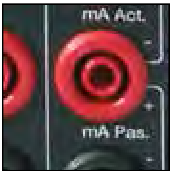
AMETEK®
SENSORS, TEST & CALIBRATION

Protocol Communication manual
Jofra RTC-PTC ASCII



AMETEK Denmark A/S

ASCII protocol specification

JOFRA RTC/PTC calibrators

Indholdsfortegnelse

1.0 Introduction

This document describes the ASCII based protocol used for communication between a PC and a JOFRA RTC/PTC calibrator. The ASCII based protocol is an alternative to the XML protocol i.e. the JOFRA RTC/PTC supports both protocols, but they can't be used at the same time.

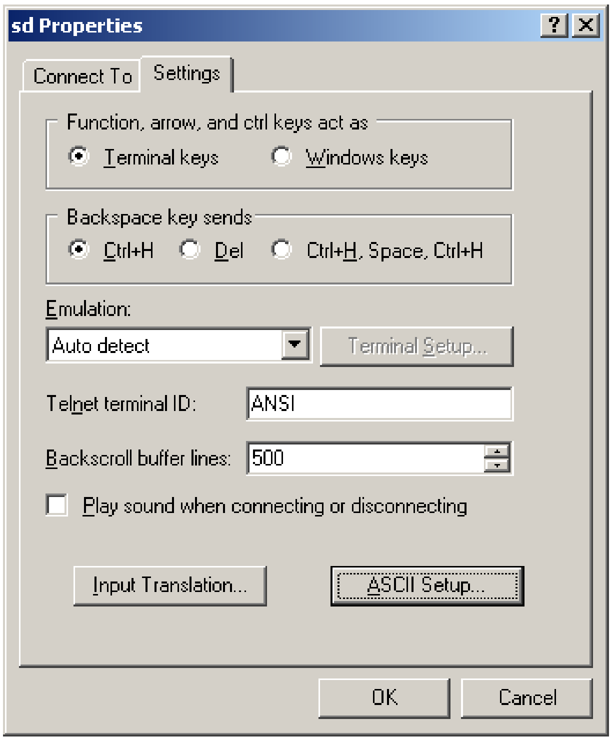
2.0 USB Communication parameters

The communication parameters are specified as follows:

- 115200 baud
- 8 data bits
- No parity
- 1 stop bit

No hardware handshake is used.

If used in a terminal, make sure that local type characters are displayed and that carriage return (CR) is appended with line feed (LF). Setting for Hyper Terminal:



sd Properties

Connect To | Settings

Function, arrow, and ctrl keys act as
● Terminal keys ○ Windows keys

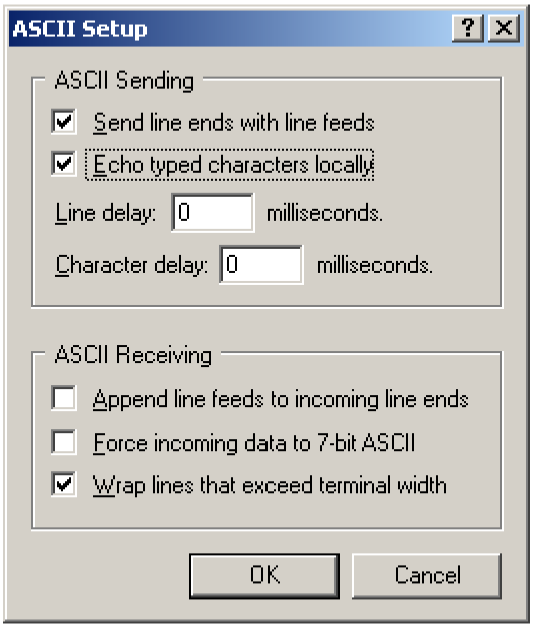
Backspace key sends
● Ctrl+H ○ Del ○ Ctrl+H, Space, Ctrl+H

Emulation: Auto detect

Telnet terminal ID: ANSI

Backscroll buffer lines: 500

□ Play sound when connecting or disconnecting



ASCII Setup

ASCII Sending
☑ Send line ends with line feeds
☑ Echo typed characters locally
Line delay: 0 milliseconds.
Character delay: 0 milliseconds.

ASCII Receiving
□ Append line feeds to incoming line ends
□ Force incoming data to 7-bit ASCII
☑ Wrap lines that exceed terminal width

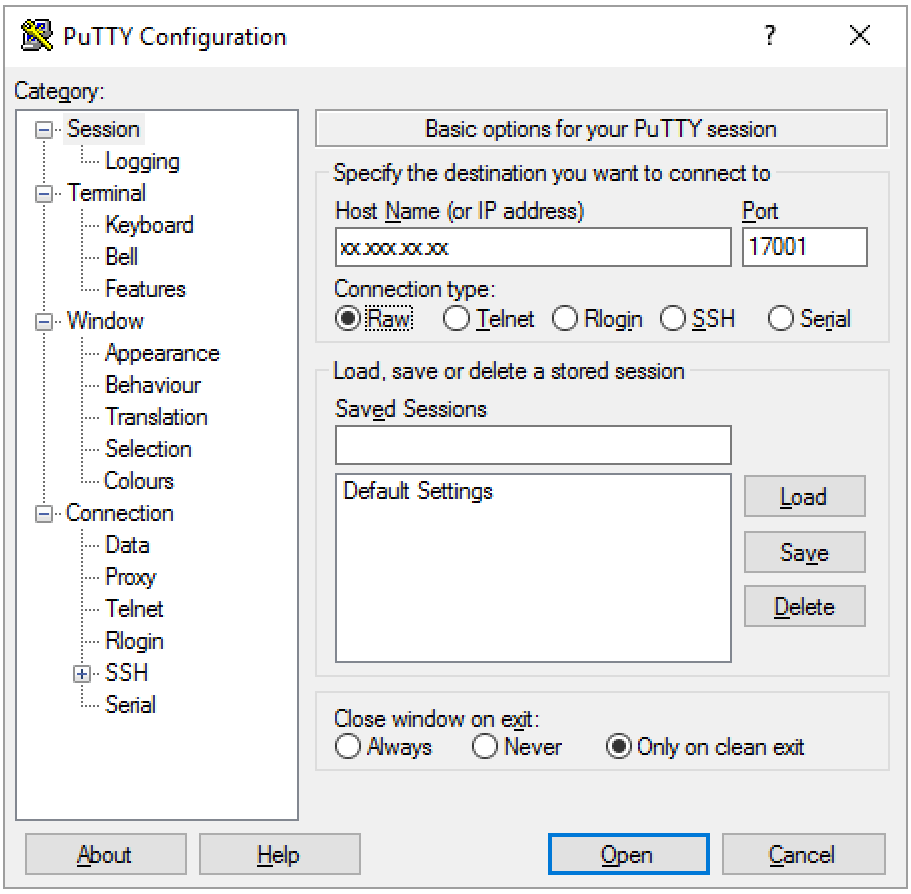
3.0 Communication RTC/PTC using TCP/IP

Communicating with RTC/PTC over ethernet using terminal program ”putty”.

Putty can be downloaded from http://www.putty.org/.

Enter the IP address of the RTC/PTC in the field “Host Name (or IP address)”.

Set “Port” to 17001 and select “Connection type: Raw”.



PuTTY Configuration

Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address): xx.xxx.xx.xx

Port: 17001

Connection type: ● Raw ○ Telnet ○ Rlogin ○ SSH ○ Serial

Load, save or delete a stored session

Saved Sessions

Default Settings

Close window on exit: ○ Always ○ Never ● Only on clean exit

Select “Open”

Enter “ASCII+” and press “ENTER”. The calibrator should respond “<ASCII protocol activated>”.

4.0 Protocol

4.1 Master/slave

The ADK-protocol is based on the Master/Slave principle. The PC is always to be regarded as the master and the calibrator as the slave. This means that the calibrator never starts the communication by itself, but solely replies to requests from the PC.

4.2 Read/write

When talking about writing and reading, the communication is seen from the PC’s point of view, which means that writing is information from the PC to the calibrator, whereas reading is information from the calibrator to the PC.

4.3 Format

The communication is based on line based string telegrams as specified in this document. The format of a telegram is dependent of it being a request (from the PC) or a reply (from the calibrator).

The requests contain the telegram name and parameters, while the reply has a start delimiter followed by the response type: GetResponse, SetResponse or Error, then telegram name and parameters (if any) and finally the end delimiter.

Eksample: To get the LogOn status the master issues:

IsLoggedOn?

The reply is:

<GetResponse IsLoggedOn True>

The protocol is case insensitive so IsLoggedOn? Is the same as isloggeDoN?

Each line must end with a CR + LF.

Example of error responses:

<Error Invalid command or argument(s)>
<Error Telegram not allowed>
<Error Temperature out of range>

4.4 Protocol switch

The calibrator is using the XML based protocol by default. To switch between the protocols use the command “ascii” followed by a ‘+’ or a ’-‘.

4.4.1 Enable

Command	ascii+
Response	<ASCII protocol activated>

4.4.2 Disable

Command	ascii-
Response	-

5.0 Telegram parameters

The following designations are used in this specification:

Integer	Signed integer, value–32,768 – 32,767.
Double	Floating point number.
Boolean	False and True.
String	String of ASCII characters, a string must not contain space as this character is used as token delimiter.

Temperature values are in Kelvin.

6.0 Telegrams

6.1 Logon

In order to write parameters to the calibrator, the master must issue the logon command. It is possible to read parameters with being logged on. When logged on the user interface of the calibrator is disabled.

If a write parameter command is issued without being logged on, a <Error Telegram not allowed> response is received.

Command	LogOn
Response	<CallResponse TelegramValue`1>

6.2 Logoff

The log off command disables the possibility to write parameters, at the same time the user interface of the calibrator is enabled.

Command	LogOff
Response	<CallResponse LogOff>

6.3 Isloggedon

This command returns the status of the logon.

Read	Isloggedon?
Response	<GetResponse Isloggedon[loggedon]>
Write	-
[loggedon]	Boolean: True|False

6.4 LiveSensors

This command returns the status and readings of the calibrator. All temperature values are in Kelvin.

Read	LiveSensors?
Response	<GetResponse LiveSensors [READConvertToTemperature] [READInputType] [READInputValue] [READInputTemperatureValue] [READStabilityTolerance] [READStabilityRequiredSeconds] [READStabilitySeconds] [READNumberOfDecimals] [READSetFollows] [TRUEName] [TRUEConvertToTemperature] [TRUEInputType] [TRUEInputValue] [TRUEInputTemperatureValue] [TRUEStabilityTolerance] [TRUEStabilityRequiredSeconds] [TRUEStabilitySeconds] [TRUENumberOfDecimals] [TRUESetFollows] [SENSORConvertToTemperature] [SENSORInputType] [SENSORInputValue] [SENSORInputTemperatureValue] [SENSORStabilityTolerance] [SENSORStabilityRequiredSeconds] [SENSORStabilitySeconds] [SENSORNumberOfDecimals] [SENSORSetFollows]

	[XDIFFName] [XDIFFConvertToTemperature] [XDIFFInputType] [XDIFFInputValue] [XDIFFInputTemperatureValue] [XDIFFStabilityTolerance] [XDIFFStabilityRequiredSeconds] [XDIFFStabilitySeconds] [XDIFFNumberOfDecimals] [XDIFFSetFollows] [SwitchIsClosed] [NumberOfSetDecimals] [TemperatureUnit]>
Write	-
[READConvertToTemperature]	Convert value to temperature Boolean: True|False
[READInputType]	Input type String: INT_RTD
[READInputValue]	The value read Double: NAN
[READInputTemperatureValue]	Read temperature Double:
[READStabilityTolerance]	Stability tolerance Double:
[READStabilityRequiredSeconds]	Required stability time in seconds Double:
[READStabilitySeconds]	Time to stabile or time it has been stabile. If negative, it is not yet stabile Double:
[READNumberOfDecimals]	Number of decimals used by sensor Integer:
[READSetFollows]	Set follows this sensor Boolean: True|False
[TRUEName]	Name of the sensor String:
[TRUEConvertToTemperature]	Convert value to temperature Boolean: True|False
[TRUEInputType]	Input type String: REF_RTD|INT_RTD
[TRUEInputValue]	The value in ohm for the external reference. Value is NAN when TRUEInputType = INT_RTD. Double:
[TRUEInputTemperatureValue]	True temperature. Double:
[TRUEStabilityTolerance]	Stability tolerance Double:
[TRUEStabilityRequiredSeconds]	Required stability time in seconds Double:
[TRUEStabilitySeconds]	Time to stabile or time it has been stabile. If negative, it is not yet stabile Double:
[TRUENumberOfDecimals]	Number of decimals used by sensor Integer:
[TRUESetFollows]	Set follows this sensor Boolean: True|False
[SENSORConvertToTemperature]	Convert value to temperature Boolean: True|False
[SENSORInputType]	Input type String: DUMMY|DUT_RTD_400|DUT_RTD_4000| DUT_TC|DUT_U|DUT_I
[SENSORInputValue]	The sensor under test value in ohm/mV/mA/V Double:

[SENSORInputTemperatureValue]	The sensor under test value in Kelvin Double:
[SENSORStabilityTolerance]	Stability tolerance Double:
[SENSORStabilityRequiredSeconds]	Required stability time in seconds Double:
[SENSORStabilitySeconds]	Time to stabile or time it has been stabile. If negative, it is not yet stabile Double:
[SENSORNumberOfDecimals]	Number of decimals used by sensor Integer:
[SENSORSetFollows]	Set follows this sensor Boolean: True|False
[XDIFFName]	Name of the sensor String:
[XDIFFConvertToTemperature]	Convert value to temperature Boolean: True|False
[XDIFFInputType]	Input type String: REF_TC
[XDIFFInputValue]	DLC reading in mV Double:
[XDIFFInputTemperatureValue]	DLC temperature Double:
[XDIFFStabilityTolerance]	Stability tolerance Double:
[XDIFFStabilityRequiredSeconds]	Required stability time in seconds Double:
[XDIFFStabilitySeconds]	Time to stabile or time it has been stabile. If negative, it is not yet stabile Double:
[XDIFFNumberOfDecimals]	Number of decimals used by sensor Integer:
[XDIFFSetFollows]	DLC activated Boolean: True|False
[SwitchIsClosed]	Status of switch, is switch closed Boolean: True|False
[NumberOfSetDecimals]	Number of decimals used in SET temperature Integer:
[TemperatureUnit]	Current temperature unit String: Kelvin|Celsius|Fahrenheit
Example	<GetResponse LiveSensors True INT_RTD NaN 296.315687561035 NaN 300 -180.914 2 False False REF_RTD NaN NaN 0.05 600 NaN 2 True True DUT_TC NaN NaN NaN 0 NaN 2 False null False REF_TC NaN NaN NaN 0 493.959 2 False False 2 Celsius>

6.5 SetTemperature

Sets/reads the SET-temperature in Kelvin.

Read	Settemperature?
Response	<GetResponse Settemperature[temperature]>
Write	Settemperature[temperature]
Response	<SetResponse SETTemperature>
[temperature]	Double: temperature in Kelvin
Example	SetTemperature 300 <SetResponse SETTemperature>

6.6 CalibratorDevice

The returned data identify the instrument type, the protocol consulted and the version number of the software of the calibrator

Read	CalibratorDevice?
Response	<GetResponse CalibratorDevice [serial] [protocolVersion] [modelId] [softwareVersion] [hardwareVersion] [model] [modelVariant] [hasSilentMode] [hasFpsc] [hasStirrer] [factoryMaxTemperature] [factoryMinTemperature] [maxSetTemperature] [minSetTemperature] [mainFrequency] [mainFrequencyAccepted] [refInputFailed] [sensorInputFailed] [isRefCalibrated] [isSensorCalibrated]>
Write	-
[serial]	Serial number of the instrument String:
[protocolVersion]	Version of the control board protocol Integer:
[modelId]	Model id of the instrument Integer:
[softwareVersion]	SW version of the control board Integer:
[hardwareVersion]	HW version of the control board Integer:
[model]	Model type possibilities String: RTC_700|RTC_600|RTC_250|RTC_159|RTC_158|RTC_157|RTC_156| RTC_250| PTC_660|PTC_350|PTC_155|PTC_125
[modelVariant]	Model variant possibilities: String: A|B|C
[hasSilentMode]	Has the instrument got silent mode Boolean: True|False
[hasFpsc]	Has the instrument got FPSC Boolean: True|False
[hasStirrer]	Has the instrument got a stirrer Boolean: True|False
[factoryMaxTemperature]	Factory max temperature Double:
[factoryMinTemperature]	Factory min temperature Double:
[maxSetTemperature]	User defined max temperature Double:
[minSetTemperature]	User defined min temperature Double:
[mainFrequency]	The mains frequency type of the instrument String: Any|Only50Hz|Only60Hz

[mainFrequencyAccepted]	Mains frequency accepted Boolean: True|False
[refInputFailed]	Failed to initialize the reference input board Boolean: True|False
[sensorInputFailed]	Failed to initialize the sensor input board Boolean: True|False
[isRefCalibrated]	Is reference calibrated Boolean: True|False
[isSensorCalibrated]	Is sensor calibrated Boolean: True|False
Example	CalibratorDevice? <GetResponse CalibratorDevice 350158-00001 208 4122 233 3 RTC_158 B True False True 428.15 233.15 428.15 233.15 Only50Hz True False False True True>

6.7 TemperatureUnit

This command sets/reads the temperature unit.

Read	TemperatureUnit?
Response	<GetResponse TemperatureUnit [unit]>
Write	TemperatureUnit [unit]
Response	<SetResponse TemperatureUnit>
[unit]	Temperature unit to use String: Kelvin|Celsius|Fahrenheit
Example	TemperatureUnit? <GetResponse TemperatureUnit Celsius>

6.8 NumberOfSetDecimals

This command sets/reads the number of decimals for the SET-temperature.

Read	NumberOfSetDecimals?
Response	<GetResponse NumberOfSetDecimals [number]>
Write	NumberOfSetDecimals [number]
Response	<SetResponse NumberOfSetDecimals>
[number]	Number of decimals used Integer: 0..3(RTC) / 0..2(PTC)

6.9 NumberOfReadDecimals

This command sets/reads the number of decimals for the Read-temperature.

Read	NumberOfReadDecimals?
Response	<GetResponse NumberOfReadDecimals [number]>
Write	NumberOfReadDecimals [number]
Response	<SetResponse NumberOfReadDecimals>
[number]	Number of decimals used Integer: 0..3(RTC) / 0..2(PTC)

6.10 NumberOfTrueDecimals

This command sets/reads the number of decimals for the True-temperature (internal or external reference).

Read	NumberOfTrueDecimals?
Response	<GetResponse NumberOfTrueDecimals [number]>
Write	NumberOfTrueDecimals [number]
Response	<SetResponse NumberOfTrueDecimals>
[number]	Number of decimals used Integer: 0..3(RTC) / 0..2(PTC)

6.11 NumberOfSensorDecimals

This command sets/reads the number of decimals for the Sensor-temperature.

Read	NumberOfSensorDecimals?
Response	<GetResponse NumberOfSensorDecimals [number]>
Write	NumberOfSensorDecimals [number]
Response	<SetResponse NumberOfSensorDecimals>
[number]	Number of decimals used Integer: 0..3(RTC) / 0..2(PTC)
Example	NumberOfSensorDecimals? <GetResponse NumberOfSensorDecimals 2>

6.12 UserMinMaxSetTemperature

This command sets/reads the user defined temperature limits of the calibrator in Kelvin.

Read	UserMinMaxSetTemperature?
Response	<GetResponse UserMinMaxSetTemperature [maxTemperature] [minTemperature]>
Write	UserMinMaxSetTemperature [maxTemperature] [minTemperature]
Response	<SetResponse UserMinMaxSetTemperature>
[maxTemperature]	Double:
[minTemperature]	Double:
Example	UserMinMaxSetTemperature? <GetResponse UserMinMaxSetTemperature 428.15 233.15>

6.13 SlopeRate

This command sets/reads the slope rate.

Read	SlopeRate?
Response	<GetResponse SlopeRate [rate]>
Write	SlopeRate [rate]
Response	<SetResponse SlopeRate>
[rate]	Double:
Example	SlopeRate? <GetResponse SlopeRate 0>
Example	SlopeRate 0.02 <SetResponse SlopeRate>

6.14 StabilitySetup

This command sets/reads the stability criteria settings.

Read	StabilitySetup?
Response	<GetResponse StabilitySetup [irefTime] [irefTolerance] [irefExtTime] [xrefTime] [xrefTolerance] [sensorTime] [sensorTolerance] [sensorEnabled]>
Write	StabilitySetup [irefTime] [irefTolerance] [irefExtTime] [xrefTime] [xrefTolerance] [sensorTime] [sensorTolerance] [sensorEnabled]
Response	<SetResponse StabilitySetup>
[irefTime]	IREF stability time in seconds Double:
[irefTolerance]	IREF stability tolerance Double:
[irefExtTime]	Extended stability time for IREF. Used to extend default IREF stability Double:
[xrefTime]	XREF stability time in seconds Double:
[xrefTolerance]	XREF stability tolerance Double:
[sensorTime]	SENSOR stability time in seconds Double:
[sensorTolerance]	Sensor stability tolerance Double:
[sensorEnabled]	Enable sensor stability time Boolean:
Example	StabilitySetup? <GetResponse StabilitySetup 300 0.0199999995529652 0 600 0.05 600 0.1 False>

6.15 FactoryMinMaxSetTemperature

This command reads the calibrator default temperature limits in Kelvin.

Read	FactoryMinMaxSetTemperature?
Response	<GetResponse FactoryMinMaxSetTemperature [maxTemperature] [minTemperature]>
Write	-
[maxTemperature]	Double:
[minTemperature]	Double:
Example	FactoryMinMaxSetTemperature? <GetResponse FactoryMinMaxSetTemperature 428.15 233.15>

6.16 SibPort

This command sets/reads the input used for SUT – sensor under test.

Read	SibPort?
Response	<GetResponse SibPort [sensorType]>
Write	SibPort [sensorType]
Response	<SetResponse SibPort>
[sensorType]	String: None|Voltage|Ampere|RTD|TC
Example	SibPort? <GetResponse SibPort Ampere>

6.17 SibTCPort

This command sets/reads the SUT – sensor under test thermocouple (mV) type.

Read	SibTCPort?
Response	<GetResponse SibTCPort [compensationMode] [manualTemperature] [sensorType]>
Write	SibTCPort [compensationMode] [manualTemperature] [sensorType]
Response	<SetResponse SibTCPort>
[compensationMode]	The compensation mode to use String: Automatic|Manual
[manualTemperature]	The temperature to use for the manual temperature compensation Double:
[sensorType]	The thermocouple sensor type String: B|E|J|K|L|N|R|S|T|U
Example	SibTCPort? <GetResponse SibTCPort Automatic 296.15 E>
Example	SibTCPort Manual 296.14 B <SetResponse SibTCPort>

6.18 SibRTDPort

This command sets/reads the setting for the SUT – sensor under test resistance temperature detector (ohm) type.

Read	SibRTDPort?
Response	<GetResponse SibRTDPort [wires] [sensorType]>
Write	SibRTDPort [wires] [sensorType]
Response	<SetResponse SibRTDPort>
[wires]	Number of wires in RTD Integer: 2..4
[sensorType]	The RTD sensor type String: P10(90)385
Example	SibRTDPort? <GetResponse SibRTDPort 4 P10(90)385>

6.19 SibVoltagePort

This command sets/reads the setting for the SUT – sensor under test voltage (V) type.

Read	SibVoltagePort?
Response	<GetResponse SibVoltagePort [lowTemperature] [highTemperature] [lowInput] [highInput]>
Write	SibVoltagePort [lowTemperature] [highTemperature] [lowInput] [highInput]
Response	<SetResponse SibVoltagePort>
[lowTemperature]	Low temperature Double:
[highTemperature]	High temperature Double:
[lowInput]	Value corresponding to the low temperature Double:
[highInput]	Value corresponding to the high temperature Double:
Example	SibVoltagePort? <GetResponse SibVoltagePort 273.15 373.15 0 10>

6.20 SibCurrentPort

This command sets/reads the setting for the SUT – sensor under test current (mA) type.

Read	SibCurrentPort?
Response	<GetResponse SibCurrentPort [lowTemperature] [highTemperature] [lowInput] [highInput]>
Write	SibCurrentPort [lowTemperature] [highTemperature] [lowInput] [highInput]
Response	<SetResponse SibCurrentPort>
[lowTemperature]	Low temperature Double:
[highTemperature]	High temperature Double:
[lowInput]	Value corresponding to the low temperature Double:
[highInput]	Value corresponding to the high temperature Double:
Example	SibCurrentPort? <GetResponse SibCurrentPort 274.15 373.16 0.004 0.02>

6.21 UseExternalReferenceSensor

This command sets/reads the setting for the True temperature. If “False”, the internal reference (Read) temperature is displayed as “True”.

Read	UseExternalReferenceSensor?
Response	<GetResponse UseExternalReferenceSensor [use]>
Write	UseExternalReferenceSensor [use]
Response	<SetResponse UseExternalReferenceSensor>
[use]	Boolean: True|False
Example	UseExternalReferenceSensor? <GetResponse UseExternalReferenceSensor True>

6.22 EnableSetFollowsXREF

This command sets/reads the “SET-follows-true“ feature for the external reference sensor.

Read	EnableSetFollowsXREF?
Response	<GetResponse EnableSetFollowsXREF [enable]>
Write	EnableSetFollowsXREF [enable]
Response	<SetResponse EnableSetFollowsXREF>
[enable]	Boolean: True|False
Example	EnableSetFollowsXREF? <GetResponse EnableSetFollowsXREF True>

6.23 EnableSetFollowsXDIFF

This command sets/reads the “DLC“ feature.

Read	EnableSetFollowsXDIFF?
Response	<GetResponse EnableSetFollowsXDIFF [enable]>
Write	EnableSetFollowsXDIFF [enable]

Response	<SetResponse EnableSetFollowsXDIFF>
[enable]	Boolean: True|False
Example	EnableSetFollowsXDIFF? <GetResponse EnableSetFollowsXDIFF False>

6.24 SensorConvertToTemperature

This command sets/reads the “Convert to temperature “ feature for the sensor under test.

Read	SensorConvertToTemperature?
Response	<GetResponse SensorConvertToTemperature [convert]>
Write	SensorConvertToTemperature [convert]
Response	<SetResponse SensorConvertToTemperature>
[convert]	Boolean: True|False
Example	SensorConvertToTemperature? <GetResponse SensorConvertToTemperature True>

6.25 XRefConvertToTemperature

This command sets/reads the “Convert to temperature “ feature for the external reference sensor.

Read	XRefConvertToTemperature?
Response	<GetResponse XRefConvertToTemperature [convert]>
Write	XRefConvertToTemperature [convert]
Response	<SetResponse XRefConvertToTemperature>
[convert]	Boolean: True|False
Example	XRefConvertToTemperature? <GetResponse XRefConvertToTemperature True>

6.26 StirrerSpeed

This command sets/reads the stirrer speed (RTC-158/RTC-250).

Read	StirrerSpeed?
Response	<GetResponse StirrerSpeed [speed]>
Write	StirrerSpeed [speed]
Response	<SetResponse StirrerSpeed>
[speed]	Integer:
Example	StirrerSpeed? <GetResponse StirrerSpeed 0>
Example	StirrerSpeed 100 <SetResponse StirrerSpeed>

6.27 CalibrationDate

This command reads the calibration date of the temperature source.

Read	CalibrationDate?
Response	<GetResponse CalibrationDate [year month date]>
Write	-
[year]	Year of calibration Integer:
[month]	Month of calibration String: January|February|March|April|June|July|August|September|October|November|December
[date]	Date (in month) of calibration Integer: 1..31
Example	CalibrationDate? <GetResponse CalibrationDate 2010 November 29>

6.28 RibCalibrationDate

This command reads the calibration date of the reference input module.

Read	RibCalibrationDate?
Response	<GetResponse RibCalibrationDate [year month date]>
Write	-
[year]	Year of calibration Integer:
[month]	Month of calibration String: January|February|March|April|June|July|August|September|October|November|December
[date]	Date (in month) of calibration Integer: 1..31
Example	RibCalibrationDate? <GetResponse RibCalibrationDate 2010 November 29>

6.29 SibCalibrationDate

This command reads the calibration date of the sensor under test input module.

Read	SibCalibrationDate?
Response	<GetResponse SibCalibrationDate [year month date]>
Write	-
[year]	Year of calibration Integer:
[month]	Month of calibration String: January|February|March|April|June|July|August|September|October|November|December
[date]	Date (in month) of calibration Integer: 1..31
Example	SibCalibrationDate? <GetResponse SibCalibrationDate 2010 November 29>

6.30 RibXREFPort

Read	RibXREFPort?
Response	<GetResponse RibXREFPort [serialNumberExtented maxPhysVal minPhysVal calYear calMonth calDate calInitials calPeriod coefType RTPW ALR BLR CLR C2LR C3LR C4LR C5LR AHR BHR CHR DHR W]> **OR** <GetResponse RibXREFPort [serialNumberExtented maxPhysVal minPhysVal calYear calMonth calDate calInitials calPeriod coefType R0 A B C]>
Write	-
[serialNumberExtented]	Extented serial number String:
[maxPhysVal]	Double:
[minPhysVal]	Double:
[calYear]	Year of calibration Integer:
[calMonth]	Month of calibration String: January|February|March|April|June|July|August|September|October|November|December
[calDate]	Date (in month) of calibration Integer: 1..31
[calInitials]	Initials of calibrator String:
[calPeriod]	Calibration period in days Integer:
[coefType]	String: RtdCoefIts90 | RtdCoefCvD
[RTPW]	Float:
[ALR]	Float:
[BLR]	Float:
[CLR]	Float:
[C2LR]	Float:
[C3LR]	Float:
[C4LR]	Float:
[C5LR]	Float:
[AHR]	Float:
[BHR]	Float:
[CHR]	Float:
[DHR]	Float:
[W]	Float:
[R0]	Float:
[A]	Float:
[B]	Float:
[C]	Float:
Example	RibXREFPort? <GetResponse RibXREFPort 547383-01 400 -50 2008 February 01 X 365 RtdCoefIts90 9 9.92758 -0.0174969 0.00280391 0 0 0 0 0 -0.0182356 -0.0001042 0 0 0>

6.31 RibXDIFFPort

Read	RibXDIFFPort?
Response	<GetResponse RibXDIFFPort [serialNumberExtented maxPhysVal minPhysVal calYear calMonth calDate calPeriod coefType coefA coefB coefC]>
Write	-
[serialNumberExtented]	Extented serial number String:
[maxPhysVal]	Double:
[minPhysVal]	Double:
[calYear]	Year of calibration Integer:
[calMonth]	Month of calibration String: January|February|March|April|June|July|August|September|October|November|December
[calDate]	Date (in month) of calibration Integer: 1..31
[calPeriod]	Calibration period in days Integer:
[coefType]	String:
[coefA]	Float:
[coefB]	Float:
[coefC]	Float:
Example	RibXDIFFPort? <GetResponse RibXDIFFPort 599353-02 155 -100 2012 June 06 365 T -1.357654E-07 1. 000047 -0.003140619>

AMETEK®

SENSORS, TEST & CALIBRATION

www.ametekcalibration.com

UK
Tel +44 (0)1243 833 302
stc.uk@ametek.com

France
Tel +33 (0)1 30 68 89 40
general.lloyd-instruments@ametek.fr

Germany
Tel +49 (0)2159 9136 510
info.mct-de@ametek.de

Denmark
Tel +45 4816 8000
jofra@ametek.com

USA - California
Tel +1 (800) 444 1850
crystal@ametek.com

USA - Florida
Tel +1 (800) 527 9999
cal.info@ametek.com

India
Tel +91 22 6196 8200
jofra@ametek.com

Singapore
Tel +65 6484 2388
jofra@ametek.com

China - Beijing
Tel +86 10 8526 2111 ext. 19
stc.china@ametek.com