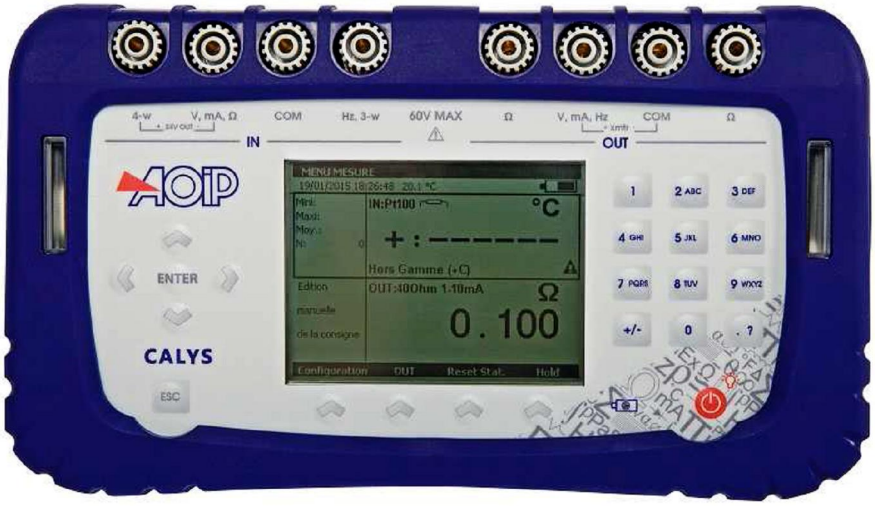
Calys 50/75/100

Calibrateur de Process Multifonction
Multifunction Process Calibrator



SCPI Commands

Nota :
In the command list , first part of of keywords, written in capital letters is mandatory. Part in "lower-case letters, is facultative. Key words between [ ] is "facultative".

1. commands List

CALibration

[:SENSe]

:VOLTage

:MEASure? {100mV | 1V | 10V| 50V} [,<*number of averaged measurements*>]

:GAIN {100mV | 1V | 10V| 50V}, < *gain value*>

:GAIN? {100mV | 1V | 10V| 50V}

:OFFSet {100mV | 1V | 10V| 50V}, <*offset value*>

:OFFSet? {100mV | 1V | 10V| 50V}

:DATE ? {100mV | 1V | 10V| 50V} // Last range adjustment date

:CURRent

:MEASure? [<*number of averaged measurements* >]

:GAIN < *gain value* >

:GAIN?

:OFFSet < *offset value* >

:OFFSet?

:DATE ? // Last range adjustment date

:RESistance

:MEASure? { 400 OHM | 4000 OHM} [,<*number of averaged measurements* >]

:GAIN { 400 OHM | 4000 OHM }, { WIRES_2|WIRES_3|WIRES_4}, < *gain value* >

:GAIN? { 400 OHM | 4000 OHM }, { WIRES_2|WIRES_3|WIRES_4}

:OFFSet { 400 OHM | 4000 OHM }, { WIRES_2|WIRES_3|WIRES_4}, < *offset value* >

:OFFSet? { 400 OHM | 4000 OHM }, { WIRES_2|WIRES_3|WIRES_4}

:DATE? {400 OHM | 4000 OHM }}, { WIRES_2|WIRES_3|WIRES_4}

:FREQuency

:MEASure?

:GAIN <*gain value*>

:GAIN?

:DATE ? // Last range adjustment date

:TCouple

:MEASure? < couple type >[, <Nber of measurement to be averaged>]

(performs TC measurement with internal CJC, but without applying CJC correction offset)

:RJUNction

:OFFSet <offset value>

:OFFSet?

CALibration

:SOURce

:VOLTage {100mV | 2V | 20V} , <*value to be sent with unit*>

:VOLTage

:ADJust? {100mV | 2V | 20V } // auto calibration of DACs 10 bits

:DISConnect // Disconnect generation output

:MEASure? {100mV | 2V | 20V} [,<*number of averaged measurements* >]

:GAIN {100mV | 2V | 20V} , < *gain value* >

:GAIN? {100mV | 2V | 20V}

:OFFSet {100mV | 2V | 20V} , < *offset value* >

:OFFSet? {100mV | 2V | 20V}

:DATE? {100mV | 2V | 20V } // Last range adjustment date

:DACLine? {100mV | 2V | 20V} // -> 0 and 50% of DACS range

:DACPoints? <1st point> [,<pts numbers] // DACs calibration points

:CURRent < *value to be sent with unit* >

:CURRent

:ADJust? // auto calibration of DACs 10 bits

:DISConnect // Déconnecter la sortie d'émission

:MEASure? [<*number of averaged measurements* >]

:GAIN < *gain value* >

:GAIN?

:OFFSet , < *offset value* >

:OFFSet?

:DATE ? // auto calibration of DACs 10 bits

:DACLine? // -> 0 and 50% of DACS range

:RESistance

:ADJust? { DAC_A | DAC_B } // Reply with calculated values

:ADJUST:DATE? { DAC_A | DAC_B}

[:AMPLitude] {40 OHM| 400 OHM | 4000 OHM}, { 1mA|10mA}, <*value to sent*>

:GAIN {40 OHM| 400 OHM | 4000 OHM}, { 1mA|10mA}, <*gain value*>

:GAIN? {40 OHM| 400 OHM | 4000 OHM}, { 1mA|10mA}

:OFFSet {40 OHM| 400 OHM | 4000 OHM}, { 1mA|10mA}, <*offset value*>

:OFFSet? {40 OHM| 400 OHM | 4000 OHM}, { 1mA|10mA}

:DATE ? {40 OHM| 400 OHM | 4000 OHM}, { 1mA|10mA} // Last Range adjustment date

:FREQuency? <frequency to be sent> respone : Frequency really sent

:FREQuency

:GAIN <*gain value*>

:GAIN?

:DATE ? // Last Range adjustment date

:TCouple

:MEASure? < couple type >[, <number of measurement to be averaged>]

(performs TC measurement with internal CJC, but without applying CJC correction offset)

:RJUNction

:OFFSet <offset value>

:OFFSet?

:DATE?

:DATE <*year*>, <*month*>,<*day*> // calibration date

:DATE?

:REPort <Chain of 0 to 50 characters between inverted commas> // Calibration report reference

:REPort?

MEASure

:VOLTage? [{100mV | 1V | 10V| 50V} [,<*number of measurement to be averaged*>]]

:CURRent? [<*number of measurement to be averaged* >]

:RESistance? [{ 400 OHM | 4000 OHM }[,< *number of measurement to be averaged* >]]
:FREQuency? [<*number of measurement to be averaged* >]
:PRESsure? [<*number of measurement to be averaged* >]
:TEMPerature? { TC | RTD }[, <*probe type*>[,< *number of measurement to be averaged*]]]
:RJUNction? [{ SENSe | SOURce }]

SENSe

:FUNCtion {VOLTage | CURRent | RESistance | TCouple | RTD | FREQuency}
:FUNCtion?

:HOLD
:TRIG
:RUN

:VOLTage
:RANGe {100mV | 1V | 10V| 50V}
:RANGe?
:AUTO { ON | OFF }
: AUTO?

:CURRent
:RANGe { 0mA | 4mA | 50mA}
:RANGe?
:SUPPly { ON|OFF }
:SUPPly?
:SCALe { LINEAR | QUADRADIC }
:SCALe?
:HART { ON | OFF }
:HART?

:RESistance
:RANGe { 400 OHM | 4000 OHM }
:RANGe?
:AUTO { ON | OFF }
: AUTO?
:WIRes? // Response: 2,3 or 4

[:TEMPerature]
[:TRANsducer]
:TCouple
:TYPE { K | T | J | E | N | U | L | S | R | B | C | PL | MO}
:TYPE?
:RJUNction <Value>
:TYPE {INTernal | DISabled | FIXed}
:TYPE?
:RJUNction?
:RTD
:TYPE { PT50 | PT100 | PT200 | PT500 | PT1000
| PT100_3916 | PT100_3926
| NI100 | NI120 | NI1000
| CU10 | CU50 }
:TYPE?

SENSe

```
:FILTer { ON | OFF }
        :COUNT <number of measurement>
        :COUNT?                                  // Response: number of measures
:FILTer?                                         // Response: 0 or 1

:NULL { ON | OFF }
        :TARE                                    // display to  0
        :AMPLitude  { < value>}
        :AMPLitude?                              // Response: tare current value
:NULL?                                           // Response: 0 or 1

:SCALing { ON | OFF }
        :SIZE <number of points of the scaling: 2 to 10>
        :SIZE?
        :POINt <num(1 à 10)>, <Xvalue Nrf>, <Yvalue Nrf>
        :POINt? <num>                            // Response : Value of X, Value of Y
        :UNIT <chain of 1 of 4 characters between ñò>
        :UNIT?
        :ACCuracy <number of resolution>
        :ACC?

:SCALing?                                        // Response: 0 or 1

:STATistics
        :INITialize
        :MAXimum?
        :MINimum?
        :AVERage?
```

SOURce

:ADCDisplay {ON|OFF} // disp. DAC measurement in the emission window
:CONTrol {ON | OFF} // Switches OFF servosystem (VDC et IDC)
:CONTrol?

:FUNCtion {VOLT | CURRent | RESistance | TCouple | RTD | FREQuency}
:FUNCtion?

:VOLTage
[:LEVel]
[:IMMediate]
[:AMPLlitude] <value> Rq: 1-value in V, 2- current range
:RANGe { 100mV | 2V | 20V }
:RANGe?

:CURRent
[:LEVel]
[:IMMediate]
[:AMPLlitude] <value> Rq: Unit to be used if not specified ?
:RANGe { 0mA | 4mA | 24mA}
:RANGe?
:SUPPly { ON|OFF }
:SUPPly?
:SCALe { LINEAR | QUADRADIC }
:SCALe?

:RESistance
[:LEVel]
[:IMMediate]
[:AMPLlitude] <value> Rq: in Ohm by default
:RANGe { 400OHM | 4000OHM }, { 1mA | 10mA }
:RANGe?

[:TEMPerature]
[:TRANsducer]
:TCouple
[:LEVel]
[:IMMediate]
[:AMPLlitude] <value> Rq: in °C if not specified
:TYPE { K | T | J | E | N | U | L | S | R | B | C | PL | MO**| XA_K | XK_L | XK68** }
:TYPE?

:RJUNction <Value> Rq: in °C if not specified
:TYPE {INTernal | DISabled | FIXed}
:TYPE?
:RJUNction?
:RTD
[:LEVel]
[:IMMediate]
[:AMPLlitude] <value>
:TYPE { PT50 | PT100 | PT200 | PT500 | PT1000
| PT100_3916 | PT100_3926
| NI100 | NI120 | NI1000
| CU10 | CU50 },
| PTP46_1_3910 | PTP50_1_3911 | P_50P_1_3911
| PTP100_1_3910
| P_100P_1_3911 | PTP500_1_3910
| CU50_1_4260 | CUP50_1_4280 | P_50M_1_4280
| CU53_1_4260 | CU100_1_4260 | CUP100_1_4280 | P_100M_1_4280
} , [{ 1mA | 10mA}]

:TYPE?

:FREQuency <frequency value> // uses Current range
:RANGe { 1000Hz | 10KHZ }
:LEVel
[:IMMediate]
[:AMPLitude] <value> Rq: in V

SOURce

:NULL { ON | OFF }
:TARE // disp to 0
:AMPLitude { < value>}
:AMPLitude? // Response: Tare Current value
:NULL? // Response: 0 or 1

:SCALing { ON | OFF }
:SIZE < number of points of the scaling: 2 to 10>
:SIZE?
:POINt <num(1 à 10)>, <Xvalue Nrf>, <Yvalue Nrf>
:POINt? <num> // Response : Value of X, Value of Y
:UNIT < chain of 1 of 4 characters between ñ>
:UNIT?
:ACCuracy < chain of 1 of 4 characters between ñ>>
:ACC?

:SCALing? // Response: 0 or 1

:SETDacs? <value DACA>, <value DACB> // Response : measurement of voltage generated by DACs

STEPs

:LOW <low value>
:HIGH <high value>
:INCRement <step value>
:TIME <step time , in secondes>
:DELay <start delay after START>

:PLAY {UP | DOWN}
:HOLD
:NEXT
:PREVious
:CONTinue
:STOP

RAMP

:LOW <Low value>
:HIGH <High value>
:TIME <ramp total time, in seconds>
:DELay < start delay after START >

:PLAY {UP | DOWN}
:HOLD
:CONTinue
:STOP

CRAMP // Cyclical ramp

:LOW < low value >
:HIGH < high value >
:LTIMe < Low value step time , in seconds >
:RTIMe <rising time, in seconds>
:HTIMe <High value step time, in seconds>
:FTIMe <falling time, in seconds>
:REPeat <number of occurrence>
:DELay < start delay after START >

:PLAY {UP | DOWN} [,<number of occurrence >]
:HOLD
:CONTinue
:STOP

SYNThetizer

:POINt <num(1 to 100)>, <Value Nrf>
:TIME <step time, in seconds>
:REPeat <number of occurrence>
:DELay < start delay after START >

:PLAY [<1st point>[, <last point>[, <number of occurrence>]]]
:HOLD
:NEXT
:PREVious
:CONTinue
:STOP

CONFig

:SAVE <number between 1 and 9 > [,<optional name, 19 characters max.>]
:LOAD < number between 1 and 9 >

DISPlay

:CONTrast < number between 0 and 1 >
:LIGHting { ON | OFF }

[SYSTem]

:DATE <year>, <month>, <day>
:DATE?
:TIME <hour>, <minute>, <second>
:TIME?

:REMote // Switch instrument in remote mode
:LOCal // Switch instrument in local mode

:ERRor [:NEXT]? // Extract from FIFO the older error code

*CLS Clear Status ï erase from FIFO error codes
*IDN? Identify
*RST Initialise to a defined status all emission and measurement functions

2. GENERAL PROTOCOL

Transmission format : 8 bits, 1 stop, No parity, no flow control
Baudrate : 115200

Commands format : type IEEE 488-2, «SCPI-like»

- Last character command line is «\n» (decimal code10, hexadecimal 0x0A)
- It can have one or more commands separated by character «;»
- Each command has a header followed by one or more spaces and a 0 or several arguments, separated by character «,»
- Command header can be built by or more key words separated by character «:», succession of these key words are defining a tree diagram allowing to classify all the commands in subsets,é
- All command header asking for a response are finished by character «?» this character belongs to the header (No space required)
- when wrong command is sent, instrument does not reply.(for commands requiring a response).
- instrument can propose a table of the last 5 error codes, managed in the FIFO. Command « ERR? » allows to extract from the older error code. Command « *CLS » erase FIFO content.

Examples of command lines:

REM\n	switch the instrument as remote (keypads is locked)
SENS:VOLT:RANG 100mV\n	Define the voltage range
SENS:FUNC VOLT\n	select volt function
LOC\n	switch the instrument as LOCAL and an lock the keypad.
*IDN?\n	Instrument identification request. The command is finished by «?» so a response is needed.

MEAS:VOLT? 1V\n

Instrument responses are finished by «\n»

2.1 Communication general principles

- Switch the instrument in Remote sending « REM » before sending other commands.
- Erase FIFO of old error codes using command « *CLS »
- Send a command
- If command is not a request , verify it has been performed sending command ERR? And waiting for the response.

- Before closing the connexion , switch the instrument in Local mode (command «LOC»)

3. Measurement and emission setting

Commands for parameter asking are same as parameter setting with a ?

3.1 Measurement function selection

SENSe:FUNCtion { VOLTage | CURRent | RESistance | TCouple | RTD
| CONTinuity | FREQuency | COUNter | PRESssure }

3.2 Measurement function setting selection

SENSe :VOLTage **:RANGe {100mV | 1V | 10V| 50V}**
:AUTO { ON | OFF }

SENSe:CURRent **:RANGe { 0mA | 4mA | 50mA}**
:SUPPly { ON | OFF }
:SCALe { LINear | QUADradic }
:HART { ON | OFF }
0mA-> range 0-20mA; 4mA-> range 4-20mA

SENSe:RESistance **:RANGe { 400 OHM | 4000 OHM }**
:AUTO { ON | OFF }

SENSe:TCouple **:TYPE { K | T | J | E | N | U | L | S | R | B | C | PL | MO | XA_K | XK_L | XK68 }**
:RJUNction <Value>
:TYPE {INTernal | FIXed | NO}

SENSe:RTD **:TYPE { PT50 | PT100 | PT200 | PT500 | PT1000**
| PT100_3916 | PT100_3926
| NI100 | NI120 | NI1000
| CU10 | CU50
| PTP46_1_3910 | PTP50_1_3911 | P_50P_1_3911
| PTP100_1_3910
| P_100P_1_3911 | PTP500_1_3910
| CU50_1_4260 | CUP50_1_4280 | P_50M_1_4280
| CU53_1_4260 | CU100_1_4260 | CUP100_1_4280 | P_100M_1_4280
}

3.3 Emission function selection

SOURce:FUNCtion {VOLTage | CURRent | RESistance | TCouple | RTD | FREQuency | PRESsure}

3.4 Emission function settings selection

SOURce:VOLTage **:RANGe {100mV | 2V | 20V}**

SOURce:CURRent **:RANGe { 0mA | 4mA | 24mA}**
:SUPPly { ON | OFF }
:SCALe { LINear | QUADradic }
0mA-> range 0-20mA; 4mA-> range 4-20mA

SOURce:RESistance **:RANGe { 400 OHM | 4000 OHM }, {1mA | 10mA}**

SOURce:TCouple **:TYPE { K | T | J | E | N | U | L | S | R | B | C | PL | MO}**
:RJUNction <Value>
:TYPE {INTernal | FIXed | NO}

SOURce:RTD :TYPE { PT50 | PT100 | PT200 | PT500 | PT1000
| PT100_3916 | PT100_3926
| NI100 | NI120 | NI1000
| CU10 | CU50 } , {1mA|10mA}

4. Measurements commands

:VOLTage? [{100mV | 1V | 10V| 50V} [,<Number of measurement to be averaged]]
:CURRent? [<Number of measurement to be averaged]]
:RESistance? [{ 400 OHM | 4000 OHM }[,< Number of measurement to be averaged]]
:FREQuency? [<Number of measurement to be averaged]]
:TEMPerature? { TC | RTD }[, <*type de la sonde*>[,< Number of measurement to be averaged]]]
:PRESsure ? [<Number of measurement to be averaged]]

<*Sensor type*> = **{ K | T | J | E | N | U | L | S | R | B | C | PL | MO}** if TC measurement
<*sensor type*> = **{ PT50 | PT100 | PT200 | PT500 | PT1000**
| PT100_3916 | PT100_3926
| NI100 | NI120 | NI1000
| CU10 | CU50 } if RTD measurement

The Instrument

- switches in Hold
- Select measurement function
- If there is a range indicated , it selects it, otherwise it uses the current range for the indicated measurement function (except for Current measure: range 50mA)
- Performs indicated number of measures (or only one if nothing indicated)
- Send the average of measurements and the range unit

In case of RTD, the unit setting (degree or Ohm) is the one define in the instrument SETUP.
In case of TC, CJC setting is the current setting (INT, FIXED, NO), unit setting (degree or mV) is the one defined in the instrument SETUP.

Examples :

MEAS:VOLT? 100mV, 8	->	95.123, mV
MEAS:VOLT?	->	95.123, mV (ou 0.09512, V if range 1V)
MEAS:CURR?	->	20.123, mA
MEAS:RES?	->	300.123, Ohm
MEAS:FREQ?	->	1234.567, Hz
MEAS:TEMP? TC, K	->	100.25, CEL
MEAS:TEMP? TC	->	100.25, CEL (Couple type not indicated -> current couple)
MEAS:TEMP? RTD, PT100	->	100.25, CEL
MEAS:TEMP? RTD	->	100.25, CEL
MEAS:PRES?	->	30.123, BAR

Request of the CJC temperature (IN or OUT connector)

MEAS:RJUNction? [{ SENSe | SOURce }]

Instrument does not trigger measurement but reply sending the last measured value (not older than 5 seconds). Value is always in degree Celsius (whatever is the nit defined in the SETUP).
Argument SENSe or SOURce indicates the selected connector. If nothing is asked , the temperature displayed is the one of the connector SENSe.
Example :

MEAS:RJUN?	->	20.5, CEL
MEAS:RJUN? SENS	->	20.5, CEL
MEAS:RJUN? SOUR	->	20.7, CEL

5. Commande emission of a value

SOURce
:VOLTage <value>
:CURRent[<value>
:RESistance <value>

:FREQuency <value>
:TEMPerature { TC | RTD}, <value>

<value> must have a unit. Without this unit, default units are :
Volt, Ampere, Ohm and Hz. For temperature, default unit is the one defined in the instrument setup

Examples :
SOUR:VOLT 1.234 -> send 1.234 Volt (if range OK)
SOUR :VOLT 1.234 mV
SOUR :CURR 1.234 mA
SOUR :CURR 0.001234 -> send 1.234 mA

SOUR:RES 200.45 OHM
SOUR:RES 200.45
SOUR:RES 0.20045 KOHM -> send 200.45 Ohm

SOUR:FREQ 1000
SOUR:FREQ 1000 Hz
SOUR:FREQ 1 kHz -> send 1000 Hz

6. Measurement memory setting and readind.

6.1 setting

TRACe

:SIZE <number of measurements> memory size
:TIMer <period> in seconds
:TRIGger
:SOURce **{ IMMediate | MANual | INTernal}** Event triggering POST counting
:LEVel <level> (if SOURce = INTernal)
:SLOPe **{ POSitive | NEGative }**
:POST < number of measurements > Number of measurement counted after the Trig

TIMer period can only have the following values :
0.5s, 1s, 2s, 5s, 10s, 20s, 30s, 1mn, 2mn, 5mn, 10mn, 20mn, 30mn
If another value is indicated, it will be the lower valid value that will be used.
Example : TRACe :TIMer 3mn -> Used period will be 2 and not 3

LEVel and SLOPe are taken into account only if SOURce = INTernal.
LEVel the input level (in current unit) corresponding to the trigger.
If SLOPE = POSitive, trigger is performed when measurement is upper or = to the LEVel value
if SLOPE = NEGative, trigger is performed when measurement is lower or = to the LEVel value

POST is taking into account when SOURce = MANual or INTernal
Programmed value is the number of measurements to be stored after TRIG detection.

Example :
TRAC :SIZE 100 ; TIM 0.5s ; TRIG :SOUR INT; LEV 100.5; SLOP POS; POST 50
- Buffer size is 100 measurements
- A value is stored every 0.5 seconds
- Trigger is performed when measurement is upper than 100,5
- 50 measurement have to be stored before stopping the record.

6.2 Acquisition

INITiate

Erase trace and start recording ad trigger monitoring (if TRIG :SOUR = MAN or INT)
if TRIG :SOUR = IMM, POST value is not taken into account and trace records exactly before stop.
(equal keyboard command **Measures | Run**)

ABORt

Stops records in the trace buffer
(equal keyboard command **Measures | Stop**)

***TRG**

if TRIGger:SOURce = MANual, start counting POST records before stop.

6.3 Measurement reading

DATA :POINts? Give the number of measurements available in the trace buffer.

DATA :HEADer?

Give the trace header as binary block with a defined length.

Example :

#297\n	97 characters
W/O NAME\n	Burst name (15 characters max)
300 POINTS\n	Number of values
PROG\n	Recording type : PROG or FREE
10/05/2005 14 :40 :00\n	Date and time of 1st record
10/05/2005 14 :45 :00\n	Date and time of last record
TC K\n	Measurement Function and range
°C\n	Measurement Unit (to be confirmed line after line if Vdc Auto)
2\n	Number of resolution point (to be confirmed by the range if Vdc Auto)
SCALING OFF\n	Scaling : SCALING ON or SCALING OFF
TARE OFF\n	TARE ON or TARE OFF
\n	

DATA? [*<1st measurement index>*, [*<number of measurement>*]]

<1st measurement index > : between 1 and the number returned by DATA :POINts ? (1 if not specified)
< number of measurement > : number of measurement specified (1 if nothing specified)

Return timed and dated measurements as a binary block with a defined length.
Example :
#273\n
000000.0\t123.56789\tUNIT\n
000000.5\t123.56789UNIT\n
000001.0\t123.56789UNIT\n
\n

each measurement with time and date is 24 bytes :

- 8 bytes time in seconds : 0.1 to 999999.9s
- 1 tabulation
- 9 bytes for measurement value
- 1 tabulation
- 4 bytes for unit
- 1 end of line

1 character «end of line' is sent between size indicator #ndddd and first measurement result. This character is numbered in the indicated number ddd.

Character 'End of line' «eventually sent after the last measurement (in case instrument has nothing else to send) is not numbered in the number ddd.

6.4 Recording in eeprom and reading

MEMory :DATA:SAVE "<Name>"
Record Trace in EEProm as a specified name

MEMory :DATA :COUNt?
Return number of traces in memory

MEMory :DATA :HEADer? *<number>*
Return trace header position <number>
<number> is a recording rank number in the memory trace
Most recent trace is number 1.
The older trace has the number returned by command MEMory :DATA?

Trace header is transmitted as a binary block with a defined length with the same structure as the one returned in response to DATA :HEADer ?

MEMory :DATA:LOAD *<number>*
Load the trace with the number indicated in the trace buffer.
<number> is a recording rank number in the memory trace.
Most recent trace is number 1.
The older trace has the number returned by command MEMory :DATA?
Once loaded in the trace buffer, burst is read using command DATA as described in chapter 9.3

MEMory :FREE?
Return the number of free bytes in the memory and the number of occupied bytes.
<Free Bytes>,*<occupied bytes>*

MEMory :DATA :DELETE *<number>*
Delete from memory the trace with this number
(Rank number of all burst is then decremented)

MEMory :DATA :DELETE:ALL
Delete all records of the memory traces.

7. Setting and exploitation of calibration procedures

7.1 Setting

PROCedure
:NEW initialisation of the fields in default value? O calibration points

:INSTrument
:NAME "<chain with 1 to 15 characters >"
:MANUfacturer "< chain with 1 to 15 characters >"
:SENSe
:FUNCtion { VOLTage | CURRent | RESistance | TCouple | RTD | FREQuency | PRESsure }
:VOLTage
:RANGe {100mV | 2V | 20V}
:CURRent
:RANGe { 0mA | 4mA | 24mA} 0mA-> range 0-20mA; 4mA-> range 4-20mA
:SUPPly { ON | OFF }
:SCALe { LINear | QUADradic }
:RESistance
:RANGe { 400 OHM | 4000 OHM }, {1mA | 10mA}
:TCouple
:TYPE { K | T | J | E | N | U | L | S | R | B | C | PL | MO}
:RJUNction <Value>
:TYPE { INTernal | FIXed | NO }
:RTD
:TYPE { PT50 | PT100 | PT200 | PT500 | PT1000
| PT100_3916 | PT100_3926
| NI100 | NI120 | NI1000
| CU10 | CU50 } , {1mA|10mA}
:FREQuency
:RANGe { 1000HZ | 10KHZ }
:UNIT { HZ | CPM }
:OUTPut { VOLTage | CONTact }
:LEVel <value> Rq: En V
:SOURce
:FUNCtion { VOLTage | CURRent | NONE }
:CURRent
:RANGe { 0mA | 4mA } 0mA-> range 0-20mA; 4mA-> range 4-20mA
:SUPPly { ON | OFF }
:SCALe { LINear | QUADradic }
:HART { ON | OFF }
:SCALing { ON | OFF }
:POINt <num(1 à 10)>, <Xvalue Nrf>, <Yvalue Nrf>
:TABLe
:SIZE <Number of points>
:POINt <Number of the point: 1 to SIZE>, <value>
:EXECution { UP | UPDown | MANual }
:STIMe <stabilisation time in seconds>
:RLIMit <Relative error in %>
:ALIMit <Absolute maximum error>

7.2 Recording in eeprom and reading

MEMory:PROCedure:SAVE save current procedure in memory

MEMory:PROCedure:COUNt? Return number of procedure saved in memory

MEMory:PROCedure:SUMMary?

Return a summary of saved procedures, as a block with a defined length, with for each, manufacturer name and number of associated reports.

#XYYY\n
<Order number>\t<instrument name >\t<manufacturer name >\t <report numbers>\n
é .
< Order number >\t < instrument name >\t< manufacturer name >\t < report numbers >\n

Each line has exactly 39 bytes + \n = 40
(3+1+15+1+15+1+3+1 = 40)
Example :
#3161\n
001\tNOM_INSTRUMENT1\tNOM_FABRICANT01\t000\n
002\tNOM_INSTRUMENT2\tNOM_FABRICANT02\t005\n
003\tNOM_INSTRUMENT3\tNOM_FABRICANT03\t010\n
004\tNOM_INSTRUMENT4\tNOM_FABRICANT04\t002\n

MEMory:PROCedure? <number>

Return procedure configuration as a block of bytes with a defined length

Example :	
#3146\n	146 c
NOM_INSTRUMENT1\n	instrument name (15 characters max)
NOM_FABRICANT01\n	Manufacturer name (15 characters max)
TC,K,RJ FIXed,18.6\n	Instrument input function and range
CURR,0.004,SUPP OFF,SCAL LINear\n	instrument output function and range
4.0\t-100.0\n	low point of scale, input and output
20.0\t500.0\n	High point of scale input and output
3\n	Number of setpoints
10.0\n	Point n°1
50.0\n	Point n°2
80.0\n	Point n°3
UPDown\n	Emission mode
60\n	Stabilisation time (in s)
1.5\t2.2\n	Allowable relative deviation (%), absolute
10\n	Nbr of associated reports

MEMory:PROCedure :DELete <number>

Delete the mentioned procedure

MEMory:PROCedure:DELete:ALL

7.3 procedure associated to a report reading

MEMory:PROCedure:PV? <procedure number>, <report number>
Return report as a bytes block with a defined legth

Exemple :

#3212\n	212 characters
NOM_INSTRUMENT1\n	instrument name (15 characters max)
NOM_FABRICANT01\n	Manufacturer name (15 characters max)
1458045\n	Serial number (15 characters max)
Calys75\n	Calys model number (15 characters max)
1001\n	calys serial number (15 characters max)
10/01/2005 10:35:00\n	calys adjustment date
10/05/2005 14:40:00\n	calys calibration date
AHZ45012\n	Calys Calibration certificate number (50 characters max)
581475\n	Keller sensor serial number (15 characters max)
DUPONT\n	User name (15 characters max)
Instabilité\n	Comment (15 characters max)
AS_FOUND\n	Step before/after adjustment
10/06/2005 16:25:00\n	Performing date
OK\n	Result
5\n	Number of calibrated points
10.0\t10.2\n	Point n°1 (true value / read value)
50.0\t48.8\n	Point n°2
80.0\t80.5\n	Point n°3
50.0\t50.1\n	Point n°4
10.0\t9.9\n	Point n°5

MEMory:PROCedure:PV;DELete <procedure number>, <report number>
MEMory:PROCedure:PV;DELete:ALL