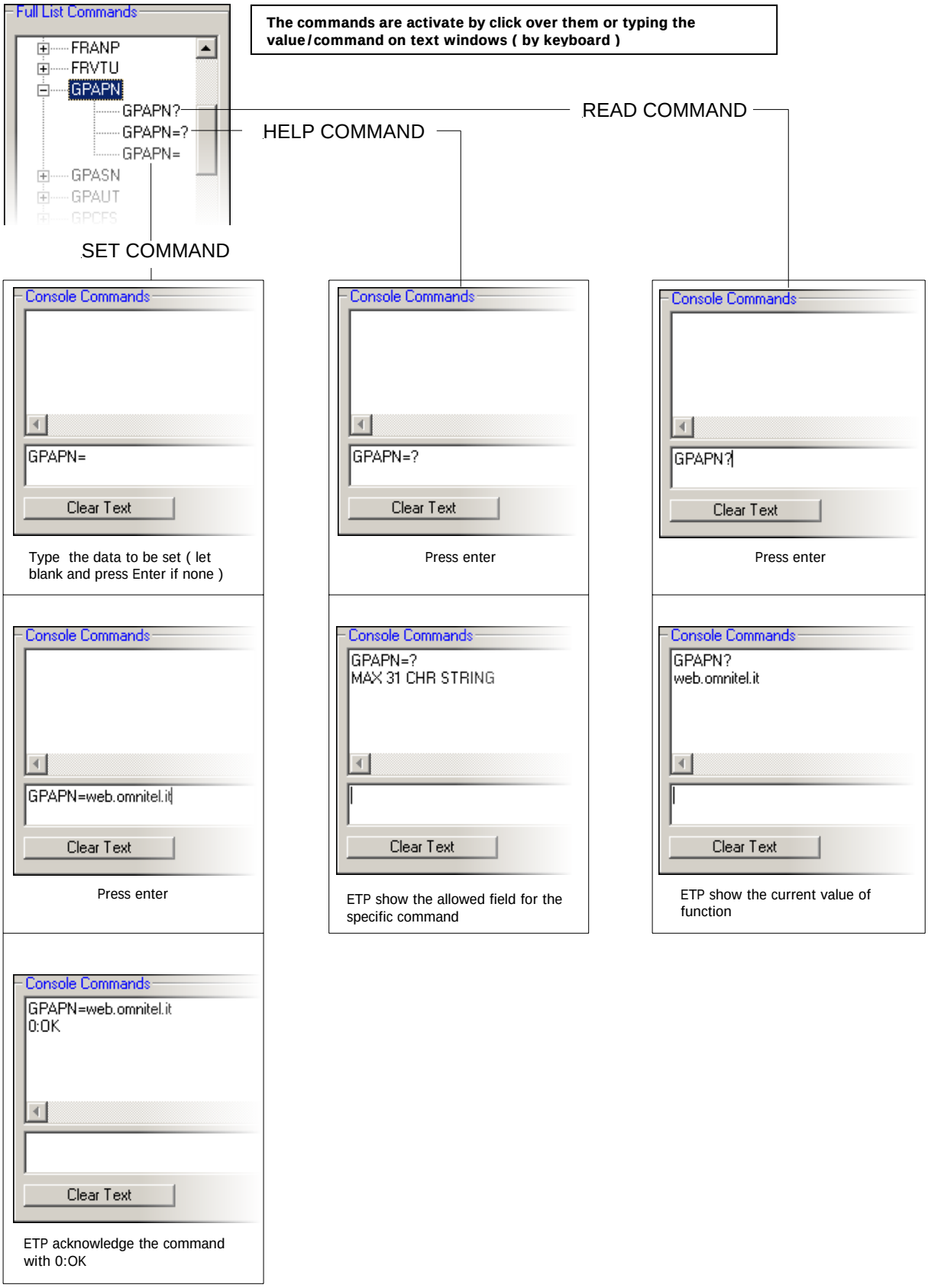
N.B.: ETP save and load automatically **ALL** functions from/to converter, therefore if is necessary to restore only some data from functions it's recommend to open the file with bottom "View file" and edit this file before reload on a new converter (**delete the functions that are not necessary , i.e. Dn , Ka , ...**)

SINTAX COMMANDS

In the ETP software there are three command : ? ; =? ; = ; with the following meaning :

[command]? : READ COMMAND
[command]=? : HELP COMMAND
[command]= SET COMMAND

Following there are one example of command (function GPAPN) :



COMMANDS AND FUNCTIONS EQUIVALENT TO THE "1-SENSOR" MENU		
Name and description	**Modes**	**Returned values or codes**
PDIMV **(Pipe DIaMeter Value) Reads or sets the nominal diameter of the sensor.**	**Read command:** PDIMV? **Required access level:** 2	**value of nominal diameter if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: PDIMV=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: PDIMV=? **Required access level:** 2	*min <> max (unit)* **if accepted** 5:ACCESS ERR **if insufficient access level**
CFFKA **(CoeFFicient KA) Reads or sets the value of the gain coefficient KA**	**Read command:** CFFKA? **Required access level:** 2	**Value of the coefficient if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: CFFKA=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: CFFKA=? **Required access level:** 2	*min <> max* **if accepted** 5:ACCESS ERR **if insufficient access level**
SMODL **(Sensor MODeL) Reads or sets the value of the sensor model**	**Read command:** SMODL? **Required access level:** 2	**Value of the sensor model if accepted** 5:ACCESS ERR **if insufficient access level** 5:ACCESS ERR **if insufficient access level**
	Set command: SMODL=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: SMODL=? **Required access level:** 2	*min <> max* **if accepted** 5:ACCESS ERR **if insufficient access level**
SIPOS **(Sensor Insertion POSition) Reads or sets the value that identifies the insertion position for that type of sensor**	**Read command:** SIPOS? **Required access level:** 2	**Value of the insertion position if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: SIPOS=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: SIPOS=? **Required access level:** 2	*min <> max* **if accepted** 5:ACCESS ERR **if insufficient access level**
CKLP0, CKLP1, CKLP2, CKLP3, CKLP4, CKLP5 **(Coefficient KL Positive 0, 1, 2, 3, 4 an 5) Reads or sets the value for the six coefficients for the linearization of the positive flow rate range.** NOTE: **the examples refer to the command** CKLP0**, but the others are the same**	**Read command:** CKLP0? **Required access level:** 2	**Value of the coefficient if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: CKLP0=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: CKLP0=? **Required access level:** 2	*min <> max* **if accepted** 5:ACCESS ERR **if insufficient access level**
CKLN0, CKLN1, CKLN2, CKLN3, CKLN4, CKLN5 **(Coefficient KL Negative 0, 1, 2, 3, 4 an 5) Reads or sets the value for the six coefficients for the linearization of the negative flow rate range.** NOTE: **the examples refer to the command** CKLN0**, but the others are the same**	**Read command:** CKLN0? **Required access level:** 2	**Value of the coefficient if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: CKLN0=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: CKLN0=? **Required access level:** 2	*min <> max* **if accepted** 5:ACCESS ERR **if insufficient access level**
SFREQ **(Sampling FREQuency) Reads or sets the measure sampling frequency of the instrument.**	**Read command:** SFREQ? **Required access level:** 2	**Value of the measure sampling frequency** 5:ACCESS ERR **if insufficient access level**

	Set command: SFREQ=*n* **Required access level:** 3	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: SIPOS=? **Required access level:** 2	*min <> max (units)* **if accepted** 5:ACCESS ERR **if insufficient access level**
CRVRF **(Coil Regulator Voltage ReFerence) Reads or sets the value of the reference for the coil current regulator.**	**Read command:** CRVRF? **Required access level:** 2	**Value of the coil regulator voltage reference** 5:ACCESS ERR **if insufficient access level**
	Set command: CRVRF=*n* **Required access level:** 3	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: CRVRF=? **Required access level:** 2	*min <> max* **if accepted** 5:ACCESS ERR **if insufficient access level**
CRRMA **(Coil Regulator Regulation MArgin) Reads or sets the value of the regulation margin for the coil current regulator.**	**Read command:** CRRMA? **Required access level:** 2	**Value of the coil regulator margin** 5:ACCESS ERR **if insufficient access level**
	Set command: CRRMA=*n* **Required access level:** 3	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: CRRMA=? **Required access level:** 2	*min <> max* **if accepted** 5:ACCESS ERR **if insufficient access level**
EPDEN **(Empty Pipe Detection ENable) Enables or disables the empty pipe detection circuit.**	**Read command:** EPDEN? **Required access level:** 2	**State of the pipe detection circuit** 5:ACCESS ERR **if insufficient access level**
	Set command: EPDEN=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: EPDEN=? **Required access level:** 2	0:OFF,1:ON **if accepted** 5:ACCESS ERR **if insufficient access level**
EPDGV **(Empty Pipe Detection Gain Value) Reads or sets the value of the gain for the empty pipe detection circuit.**	**Read command:** EPDGV? **Required access level:** 2	**Value of the empty pipe detection gain** 5:ACCESS ERR **if insufficient access level**
	Set command: EPDGV=*n* **Required access level:** 3	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: EPDGV=? **Required access level:** 2	*min <> max* **if accepted** 5:ACCESS ERR **if insufficient access level**
EPDCC **(Empty Pipe Detection Calibration Command) Executes the calibration of the empty pipe detection circuit.**	**Read command: NOT SUPPORTED**	1:CMD ERR **if read command is issued** 5:ACCESS ERR **if insufficient access level**
	Set command: EPDCC=1 **Required access level:** 2	0:OK **if execution ok** 2:PARAM ERR **if parameter not equal to 1**
	Help command: EPDCC=? **Required access level:** 2	1:EXECUTE **if accepted**
SZPCC **(Sensor Zero Point Calibration Command) Executes the calibration of the sensor zero point.**	**Read command:** SZPCC? **Required access level:** 2	0 **if the calibration is terminated** 1 **if the calibration is in progress** 5:ACCESS ERR **if insufficient access level**
	Set command: SZPCC=1 **Required access level:** 2	0:OK **if execution ok** 2:PARAM ERR **if parameter not equal to 1** 5:ACCESS ERR **if insufficient access level**
	Help command: SZPCC=? **Required access level:** 2	1:EXECUTE **if accepted** 5:ACCESS ERR **if insufficient access level**
SZPCR **(Sensor Zero Point Calibration Reset) Resets the zero point calibration value**	**Read command: NOT SUPPORTED**	1:CMD ERR **if read command is issued** 5:ACCESS ERR **if insufficient access level**

evaluated with the preceding instruction SZPCC.	**Set command:** SZPCR=1 **Required access level:** 2	0:OK **if execution ok** 2:PARAM ERR **if parameter not equal to 1** 5:ACCESS ERR **if insufficient access level**
	Help command: SZPCR=? **Required access level:** 2	1:EXECUTE **if accepted** 5:ACCESS ERR **if insufficient access level**

COMMANDS AND FUNCTIONS EQUIVALENT TO THE "2-SCALES" MENU		
Name and description	**Modes**	**Returned values or codes**
FRMUT **(Flow Rate Measure Unit Type) Reads or sets the type of measure units relative to the flow rate. Possible values are:** 0:VM **= Volume, Metric** 1:WM **= Weight, Metric** 2:VI **= Volume, Imperial or American** 3:WI **= Weight, Imperial or American**	**Read command:** FRMUT? **Required access level:** 2	**Value of measure unit if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: FRMUT=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: FRMUT=? **Required access level:** 2	**List of** *num:description* **if accepted** 5:ACCESS ERR **if insufficient access level**
FRMUV **(Flow Rate Measure Unit Value) Reads or sets the value of measure units relative to the flow rate.**	**Read command:** FRMUV? **Required access level:** 2	**Value of measure unit if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: FRMUV=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: FRMUV=? **Required access level:** 2	**List of** *num:description* **if accepted** 5:ACCESS ERR **if insufficient access level**
FRFS1 **(Flow Rate Full Scale 1) Reads or sets the value of flow rate full scale 1.**	**Read command:** FRFS1? **Required access level:** 2	**Value of flow rate full scale 1 if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: FRFS1=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: FRFS1=? **Required access level:** 2	*min <> max (units)* **if accepted** 5:ACCESS ERR **if insufficient access level**
FRFS2 **(Flow Rate Full Scale 2) Reads or sets the value of flow rate full scale 2.**	**Read command:** FRFS2? **Required access level:** 2	**Value of flow rate full scale 2 if accepted** 1:CMD ERR **if full scale 2 not enabled** 5:ACCESS ERR **if insufficient access level**
	Set command: FRFS2=*n* **Required access level:** 2	0:OK **if parameters accepted** 1:CMD ERR **if full scale 2 not enabled** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: FRFS2=? **Required access level:** 2	*min <> max (units)* **if accepted** 1:CMD ERR **if full scale 2 not enabled** 5:ACCESS ERR **if insufficient access level**
VTMUT **(Volume Totalizer Measure Unit Type) Reads or sets the type of measure units relative to the totalizers. Possible values are:** 0:VM **= Volume, Metric** 1:WM **= Weight, Metric** 2:VI **= Volume, Imperial or American** 3:WI **= Weight, Imperial or American**	**Read command:** VTMUT? **Required access level:** 2	**Type of measure unit if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: VTMUT=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: VTMUT=? **Required access level:** 2	**List of** *num:description* **if accepted** 5:ACCESS ERR **if insufficient access level**
VTMUV **(Volume Totalizer Measure Unit Value) Reads or sets the value of measure units relative to the totalizers.**	**Read command:** VTMUV? **Required access level:** 2	**Value of measure unit if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: VTMUV=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of rang** 5:ACCESS ERR **if insufficient access level**
	Help command: VTMUV=? **Required access level:** 2	**List of** *num:description* **if accepted** 5:ACCESS ERR **if insufficient access level**
VTDPP **(Volume Totalizers Decimal**	**Read command:** VTDPP? **Required access level:** 2	**Value of decimal digits if accepted** 5:ACCESS ERR **if insufficient access level**

Point Position) Reads or sets the value representing the number of decimal digits for representing the volume totalizers.	**Set command:** VTDPP=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: VTDPP=? **Required access level:** 2	*min <> max* **if accepted** 5:ACCESS ERR **if insufficient access level**
CH1PV **(CHannel** 1 **Pulse** V**alue) Reads or sets the value representing the volume of one totalization pulse for the channel 1.**	**Read command:** CH1PV? **Required access level:** 2	**Value of volume pulse for channel 1** 1:CMD ERR **if channel 1 not enabled** 5:ACCESS ERR **if insufficient access level**
	Set command: CH1PV=*n* **Required access level:** 2	0:OK **if parameters accepted** 1:CMD ERR **if channel 1 not enabled** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: CH1PV=? **Required access level:** 2	*min <> max (units)* **if accepted** 1:CMD ERR **if channel 1 not enabled** 5:ACCESS ERR **if insufficient access level**
CH2PV **(CHannel** 2 **Pulse** V**alue) Reads or sets the value representing the volume of one totalization pulse for the channel 2.**	**Read command:** CH2PV? **Required access level:** 2	**Value of volume pulse for channel 2** 1:CMD ERR **if channel 2 not enabled** 5:ACCESS ERR **if insufficient access level**
	Set command: CH2PV=*n* **Required access level:** 2	0:OK **if parameters accepted** 1:CMD ERR **if channel 2 not enabled** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: CH2PV=? **Required access level:** 2	*min <> max (units)* **if accepted** 1:CMD ERR **if channel 2 not enabled** 5:ACCESS ERR **if insufficient access level**
CH1PT **(CHannel** 1 **Pulse Time) Reads or sets the value representing the time duration pulse for the channel 1.**	**Read command:** CH1PT? **Required access level:** 2	**Value of pulse time for channel 1** 1:CMD ERR **if channel 1 not enabled** 5:ACCESS ERR **if insufficient access level**
	Set command: CH1PT=*n* **Required access level:** 2	0:OK **if parameters accepted** 1:CMD ERR **if channel 1 not enabled** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: CH1PT=? **Required access level:** 2	*min <> max (units)* **if accepted** 1:CMD ERR **if channel 1 not enabled** 5:ACCESS ERR **if insufficient access level**
CH2PT **(CHannel** 2 **Pulse Time) Reads or sets the value representing the time duration pulse for the channel 2.**	**Read command:** CH2PT? **Required access level:** 2	**Value of pulse time for channel 2** 1:CMD ERR **if channel 2 not enabled** 5:ACCESS ERR **if insufficient access level**
	Set command: CH2PT=*n* **Required access level:** 2	0:OK **if parameters accepted** 1:CMD ERR **if channel 2 not enabled** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: CH2PT =? **Required access level:** 2	*min <> max (units)* **if accepted** 1:CMD ERR **if channel 2 not enabled** 5:ACCESS ERR **if insufficient access level**
VMSGC **(Volume to Mass Specific Gravity Coefficient) Reads or sets the value representing the transformation coefficient between volume and mass.**	**Read command:** VMSGK? **Required access level:** 2	**Value of volume-to-mass coefficient** 1:CMD ERR **if weight units not enabled** 5:ACCESS ERR **if insufficient access level**
	Set command: VMSGK=*n* **Required access level:** 2	0:OK **if parameters accepted** 1:CMD ERR **if weight units not enabled** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: VMSGK=? **Required access level:** 2	*min <> max (units)* **if accepted** 1:CMD ERR **if weight units not enabled** 5:ACCESS ERR **if insufficient access level**
A1CSY **(Analog input** 1 **Custom SYmbol) Reads or sets the value representing the custom measure unit used for the analog input 1. If the values is to be set, a fixed 3-characters length string must be supplied.**	**Read command:** A1CSY? **Required access level:** 2	**Value of analog input 1 measure unit symbol** 1:CMD ERR **if analog input 1 not enabled** 5:ACCESS ERR **if insufficient access level**
	Set command: A1CSY=*string* **Required access level:** 2	0:OK **if parameters accepted** 1:CMD ERR **if analog input 1 not enabled** 2:PARAM ERR **if *string* length is not 3 chars** 5:ACCESS ERR **if insufficient access level**
	Help command: A1CSY=? **Required access level:** 2	*3 CHR STRING* **if accepted** 1:CMD ERR **if analog input 1 not enabled** 5:ACCESS ERR **if insufficient access level**
A1MUV **(Analog input** 1 **Measure Unit Value) Reads or sets the value of measure units relative to the totalizers.**	**Read command:** A1MUV? **Required access level:** 2	**Value of measure unit if accepted** 1:CMD ERR **if analog input 1 not enabled** 5:ACCESS ERR **if insufficient access level**
	Set command: A1MUV=*n* **Required access level:** 2	0:OK **if parameters accepted** 1:CMD ERR **if analog input 1 not enabled** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: A1MUV=? **Required access level:** 2	**List of** *num:description* **if accepted** 1:CMD ERR **if analog input 1 not enabled** 5:ACCESS ERR **if insufficient access level**

A1SSV **(Analog input 1 Start Scale Value) Reads or sets the value of the start scale point for the analog input 1.**	**Read command:** A1SSV? **Required access level:** 2	**Start scale value of analog in.1 if accepted** 1:CMD ERR **if analog input 1 not enabled** 5:ACCESS ERR **if insufficient access level**
	Set command: A1SSV=*n* **Required access level:** 2	0:OK **if parameters accepted** 1:CMD ERR **if analog input 1 not enabled** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: A1SSV=? **Required access level:** 2	*min <> max (units)* **if accepted** 1:CMD ERR **if analog input 1 not enabled** 5:ACCESS ERR **if insufficient access level**
A1FSV **(Analog input 1 Full Scale Value) Reads or sets the value of the full scale point for the analog input 1.**	**Read command:** A1FSV? **Required access level:** 2	**Full scale value of analog in.1 if accepted** 1:CMD ERR **if analog input 1 not enabled** 5:ACCESS ERR **if insufficient access level**
	Set command: A1FSV=*n* **Required access level:** 2	0:OK **if parameters accepted** 1:CMD ERR **if analog input 1 not enabled** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: A1FSV=? **Required access level:** 2	*min <> max (units)* **if accepted** 1:CMD ERR **if analog input 1 not enabled** 5:ACCESS ERR **if insufficient access level**

COMMANDS AND FUNCTIONS EQUIVALENT TO THE "3-MEASURES" MENU		
Name and description	**Modes**	**Returned values or codes**
MFTCV **(Measure Filter Time Constant Value) Reads or sets the value of the measure filter time constant.**	**Read command:** MFTCV? **Required access level:** 2	**Value of measure time constant if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: MFTCV=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: MFTCV=? **Required access level:** 2	*min <> max (units)* **if accepted** 5:ACCESS ERR **if insufficient access level**
MFSKT **(Measure Filter SKip Threshold) Reads or sets the value of the measure filter skip threshold.**	**Read command:** MFSKT? **Required access level:** 2	**Value of measure skip threshold if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: MFSKT=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: MFSKT=? **Required access level:** 2	*min <> max (units)* **if accepted** 5:ACCESS ERR **if insufficient access level**
MFMXT **(Measure Filter MaX variation Threshold) Reads or sets the value of the measure filter maximum variation threshold.**	**Read command:** MFMXT? **Required access level:** 2	**Value of max. variation threshold if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: MFMXT=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: MFMXT=? **Required access level:** 2	*min <> max (units)* **if accepted** 5:ACCESS ERR **if insufficient access level**
MFCUT **(Measure Filter Cut-off Threshold) Reads or sets the value of the measure filter cut-off threshold.**	**Read command:** MFCUT? **Required access level:** 2	**Value of measure cut-off thr. if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: MFCUT=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: MFCUT=? **Required access level:** 2	*min <> max (units)* **if accepted** 5:ACCESS ERR **if insufficient access level**
ACALE **(Auto-CALibration Enable) Enables or disables the auto-calibration feature of the instrument.**	**Read command:** ACALE? **Required access level:** 2	**Enable/disable status if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: ACALE=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: ACALE=? **Required access level:** 2	**List of** *num:description* **if accepted** 5:ACCESS ERR **if insufficient access level**

ARNGE **(Auto-RaNGe Enable) Enables or disables the auto-range feature of the instrument.**	**Read command:** ARNGE? **Required access level:** 2	**Enable/disable status if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: ARNGE=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: ARNGE=? **Required access level:** 2	**List of** *num:description* **if accepted** 5:ACCESS ERR **if insufficient access level**
ENSVE **(ENergy-SaVing Enable) Enables or disables the energy-saving feature of the instrument.**	**Read command:** ENSVE? **Required access level:** 2	**Enable/disable status if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: ENSVE=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: ENSVE=? **Required access level:** 2	**List of** *num:description* **if accepted** 5:ACCESS ERR **if insufficient access level**
ENSST **(ENergy-Saving Sample Time) Reads or sets the value of the measure sampling time when the energy-saving feature is active.**	**Read command:** ENSST? **Required access level:** 2	**Value of measure sampling time if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: ENSST=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: ENSST=? **Required access level:** 2	*min <> max (units)* **if accepted** 5:ACCESS ERR **if insufficient access level**

COMMANDS AND FUNCTIONS EQUIVALENT TO THE "4-ALARMS" MENU		
Name and description	**Modes**	**Returned values or codes**
FRAXP **(Flow Rate Alarm maX Positive threshold) Reads or sets the value of the maximum positive flow rate alarm threshold.**	**Read command:** FRAXP? **Required access level:** 2	**Value of positive max threshold if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: FRAXP=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: FRAXP=? **Required access level:** 2	*min <> max (units)* **if accepted** 5:ACCESS ERR **if insufficient access level**
FRAXN **(Flow Rate Alarm maX Negative threshold) Reads or sets the value of the maximum positive flow rate alarm threshold.**	**Read command:** FRAXN? **Required access level:** 2	**Value of negative max threshold if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: FRAXN=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: FRAXN=? **Required access level:** 2	*min <> max (units)* **if accepted** 5:ACCESS ERR **if insufficient access level**
FRANP **(Flow Rate Alarm miN Positive threshold) Reads or sets the value of the minimum positive flow rate alarm threshold.**	**Read command:** FRANP? **Required access level:** 2	**Value of positive min threshold if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: FRAXP=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: FRAXP=? **Required access level:** 2	*min <> max (units)* **if accepted** 5:ACCESS ERR **if insufficient access level**
FRANN **(Flow Rate Alarm miN Negative threshold) Reads or sets the value of the minimum positive flow rate alarm threshold.**	**Read command:** FRANN? **Required access level:** 2	**Value of negative min threshold if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: FRANN=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: FRANN=? **Required access level:** 2	*min <> max (units)* **if accepted** 5:ACCESS ERR **if insufficient access level**
ATHYS **(Alarm Thresholds HYSteresis) Reads or sets the value of the alarm threshold hysteresis.**	**Read command:** ATHYS? **Required access level:** 2	**Value of hysteresis if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: ATHYS=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: ATHYS=? **Required access level:** 2	*min <> max (units)* **if accepted** 5:ACCESS ERR **if insufficient access level**
EPDAT **(Empty Pipe Detection Alarm Threshold) Reads or sets the value of the alarm threshold for the empty pipe detection system.**	**Read command:** EPDAT? **Required access level:** 2	**Value of empty pipe threshold if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: EPDAT=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: EPDAT=? **Required access level:** 2	*min <> max (units)* **if accepted** 5:ACCESS ERR **if insufficient access level**

OCACV **(Output Current Alarm Condition Value) Reads or sets the value of the alarm value at which the current output will be set in case of error conditions. Valid only if the current output is installed and enabled.**	**Read command:** OCACV? **Required access level:** 2	**Value of alarm current output if accepted** 1:CMD ERR **if current output not enabled** 5:ACCESS ERR **if insufficient access level**
	Set command: OCACV=*n* **Required access level:** 2	0:OK **if parameters accepted** 1:CMD ERR **if current output not enabled** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: OCACV=? **Required access level:** 2	*min <> max (units)* **if accepted** 1:CMD ERR **if current output not enabled** . 5:ACCESS ERR **if insufficient access level**

COMMANDS AND FUNCTIONS EQUIVALENT TO THE "5-INPUTS" MENU		
Name and description	**Modes**	**Returned values or codes**
VTTPE **(Volume Totalizer Total Positive reset Enable) Enables or disables the consent for resetting the total positive volume totalizer when receiving an external signal on the digital input. Valid only if the digital input is installed and enabled.**	**Read command:** VTTPE? **Required access level:** 2	**Status of reset consent if accepted** 1:CMD ERR **if digital input not enabled** 5:ACCESS ERR **if insufficient access level**
	Set command: VTTPE=*n* **Required access level:** 2	0:OK **if parameters accepted** 1:CMD ERR **if digital input not enabled** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: VTTPE=? **Required access level:** 2	**List of** *num:description* **if accepted** 1:CMD ERR **if digital input not enabled** 5:ACCESS ERR **if insufficient access level**
VTPPE **(Volume Totalizer Partial Positive reset Enable) Enables or disables the consent for resetting the partial positive volume totalizer when receiving an external signal on the digital input. Valid only if the digital input is installed and enabled.**	**Read command:** VTPPE? **Required access level:** 2	**Status of reset consent if accepted** 1:CMD ERR **if digital input not enabled** 5:ACCESS ERR **if insufficient access level**
	Set command: VTPPE=*n* **Required access level:** 2	0:OK **if parameters accepted** 1:CMD ERR **if digital input not enabled** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: VTPPE=? **Required access level:** 2	**List of** *num:description* **if accepted** 1:CMD ERR **if digital input not enabled** 5:ACCESS ERR **if insufficient access level**
VTTNE **(Volume Totalizer Total Negative reset Enable) Enables or disables the consent for resetting the total negative volume totalizer when receiving an external signal on the digital input. Valid only if the digital input is installed and enabled.**	**Read command:** VTTNE? **Required access level:** 2	**Status of reset consent if accepted** 1:CMD ERR **if digital input not enabled** 5:ACCESS ERR **if insufficient access level**
	Set command: VTTNE=*n* **Required access level:** 2	0:OK **if parameters accepted** 1:CMD ERR **if digital input not enabled** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: VTTNE=? **Required access level:** 2	**List of** *num:description* **if accepted** 1:CMD ERR **if digital input not enabled** 5:ACCESS ERR **if insufficient access level**
VTPNE **(Volume Totalizer Partial Negative reset Enable) Enables or disables the consent for resetting the partial negative volume totalizer when receiving an external signal on the digital input. Valid only if the digital input is installed and enabled.**	**Read command:** VTPNE? **Required access level:** 2	**Status of reset consent if accepted** 1:CMD ERR **if digital input not enabled** 5:ACCESS ERR **if insufficient access level**
	Set command: VTPNE=*n* **Required access level:** 2	0:OK **if parameters accepted** 1:CMD ERR **if digital input not enabled** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: VTPNE=? **Required access level:** 2	**List of** *num:description* **if accepted** 1:CMD ERR **if digital input not enabled** 5:ACCESS ERR **if insufficient access level**
TCLIE **(Totalizers Count Lock Input Enable) Enables or disables the consent for locking the totalizers when receiving an external signal on the digital input. Valid only if the digital input is installed and enabled.**	**Read command:** TCLIE? **Required access level:** 2	**Status of locking consent if accepted** 1:CMD ERR **if digital input not enabled** 5:ACCESS ERR **if insufficient access level**
	Set command: TCLIE=*n* **Required access level:** 2	0:OK **if parameters accepted** 1:CMD ERR **if digital input not enabled** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: TCLIE=? **Required access level:** 2	**List of** *num:description* **if accepted** 1:CMD ERR **if digital input not enabled** 5:ACCESS ERR **if insufficient access level**
CALIE **(CALibration Input Enable) Enables or disables the consent for calibrating when receiving an external signal on the digital input. Valid only if the digital input is installed and enabled.**	**Read command:** CALIE? **Required access level:** 2	**Status of calibration consent if accepted** 1:CMD ERR **if digital input not enabled**
	Set command: CALIE=*n* **Required access level:** 2	0:OK **if parameters accepted** 1:CMD ERR **if digital input not enabled** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: CALIE=? **Required access level:** 2	**List of** *num:description* **if accepted** 1:CMD ERR **if digital input not enabled** 5:ACCESS ERR **if insufficient access level**

SRCIE **(Scale Range Change Input Enable) Enables or disables the consent for changing the scale range when receiving an external signal on the digital input. Valid only if the digital input is installed and enabled.**	**Read command:** SRCIE? **Required access level:** 2	**Status of range change consent if accepted** 1:CMD ERR **if digital input not enabled** 5:ACCESS ERR **if insufficient access level**
	Set command: SRCIE=*n* **Required access level:** 2	0:OK **if parameters accepted** 1:CMD ERR **if digital input not enabled** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: SRCIE=? **Required access level:** 2	**List of** *num:description* **if accepted** 1:CMD ERR **if digital input not enabled** 5:ACCESS ERR **if insufficient access level**
WKUIE **(WaKe-Up Input Enable) Enables or disables the consent for waking-up from the energy saving standby mode when receiving an external signal on the digital input. Valid only if the digital input is installed and enabled.**	**Read command:** WKUIE? **Required access level:** 2	**Status of wake-up consent if accepted** 1:CMD ERR **if digital input not enabled** 5:ACCESS ERR **if insufficient access level**
	Set command: WKUIE =*n* **Required access level:** 2	0:OK **if parameters accepted** 1:CMD ERR **if digital input not enabled** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: WKUIE =? **Required access level:** 2	**List of** *num:description* **if accepted** 1:CMD ERR **if digital input not enabled** 5:ACCESS ERR **if insufficient access level**

COMMANDS AND FUNCTIONS EQUIVALENT TO THE "6-OUTPUTS" MENU		
Name and description	**Modes**	**Returned values or codes**
OUT1F **(OUTput 1 Function) Sets the function type related to the digital output 1. Valid only if the digital outputs are installed and enabled.**	**Read command:** OUT1F? **Required access level:** 2	**Type of function if accepted** 1:CMD ERR **if digital outputs not enabled** 5:ACCESS ERR **if insufficient access level**
	Set command: OUT1F=*n* **Required access level:** 2	0:OK **if parameters accepted** 1:CMD ERR **if digital outputs not enabled** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: OUT1F=? **Required access level:** 2	**List of** *num:description* **if accepted** 1:CMD ERR **if digital outputs not enabled** 5:ACCESS ERR **if insufficient access level**
OUT2F **(OUTput 2 Function) Sets the function type related to the digital output 2. Valid only if the digital outputs are installed and enabled.**	**Read command:** OUT2F? **Required access level:** 2	**Type of function if accepted** 1:CMD ERR **if digital outputs not enabled** 5:ACCESS ERR **if insufficient access level**
	Set command: OUT2F=*n* **Required access level:** 2	0:OK **if parameters accepted** 1:CMD ERR **if digital outputs not enabled** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: OUT2F=? **Required access level:** 2	**List of** *num:description* **if accepted** 1:CMD ERR **if digital outputs not enabled** 5:ACCESS ERR **if insufficient access level**
CO1FS **(Current Output 1 Full Scale) Sets the full scale value for the current output 1: it can be 20 or 22 mA. Valid only if the current output1 is installed and enabled.**	**Read command:** CO1FS? **Required access level:** 2	**Full scale value if accepted** 1:CMD ERR **if current output not enabled** 5:ACCESS ERR **if insufficient access level**
	Set command: CO1FS=*n* **Required access level:** 2	0:OK **if parameters accepted** 1:CMD ERR **if current output not enabled** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: CO1FS=? **Required access level:** 2	**List of** *num:description* **if accepted** 1:CMD ERR **if current output not enabled** 5:ACCESS ERR **if insufficient access level**
CO1SS **(Current Output 1 Start Scale) Sets the start scale value for the current output 1: it can be 0 or 4 mA. Valid only if the current output1 is installed and enabled.**	**Read command:** CO1SS? **Required access level:** 2	**Start scale value if accepted** 1:CMD ERR **if current output not enabled** 5:ACCESS ERR **if insufficient access level**
	Set command: CO1SS=*n* **Required access level:** 2	0:OK **if parameters accepted** 1:CMD ERR **if current output not enabled** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: CO1SS=? **Required access level:** 2	**List of** *num:description* **if accepted** 1:CMD ERR **if current output not enabled** 5:ACCESS ERR **if insufficient access level**
CO1FM **(Current Output 1 Field Mode) Sets the field mode value for the current output 1: it can be +, -, +/-, –0+. Valid only if the current output1 is installed and enabled.**	**Read command:** CO1FM? **Required access level:** 2	**Field mode for current output 1 if accepted** 1:CMD ERR **if current output not enabled**
	Set command: CO1FM=*n* **Required access level:** 2	0:OK **if parameters accepted** 1:CMD ERR **if current output not enabled** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: CO1FM=? **Required access level:** 2	**List of** *num:description* **if accepted** 1:CMD ERR **if current output not enabled** 5:ACCESS ERR **if insufficient access level**

COMMANDS AND FUNCTIONS EQUIVALENT TO THE "7-COMMUNICATION" MENU		
Name and description	**Modes**	**Returned values or codes**
IF2PT **(**IF2 P**rotocol** T**ype) Reads or sets the protocol type for the IF2 port: it can be DPP or HTP.**	**Read command:** IF2PT? **Required access level:** 2	**Type of protocol if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: IF2PT=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: IF2PT=? **Required access level:** 2	**List of** *num:description* **if accepted** 5:ACCESS ERR **if insufficient access level**
DVADR **(**D**e**V**ice** AD**d**R**ess) Reads or sets the device address. Valid only if the RS232 port is installed and enabled.**	**Read command:** DVADR? **Required access level:** 2	**Device address if accepted** 1:CMD ERR **if RS232 port not enabled** 5:ACCESS ERR **if insufficient access level**
	Set command: DVADR=*n* **Required access level:** 2	0:OK **if parameters accepted** 1:CMD ERR **if RS232 port not enabled** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: DVADR=? **Required access level:** 2	*min <> max* **if accepted** 1:CMD ERR **if RS232 port not enabled** 5:ACCESS ERR **if insufficient access level**
232SP **(rs**232 SP**eed) Reads or sets the RS232 speed in bps. Valid only if the RS232 port is installed and enabled.**	**Read command:** 232SP? **Required access level:** 2	**RS232 speed if accepted** 1:CMD ERR **if RS232 port not enabled** 5:ACCESS ERR **if insufficient access level**
	Set command: 232SP=*n* **Required access level:** 2	0:OK **if parameters accepted** 1:CMD ERR **if RS232 port not enabled** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: 232SP=? **Required access level:** 2	**List of** *num:description* **if accepted** 1:CMD ERR **if RS232 port not enabled** 5:ACCESS ERR **if insufficient access level**
232PT **(rs**232 P**rotocol** T**ype) Reads or sets the protocol type for the RS232 port: it can be DPP or HTP. Valid only if the RS232 port is installed and enabled.**	**Read command:** 232PT? **Required access level:** 2	**RS232 protocol type if accepted** 1:CMD ERR **if RS232 port not enabled** 5:ACCESS ERR **if insufficient access level**
	Set command: 232PT=*n* **Required access level:** 2	0:OK **if parameters accepted** 1:CMD ERR **if RS232 port not enabled** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: 232PT=? **Required access level:** 2	**List of** *num:description* **if accepted** 1:CMD ERR **if RS232 port not enabled** 5:ACCESS ERR **if insufficient access level**

COMMANDS AND FUNCTIONS EQUIVALENT TO THE "8-DISPLAY" MENU		
Name and description	**Modes**	**Returned values or codes**
LLANG **(**L**ayout** LANG**uage) Reads or sets the layout language used for all the display messages.**	**Read command:** LLANG? **Required access level:** 2	**Layout language if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: LLANG=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: LLANG=? **Required access level:** 2	**List of** *num:description* **if accepted** 5:ACCESS ERR **if insufficient access level**

VTTPR **(Volume Totalizer Total Positive Reset) Resets the total positive volume totalizer.**	**Read command: NOT SUPPORTED**	1:CMD ERR **if read command is issued** 5:ACCESS ERR **if insufficient access level**
	Set command: VTTPR=1 **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* not equal to 1** 5:ACCESS ERR **if insufficient access level**
	Help command: VTTPR=? **Required access level:** 2	1:EXECUTE **if accepted** 5:ACCESS ERR **if insufficient access level**
VTPPR **(Volume Totalizer Partial Positive Reset) Resets the partial positive volume totalizer.**	**Read command: NOT SUPPORTED**	1:CMD ERR **if read command is issued** 5:ACCESS ERR **if insufficient access level**
	Set command: VTPPR=1 **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* not equal to 1** 5:ACCESS ERR **if insufficient access level**
	Help command: VTPPR=? **Required access level:** 2	1:EXECUTE **if accepted** 5:ACCESS ERR **if insufficient access level**
VTTNR **(Volume Totalizer Total Negative Reset) Resets the total negative volume totalizer.**	**Read command: NOT SUPPORTED**	1:CMD ERR **if read command is issued** 5:ACCESS ERR **if insufficient access level**
	Set command: VTTNR=1 **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* not equal to 1** 5:ACCESS ERR **if insufficient access level**
	Help command: VTTNR=? **Required access level:** 2	1:EXECUTE **if accepted** 5:ACCESS ERR **if insufficient access level**
VTPNR **(Volume Totalizer Partial Negative Reset) Resets the partial negative volume totalizer.**	**Read command: NOT SUPPORTED**	1:CMD ERR **if read command is issued** 5:ACCESS ERR **if insufficient access level**
	Set command: VTPNR=1 **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* not equal to 1** 5:ACCESS ERR **if insufficient access level**
	Help command: VTPNR=? **Required access level:** 2	1:EXECUTE **if accepted** 5:ACCESS ERR **if insufficient access level**
ENSDT **(ENergy Saving Display Time) Reads or sets the value of the display on time used in the energy saving mode. This time is also used to wait for the incoming of new sms if they are enabled.**	**Read command:** ENSDT? **Required access level:** 2	**Value of display on time if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: ENSDT=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: ENSDT=? **Required access level:** 2	*min <> max (units)* **if accepted** 5:ACCESS ERR **if insufficient access level**
QSTME **(Quick STart Menu Enable)**	**Read command:** QSTME? **Required access level:** 2	**Status of functions if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: QSTME =*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: QSTME =? **Required access level:** 2	**List of** *num:description* **if accepted** 5:ACCESS ERR **if insufficient access level**
TTNVE **(TOTalizers Net Value Enable) enable disable the visualization of Net totalizer**	**Read command:** TTNVE? **Required access level:** 2	**Status of functions if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: TTNVE =*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: TTNVE =? **Required access level:** 2	**List of** *num:description* **if accepted** 5:ACCESS ERR **if insufficient access level**
TCMDE **(Totalizers Currency Mode Display Enable) Enables or disables the displaying of the currency values for the totalizers.**	**Read command:** TCMDE? **Required access level:** 2	**Status of currency mode display if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: TCMDE=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: TCMDE=? **Required access level:** 2	**List of** *num:description* **if accepted** 5:ACCESS ERR **if insufficient access level**
CUSYT **(CUrrency SYmbol Type) Reads or sets the type of currency symbol used to represent the values converted from the totalizers.**	**Read command:** CUSYT? **Required access level:** 2	**Type of currency symbol if accepted** 1:CMD ERR **if currency not enable** 5:ACCESS ERR **if insufficient access level**
	Set command: CUSYT=*n* **Required access level:** 2	0:OK **if parameters accepted** 1:CMD ERR **if currency not enabled** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**

	Help command: CUSYT=? **Required access level:** 2	**List of** *num:description* **if accepted** 1:CMD ERR **if currency not enabled** 5:ACCESS ERR **if insufficient access level**
CUCSS **(CUrrency CuStom Symbol) Reads or sets the custom symbol used to represent the currency. If the values is to be set, a fixed 3-characters length string must be supplied.**	**Read command:** CUCSS? **Required access level:** 2	**Custom currency symbol if accepted** 1:CMD ERR **if currency not enabled** 5:ACCESS ERR **if insufficient access level**
	Set command: CUCSS=*string* **Required access level:** 2	0:OK **if parameters accepted** 1:CMD ERR **if currency not enabled** 2:PARAM ERR **if *string* length is not 3 chars** 5:ACCESS ERR **if insufficient access level**
	Help command: CUCSS=? **Required access level:** 2	*3 CHR STRING* **if accepted** 1:CMD ERR **if currency not enabled** 5:ACCESS ERR **if insufficient access level**
CUDEC **(CUrrency DEcimal Ciphers) Reads or sets the value of the decimal ciphers for representing the volume totalizers converted to currency.**	**Read command:** CUDEC? **Required access level:** 2	**Value of decimal ciphers if accepted** 1:CMD ERR **if currency not enabled** 5:ACCESS ERR **if insufficient access level**
	Set command: CUDEC=*n* **Required access level:** 2	0:OK **if parameters accepted** 1:CMD ERR **if currency not enabled** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: CUDEC=? **Required access level:** 2	*min <> max* **if accepted** 1:CMD ERR **if currency not enable** 5:ACCESS ERR **if insufficient access level**
CUPCF **(CUrrency Positive Conversion Factor) Reads or sets the value of the conversion factor coefficient used to convert the partial positive totalizer to currency.**	**Read command:** CUPCF? **Required access level:** 2	**Value of positive conv. factor if accepted** 1:CMD ERR **if currency not enabled** 5:ACCESS ERR **if insufficient access level**
	Set command: CUPCF=*n* **Required access level:** 2	0:OK **if parameters accepted** 1:CMD ERR **if currency not enabled** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: CUPCF=? **Required access level:** 2	*min <> max (units)* **if accepted** 1:CMD ERR **if currency not enable** 5:ACCESS ERR **if insufficient access level**
CUNCF **(CUrrency Negative Conversion Factor) Reads or sets the value of the conversion factor coefficient used to convert the partial negative totalizer to currency.**	**Read command:** CUNCF? **Required access level:** 2	**Value of negative conv. factor if accepted** 1:CMD ERR **if currency not enabled** 5:ACCESS ERR **if insufficient access level**
	Set command: CUNCF=*n* **Required access level:** 2	0:OK **if parameters accepted** 1:CMD ERR **if currency not enabled** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: CUNCF=? **Required access level:** 2	*min <> max (units)* **if accepted** 1:CMD ERR **if currency not enabled** 5:ACCESS ERR **if insufficient access level**

COMMANDS AND FUNCTIONS EQUIVALENT TO THE "10-DIAGNOSTIC" MENU		
Name and description	**Modes**	**Returned values or codes**
CALIC **(CALibration Immediate Command) performs an immediate calibration cycle and reads the status of the operation (1 = in progress , 0 = terminated).**	**Read command:** CALIC? **Required access level:** 2	**Status of calibration operation if accepted**
	Set command: CALIC=1 **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* not equal to 1**
	Help command: CALIC=? **Required access level:** 2	1:EXECUTE **if accepted**
ATSIC **(Auto-TeSt Immediate Command) Performs an auto-test cycle.**	**Read command: NOT SUPPORTED**	1:CMD ERR **if command is issued**
	Set command: ATSIC=1 **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* not equal to 1** 5:ACCESS ERR **if insufficient access level**
	Help command: ATSIC=? **Required access level:** 2	1:EXECUTE **if accepted** 5:ACCESS ERR **if insufficient access level**
MSIEN **(Measure SImulation ENable) Enables or disables the measure simulation function.**	**Read command:** MSIEN? **Required access level:** 2	**Status of measure sim. function if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: MSIEN=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: MSIEN=? **Required access level:** 2	**List of** *num:description* **if accepted** 5:ACCESS ERR **if insufficient access level**
STBYC **(STandBY Command) set converter in stand-by mode**	**Read command:** STBYC =? **Required access level:** 2	**List of** *num:description* **if accepted** 5:ACCESS ERR **if insufficient access level**

Once this command is issued , the only way to wakeup the converter is by the keyboard	**Set command:** STBYC =*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**

COMMANDS AND FUNCTIONS EQUIVALENT TO THE "11-INTERNAL DATA" MENU		
Name and description	**Modes**	**Returned values or codes**
L2ACD **(Level 2 Access CoDe) Reads or sets the level 2 programmable access code. It can be set to zero to disable all L2 access requests.** WARNING: do not forget the code entered!	**Read command:** L2ACD? **Required access level:** 2	**Programmed code if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: L2ACD=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: L2ACD=? **Required access level:** 2	*min <> max* **if accepted** 5:ACCESS ERR **if insufficient access level**
LFDIC **(Load Factory Defaults Immediate Command) Loads the factory default parameters.** WARNING: all the current working parameters eventually modified by the user will be lost!	**Read command: NOT SUPPORTED**	1:CMD ERR **if command is issued**
	Set command: LFDIC=1 **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* not equal to 1** 3:EXEC ERR **if default data corrupted** 5:ACCESS ERR **if insufficient access level**
	Help command: LFDIC=? **Required access level:** 2	1:EXECUTE **if accepted** 5:ACCESS ERR **if insufficient access level**
LUDIC **(Load User Data Immediate Command) Loads the user preset parameters.** WARNING: all the current working parameters eventually modified by the user will be lost!	**Read command: NOT SUPPORTED**	1:CMD ERR **if command is issued**
	Set command: LUDIC=1 **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* not equal to 1** 3:EXEC ERR **if user data corrupted** 5:ACCESS ERR **if insufficient access level**
	Help command: LUDIC=? **Required access level:** 2	1:EXECUTE **if accepted** 5:ACCESS ERR **if insufficient access level**
SUDIC **(Save User Data Immediate Command) Saves the current working parameters as user preset data.**	**Read command: NOT SUPPORTED**	1:CMD ERR **if command is issued**
	Set command: SUDIC=1 **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* not equal to 1** 3:EXEC ERR **if memory failure** 5:ACCESS ERR **if insufficient access level**
	Help command: SUDIC=? **Required access level:** 2	1:EXECUTE **if accepted** 5:ACCESS ERR **if insufficient access level**
SFDIC **(Save Factory Defaults Immediate Command) Saves the current working parameters as factory defaults preset data.**	**Read command: NOT SUPPORTED**	1:CMD ERR **if command is issued**
	Set command: SFDIC=1 **Required access level:** 3	0:OK **if parameters accepted** 2:PARAM ERR **if *n* not equal to 1** 3:EXEC ERR **if memory failure** 5:ACCESS ERR **if insufficient access level**
	Help command: SFDIC=? **Required access level:** 2	1:EXECUTE **if accepted** 5:ACCESS ERR **if insufficient access level**
CMRIC **(Complete Memory Reset Immediate Command) Resets all the working parameters to the default values.** WARNING: all the current working parameters eventually modified by the user and the calibration coefficients will be lost!	**Read command: NOT SUPPORTED**	1:CMD ERR **if command is issued**
	Set command: CMRIC=1 **Required access level:** 4	0:OK **if parameters accepted** 2:PARAM ERR **if *n* not equal to 1** 3:EXEC ERR **if memory failure** 5:ACCESS ERR **if insufficient access level**
	Help command: CMRIC=? **Required access level:** 2	1:EXECUTE **if accepted** 5:ACCESS ERR **if insufficient access level**
SRNUM **(SeRial NUMber) Reads the device serial number.**	**Read command:** SRNUM? **Required access level:** 2	**Value of serial number if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: NOT SUPPORTED	1:CMD ERR **if command is issued**
	Help command: NOT SUPPORTED	1:CMD ERR **if command is issued**
TONTM **(Total ON TiMe) Reads the total functioning time of the device.**	**Read command:** TONTM? **Required access level:** 2	**Value of total on time if accepted**
	Set command: NOT SUPPORTED	1:CMD ERR **if command is issued**
	Help command: NOT SUPPORTED	1:CMD ERR **if command is issued**
CFFKT **(CoeFFicient KT) Reads or sets the value of the gain coefficient KT**	**Read command:** CFFKT? **Required access level:** 2	**Value of the coefficient if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: CFFKT=*n* **Required access level:** 3	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: CFFKT=? **Required access level:** 2	*min <> max* **if accepted** 5:ACCESS ERR **if insufficient access level**

CFFKR **(CoeFFicient** KR**) Reads or sets the value of the gain coefficient KR**	**Read command:** CFFKR? **Required access level:** 2	**Value of the coefficient if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: CFFKR=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: CFFKR=? **Required access level:** 2	*min <> max* **if accepted** 5:ACCESS ERR **if insufficient access level**
CFFKS **(CoeFFicient** KS**) Reads or sets the value of the gain coefficient KS**	**Read command:** CFFKS? **Required access level:** 2	**Value of the coefficient if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: CFFKS=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: CFFKS=? **Required access level:** 2	*min <> max* **if accepted** 5:ACCESS ERR **if insufficient access level**
CFFKZ **(CoeFFicient** KZ**) Reads or sets the value of the zero coefficient KZ**	**Read command:** CFFKZ? **Required access level:** 2	**Value of the coefficient if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: CFFKZ=*n* **Required access level:** 3	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: CFFKZ=? **Required access level:** 2	*min <> max* **if accepted** 5:ACCESS ERR **if insufficient access level**
ICALE **(Ignore** CAL**ibration Errors) Enables or disables the calibration errors recognition.**	**Read command:** ICALE? **Required access level:** 2	**Status of cal. error recognition if accepted** 5:ACCESS ERR **if insufficient access level**
	Set command: ICALE=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: ICALE=? **Required access level:** 2	**List of** *num:description* **if accepted** 5:ACCESS ERR **if insufficient access level**
C1CP1 **(Current output** 1 **Calibration Point** 1**) Reads or sets the value of the current output calibration point 1 (4 mA).**	**Read command:** C1CP1? **Required access level:** 2	**Value of the cal. point if accepted** 1:CMD ERR **if current output not enabled** 5:ACCESS ERR **if insufficient access level**
	Set command: C1CP1=*n* **Required access level:** 3	0:OK **if parameters accepted** 1:CMD ERR **if current output not enabled** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: C1CP1=? **Required access level:** 2	*min <> max* **if accepted** 1:CMD ERR **if current output not enabled** 5:ACCESS ERR **if insufficient access level**
C1CP2 **(Current output** 1 **Calibration Point** 2**) Reads or sets the value of the current output calibration point 2 (20 mA).**	**Read command:** C1CP2? **Required access level:** 2	**Value of the cal. point if accepted** 1:CMD ERR **if current output not enabled** 5:ACCESS ERR **if insufficient access level**
	Set command: C1CP2=*n* **Required access level:** 3	0:OK **if parameters accepted** 1:CMD ERR **if current output not enabled** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: C1CP2=? **Required access level:** 2	*min <> max* **if accepted** 1:CMD ERR **if current output not enabled** 5:ACCESS ERR **if insufficient access level**
A1CP1 **(Analog input** 1 **Calibration Point** 1**) Reads or sets the value of the current output calibration point 1 (4 mA or min. voltage value).**	**Read command:** A1CP1? **Required access level:** 2	**Value of the cal. point if accepted** 1:CMD ERR **if analog input not enabled** 5:ACCESS ERR **if insufficient access level**
	Set command: A1CP1=*n* **Required access level:** 3	0:OK **if parameters accepted** 1:CMD ERR **if analog input not enabled** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: A1CP1=? **Required access level:** 2	*min <> max* **if accepted** 1:CMD ERR **if analog input not enabled** 5:ACCESS ERR **if insufficient access level**
A1CP2 **(Analog input** 1 **Calibration Point** 2**) Reads or sets the value of the current output calibration point 2 (20 mA or max. voltage value).**	**Read command:** A1CP2? **Required access level:** 2	**Value of the cal. point if accepted** 1:CMD ERR **if analog input not enabled** 5:ACCESS ERR **if insufficient access level**
	Set command: A1CP2=*n* **Required access level:** 3	0:OK **if parameters accepted** 1:CMD ERR **if analog input not enabled** 2:PARAM ERR **if *n* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: A1CP2=? **Required access level:** 2	*min <> max* **if accepted** 1:CMD ERR **if analog input not enabled** 5:ACCESS ERR **if insufficient access level**

COMMANDS AND FUNCTIONS USED FOR READING THE PROCESS PARAMETERS

Name and description	Modes	Returned values or codes

FRSRN **(Flow Rate Scale Range Number) Reads the scale range currently in use on the instrument. Valid only if the dual-range function is active.**	**Read command:** FRSRN?	**Scale number: (1 or 2) if dual range active** 1:CMD ERR **if dual range not enabled**
	Set command: NOT SUPPORTED	1:CMD ERR **if command is issued**
	Help command: NOT SUPPORTED	1:CMD ERR **if command is issued**
FRVPC **(Flow Rate Value PerCent) Reads or sets the flow rate value in the percent form. Returns two comma-separated fields, the percent symbol and the numeric value. Percent value is referred to the full scale currently active. The flow rate value can be set only when the simulation mode is active.**	**Read command:** FRVPC?	**Returns *%,value* string**
	Set command: FRVPC=*n* **Required access level:** 2	0:OK **if parameters accepted** 2:PARAM ERR **if *n* out of range or simulation mode not enabled**
	Help command: FRVPC=?	*min <> max (%)*
FRVTU **(Flow Rate Value Technical Units) Reads the flow rate value and its technical measure unit. Returns two comma-separated fields, the technical unit symbol and the numeric value.**	**Read command:** FRVTU?	**Returns *measure-unit,value* string**
	Set command: NOT SUPPORTED	1:CMD ERR **if command is issued**
	Help command: NOT SUPPORTED	1:CMD ERR **if command is issued**
VTTPV **(Volume Totalizer Total Positive Value) Reads the positive total totalizer value and its technical measure unit. Returns two comma-separated fields, the technical unit symbol and the numeric value.**	**Read command:** VTTPV?	**Returns *measure-unit,value* string**
	Set command: NOT SUPPORTED	1:CMD ERR **if command is issued**
	Help command: NOT SUPPORTED	1:CMD ERR **if command is issued**
VTPPV **(Volume Totalizer Partial Positive Value) Reads the positive partial totalizer value and its technical measure unit. Returns two comma-separated fields, the technical unit symbol and the numeric value.**	**Read command:** VTPPV?	**Returns *measure-unit,value* string**
	Set command: NOT SUPPORTED	1:CMD ERR **if command is issued**
	Help command: NOT SUPPORTED	1:CMD ERR **if command is issued**
VTTNV **(Volume Totalizer Total Negative Value) Reads the negative total totalizer value and its technical measure unit. Returns two comma-separated fields, the technical unit symbol and the numeric value.**	**Read command:** VTTNV?	**Returns *measure-unit,value* string**
	Set command: NOT SUPPORTED	1:CMD ERR **if command is issued**
	Help command: NOT SUPPORTED	1:CMD ERR **if command is issued**
VTPNV **(Volume Totalizer Partial Negative Value) Reads the negative partial totalizer value and its technical measure unit. Returns two comma-separated fields, the technical unit symbol and the numeric value.**	**Read command:** VTPNV?	**Returns *measure-unit,value* string**
	Set command: NOT SUPPORTED	1:CMD ERR **if command is issued**
	Help command: NOT SUPPORTED	1:CMD ERR **if command is issued**
CUTPV **(CUrrency Totalizer Positive Value) Reads the converted currency value relative to the partial positive totalizer and its currency unit. Returns two comma-separated fields, the unit symbol and the numeric value.**	**Read command:** CUTPV?	**Returns *currency-symbol,value* string** 1:CMD ERR **if currency mode not enabled**
	Set command: NOT SUPPORTED	1:CMD ERR **if command is issued**
	Help command: NOT SUPPORTED	1:CMD ERR **if command is issued**
CUTNV **(CUrrency Totalizer Negative Value) Reads the converted currency value relative to the partial negative totalizer and its currency unit. Returns two comma-separated fields, the unit symbol and the numeric value.**	**Read command:** CUTNV?	**Returns *currency-symbol,value* string** 1:CMD ERR **if currency mode not enabled**
	Set command: NOT SUPPORTED	1:CMD ERR **if command is issued**
	Help command: NOT SUPPORTED	1:CMD ERR **if command is issued**
AIN1V **(Analog INput 1 Value) Reads the value of the analog input 1 and its associated technical unit. Returns two comma-separated fields, the unit symbol and the numeric value.**	**Read command:** AIN1V?	**Returns *measure-unit,value* string** 1:CMD ERR **if analog input 1 not enabled**
	Set command: NOT SUPPORTED	1:CMD ERR **if command is issued**
	Help command: NOT SUPPORTED	1:CMD ERR **if command is issued**
BATTS **(BATTery Status) Reads the estimated residual capacity of the battery. Returns the percent symbol and value separated by the comma.**	**Read command:** BATTS?	**Returns *%,value* string**
	Set command: NOT SUPPORTED	1:CMD ERR **if command is issued**
	Help command: NOT SUPPORTED	1:CMD ERR **if command is issued**
ALARM **(ALARM status) Reads the alarm status of the instrument. Returns a CSV list of all active alarm.**	**Read command:** ALARM?	**Returns all active alarms in CSV format**
	Set command: NOT SUPPORTED	1:CMD ERR **if command is issued**
	Help command: NOT SUPPORTED	1:CMD ERR **if command is issued**

AUXILIARY SET COMMAND

Name and description	Modes	Returned values or codes
ACODE **(Access CODE)**	**Read command:** ACODE? **Required access level:** 2	**String value if command accepted** 1:CMD ERR **if GPRS not enabled** 5:ACCESS ERR **if insufficient access level**
	Set command: ACODE =*string* **Required access level:** 2	0:OK **if parameters accepted** 1:CMD ERR **if GPRS not enabled** 2:PARAM ERR **if *string* out of range** 5:ACCESS ERR **if insufficient access level**
	Help command: ACODE =? **Required access level:** 2	NUMERIC RANGE **if accepted** 1:CMD ERR **if GPRS not enabled** 5:ACCESS ERR **if insufficient access level**
MODSV **(MODel and Software Version)**	**Read command:** MODSV?	**String value if command accepted** 1:CMD ERR **if GPRS not enabled** 5:ACCESS ERR **if insufficient access level**