PRESYS®

Advanced Calibrators
Web API

Technical Manual

EM0348-00

Table of Contents

1 - Introduction

The Advanced Calibrators Family consists of devices developed using advanced technology and features that facilitate the calibration environment. In addition to having an easy-to-use and intuitive interface, they allow network use through various forms of communication described in this manual.

The Advanced Calibrators Family are:

- Universal Process Calibrator MCS-XV;
- Advanced Pressure Calibrator PCA-570;
- Advanced Temperature Calibrator TCA-520;
- TA Dry blocks Calibrators;
- Pressure Controller PCON-Y17;
- HART® Configurator FCY-15;
- Calibration Cell Station Modules.

They have features such as automated tasks and data loggers, which generate files that can be stored in the instrument's internal memory or can be exported to a network or other type of storage such as a computer or a pen drive.

Automated tasks can be created in the calibrator itself using the task creation option - through the Isoplan® Software or the user can create the task .xml file and integrate with their existing systems.

2 - Network connection and configuration

In order to have access to Web API, the calibrator must be connected to a network which can be either wired or wireless.

2.1.1 - Network connection

The calibrator can be connected to a network in two ways:

i) Wired connection: the calibrator can be connected to a wired network using its RJ-45 connection (labelled ETHERNET).

ii) Wireless connection: the calibrator can be connected to a wireless network using a Wi-Fi USB Adapter plugged to its USB Host port. This adapter is provided by Presys. If you do not have one, please contact Presys.

2.1.2 - Network configuration

Go to *Main Screen->Settings*:

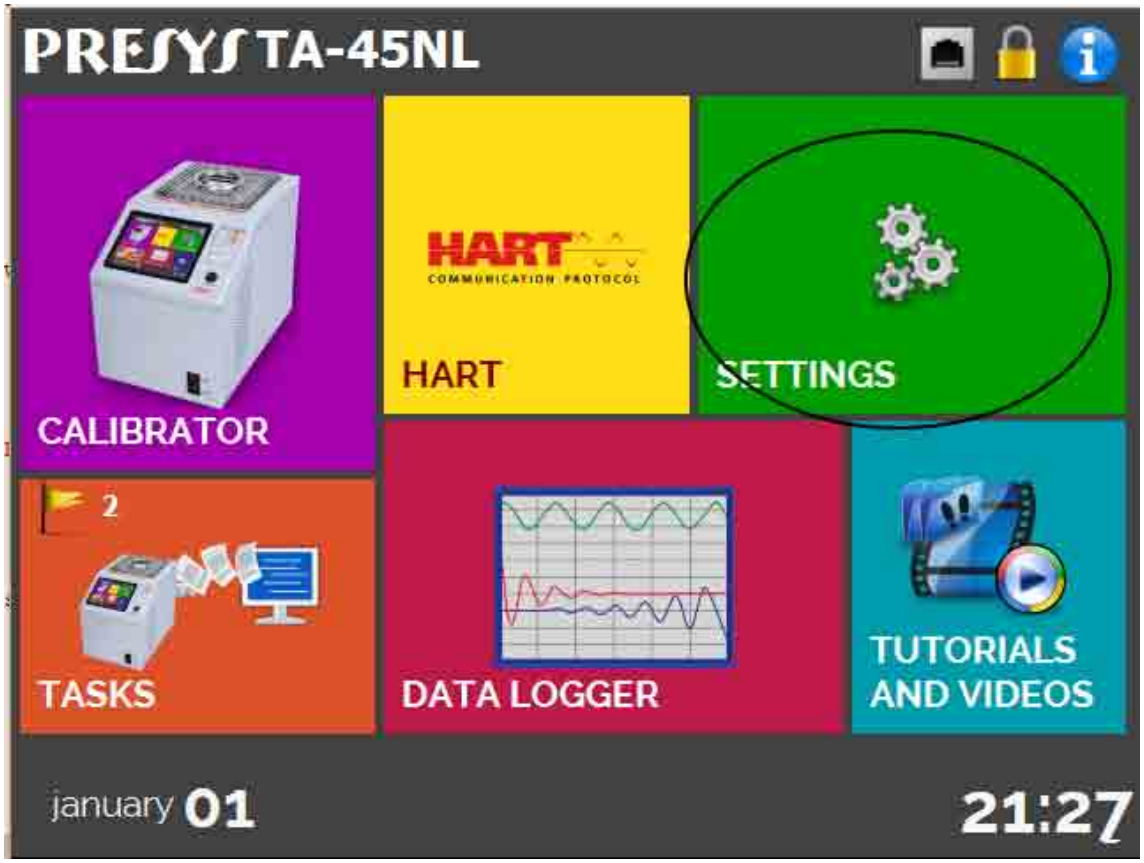

Fig. 1 – Main Screen

Next, go *Network* tab:

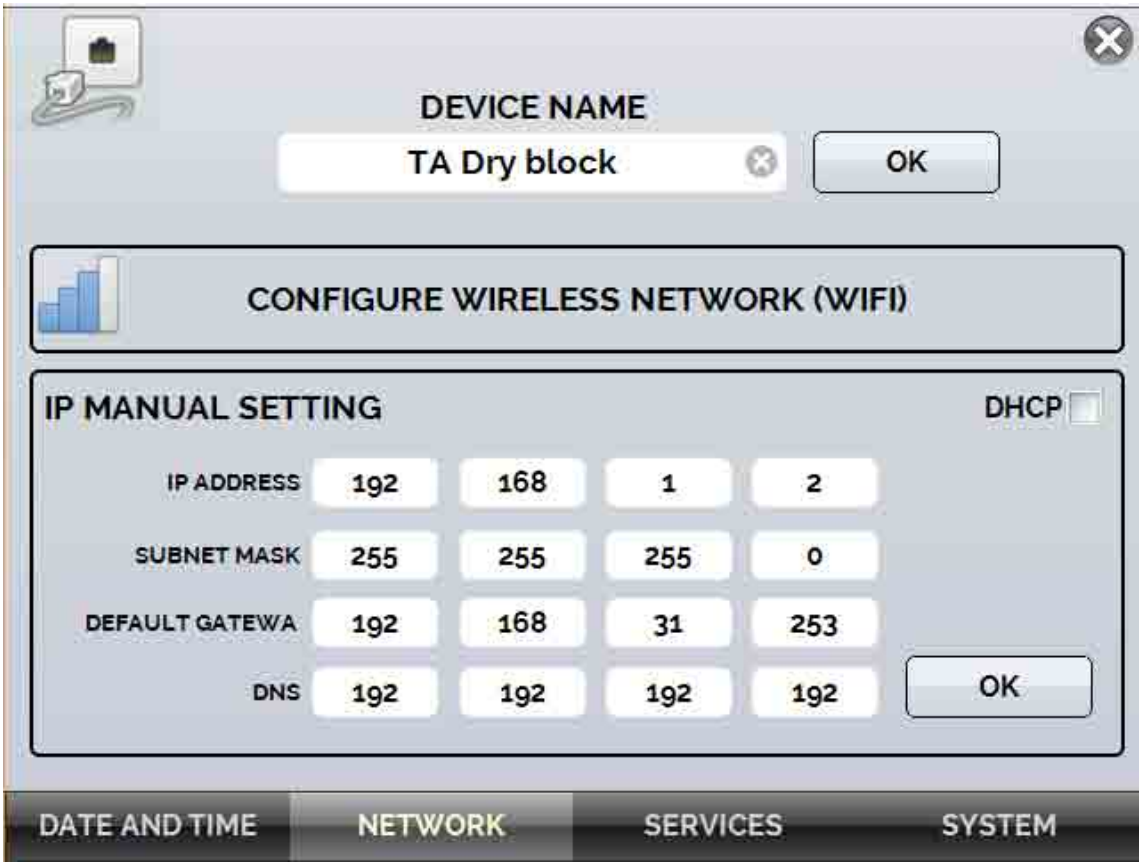

Fig.2 – Network tab

It is possible to enable or disable DHCP, checking or unchecking “DHCP” option. If DHCP is disabled, the user can edit IP Address, Subnet Mask, Default Gateway, and DNS settings. In order to confirm the configuration, click on OK. In case of wireless network, DHCP is always enabled.

In order to enable the wireless network, connect the Wi-Fi adapter into USB Host port of the calibrator and then click on CONFIGURE WIRELESS NETWORK (WIFI) button. If there is no Wi-Fi adapter, an error message will appear on the screen.

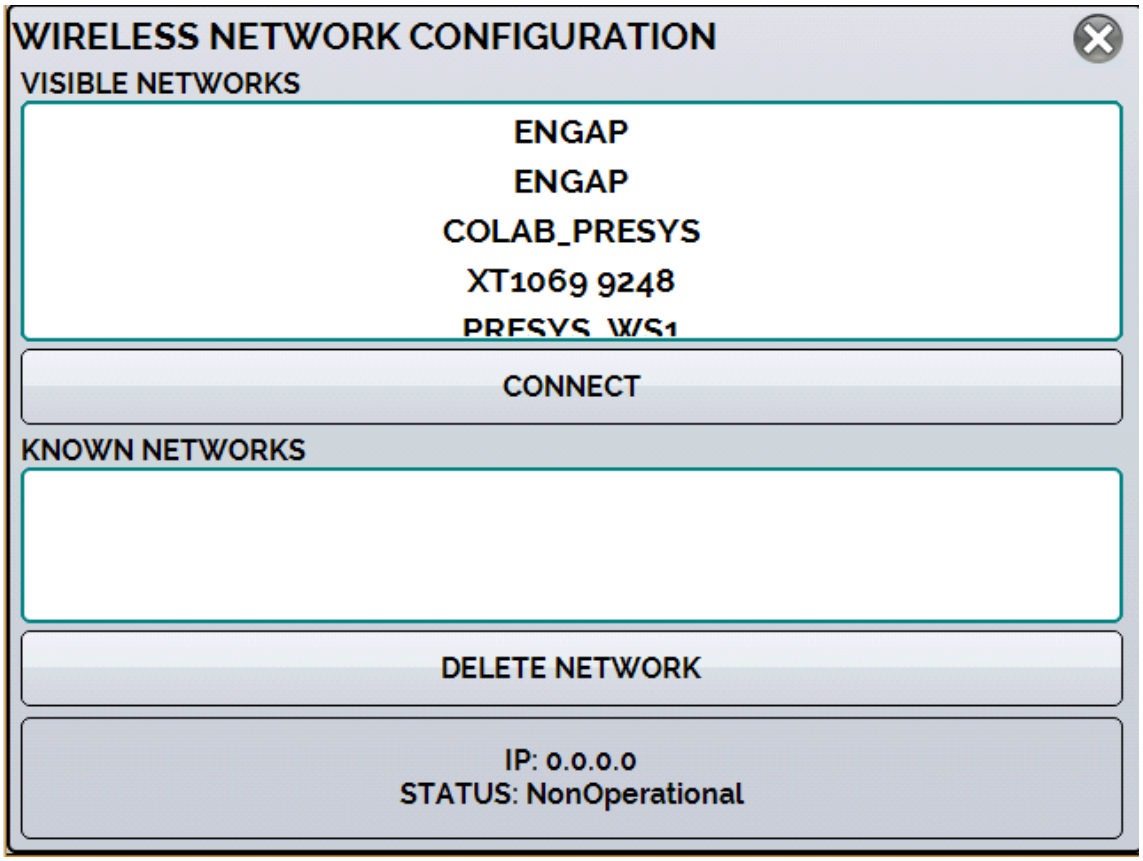

Fig.3 – Wireless Network Configuration Screen

Next, select an available network in the Visible Networks panel and the click on CONNECT button.

Fig.4 – Wireless Network Setting

Enter network password and security mode and then click on OK button. The network that has been connected is added into KNOWN NETWORKS and IP and STATUS are updated. In case of failure, IP and STATUS are not updated.

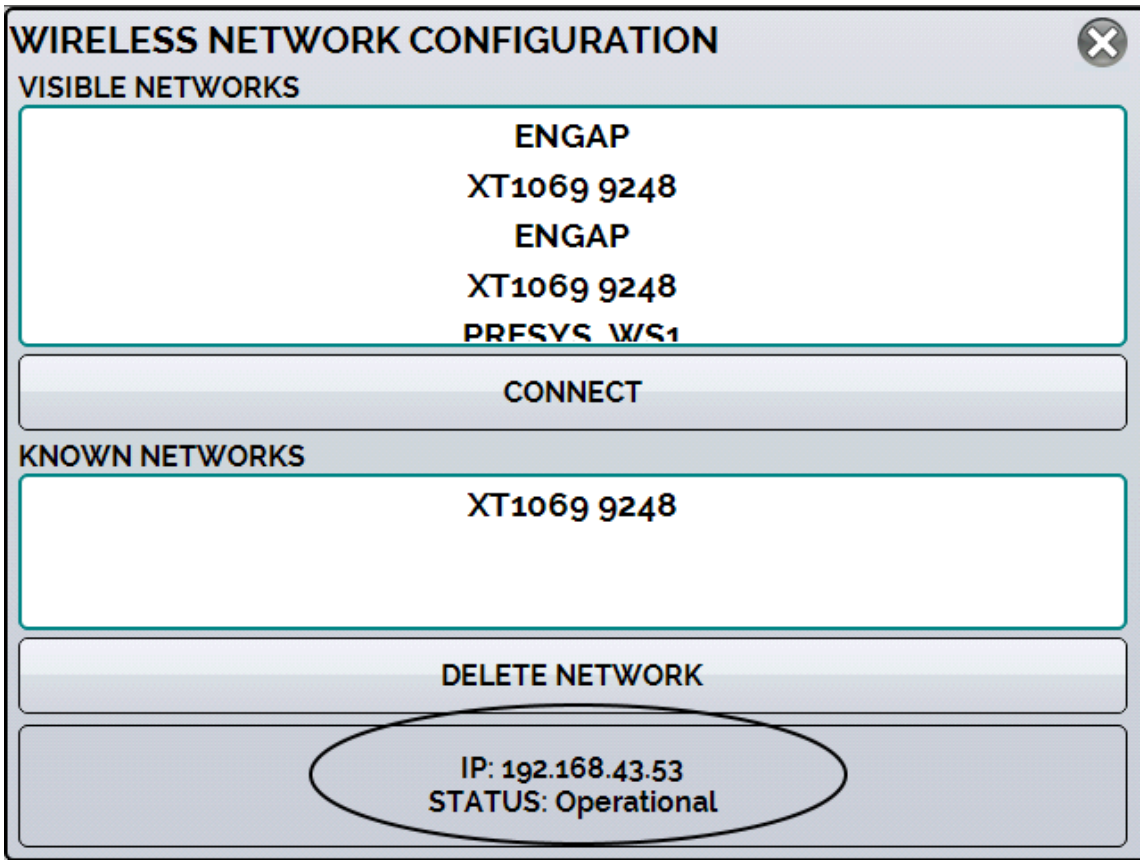

Fig.5 – Wireless Network Connected Succesfully

On Main Screen, the user can check the network connection status. If the calibrator is connected to both wired and wireless network only the Wi-Fi status icon will appear.

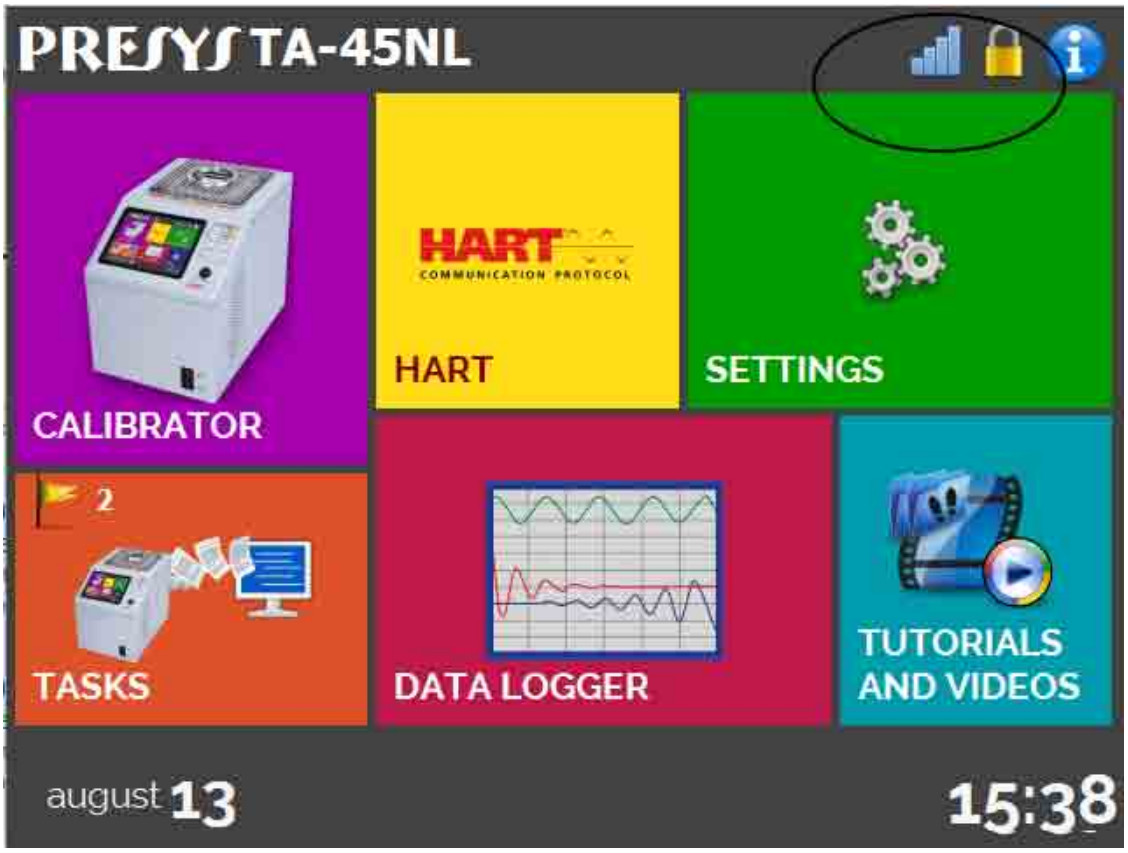

Fig.6 – Wireless Network Status Icon

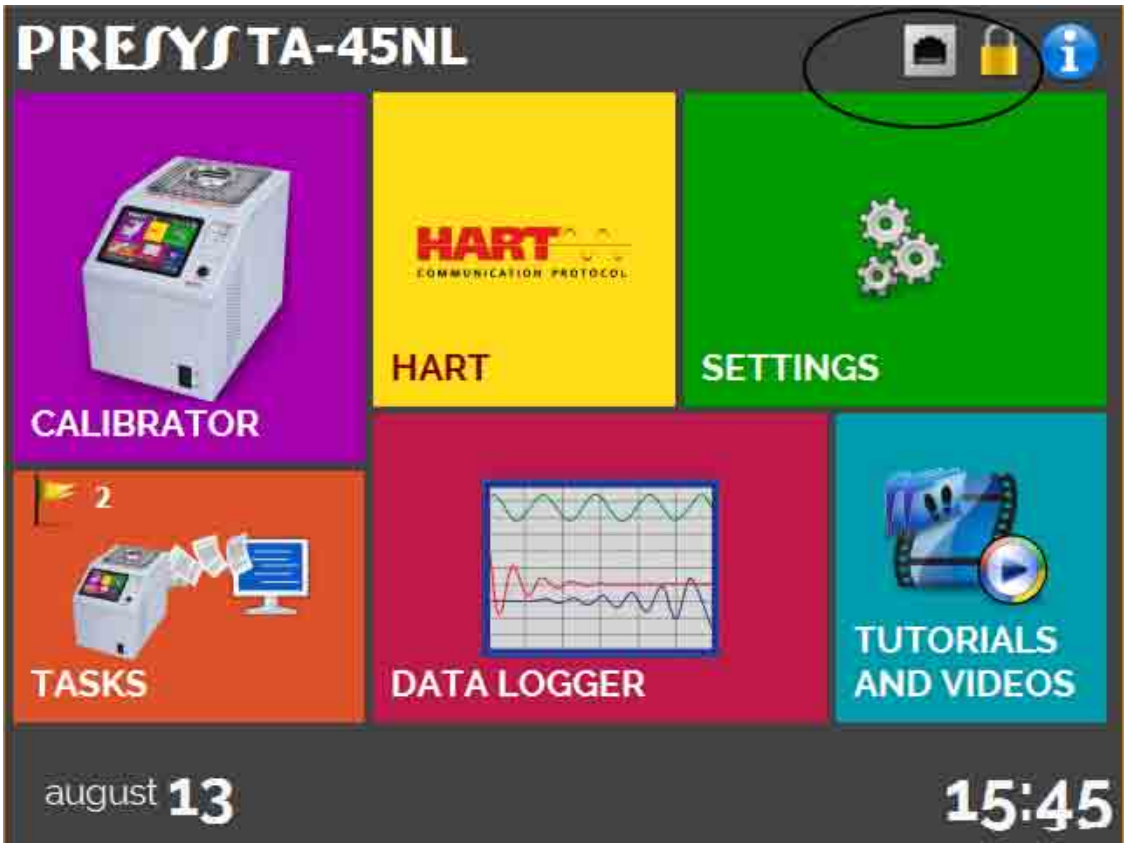

Fig.7 – Wired Network Status Icon

Clicking on the status icon, a network status popup window will appear:

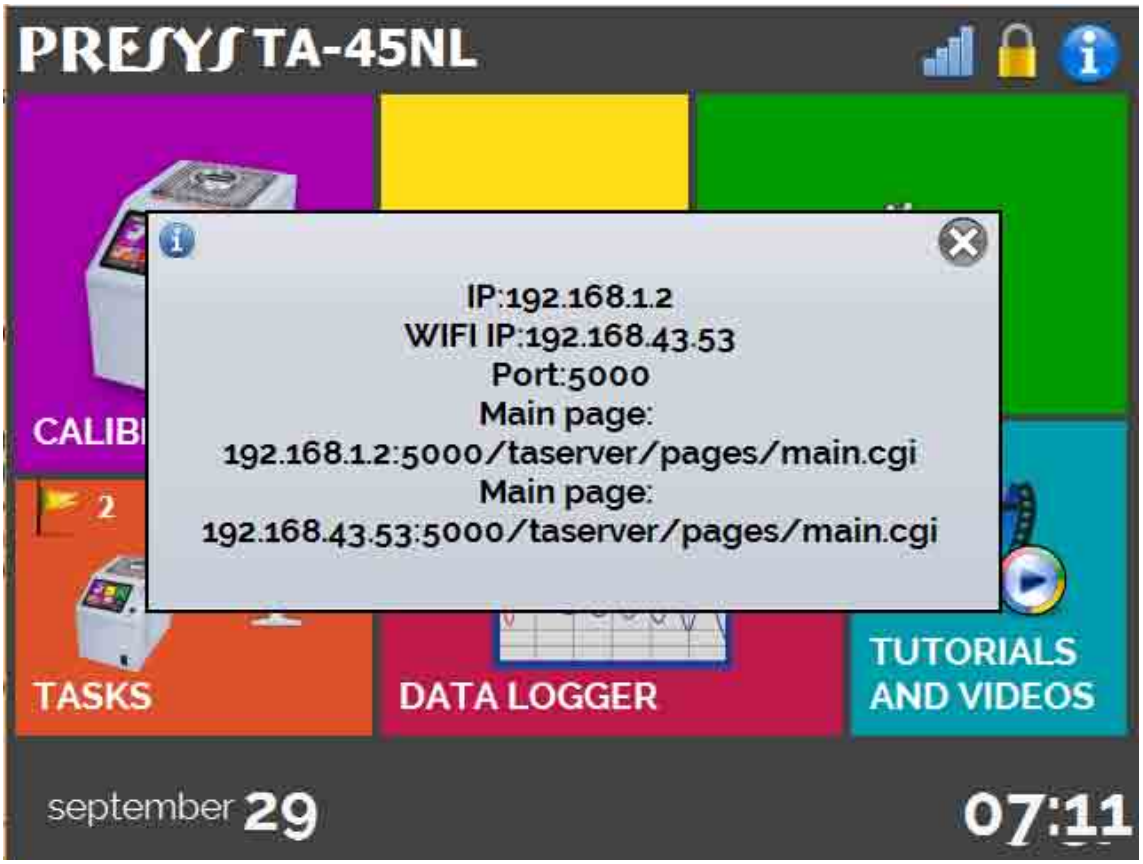

Fig.8 – Network Status Popup

The Main Page URL can be typed in a browser (Chrome, Firefox etc) to run a sample application that access the Web API described in this manual. The browser will ask for user name/password:

User Name: admin
Password: xvmaster

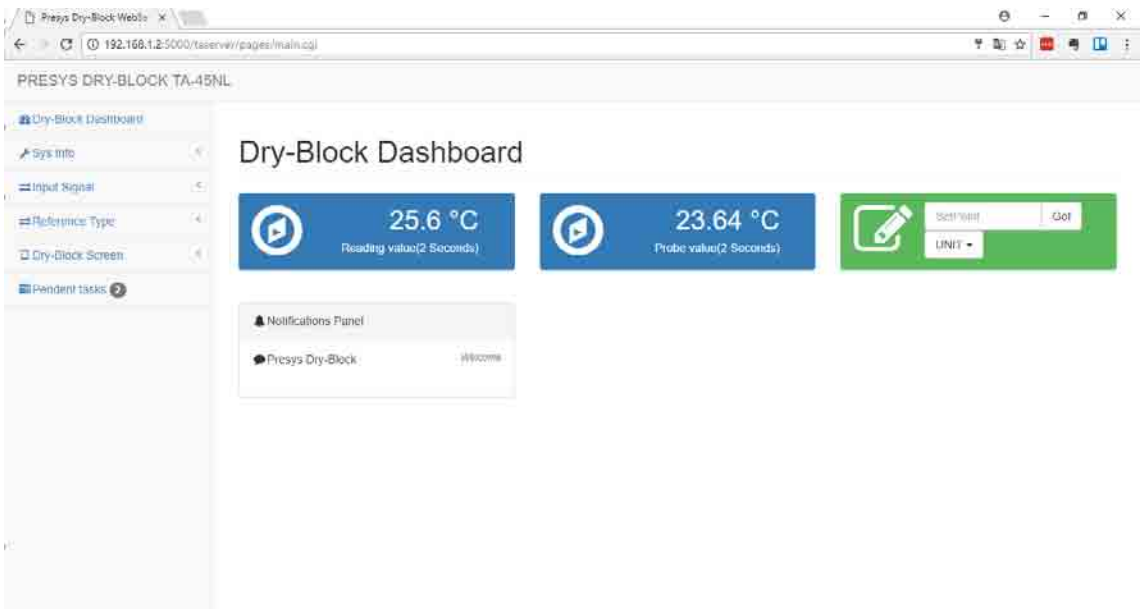

Fig.9 – Sample Application Running in a Browser

3 - Web API

The calibrators provide a HTTP-based web service which can be consumed easily by the clients. Through the web service, it is possible either to control a calibration remotely or transmit a task to/from calibrator. Thus, the user can develop their own application using the most diverse modern languages (C#, Java, Python etc) in order to control the calibrator.

The HTTP commands used are GET and POST with basic authentication scheme as defined in RFC 7617. For all commands, the credentials to be transmitted are:

User ID: admin
Password: xvmaster

The format data is proprietary, so the user has to implement a parser in order to manage it properly. Most of the time, the data is a text separated by “pipes” (|) and the application must split it.

3.1 - API Functions

In order to consume a service, the user has to send a HTTP request to the calibrator. For all functions, the URL format follows the format:

http://<X.X.X.X>:<YYYY>/<calibratorserver>/pages/<command>.cgi

When:

<X.X.X.X> - IP address
<YYYY> - TCP port
<calibratorserver> - ***mcsxvserver***, in case of MCS-XV
- ***pconserver****,* in case of *PCON*
- ***taserver****,* in case of TA Dry block

<command> - depending on function.

In order to simplify and avoid repetition, consider the IP address and TCP port values constants. The chosen values for these parameters are:

IP Address = 192.168.50.2
TCP port = 5000

3.1.1 - API Functions to control the calibrator

The user can control the calibrator using these functions. It is possible to change the input/output type of signal, read the current input, change the setpoint etc. Thereby, the user can develop an application that performs a complete calibration controlled remotely by another device such as a PC or a smartphone.

3.1.1.1 - Input Value Reading

Function:	Input Value Reading
HTTP Method	GET
Definition	Gets the value of auxiliary input 1 of the calibrator.
Available to	MCS-XV, PCON and TA Dry block
Command	getinput.cgi
Response Data Type	Text
Response Data Format	Value|Unit Value – Auxiliar Input 1 Value Unit – Auxiliar Input 1 Unit
Examples	Reading the auxiliary input 1 of a MCS-XV. The auxiliary input is set for reading thermocouple type-J, ITS-90 and internal CJC. The display shows -91.8 °C. GET http://192.168.50.2:5000/mcsxvserver/pages/getinput.cgi Response: -91.8|°C

3.1.1.2 - Output or Input Value Reading

Function:	Output or Input 2 Value Reading
HTTP Method	GET
Definition	Gets the value of auxiliary input 2 or output of the calibrator.
Available to	MCS-XV and PCON
URL	getoutput.cgi
Response Data Type	Text
Response Data Format	Value|Unit|Mode Value – Auxiliar Input 2 Value or Output Value Unit – Auxiliar Input 2 Unit or Output Value Mode – 1=output value / 2=input 2 value
Examples	i) Reading the auxiliary input 2 of a PCON. The auxiliary input is set for reading pressure. The display shows 100.123 psi GET http://192.168.50.2:5000/pconserver/pages/getoutput.cgi Response: 100.123|psi|2 ii) Reading the output of a MCS-XV. The output is set for generating voltage. The setpoint is 1.2345 V GET http://192.168.50.2:5000/pconserver/pages/getoutput.cgi Response: 1.2345|V|2

3.1.1.3 - Change Input Type

Function:	Change Input Type
HTTP Method	GET
Definition	Changes the auxiliary input 1 type
Available to	MCS-XV, PCON and TA Dry block
URL	setinputtype.cgi?newInput=<inputtype> <inputtype> = see section 5 for details
Response Data Type	Text
Response Data Format	OK:<inputrange>, in case of success FAIL: <error message>, in case of error
Examples	Changing the auxiliary input 1 of a TA Dry block type to mA. GET http://192.168.50.2:5000/taserver/pages/setinputtype.cgi?newInput=General:mA Response: OK:RANGE: -1 TO 24.5 mA

3.1.1.4 - Change Output Type

Function:	Change Output Type
HTTP Method	GET
Definition	Changes the output type
Available to	MCS-XV and PCON
URL	setoutputtype.cgi?newOutput=<outputtype> <ouputtype> = see section 5 for details
Response Data Type	Text
Response Data Format	OK:<outputrange>, in case of success FAIL: <error message>, in case of error
Examples	Changing the output of a MCS-XV type to V. GET http://192.168.50.2:5000/mcsxvserver/pages/setoutputtype.cgi?newOuput=General:V Response: OK:RANGE: -0.5 TO 12 V

3.1.1.5 - Change Dry block reference temperature

Function:	Change Dry block reference temperature
HTTP Method	GET
Definition	Changes the reference temperature of a Dry block
Available to	TA Dry block
URL	setoutputtype.cgi?newOutput=<outputtype> <ouputtype> = see section 5 for details
Response Data Type	Text
Response Data Format	OK:< outputrange >, in case of success FAIL: <error message>, in case of error
Examples	Changing the temperature reference of a TA-45NL to internal GET http://192.168.50.2:5000/taserver/pages/setoutputtype.cgi?newOuput=DryBlock:STD:Internal Response: OK:RANGE: -45.00 TO 140.00 °C

3.1.1.6 - Write Dry block Setpoint

Function:	Write Dry block Setpoint
HTTP Method	GET
Definition	Writes a setpoint to a Dry block
Available to	TA Dry block
URL	setpoint.cgi?spValue=<setpointvalue> <setpointvalue> = numerical value of setpoint
Response Data Type	Text
Response Data Format	OK:<newsetpointvalue>, in case of success FAIL: <error message>, in case of error
Examples	Changing the setpoint of a TA Dry block to 60.12°C GET http://192.168.50.2:5000/taserver/pages/setpoint.cgi?spValue=60.12 Response: OK:NEW SETPOINT VALUE: 60.12

3.1.1.7 - Read Dry block Temperature

Function:	Read Dry block Temperature
HTTP Method	GET
Definition	Reads the effective temperature that Dry block is generating
Available to	TA Dry block
URL	getpbvalue.cgi
Response Data Type	Text
Response Data Format	Value|Unit Value – Temperature Value Unit – Unit Value
Examples	Reading the Dry block temperature. The temperature is 31.23 °C GET http://192.168.50.2:5000/taserver/pages/getpbvalue.cgi Response: 31.23|°C

3.1.1.8 - Change Dry block Temperature Unit

Function:	Change Dry block Temperature Unit
HTTP Method	GET
Definition	Changes the temperature unit of the Dry block
Available to	TA Dry block
URL	changetempunit.cgi?unit=<unit> **unit:** °C, °F or K Note: the degrees signal (°) matches the code character 176 of the table ISO 8859-1.
Response Data Type	Text
Response Data Format	OK:<newunit>, in case of success FAIL: <error message>, in case of error
Examples	Changing the temperature unit of a TA Dry block to °C GET http://192.168.50.2:5000/taserver/pages/changetempunit.cgi?unit=°C Response: OK:NEW UNIT: °C

3.1.1.9 - Write PCON Setpoint

Function:	Write Dry block Setpoint
HTTP Method	GET
Definition	Writes a setpoint to a PCON
Available to	PCON
URL	genpressure.cgi/?pressureValue=<setpointvalue> <setpointvalue> = numerical value of setpoint
Response Data Type	Text
Response Data Format	OK:<newsetpointvalue>, in case of success FAIL: <error message>, in case of error
Examples	Changing the setpoint of a PCON to 100.123 psi GET http://192.168.50.2:5000/pconserver/pages/genpressure.cgi/?pressureValue=100.123 Response: OK:NEW SETPOINT VALUE: 100.123

3.1.1.10 - Read PCON Pressure

Function:	Read PCON Effective Pressure Generated
HTTP Method	GET
Definition	Reads the effective pressure that the controller is generating
Available to	PCON
URL	getpressureinput.cgi
Response Data Type	Text
Response Data Format	Value|Unit|Stable Value – Pressure Value Unit – Pressure Unit Stable – Flag that sinalizes if the pressure is stable (true) or not (false).
Examples	Reading the generated pressure of a PCON. The pressure generated is 100.123 psi and is stable. GET http://192.168.50.2:5000/pconserver/pages/getpressureinput.cgi Response: 100.123|psi|true

3.1.1.11- Change PCON Pressure Unit

Function:	Change PCON Pressure Unit
HTTP Method	GET
Definition	Changes PCON pressure unit
Available to	PCON
URL	changepressureunit.cgi?unit=<unit> **unit:** psi, bar, mbar, Mpa, kPa, Pa, atm, at, mmH2O, cmH2O, ftH2O, inH2O, inH2O@60°F, torr, mmHg, inHg, inHg@60°F, gf/cm^2, kgf/cm^2, kgf/m^2 Note: the degrees signal (°) matches the code character 176 and the signal (^{2}) to the code 178 of the table ISO 8859-1.
Response Data Type	Text
Response Data Format	OK:<unit>, in case of success FAIL: <error message>, in case of error
Examples	Changing the pressure unit of a PCON to psi. GET http://192.168.50.2:5000/pconserver/pages/changepressureunit?unit=psi Response: OK:Unit changed

3.1.1.12 - Get Input/Output String Representation

Function:	Get Input/Output String Representation
HTTP Method	GET
Definition	Gets the input/output string from the current configuration of the calibrator
Available to	MCS-XV, PCON and TA Dry block
URL	getctor.cgi?type=<type> <type> = input or output
Response Data Type	Text
Response Data Format	OK:<input/output string representation>
Examples	Given a MCS-XV configured to reads mA and generates V GET http://192.168.50.2:5000/mcsxvserver/pages/ctor.cgi?type=input Response: OK:General:mA GET http://192.168.50.2:5000/mcsxvserver/pages/ctor.cgi?type=output Response: OK:General:V *For more details about the string representation see section 5

3.1.2 - API Functions for sendind/receiving tasks

Another application can send or receive tasks to the calibrators. A task is a text document containing information about the calibration data such as calibration values and calibration procedure as points to be calibrated. This feature allows other applications to work disconnected during calibration as in the case where the calibrator is removed from the laboratory and taken to calibration in the field without access to the network. When the calibrator is returned to the laboratory, other applications can refer to all the tasks performed on the calibrator.

3.1.2.1 - Get Tasks

Function:	Get Tasks
HTTP Method	GET
Definition	Gets a list containing tasks from calibrator memory
Available to	MCS-XV, PCON and TA Dry block
URL	listtasks.cgi?complete=<complete> <complete> = true, in case of finished tasks false, in case of pending tasks *true could be omitted. See examples below for details.
Response Data Type	Text
Response Data Format	TASK LIST START <list of tasks>
Examples	There are two tasks called TT-01 and TT-02 waiting to be executed. Moreover, there are two tasks called TT-03 and TT-04 that have already been performed. The calibrator is a PCON GET http://192.168.50.2:5000/pconserver/pages/listtasks.cgi Response: TASK LIST START TT-03 TT-04 GET http://192.168.50.2:5000/pconserver/pages/listtasks.cgi?complete=false Response: TASK LIST START TT-01 TT-02

3.1.2.2 - Get Task

Function:	Get Task
HTTP Method	GET
Definition	Gets a specified task from a given task name
Available to	MCS-XV, PCON and TA Dry block
URL	gettask.cgi?taskname=<taskname>&nocomplete=true <taskname> - task name. The same name of xml file. *&nocomplete=true could be omitted. See examples below for details.
Response Data Type	XML
Response Data Format	TASK LIST START <list of tasks>
Examples	There are two tasks called TT-01 and TT-02 waiting to be executed. Moreover, there are two tasks called TT-03 and TT-04 that have already been performed. The calibrator is a PCON GET http://192.168.50.2:5000/pconserver/pages/gettask.cgi?taskname=TT-04 Response: TAGMAN DATA: {TT-04 taskdata} <EOL> GET http://192.168.50.2:5000/pconserver/pages/gettask.cgi?taskname=TT-01&nocomplete=true Response: TAGMAN DATA: {TT-01 taskdata} <EOL> See section 4 form details about taskdata

3.1.2.3 - Send Task

Function:	Send Task
HTTP Method	POST
Definition	Sends a task to calibrator
Available to	MCS-XV, PCON and TA Dry block
URL	sendtask.cgi
Message Body	<taskname> <task in XML format > where: <taskname>= task name. That is the xml file name. <task in XML format> = task in XML format according to section 4
Response Data Type	Text
Response Data Format	In case of success the response is: OK:FILE <Task Name> RECEIVED, < Total Characters > CHARS In case of failure the response is: FAIL: TASK_INVALID_OR_INCOMPLETE
Examples	The user wants to send a task, in xml format, to the calibrator. The desired task name is TT-05. The calibrator is a TA Dry block. POST http://192.168.50.2:5000/taserver/pages/sendtask.cgi Message Body: TT-05 { task in XML format } Response: OK:FILE TT-05 RECEIVED, 2112 CHARS See section 4 form details about XML format.

3.1.2.4 - Delete Task

Function:	Delete Task
HTTP Method	GET
Definition	Delete a task
Available to	MCS-XV, PCON and TA Dry block
URL	deletetask.cgi?taskname=<taskname>, for deleting a specified task given its name, pending or not; deletetask.cgi?taskname=ERASEALL, for deleting all pending tasks deletetask.cgi?taskname=ERASEALL&executed=true, for deleting all performed tasks
Response Data Type	Text
Response Data Format	OK
Examples	The user wants to delete a specified task name called TT-06. The calibrator is a TA Dry block GET http://192.168.50.2:5000/taserver/pages/deletetask.cgi?taskname=TT-06 Response: OK

4 - Calibrator Task File Format

The format of XML file used is as follows. In order to facilitate the explanation an example will be taken.

```
<!--PRESYS CALIBRATORS TAGMAN FILE-->
<tagman>
	<info>
		<exec>
			<tag>TT-001</tag>
			<serial>1234</serial>
			<created>06/02/2017 17:24</created>
			<maxerror>1.000</maxerror>
			<createdby>adminadmin</createdby>
			<executedby>MCS-XV TASK CREATOR</executedby>
			<model>dmy-2030</model>
			<manufacturer>presys</manufacturer>
			<message> Checkconnections </message>
			<localization> utilities </localization>
			<asfoundrepetitions>1</asfoundrepetitions>
			<asleftrepetitions>1</asleftrepetitions>
			<asfoundpointscount>2</asfoundpointscount>
			<asleftpointscount>2</asleftpointscount>
			<fs>20</fs>
			<errortype>span</errortype>
			<maxout>100</maxout>
			<minout>0</minout>
			<minin>4</minin>
			<db_error>0</db_error>
			<maxin>20</maxin>
		</exec>
	</info>
<input>
	<type>General:mA</type>
	<decimals>4</decimals>
	<min_range>4</min_range>
	<max_range>20</max_range>
	<manual_input>FALSE</manual_input>
	<has_scale>FALSE</has_scale>
	<scale_data />
</input>
<output>
	<outputlimits>
		<repeats>5</repeats>
		<strategy>UP</strategy>
		<interval>2</interval>
		<type>Thermoresistance:Pt-100 (IEC)|ITS-90</type>
		<tag />
	</outputlimits>
	<points_asfound>
		<value expected="4">0</value>
		<value expected="20">100</value>
```

```
        </points_asfound>
        <points_asleft>
                <value expected="4">0</value>
                <value expected="20">100</value>
        </points_asleft>
</output>
<executed_results>
        <asfound_operator>adminadmin</asfound_operator>
        <ASFOUND>
                <result expected="4" expected_gen="0"
                obtained="0.0001"point_value="0"error="-3.9999" date="2017-02-06"
                in_unit="mA" out_unit="°C" in_decimals="4" out_decimals="2" />
                <result expected="20" expected_gen="100" obtained="0.0001"
                point_value="100" error="-19.9999" date="2017-02-06" in_unit="mA"
                out_unit="°C" in_decimals="4" out_decimals="2"
                 />
        </ASFOUND>
</executed_results>
<executed_results>
        <ASLEFT>
                <result expected="4" expected_gen="0" obtained="0.0001"
                point_value="0" error="-3.9999" date="2017-02-06" in_unit="mA"
                out_unit="°C" in_decimals="4" out_decimals="2" />
                <result expected="20" expected_gen="100" obtained="0.0001"
                point_value="100" error="-19.9999" date="2017-02-06" in_unit="mA"
                out_unit="°C" in_decimals="4" out_decimals="2" />
        </ASLEFT>
<asleft_operator>adminadmin</asleft_operator>
</executed_results>
</tagman>
```

4.1 – XML Main Elements

XML Main Elements are the main elements of the task serving primarily to group other elements together. The elements values must set by the application.

Element	Description
tagman	Root element of xml file. Contains all elements of the xml file
info	Task overview
input	Input Calibrator Data*
output	Output Calibrator Data*
executed_result	Collected values

*Input/Output Calibrator refers to configuration of the physical input/output of the Calibrator. From the example above, the calibrator is configured to measures mA and generates Thermoresistance Pt-100.

4.2 – Info Element

The info element contains general information about the task as follow: The elements values must be set by the application.

Element	Description
info	Contains general information about the instrument to be calibrated.
Exec	Contains task execution data, such as Tag, maximum error, error source, etc.
tag	Tag of the instrument to be calibrated
serial	Serial number of the instrument to be calibrated
created	Task creation date
maxerror	Maximum allowed error. In case of thermoswitch, maxerror is the trip point error, in temperature unit.
createdby	Name of the person responsible for creating the task
executedby	Name of person responsible for executing the task
model	Model of the instrument to be calibrated
manufacturer	Manufacturer of the instrument to be calibrated
message	Message that will be displayed before starting the task
localization	Location where the instrument is located on the plant (e.g. Lab, Factory etc.)
asfoundrepetitions	Number of asfound repetitions
asleftrepetitions	Number of asleft repetitions
asfoundpointscount	Number of asfound points
asleftpointscount	Number of asleft points
fs	Full Scale Error
errortype	Error Base -Span=amplitude -FS=Full Scale -Reading -abs = absolute error. Only for thermoswitch calibration. -?= undefined
maxout	Maximum output of the instrument to be calibrated
minout	Minimum output value of the instrument to be calibrated
minin	Minimal input value of the instrument to be calibrated
maxin	Maximum input value of the instrument to be calibrated
db_error	Maximum error of the thermoswitch deadband, in temperature unit.

4.3 - Input Element

It contains the calibrator input data (output of the instrument to be calibrated). The elements values must be set by the application.

Element	Description
Type	String with input type, see section 5
Decimals	Decimal places
min_range	Minimum range
max_range	Maximum range
manual_input	True= Input entered by the user False= Input read by calibrator
has_scale	True= With scale False= Without scale
scale_data	Scale data

4.4 - Output Element

It contains the calibrator output data (input of the instrument to be calibrated). The elements values must be set by the application.

Element	Description
outputlimits	Output data
repeats	Repetitions (Isoplan® Software data, can be neglected)
strategy	Strategy One of the following options: UP= up DOWN = down UP/DOWN = up and down
interval	Stabilization time (in seconds). In case of themoswitch, interval means time ramp.
type	String with output type, see section 5
points_asfound	As found Calibration Points
points_asleft	As left Calibration Points
value	Element belonging to the preliminary and final calibration points. Value to be generated at calibrator output expected = attribute with expected value at calibrator input

Notes: The strategy is formed through the asfound / asleft points, so if the user wants to execute an UP / DOWN strategy, it would look like the following example:

```
<points_asfound>
    <valueexpected="0">0</value>
    <valueexpected="100">10</value>
    <valueexpected="100">10</value>
    <valueexpected="0">0</value>
```

```
        <valueexpected="0">0</value>
        <valueexpected="100">10</value>
</points_asfound>
```

4.5 - Results Element

It contains the results of the task execution. The elements values are set by the calibrator after the end of task execution.

Element	Description
asfound_operator	Name of the responsible for the as-found calibration.
asleft_operator	Name of the responsible for the as-left calibration.
ASFOUND	As-found calibration result
ASLEFT	As-left calibration result
result	Result obtained from each calibration point, see below the table of attributes

Attribute of the result element	Description
expected	Expected input value
expected_gen	Value generated at the output
obtained	Value obtained in the input
point_value	Real point value; Valid for PCON and TA
error	Error obtained
date	Date of execution
in_unit	Input unit (output of the instrument to be calibrated)
out_unit	Output unit (input of the instrument to be calibrated)
in_decimals	Input decimal places (output of the instrument to be calibrated)
out_decimals	Output decimal places (input of the instrument to be calibrated)

4.5 – Special Case: Thermoswitch Calibration in TA Dry block

Thermoswitch calibration is a special case. It is only possible to execute this kind of task in TA Dry block calibrator. The required elements for the thermoswitch calibration task are:

<errortype> = must be abs;
<maxout> = <maxin> = thermoswitch trip point, in temperature unit;
<minout> = <minin> = themoswitch deadband, in temperature unit;
<maxerror> = trip point error, in temperature unit;
<db_error> = deadband error, in temperature unit;

<type> = Switch
<decimals> = number of decimal places of the Dry block;
<min_range> = lower range of the Dry block;

<max_range> = upper range of the Dry block;

<points_asfound> = only two points are required. Trip point and deadband are the first and second point respectively.
<points_asleft> = only two points are required. Trip point and deadband are the first and second point respectively.

In order to make it clear, an example is presented below:

Thermoswitch data:
Trip point = 80 °C
Deadband = 10 °C
Trip point error = 1 °C
Deadband error = 2 °C
Time Ramp = 300 s

Dry block data:
TA-45NL

XML file:

```
<!--PRESYS CALIBRATORS TAGMAN FILE-->
<tagman>
	<info>
		<exec>
			<tag>SW-001</tag>
			<serial>001.01.01</serial>
			<created>2017-09-29 17:24</created>
			<createdby></createdby>
			<executedby>MCS-XV TASK CREATOR</executedby>
			<model> </model>
			<manufacturer></manufacturer>
			<message></message>
			<localization></localization>
			<errortype>abs</errortype>
			<maxout>80</maxout>
			<minout>10</minout>
			<maxin>80</maxin>
			<minin>10</minin>
			<maxerror>1</maxerror>
			<db_error>2</db_error>
		</exec>
	</info>
<input>
	<type>Switch</type>
	<decimals>2</decimals>
	<min_range>-25</min_range>
	<max_range>140</max_range>
</input>
<output>
	<outputlimits>
		<repeats>5</repeats>
		<strategy>UP</strategy>
```

```
                <interval>300</interval>
                <type>DryBlock:STD:Internal</type>
                <tag />
        </outputlimits>
        <points_asfound>
                <value expected="80">0</value>
                <value expected="10">100</value>
        </points_asfound>
        <points_asleft>
                <value expected="80">0</value>
                <value expected="10">100</value>
        </points_asleft>
</output>
<executed_results>
        <asfound_operator></asfound_operator>
        <ASFOUND>
                <result expected="80" expected_gen="80"
                obtained="80.12"point_value="80"error="0.12" date="2017-09-29"
                in_unit="°C" out_unit="°C" in_decimals="2" out_decimals="2" />
                <result expected="10" expected_gen="10"
                obtained="10.05"point_value="10"error="0.05" date="2017-09-29"
                in_unit="°C" out_unit="°C" in_decimals="2" out_decimals="2" />
                 />
        </ASFOUND>
</executed_results>
<executed_results>
        <ASLEFT>
                <result expected="80" expected_gen="80"
                obtained="80.12"point_value="80"error="0.12" date="2017-09-29"
                in_unit="°C" out_unit="°C" in_decimals="2" out_decimals="2" />
                <result expected="10" expected_gen="10"
                obtained="10.05"point_value="10"error="0.05" date="2017-09-29"
                in_unit="°C" out_unit="°C" in_decimals="2" out_decimals="2" />
        </ASLEFT>
<asleft_operator></asleft_operator>
</executed_results>
</tagman>
```

5 - String Specifications for Calibrator Input and Output

The calibrator input and output should be configured by means of text (string) which is used in calibration tasks and online calibration functions. The format of this text is explained below. PCON and TA Dry block configuration are considered as output.

5.1 - Input Specifications

Type of Input	Available for	String Representation
Current (mA)	MCS-XV/PCON/TA	"General:mA"
Voltage (V)	MCS-XV/PCON	"General:V"
miliVoltage (mV)	MCS-XV/PCON/TA	"General:mV"
Resistance (Ω)	MCS-XV/PCON/TA	"Resistance:{n}" n=0(2-wire measurement) n=1(3-wire measurement) n=2(4-wire measurement)
Thermoresistance (RTD)	MCS-XV/PCON/TA	"Thermoresistance:{type}|{temperature scale}:{wires}:{decimals}:{unit}" **type:** Cu-10, Ni-100, Pt-100 (IEC) or Pt-1000 **temperature scale:** ITS-90 or IPTS-68 **wires:** TWO, THREE or FOUR **decimals:** 0, 1 or 2 **unit:** °C, °F or K Note: the degrees signal (°)matches the code character 176 of the table ISO 8859-1
Thermocouple (TC)	MCS-XV/TA	"ThermoCouple:{type}|{temperature scale}:{CJC type}:{ CJC temperature}:{decimals}:{unit}" **type:** TC-J, TC-K, TC-T, TC-B, TC-R, TC-S, TC-E, TC-N, TC-U, TC-L or TC-C **temperature scale:** ITS-90 or IPTS-68

		CJC type: MANUAL or AUTO **CJC temperature:** CJC temperature value. In case of CJC type = INTERNAL, value must be 0 **decimals:** 0 or 1 **unit:** °C, °F or K Note: the degrees signal (°)matches the code character 176 of the table ISO 8859-1
Probe	MCS-XV	“Probe:{type}|{temperature scale}:{decimals}:{unit}” **type:** Pt-100 (IEC) **temperature scale:** ITS-90 or IPTS-68 **decimals:** 0, 1 or 2 **unit:** °C, °F or K Note: the degrees signal (°)matches the code character 176 of the table ISO 8859-1
Switch	MCS-XV/PCON/TA	“Switch”
Frequency	MCS-XV	“Freq:{type}:{time}” **type:** Freq or Counter **time:** Window time in counter mode. Set 0 in case of frequency mode.
Pressure	MCS-XV/PCON	Pressure:{sensor}:{decimals}:{unit} **sensor:** C1, C2, C3 or C4 **Decimal places:**

		0, 1, 2, ,3 or 4 (depending on unit) **unit:** psi, bar, mbar, Mpa, kPa, Pa, atm, at, mmH2O, cmH2O, ftH2O, inH2O, inH2O@60°F, torr, mmHg, inHg, inHg@60°F, gf/cm^2, kgf/cm^2, kgf/m^2 Note: the degrees signal (°) matches the code character 176 and the signal (^{2}) to the code 178 of the table ISO 8859-1.

5.2 - Output Specifications

Type of Output	Available for	String Representation
Current (mA)	MCS-XV	"General:mA"
Voltage (V)	MCS-XV	"General:V"
miliVoltage (mV)	MCS-XV	"General:mV"
Resistance (Ω)	MCS-XV	"Resistance:{scale}" **Scale:** R400(400 ohms maximum) R2500(2500 ohms maximum)
Thermoresistance (RTD)	MCS-XV	"Thermoresistance:{type}|{temperature scale}:{unit}" **type:** Cu-10, Ni-100, Pt-100 (IEC) or Pt-1000 **temperature scale:** ITS-90 or IPTS-68 **unit:** °C, °F or K Note: the degrees signal (°)matches the code character 176 of the table ISO 8859-1
Thermocouple (TC)	MCS-XV	"ThermoCouple:{type}|{temperature scale}:{ CJC type }:{CJC temperature CJC}: {unit}" **type:** TC-J, TC-K, TC-T, TC-B, TC-R, TC-S, TC-E, TC-N, TC-U, TC-L or TC-C **temperature scale:** ITS-90 or IPTS-68

		CJC type: MANUAL or AUTO **CJC temperature:** CJC temperature value. In case of CJC type = INTERNAL, value must be 0 **unit:** °C, °F or K Note: the degrees signal (°)matches the code character 176 of the table ISO 8859-1
Frequency	MCS-XV	"Freq:{type}:{amplitude}:{frequency}:{counts}" **type:** Freq10000; frequency up to 10000 Hz Freq100 ; frequency up to 100 Hz Counter ; counter mode **amplitude:** Integer representing the amplitude with two decimal places Ex: 1200 It corresponds to an amplitude of 12.00 V **frequency:** Frequency in counter mode. Set 0 in frequency mode. **counts:** Counts in counter mode. Set 0 in frequency mode.
Dry block Configuration (Temperature Generation)	TA	"DryBlock:{tag}:{type|control}" **tag:** Tag of the external reference. Tag must have been registered already. In case of type = INTERNAL, tag must be STD. **type:** INTERNAL or EXTERNAL **control:** true –dry block temperature is controlled by external reference. false – dry block temperature is

		controlled by internal reference. External reference is used only for indication. Note: control is not valid for TA-1200P/TA-1200PL
PCON Configuration (Pressure Generation)	PCON	“PRESSURE:{unit}:{type}” **unit:** psi, bar, mbar, Mpa, kPa, Pa, atm, at, mmH2O, cmH2O, ftH2O, inH2O, inH2O@60°F, torr, mmHg, inHg, inHg@60°F, gf/cm^2, kgf/cm^2, kgf/m^2 Note: the degrees signal (°) matches the code character 176 and the signal (^{2}) to the code 178 of the table ISO 8859-1. **type:** INVERTED, in the case of inverted calibration. For the normal calibration (non-inverted), this parameter must be omitted.

www.presys.com.br