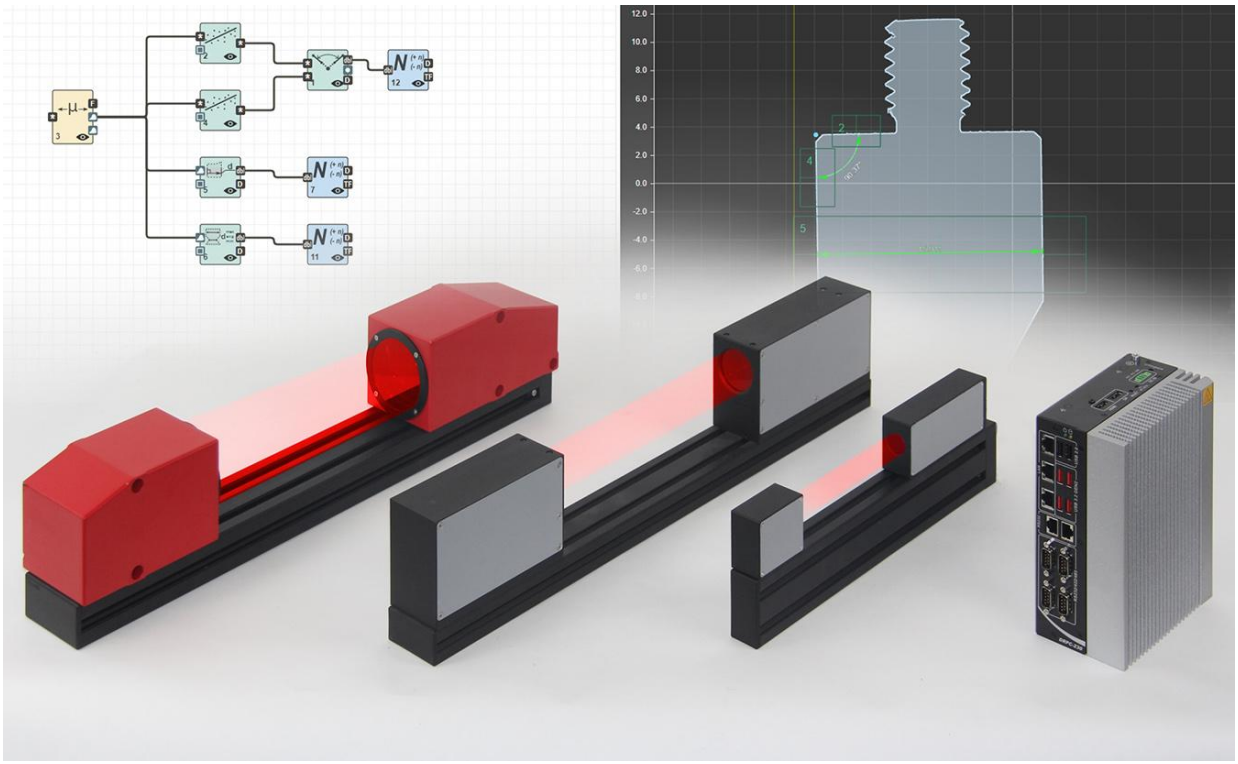
RIFTEK
Sensors & Instruments



2D OPTICAL MICROMETERS

RF65x.2D Series

User's manual

www.riftek.com
info@riftek.com

Contents

1. Safety precautions

- Use the supply voltage and interfaces given in the micrometer specifications.
- When connecting/disconnecting cables, the device must be turned off.
- Do not use the micrometer in locations close to powerful light sources.
- To obtain stable results, wait about 20 minutes after turning on the power to allow the optical sensor to warm up evenly.
- All components of the device must be grounded.

2. CE compliance

2D Optical Micrometers have been developed for use in industry and meet the requirements of the following Directives:

- EU directive 2014/30/EU. Electromagnetic compatibility (EMC).
- EU directive 2011/65/EU, “RoHS“ category 9.

3. Light source

The micrometers make use of the LED with a dominant wavelength of

- RF656.2D Series - 630 нм (red)
- RF657.2D and RF657R.2DR - 525 нм (green)

According to EN 62471:2008, the device is classified as safe.

4. General information

2D Optical Micrometers are designed for non-contact two-dimensional measurements of linear dimensions, diameters, angles, thread parameters, part shapes, etc. This Operating Manual is uniform for all series of 2D RF65x.2D optical micrometers, namely

- RF656.2D
- RF657.2D
- RF657R.2D

A single web interface is used to configure micrometers. The series differ in technical characteristics (range, speed, accuracy).

5. Structure and operating principle

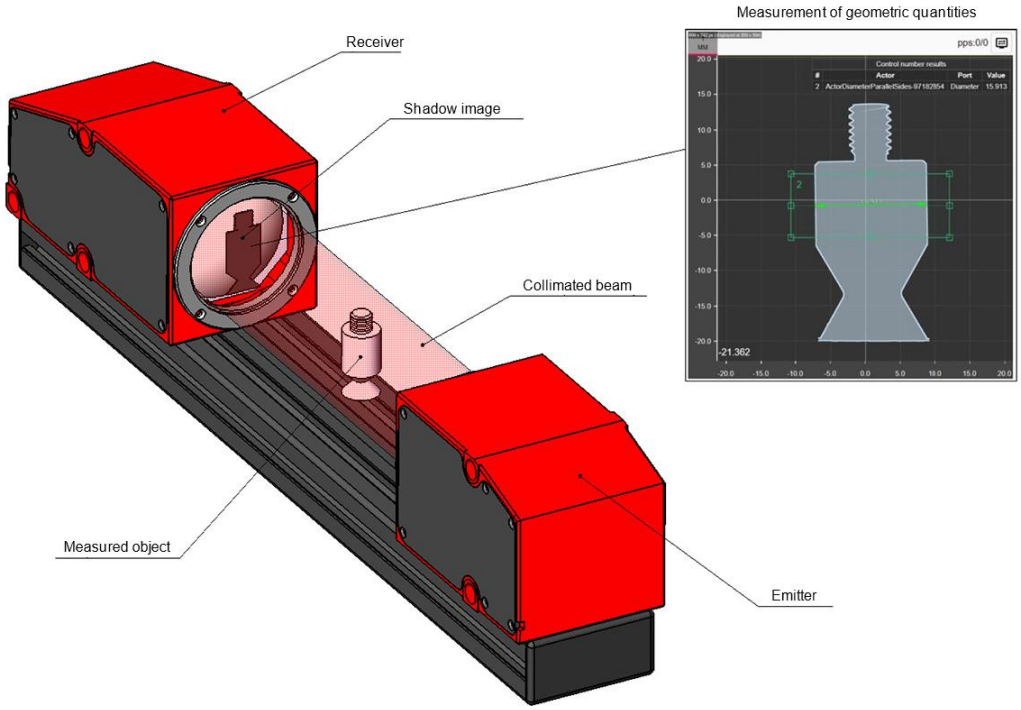
The operation of the micrometer is based on the so-called "shadow" principle.

The main components of the 2D micrometer are an optical sensor and a controller.

Optical sensor

Controller

The optical sensor of the micrometer consists of two parts - the emitter and the receiver. The light from the LED is collimated by the lens. When a product is placed in the region of a collimated beam, its shadow image is projected by the receiver lens onto the 2D CMOS sensor. According to the location of the shadow border of the image (object profile), the controller calculates the required parameters of the object.



Measurements and tolerance control are made according to the algorithm created by the user. To build the measurement algorithm, a simple and visual tool is proposed - the measurement scheme. The scheme is formed from a library of ready-made blocks. Various combinations of blocks and connections between them allow the user to create an almost unlimited number of measuring functions and measure products of varying complexity. Measurement results can be transmitted via various protocols (OPC UA, Modbus TCP, Siemens S7, UDP), as well as to the logical outputs of the micrometer for controlling actuators and signaling the suitability of the product.

6. Basic technical data

6.1. General specifications

Technical characteristics of optical sensors of RF656.2D Series:

RF656.2D	-8x10	-15x20	-25x35	-40x50
Measuring range, mm	8x10	15x20	25x35	40x50
Measurement error, µm	±1.5	±2	±2.5	±4.5
Distance along the axis at which the measurement error is applied, mm	±1	±2	±3	±4
Minimum object size, mm	0.07	0.2	0.2	0.35
Speed, measurements/s	up to 65			
Exposure time, µs	100			
Light source	LED, 630 nm, RED			
Overall dimensions, figure	1	2	2	4
Weight, not more, kg	1.1	2.3	2.3	5.6

Technical characteristics of optical sensors of RF657.2D Series:

RF657.2D	-15x20	-25x35	-40x50	-60x80
Measuring range, mm	15x20	25x35	40x50	60x80
Measurement error, µm	±0,8	±1,2	±2	±3
Distance along the axis at which the measurement error is applied, mm	±5	±10	±15	±20
Minimum object size, mm	0.13	0.13	0.2	0.3
Speed, measurements/s	24			
Exposure time, µs	15			
Light source	LED, 525 nm, GREEN			
Overall dimensions, figure	5	6	7	8
Weight, not more, kg	5	5.6	10.1	22.3

Technical characteristics of optical sensors of RF657R.2D Series:

RF657R.2D	-25	-45	-70	-100
Measuring range, diameter, mm	25	45	70	100
Measurement error, µm	±0.8	±1.2	±2	±3
Distance along the axis at which the measurement error is applied, mm	±5	±10	±15	±20
Minimum object size, mm	0.1	0.13	0.2	0.3
Speed, measurements/s	4			
Exposure time, µs	15			
Light source	LED, 525 nm, GREEN			
Overall dimensions, figure	5	6	7	8
Weight, not more, kg	5	5.6	10.13	22.3

General technical characteristics of RF65x.2D optical sensors:

Interface	
Basic interface	Ethernet / 1000 Mbps
Synchronization inputs	1 channel
Logic outputs	2 channels (1 channel is used as a strobe of active exposure)
Power supply, V	12...24
Power consumption, not more, W	6
Environmental resistance	
Enclosure rating	IP62
Vibration	20 g / 10…1000 Hz, 6 hours for each of XYZ axes
Shock	30 g / 6 ms
Operating ambient temperature, °C	-10…+50
Storage temperature, °C	-20…+80
Relative humidity, %	20-80 (no condensation)
Housing/windows material	aluminum/glass

Technical characteristics of RF65x.2D-SuM controllers:

Parameter	Value	
Speed, measurements/s	50 - RF656.2D and 24 - RF657.2D	
Interface		
Ethernet	3 x GbE, RJ-45 connectors	
COM port	4 x RS-232/422/485, DB-9 connectors. Changing the interface type can be done in the BIOS (see Annex 2. Configuring the protocol for controller serial interfaces). 2 x RS-232, RJ-45 connectors	
USB	4 x USB 3.2 Gen 2 (10 Gb/s) 2 x USB 2.0	6 x USB 3.2 Gen 2 (10 Gb/s)
Display	1 x HDMI 1 x DisplayPort	
Power		
Power supply, V	12...24	
Power consumption, not more, W	60	
Standard	AT/ATX, switchable	
Environmental resistance		
Operating ambient temperature, °C	-20…+60	
Storage temperature, °C	-40…+85	
Permissible relative humidity during use, %	10-95 (no condensation)	
Permissible relative humidity during storage, %	10-95 (no condensation)	
Shock	5G/11ms half-sine shock, 100 shocks for each of XYZ axes, IEC68-2-27	
Vibration	MIL-STD-810G 514.6C-1 (SSD)	
Weight, kg	2.9	
Overall dimensions, mm	81 x 150 x 190	

6.2. Overall dimensions

Detailed CAD documentation (2D and 3D) is available here:
https://riftek.com/upload/iblock/dd2/RF656.2D_2D_CAD.rar
https://riftek.com/upload/iblock/262/RF656.2D_3D_CAD.zip

The housing of the optical sensor is made of anodized aluminum. The overall and mounting dimensions of the sensors, as well as the field of view (measuring range) are shown in the figures below.

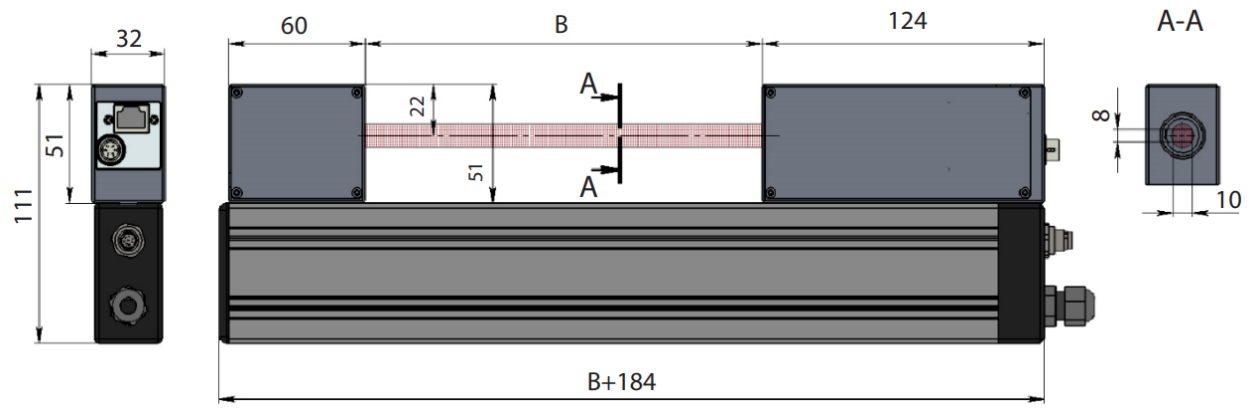


Figure 1. RF656.2D-8x10. B=20...100

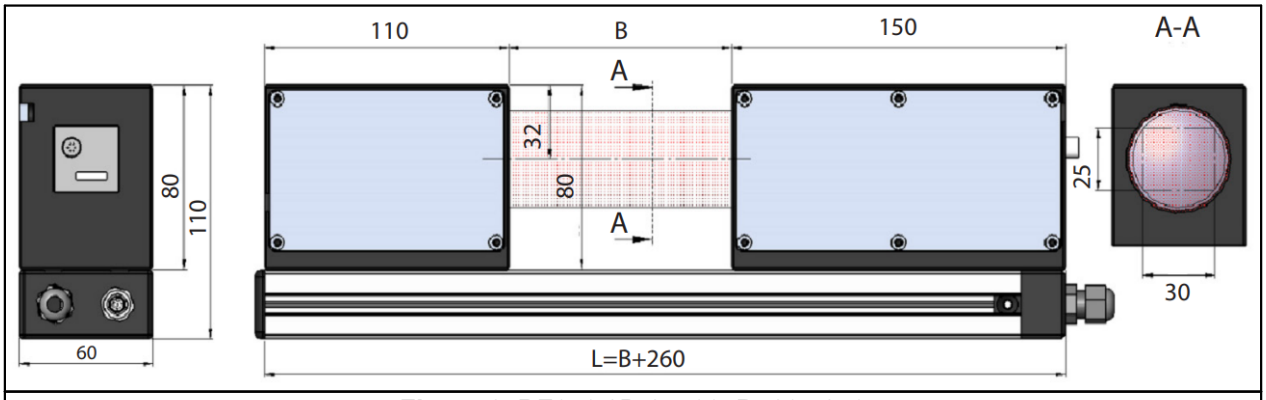


Figure 2. RF656.2D-25x30. B=20...240

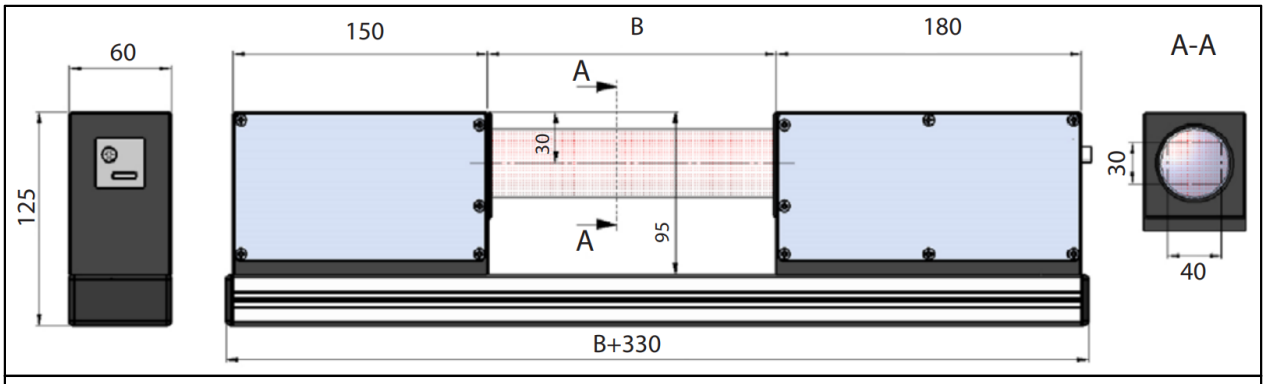


Figure 3. RF656.2D-30x40. B=20...250

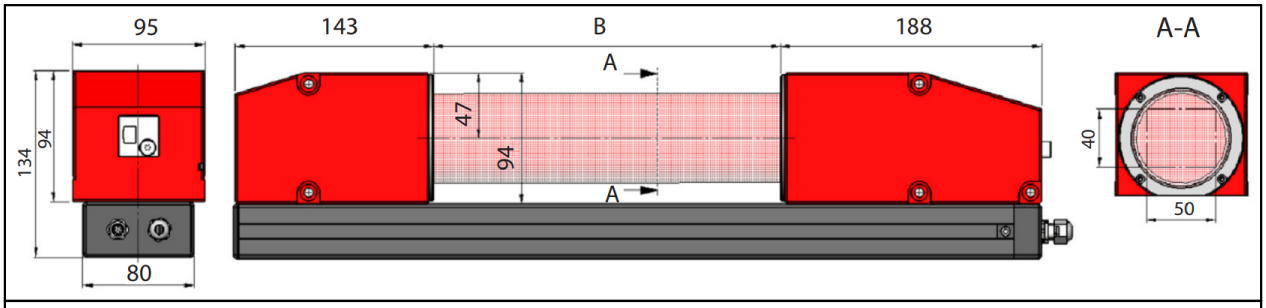


Figure 4. RF656.2D-40x50. B=20...500

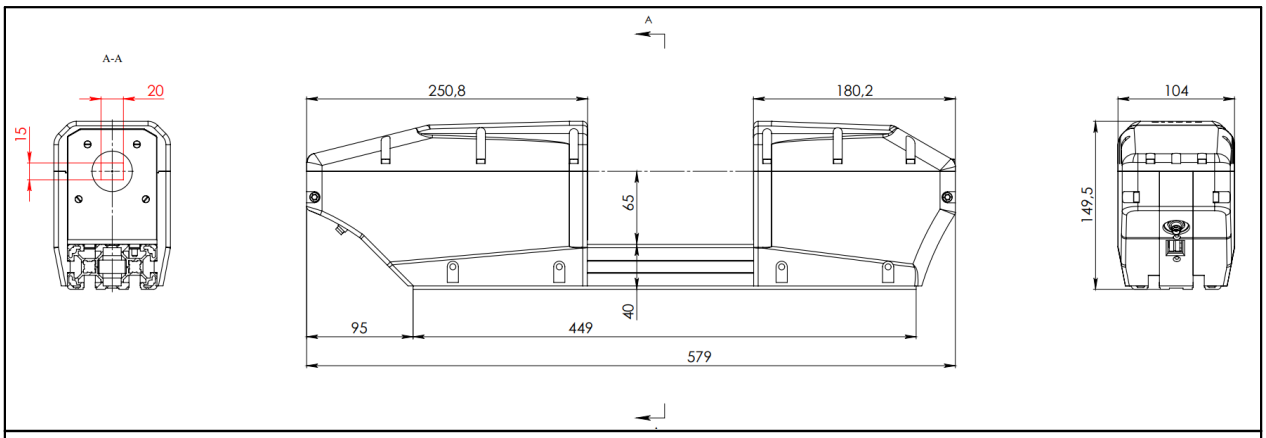


Figure 5. RF657.2D-15x20 and RF657R.2D-25

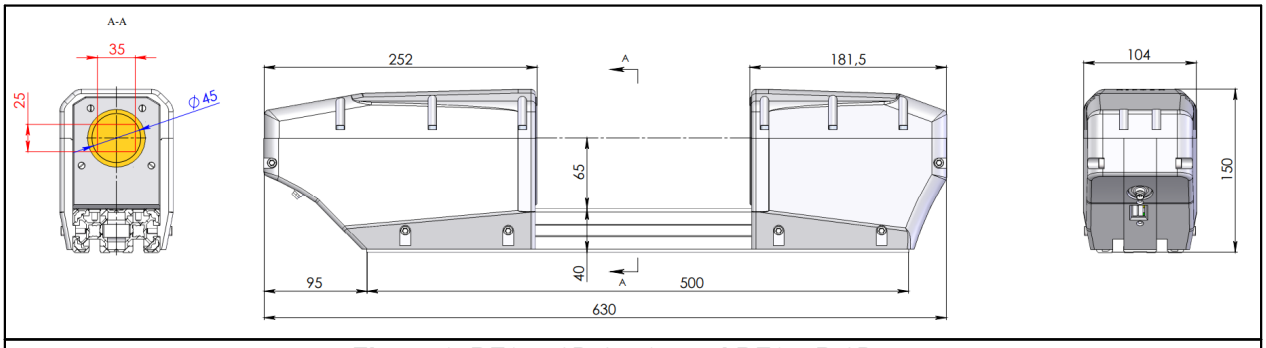


Figure 6. RF657.2D-25x35 and RF657R.2D-45

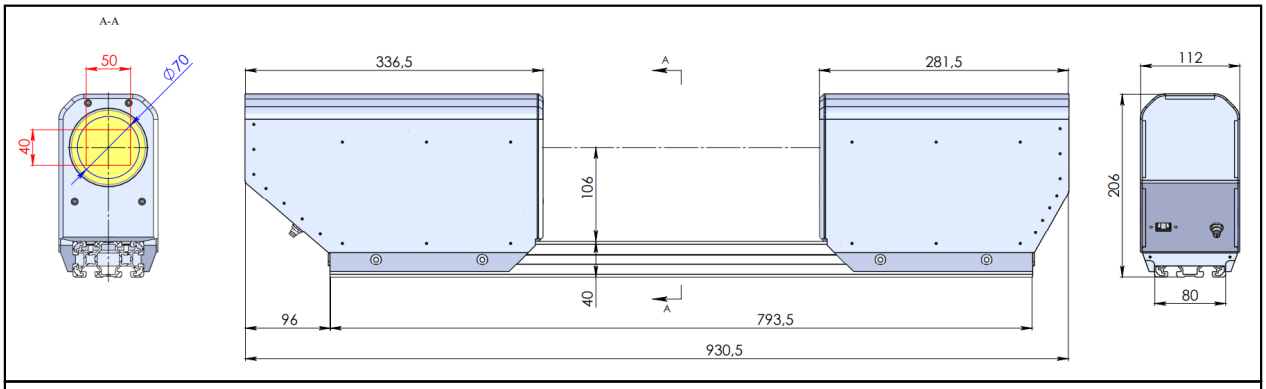


Figure 7. RF657.2D-40x50 and RF657R.2D-70

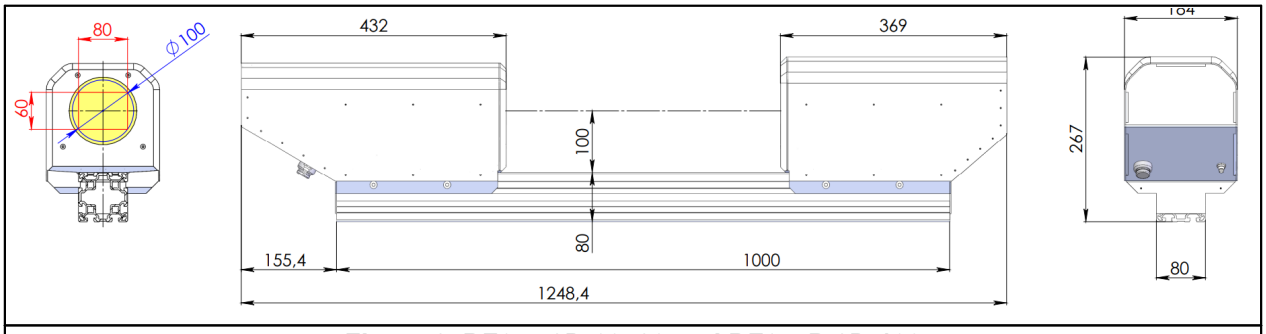


Figure 8. RF657.2D-60x80 and RF657R.2D-100

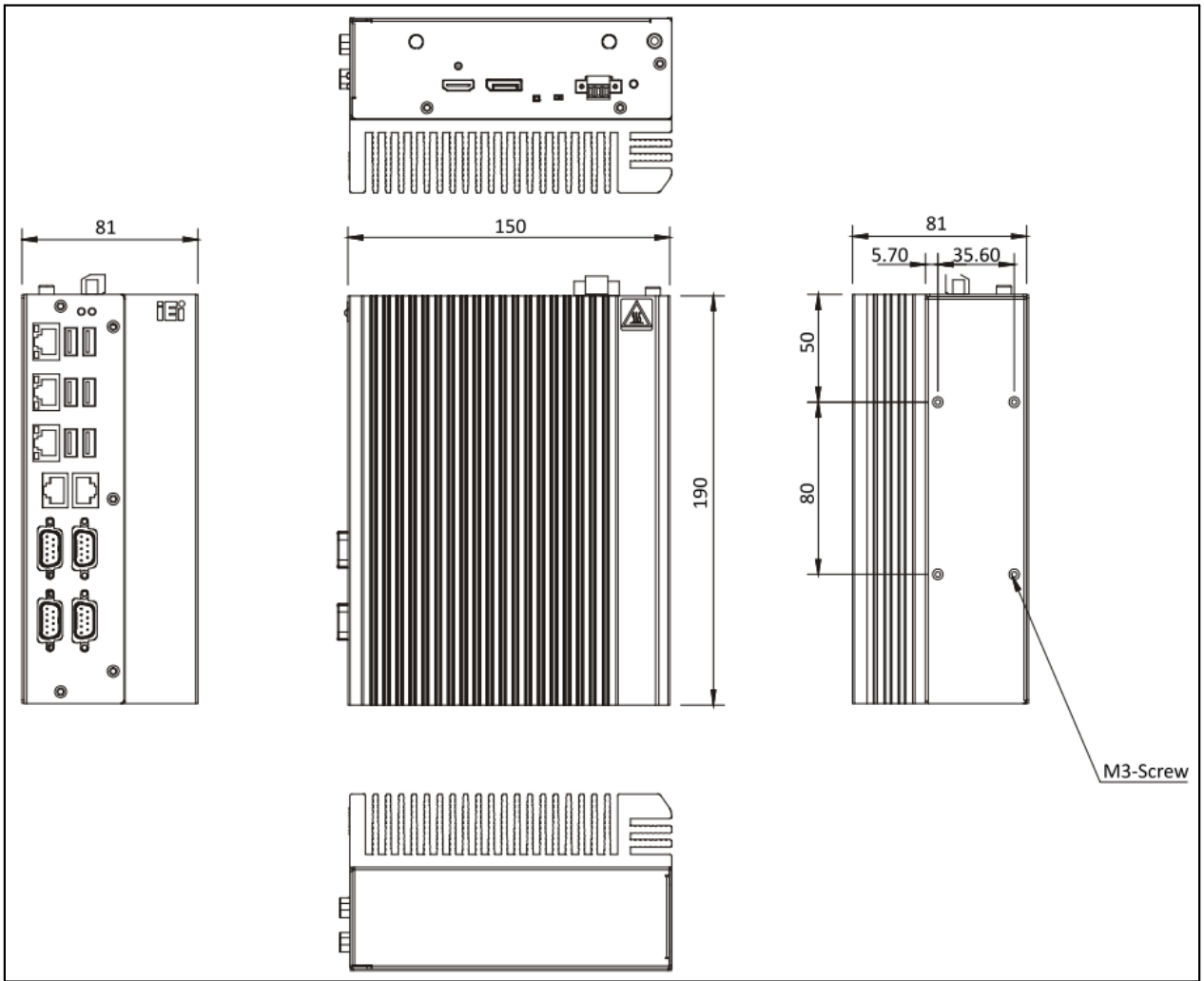
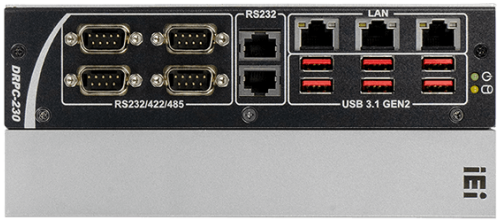


Figure 9. RF656.2D-SUM

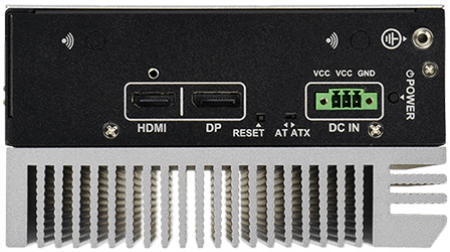
6.3. View of controller panels

The front panel of the HW1,HW2 controller includes:



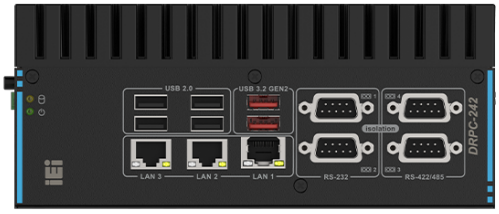
- 3 x RJ-45 Gigabit LAN with RJ-45 connectors.
- 4 x RS-232/422/485 serial ports with DB-9 connectors. Changing the interface type can be done in the BIOS (see Annex 2. Configuring the protocol for controller serial interfaces).
- 2 x RS-232 serial ports with RJ-45 connectors.
- 4 x USB 3.2 Gen 2 and 2 x USB 2 ports – for HW1 model, 6 x USB 3.2 Gen 2 ports – for HW2 model.
- Power LED – Green indicator.
- HDD LED – Yellow indicator.

The top panel of the HW1,HW2 controller includes:



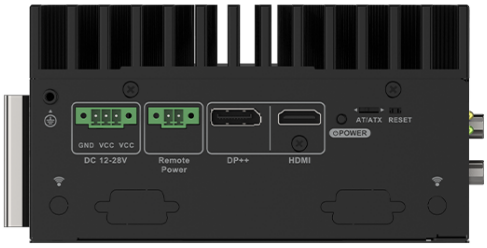
- DC IN – power connector for 12-24 V DC.
- Ground connector.
- HDMI.
- DisplayPort.
- "Power" – Power button.
- "Reset" – Reset button.
- AT/ATX switch.

The front panel of the HW3 controller includes:



- 3 x RJ-45 2.5 Gigabit LAN with RJ-45 connectors.
- 2 x RS-232 (COM1, COM2) and 2 x RS-422/485 (COM3, COM4) serial ports with DB-9 connectors. Changing the interface type can be done in the BIOS (see Annex 2. Configuring the protocol for controller serial interfaces).
- 2 x USB 3.2 Gen 2 and 4 x USB 2 ports.
- Power LED – Green indicator.
- HDD LED – Yellow indicator.

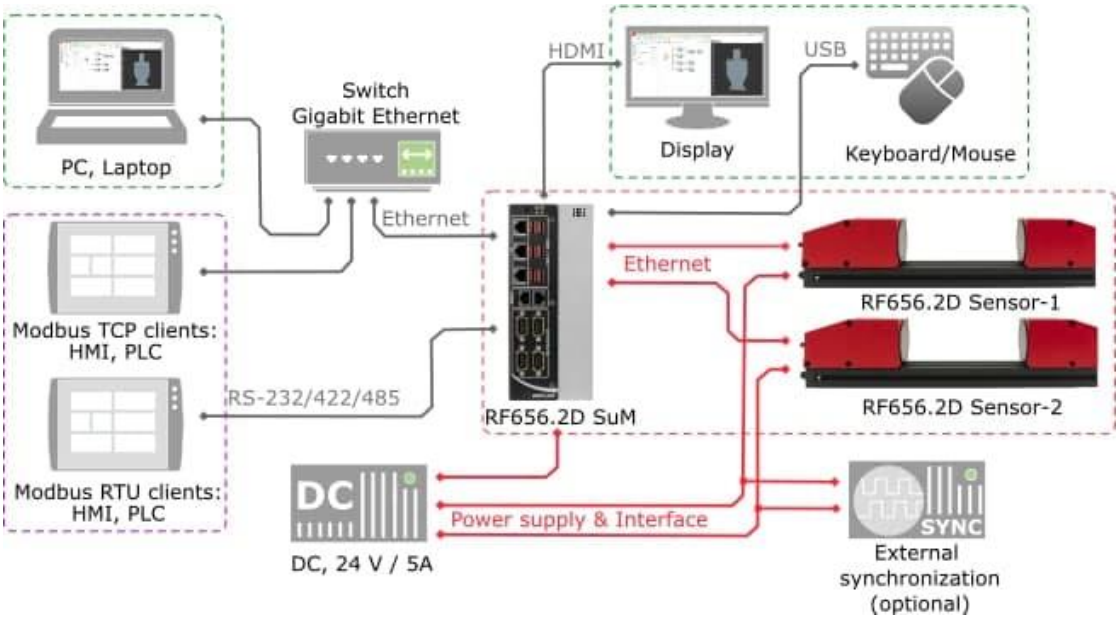
The top panel of the HW3 controller includes:



- DC IN – power connector for 12-28 V DC.
- "Remote power" – connector for an external power button.
- Ground connector.
- HDMI.
- DisplayPort.
- "Power Switch" – Power button.
- "Reset" – Reset button.
- AT/ATX switch.

7. Connection options

The block diagram of the connection options is shown in the figure.



The red box shows the standard set, which includes:

- Controller RF656.2D SuM.
- Optical sensor RF656.2D of the required range (up to four optical sensors can be connected to one controller).
- Ethernet cable for connecting the optical sensor to the controller.
- Optical sensor power cable with sync and output lines.
- Controller power cable.

NOTE. Pin assignment of connectors and cables, as well as electrical characteristics of the inputs/outputs of the optical sensor are shown in Annex 1.

The green box shows the service equipment needed to operate the micrometer. A computer or a display with a keyboard connected to the controller is used to parameterize the micrometer, generate measurement schemes, display the result, etc.

The purple box shows the process automation tools (operator panel and/or programmable logic controller) connected to the micrometer controller, if needed.

8. Example of item designation when ordering

RF656.2D-R-LP-LS-LI

Symbol	Description
x	6 or 7 or 7R
R	Measuring range of the optical sensor. RF656.2D-(FOV, height x width, mm): • 8x10 • 25x30 • 30x40 • 40x50 RF657.2D-(FOV, height x width, mm): • 15x20 • 25x35 • 40x50 • 60x80 RF657R.2D-(FOV, diameter, mm): • 25 • 45 • 70 • 100
LP	The length of the controller power cable, m.
LS	The length of the power and sync cable of the optical sensor, m.
LI	The length of the Ethernet cable, m (max. 100).

Example: RF656.2D-40x50-3-3-10 - optical sensor with measuring range 40x50 mm, controller power cable length 3 m, optical sensor power cable length 3 m, Ethernet cable length 10 m.

9. Overall demands for mounting

The optical sensor of the micrometer is installed in such a way that the controlled object is within the measuring range of the sensor. In addition, there should be no foreign objects in the area of the collimated beam.

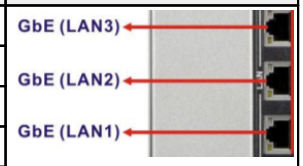
Avoid direct sunlight on the optical sensor and the measured object.

ATTENTION!
The optical sensor of the micrometer and the controller of the micrometer must be grounded. Static electricity can cause the failure of electronic components.

10. Network setup and connection

10.1. Network setup

Unless otherwise specified in the order, all controllers are shipped with the following Ethernet settings:

Parameter	LAN1 (enp1s0)	LAN2 (enp2s0)	LAN3 (eno1)	Network interfaces
mode	static - a static address is assigned.			
IP Address	192.168.1.130	192.168.3.130	192.168.2.130	
mask	255.255.255.0			
gateway	192.168.1.1	192.168.3.1	192.168.2.1	
dns	192.168.1.1	192.168.3.1	192.168.2.1	

To connect to the controller, configure the network settings of the connected PC/device as follows:

- LAN1: device address must be 192.168.1.*, mask - 255.255.255.0
- LAN2: device address must be 192.168.3.*, mask - 255.255.255.0
- LAN3: device address must be 192.168.2.*, mask - 255.255.255.0

(* is any number from 1 to 254, except 130)

Unless otherwise specified in the order, all sensors are shipped with the following factory settings:

Parameter	Value
mode	static
IP Address	192.168.3.30
mask	255.255.255.0
gateway	192.168.3.1
dns	-

The network parameters of both the controller and the sensor can be changed using the service software (SDK), service protocol, or on the device web page.

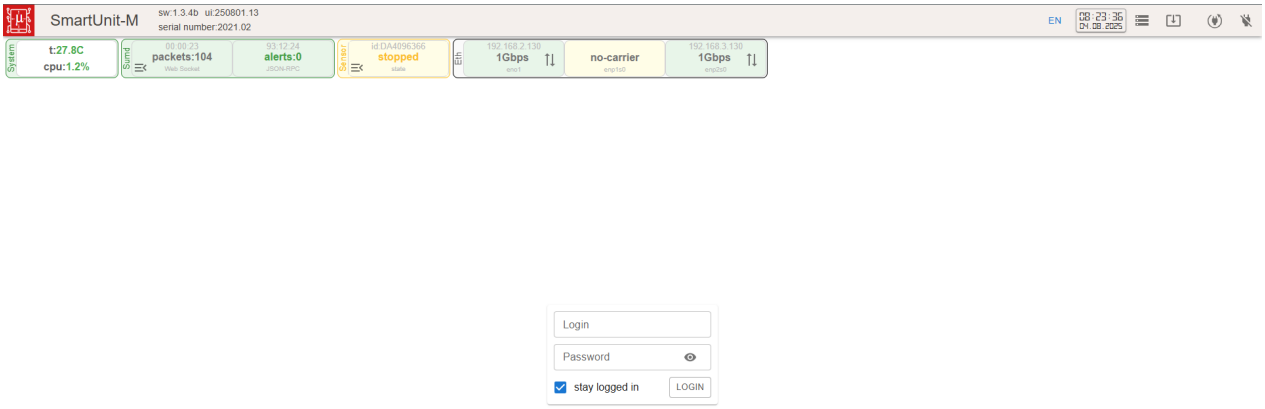
10.2. First startup

- Make network settings according to the previous paragraph.
- Connect the service equipment (PC or switch) to the LAN1 or LAN3 output of the controller.
- Connect the optical sensor to the LAN2 output of the controller.
- Connect the power supply (12...24V) to the controller (DC IN connector on the top panel of the controller).
- Connect the power supply (12...24V) to the optical sensor (red wire "+", brown wire "-").

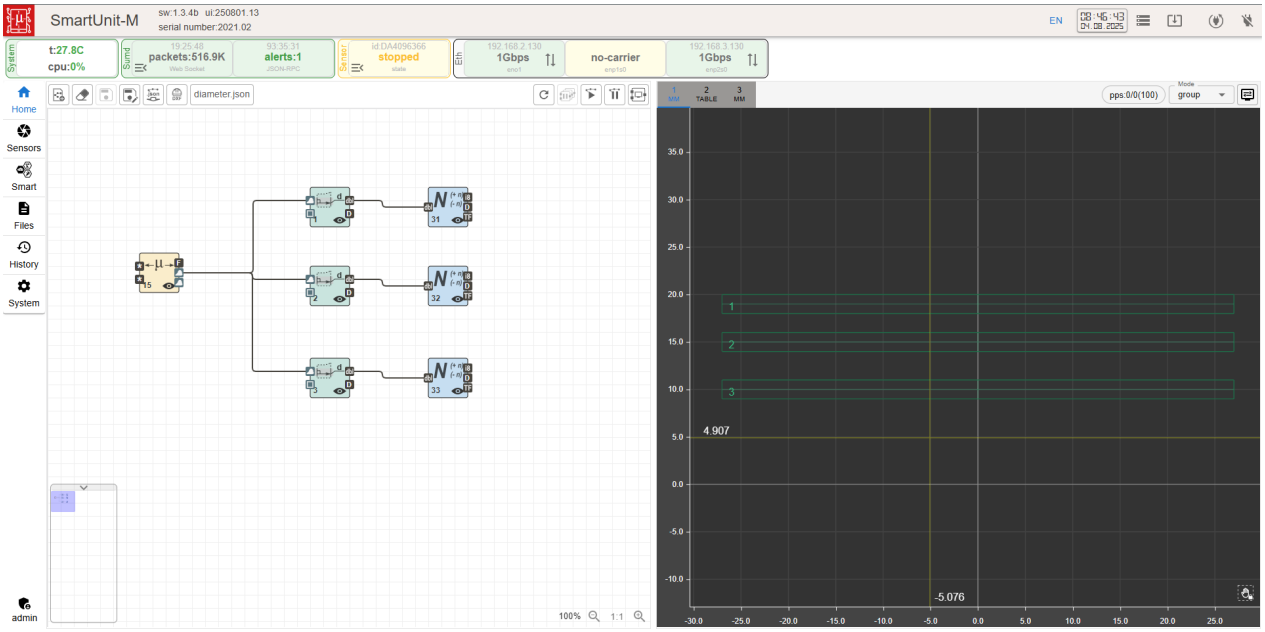
Within 15-30 seconds after the controller is turned on, the controller firmware is loaded and the Ethernet interface is initialized.

Next, it is recommended to go to the web page of the micrometer, which can be accessed from any browser - enter the network address of the controller into the address bar of the web browser, namely 192.168.1.130 when connected to LAN1 or 192.168.2.130 when connected to LAN3.

If all settings are correct, the browser will display the login page:



In the login form, enter the following credentials:
Login: admin
Password: admin
Then click the **LOGIN** button.
If the authorization is successful, the main page of the web interface will appear:



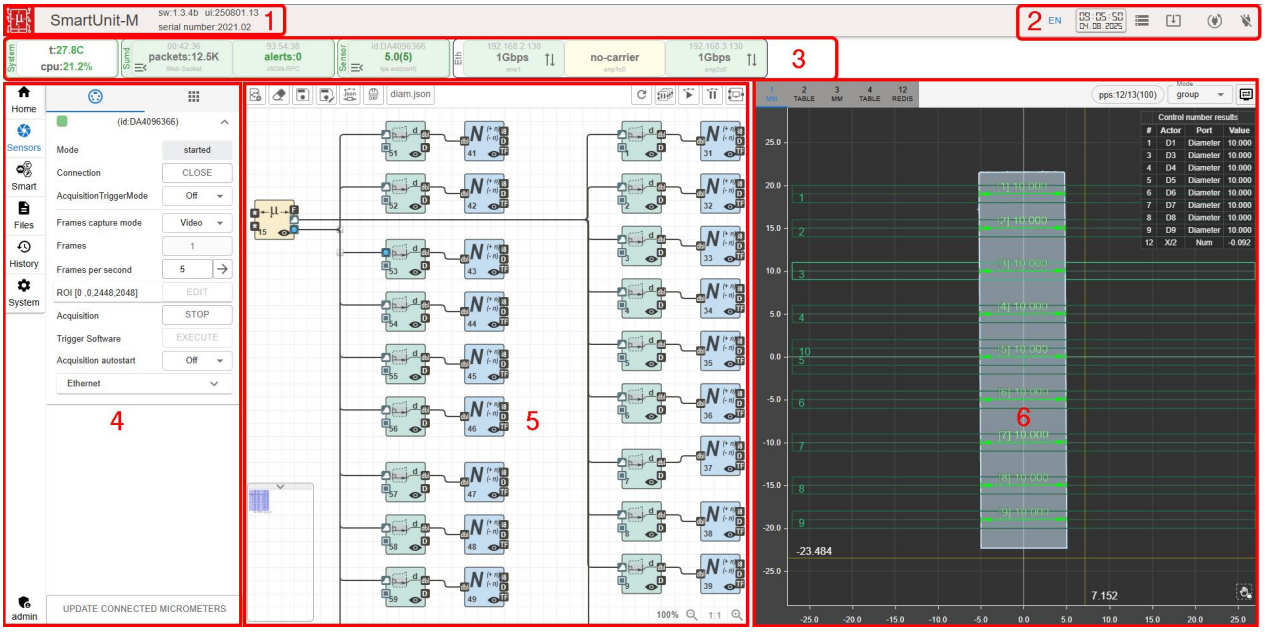
Evaluate the operation of the controller and the optical sensor by the status indicators located at the top of the web page (see Web interface).

The optical sensor is turned off by removing the supply voltage.

The controller can be turned off using the service software (SDK), the service protocol, the "Power" button on the top panel of the controller, and the web page of the micrometer.

11. Web interface

2D Optical Micrometers RF65x.2D have an embedded web page, which can be accessed from any browser by entering the network address of the controller into the address bar of the browser. The web page is intended for checking the operation of the micrometer, setting parameters, accumulating and displaying a shadow image and a profile of parts, and creating the measurement scheme.



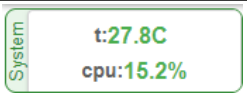
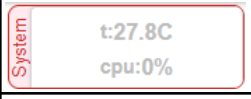
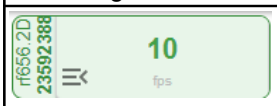
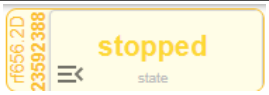
The web page is divided into six areas:
1 - General information (controller name and firmware version).
2 - Control buttons.
3 - Status indicators.
4 - Parameterization tabs.
5 - Measurement scheme.
6 - Measurement results.

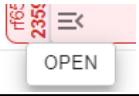
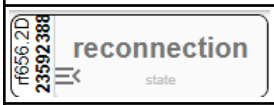
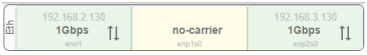
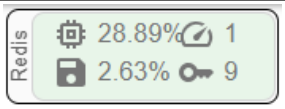
Area 1 contains the name of the controller, its serial number and firmware version. The name can be changed by the user.

Area 2 contains the following control buttons:

Button	Name	Description
EN	**Language**	Changing the interface language.
	File Browser	Opening the file manager.
	Update Firmware	Updating the controller firmware.
	System Backup	Opening the page for creating a backup of system settings and files.
	Restart device	Restarting the controller.
	Power off device	Turning off the controller.

Area 3 contains a set of status indicators for the controller and optical sensors:

Group	Description	
System	Displays the controller status. The check is performed by the availability of the SUD (Smart Unit Daemon) service.	
		The controller is running. The temperature and CPU load are displayed. This information is for reference only and is used to evaluate the operating conditions of the controller. The temperature must not be allowed to rise to 90°C or more. Temperature indicator color, t °C: • Green: 0 < t ≤ 61 • Yellow: 61 < t ≤ 91 • Red: 91 < t CPU indicator color, %: • Green: 0 < % ≤ 61 • Yellow: 61 < % ≤ 91 • Red: 91 < %
		The controller is not running. Information about the current state of the controller is not available.
SuM daemon	Displays the SUMD (Smart Unit Micrometer Daemon) service status. This service is responsible for interacting with the micrometer and performing calculations according to the scheme. The card consists of two panels that display: • connection status via websocket (first panel), • passing JSON-RPC commands (second panel). The indicator also contains a button for calling the auxiliary control panel for the SUMD service. The auxiliary panel contains different sets of buttons according to the current status of the service.	
		The service is running.
		The service is stopped or is not available.
Sensor	Displays the operating status of the optical sensor. The indicator has a button for calling the auxiliary control panel. The auxiliary panel contains different sets of buttons according to the current status of the sensor.	
		Connected. Frame capture started. The sensor is used in the measurement scheme. In this mode, the frame capture rate is additionally displayed. The auxiliary panel contains the "Stop" button to stop capturing frames. 
		Connected. Frame capture stopped. The sensor is used in the measurement scheme. The auxiliary panel contains the following buttons: • "Capture frame" - to capture one frame and then stop, • "Start video" - to start continuous frame capture. 
		Not connected. The sensor is used in the measurement scheme. The auxiliary panel contains the "Open" button to connect to the sensor. Once connected, the sensor status will change to "Stopped".

Group	Description	
		
		The connection to the sensor was lost after a successful connection. Reconnection attempts are being made. The sensor is used in the measurement scheme.
		Available for connection. The sensor is not used in the measurement scheme.
Ethernet		Status of available Ethernet interfaces. A separate panel is displayed for each available interface. Depending on the status of the interface, the panels can be of the following colors: • Green - The interface is configured and running. For the active interface, the IP address, connection speed (1Gbps, 100Mbps) and transmission type (duplex or half duplex) are additionally displayed. • Yellow - This status occurs if the interface is not physically connected to other network devices. The "no-carrier" or "dormant" message is displayed. • Red - The network interface is deactivated. The "off" message is displayed.
Redis		Displays the status of the Redis database and the Webdis service. Depending on the service status, the panel colors indicate the following: • Green - the service is operating normally. In this state, additional information is displayed, including CPU load, RAM usage, read/write throughput, number of connected clients, and number of tables in the database. • Red - Redis and/or Webdis are deactivated, and the label "off" is displayed. In this mode, a button is available to restart Redis.

Area 4 provides access to detailed settings and contains the following tabs:

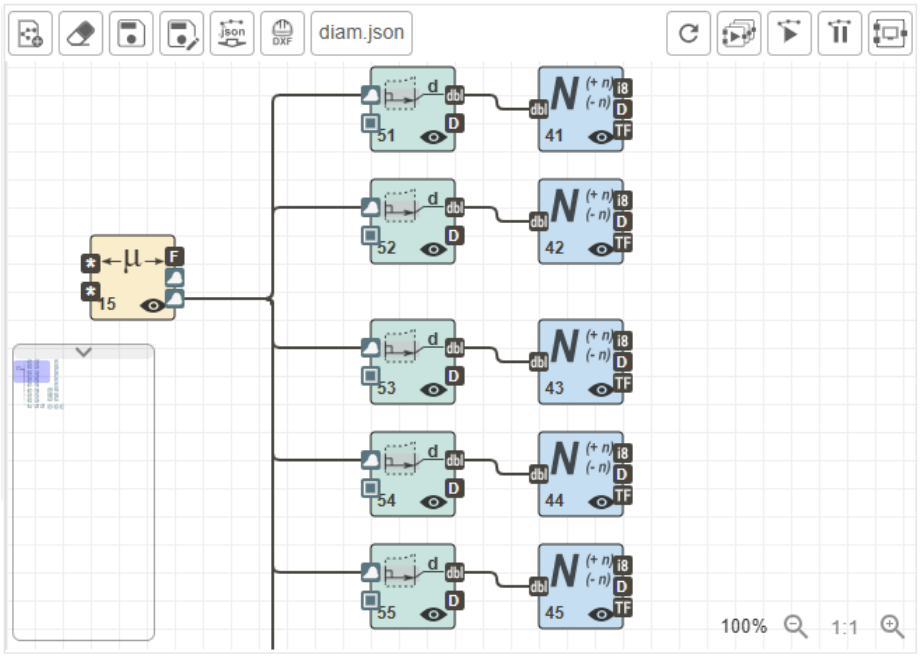
Tab	Icon	Description
Home	 home	Default tab. The auxiliary panel with settings is hidden.
Sensors	 rf656.2D	Settings for sensors and calibration tables, including settings for frame capture, gating, and Ethernet.
Smart	 Smart	Access to the functions of mathematical processing of profiles, smart blocks for measuring various geometric and statistical quantities, measurement schemes.
Files	Files	File browser: dumps, logs and calibration tables.
History	 History	Management of measurement results, including data viewing and visualization, deletion of selected records, as well as report generation and export.
System	 System	Micrometer system settings, including general information about the micrometer, system management, controller network settings, and viewing the device operation log (log file).

Area 5 is intended for the user to form an algorithm for measuring various geometric and statistical quantities of the controlled product. The controls for this area are described in par. 11.1.

Area 6 displays the results of the micrometer operation. The controls for this area are described in par. 11.1.

11.1. Measurement scheme

The following tool is provided to create, delete, upload and edit measurement schemes:



To create, save and upload measurement schemes, use the buttons located at the top of the measurement scheme area:

	
	Creating a new measurement scheme. When you create a new scheme, it is necessary to define its name. In accordance with the entered name, a file is created in the non-volatile memory of the controller.
	Clearing the current measurement scheme. All blocks will be removed.
	Saving all changes to non-volatile memory. Until you click this button, all changes made to the scheme are stored in volatile memory and will be lost when the scheme is reloaded.
	Saving the current scheme to non-volatile memory under a new name.
	Opening the dialog box for managing saved measurement schemes.
	Downloading the current measurement scheme from the controller in order to save it to the computer. The saved measurement scheme can later be used on other 2D micrometers.

	Redrawing the current measurement scheme.
	Activating/deactivating the specified block. When deactivated, this block is no longer used in calculations, i.e. internal processing loops are stopped and information on all ports of the deactivated block is ignored.
	Activating all blocks of the scheme.
	Deactivating all blocks of the scheme.
	Displaying/hiding the "Display" blocks in the measurement scheme. The "Display" blocks are designed to transfer information from the scheme blocks to the measurement results display area.

11.1.1. Managing saved schemes

Click  to open the window for managing saved measurement schemes. This window contains the following controls:

#	Description
1	This button is used to upload a json file with a measurement scheme from a computer to the micrometer controller.
2	This field displays the name of the current (uploaded) scheme.
3	This icon indicates the default scheme, i.e. the scheme loaded at controller startup.
4	This button deletes the scheme file from the non-volatile memory of the controller.
5	This button uploads the selected scheme as the current one. After the selected scheme is uploaded, its name will be shown as the current scheme (see #2).
6	This button sets the selected scheme as the default scheme, i.e. the scheme uploaded at controller startup.

11.2. Measurement results display

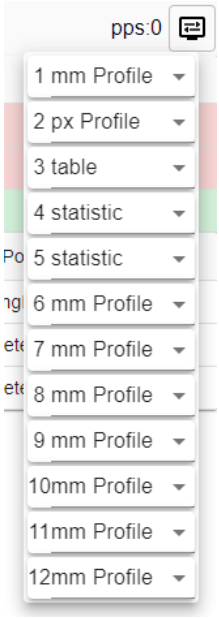
This area is designed to display the results of the smart block operation, as well as to provide visual control and customization of smart block search areas.

The area can contain up to 12 virtual displays. Each display can be configured to present information in any of the following ways:

- **2D mm** – two-dimensional rectangular coordinate system. Coordinate values are given in millimeters.
- **2D px** – two-dimensional rectangular coordinate system. Coordinate values are given in pixels.
- **Table** – tabular representation of scalar quantities.
- **Statistics** – representation of the dependence of scalar quantities on the measurement cycle.

- **Polar** – polar coordinate system. Intended for data visualization from the "Runout and Concentricity" block.
- **Monitoring** – history of the latest measurements recorded in the database.

Displays are configured in a special area that appears when you click on the **Display settings** button.



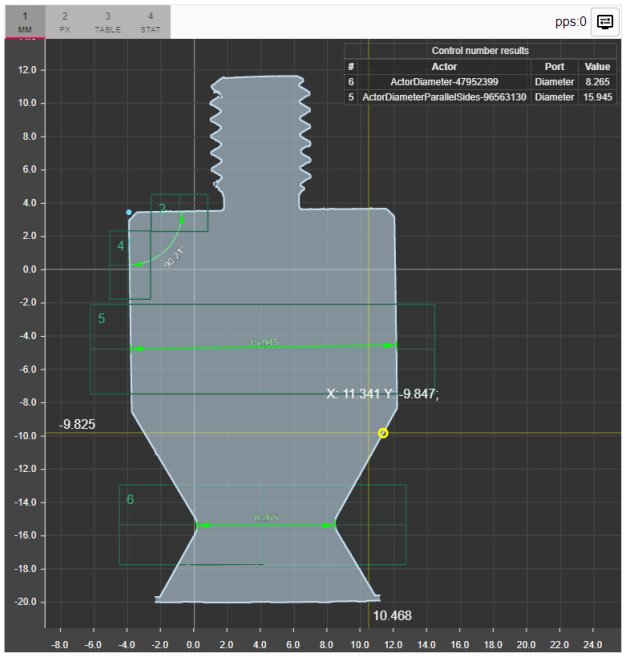
Each display shows certain types of data (see Data types):

Display type	Data type	
2D mm	TF i8 i16 i32 i64 dbl ● ■ ⊶ ／ ⁂ ◠ ◠ D	bool, int8_t, int16_t, int32_t, int64_t, double, Point2d<double>, Rect, SegmentLine, StraightLine, PolyLine, Contour, Profile, Description
2D px	TF i8 i16 i32 i64 dbl ◠ F D	bool, int8_t, int16_t, int32_t, int64_t, double, Profile, Frame. Description – only for "Tolerance" blocks.
Table Statistics	TF i8 i16 i32 i64 dbl D	bool, int8_t, int16_t, int32_t, int64_t, double. Description – only for "Tolerance" blocks.
Polar	⁂ D	Slices – only for "Runout & Concentricity" blocks. Description – only for "Runout & Concentricity" blocks.
Monitoring	TF i8 i16 i32 i64 dbl D	bool, int8_t, int16_t, int32_t, int64_t, double. Description – only for "Tolerance" blocks.

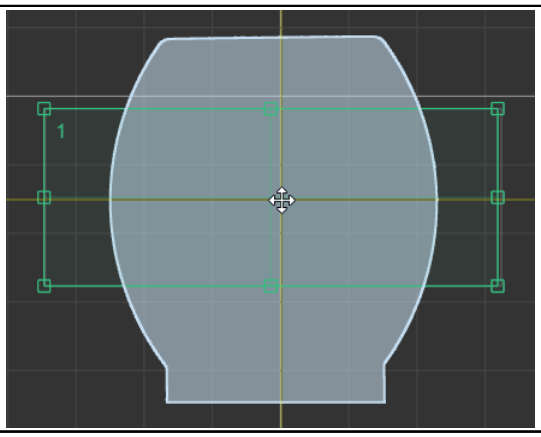
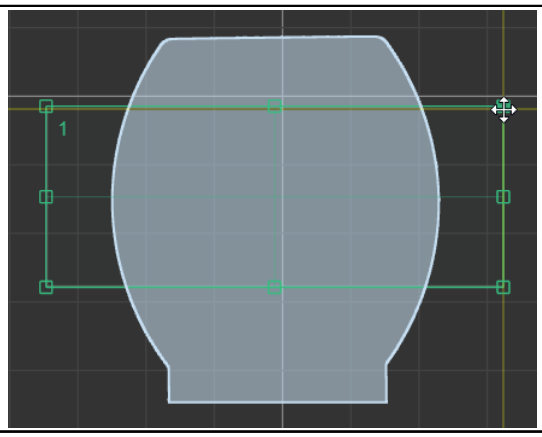
11.2.1. "2D mm" display

It is designed to display profiles, contours, polylines, straight lines, segments, points, scalars, search areas, and measurement results.

The display is also used for visual control and setting up search areas for smart blocks.



Some blocks have search areas within which block functions are executed. The user can move and resize the search area. Moving is done with the right mouse button (click on the search area and move the mouse). Resizing is carried out using special rectangles located around the perimeter of the search area:

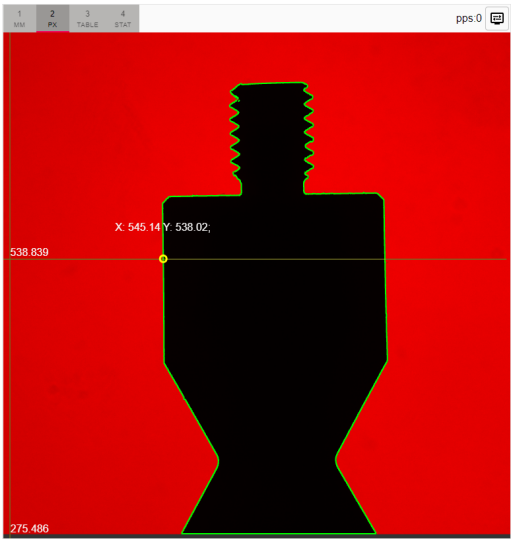
	
Moving the search area	Resizing the search area

In the lower-right corner of the display, when hovering the mouse cursor, a hidden button appears. It is used to switch the search area control mode. Three modes are available:

		
Search areas are displayed, editing enabled. In this mode, the search areas are visible on the screen and can be modified and adjusted.	**Search areas are displayed, editing disabled.** In this mode, the search areas are visible but cannot be edited.	**Search areas are hidden, editing disabled.** In this mode, the search areas are not displayed and cannot be edited.

11.2.2. "2D px" display

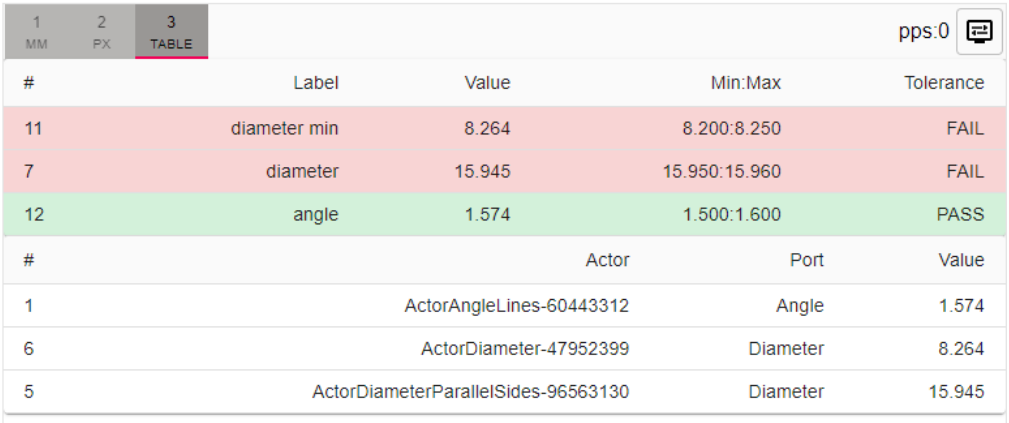
It is designed to display frames, profiles, and search areas.



A feature of this display is the ability to display the original shadow image of the object. However, be aware that when information is displayed on this display, the bandwidth requirements for the network connection between the controller and the computer increase significantly. This is due to the fact that the image from the controller is transmitted without compression. Required network bandwidth: 110 Mbps (at 10 fps optical sensor frame rate).

11.2.3. "Table" display

It is designed to display scalar values, as well as the results of checking for scalar values (measurement results) to fall within the specified range.



#	Label	Value	Min:Max	Tolerance
11	diameter min	8.264	8.200:8.250	FAIL
7	diameter	15.945	15.950:15.960	FAIL
12	angle	1.574	1.500:1.600	PASS

#	Actor	Port	Value
1	ActorAngleLines-60443312	Angle	1.574
6	ActorDiameter-47952399	Diameter	8.264
5	ActorDiameterParallelSides-96563130	Diameter	15.945

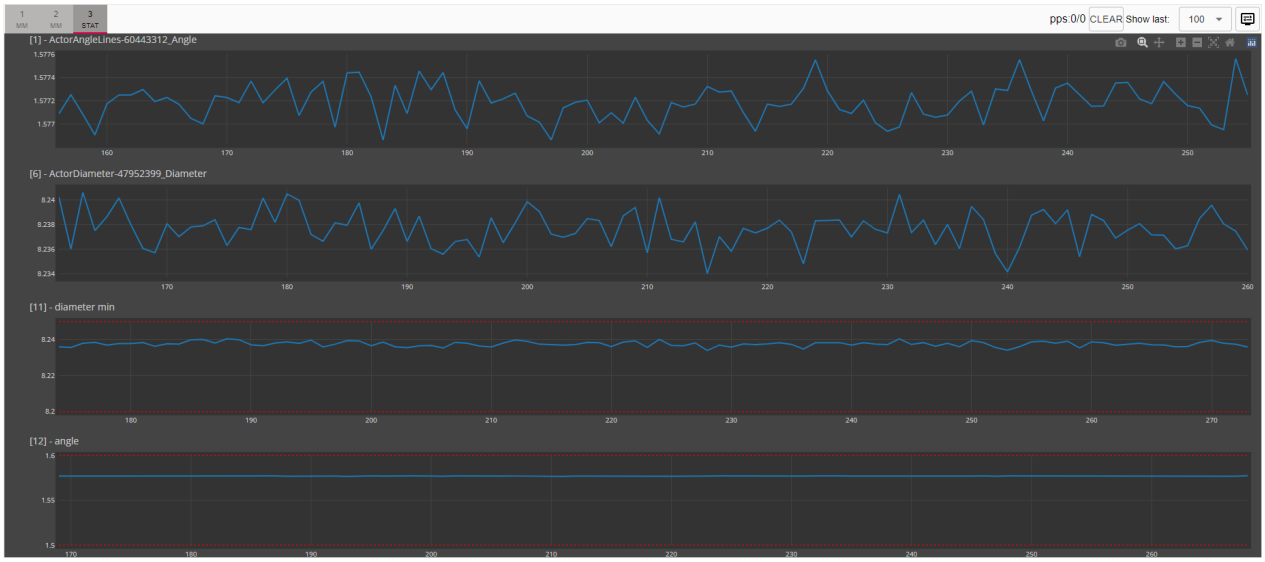
All information on the display is grouped into two tables.

The first table contains information about whether the scalar values fall within the specified range or not. This information can come from the **ResultDescription** output of the **tolerance** block. Depending on the result of the check, the rows of the table are highlighted in colors: red - if scalar values do not fall within the specified range, green - if scalar values fall within the specified range.

The second table contains scalar values.

11.2.4. "Statistics" display

It is designed to visualize the dependence of the measured quantities on the measurement number. The display allows the user to visually assess the stability of the measurement results.



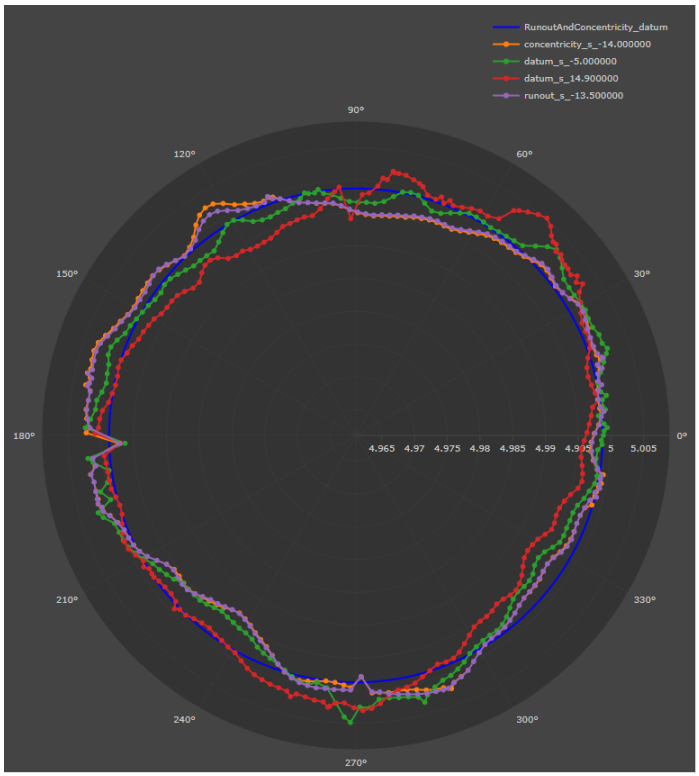
The display has the following auxiliary controls:

- **Clear** button - to clear the contents of the window. Clicking this button deletes all measurement information accumulated on the display.
- **Show last ...** list - to set the number of measurements to be displayed on the graphs. Only the numbers of the latest measurements (N) are displayed for each displayed scalar value. The following N values are available in the list: 10, 50, 100, 250, 500.

When displaying data from the **ResultDescription** output of the **tolerance** block, the upper and lower limits of the range are displayed on the graph.

11.2.5. "Polar" display

This display is designed for data visualization in a polar coordinate system. It shows measurement results obtained only from the **Runout & Concentricity** block. It is used to visually represent runout and concentricity parameters in polar coordinates.



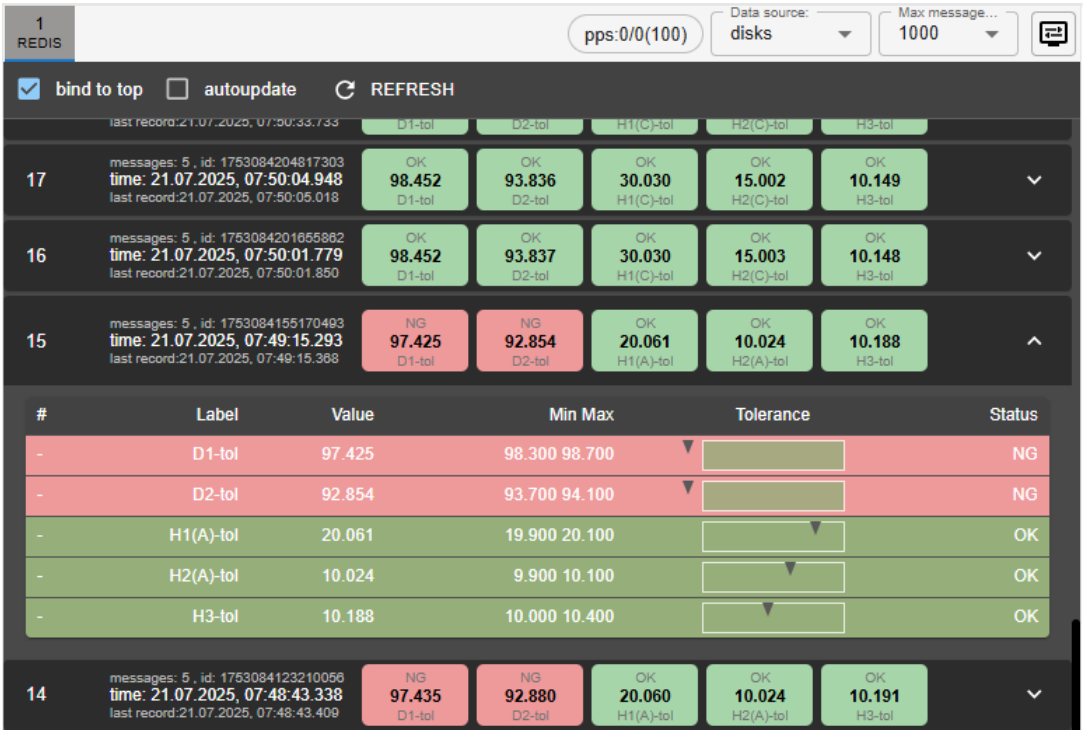
11.2.6. "Monitoring" display

This display is designed to visualize the measurement history stored in the built-in Redis database using the "database logger" block. Each row on the display corresponds to a unique profile identifier (**id**) and aggregates all measurement results associated with that profile. Thus, all records with the same **id** are shown in a single row.

For each row, the following summary information is displayed:

- messages – number of database records on which the row is based (for example, messages: 5);
- id – profile identifier;
- time – time of the first record;
- last record – time of the last record.

Data obtained from the ResultDescription output of the "tolerance" block are highlighted with a specific color, similar to how they are shown in the "Table" displays. Each row can be expanded to view detailed information in a tabular format, consistent with the layout used in the "Table" displays.



The display includes the following auxiliary control elements:

- **Data source** – dropdown list for selecting the key by which data will be displayed. The key is defined in the "label" field of the "database logger" block. When the block is created, the default name corresponds to the value ${id}, the unique identifier of the smart block.
- **Max messages** – dropdown list for selecting the number of most recent records displayed on the screen. Available values: 1000, 5000, 10000, 20000.
- **Bind to top** – checkbox that enables automatic scrolling so that the latest record is always visible on the screen.
- **Autoupdate** – checkbox that enables automatic list updates when new data appear in the database.
- **Refresh** – button for manually refreshing the list.

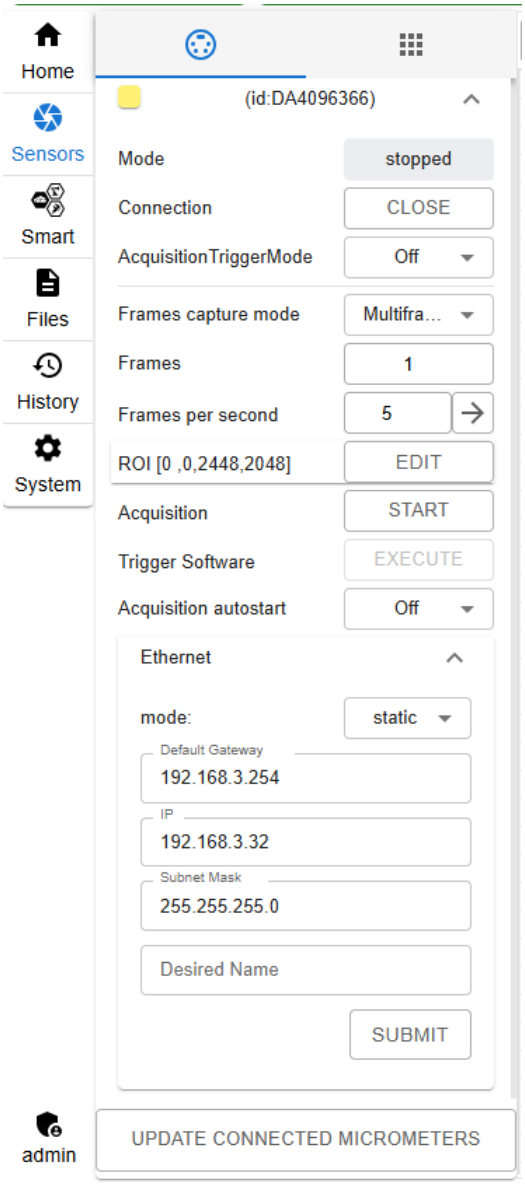
11.3. "Sensors" tab

The **Sensors** tab is designed to configure optical sensors and calibration tables for them, as well as configure frame capture, gating, and Ethernet settings for optical sensors.

This tab contains two sections:

- Sensors Settings.
- Calibration Tables.

11.3.1. "Sensors Settings" section



Parameters:

Parameter	Default value	Description
Mode	-	The current operating mode of the optical sensor. The following modes are possible: • started - Connected. Frames capture started. • stopped - Connected. Frame capture stopped. The sensor is used in the measurement scheme. • closed - Disconnected. The sensor is used in the measurement scheme.

Parameter	Default value	Description
		• reconnection - The connection has been lost. Attempts are being made to reconnect. The sensor is used in the measurement scheme. • accessible - Available for connection. The sensor is not used in the measurement scheme.
Connection	-	Button for establishing/closing the connection to the sensor.
AcquistionTriggerMode	OFF	Selecting a channel for connecting an external trigger for capturing frames. The following options are available: "Off", "Line0", "Line1", "Line2", "Line3".
Frame Rate	30	The number of frames per second captured by the sensor.
Frames	1	Number of frames captured in "Multiframe" capture mode.
Frames per second	Multiframe	Frame capture mode. The following options are available: • Multiframe - capture the number of frames specified by the "Frames" parameter, and then stop capturing. • Video - continuous frame capture.
ROI	-	Displays the image selection area (region of interest, ROI), defined by the parameters [offset X, offset Y, width, height], where: • offset X – horizontal offset of the area; • offset Y – vertical offset of the area; • width – width of the area; • height – height of the area. Clicking the **Edit** button allows you to edit these parameters.
Acquistion	-	Button to start/stop capturing frames.
Acquistion autostart	Off	Automatically start capturing frames when the controller starts and connects to the sensor.

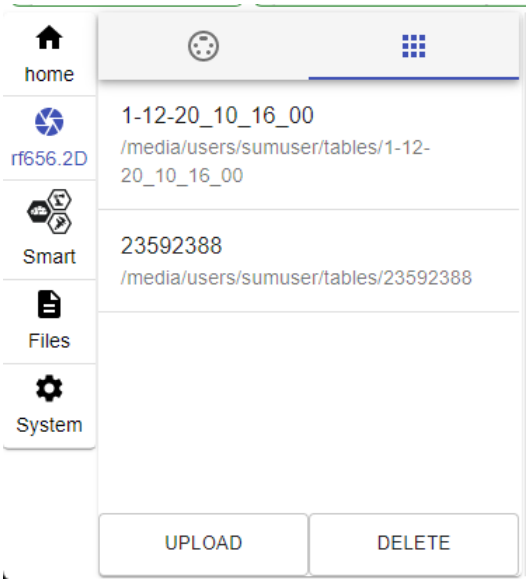
The **Sensor Settings** section also provides the interface for configuring the network settings of the sensors. All network settings are grouped in the **Ethernet** area:

Parameter	Default value	Description
mode	static	static - a static address is assigned manually, dhcp - IPv4 or IPv6 address is dynamically assigned if there is the DHCP server on the network.
IP Address	192.168.3.30	IP address of the sensor. Only for mode:static.
Subnet mask	255.255.255.0	Subnet mask. Only for mode:static.
Default gateway	192.168.3.1	Gateway network address. Only for mode:static. This parameter is optional.
Desired Name	-	Network name for the sensor.

It is necessary to click the **SUBMIT** button in order for the changes to take effect.

11.3.2. "Calibration Tables" section

This section is designed to upload new calibration tables for optical sensors, as well as delete existing tables.



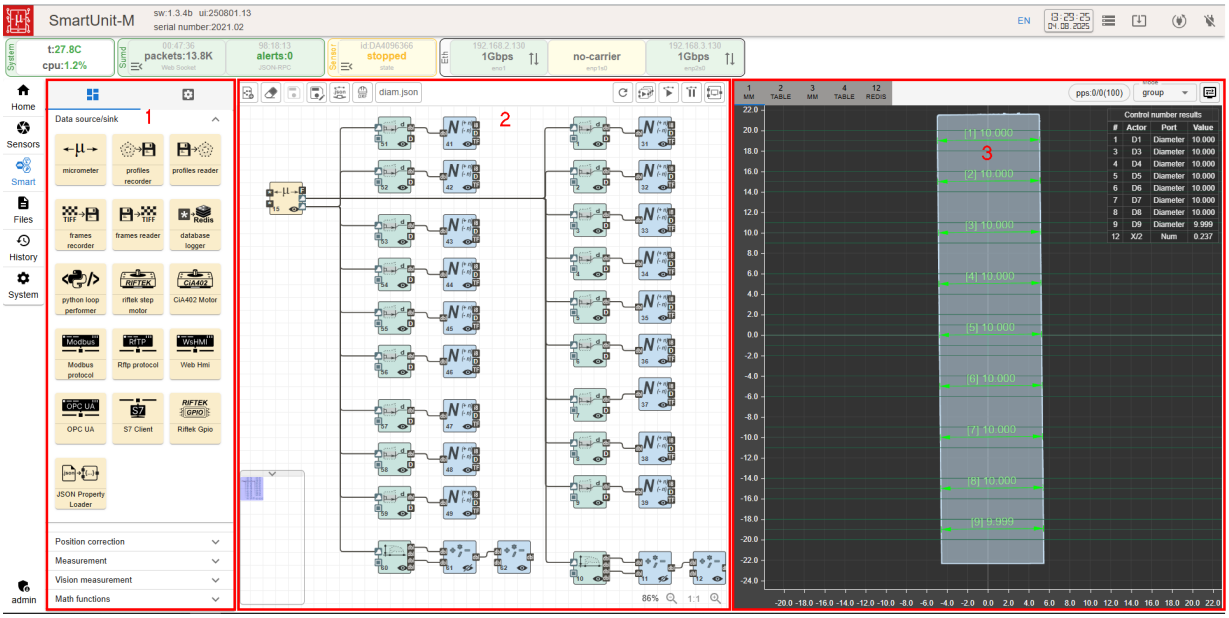
11.4. "Smart" tab

The **Smart** tab is designed to implement the smart functions of the micrometer.
Smart functions include:

- Creating an algorithm for measuring various geometrical and statistical parameters of the controlled product.
- Performing measurements in real time according to a given algorithm.
- Processing of measurement results and automatic decision-making about their being within acceptable limits (control of tolerances).
- Transmitting measurement results via industrial (Modbus TCP, Modbus RTU) and simplified (UDP, UART) protocols.
- Forming control actions (for example, pass/fail) at the physical outputs of the micrometer.

To ensure the simplicity and ease of use of smart functions, the concept of the "computation graph" is applied. The user creates the measurement scheme to solve a specific problem. The measurement scheme is an ordered sequence of operations performed by the micrometer. This sequence is presented in the form of smart blocks and links between them.

The main window of the web interface (the **Smart** tab):



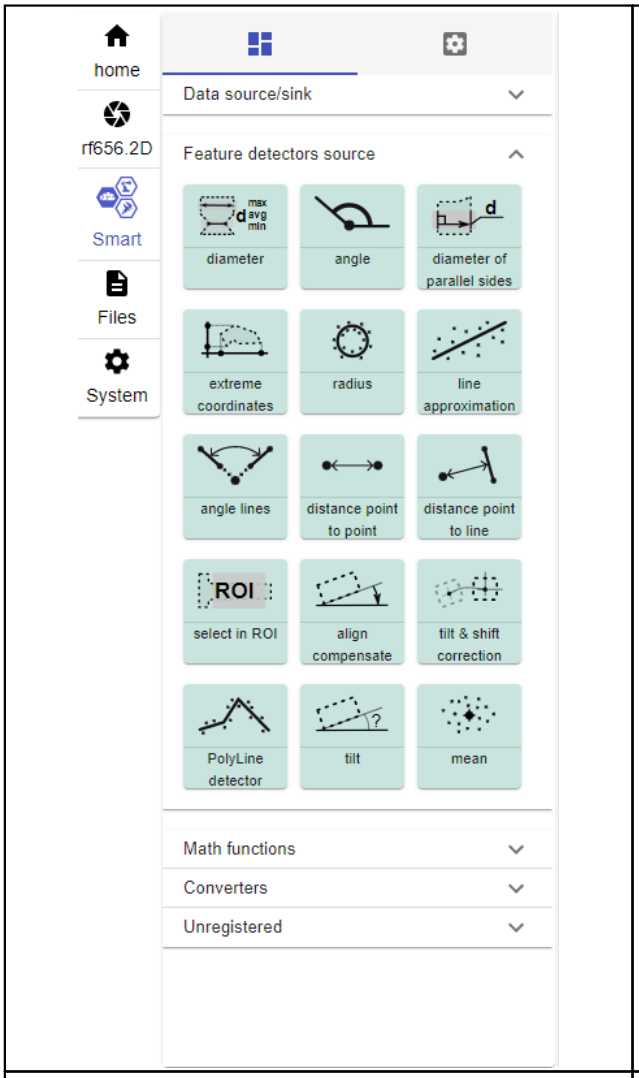
Designations:
1 - smart blocks and parameters area;
2 - measurement scheme area;
3 - measurement results display area.

11.4.1. Smart blocks and parameters

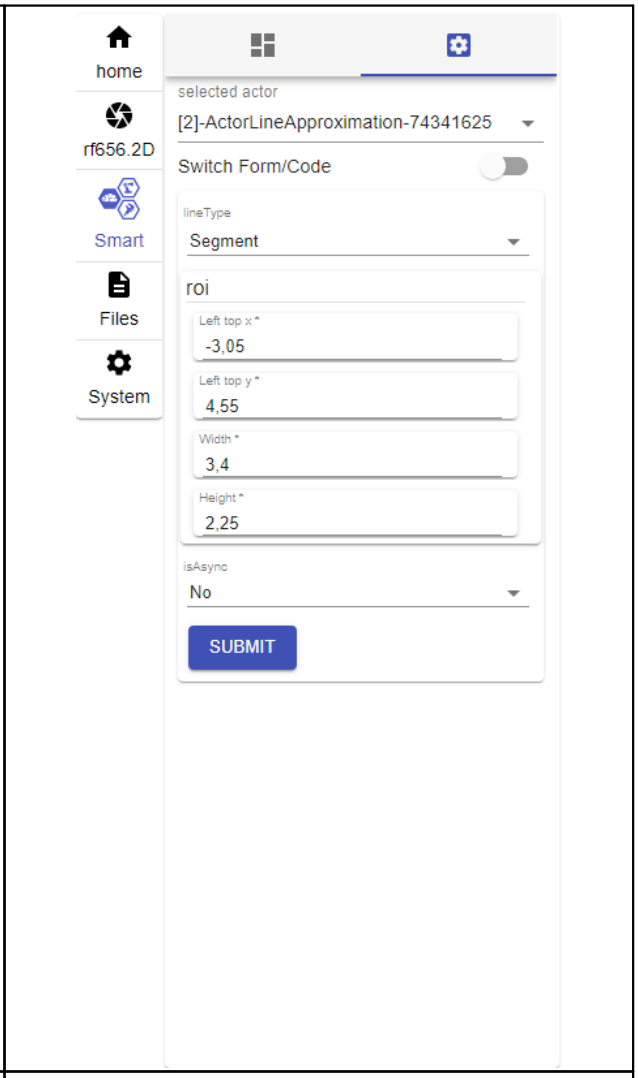
The area is intended for displaying a set of smart blocks and setting the parameters of the blocks placed on the measurement scheme.

The area contains two tabs:

- **Smart blocks** - a set of smart blocks grouped by functionality.
- **Block settings** - parameters of the block selected on the graph.



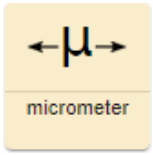
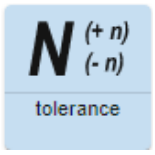
"Smart Blocks" tab



"Block Settings" tab

11.4.1.1. "Smart Blocks" tab

This tab contains smart blocks available for use. All smart blocks are logically divided into groups according to their functional purpose. The pictogram on the smart block schematically displays the function it performs. Examples:

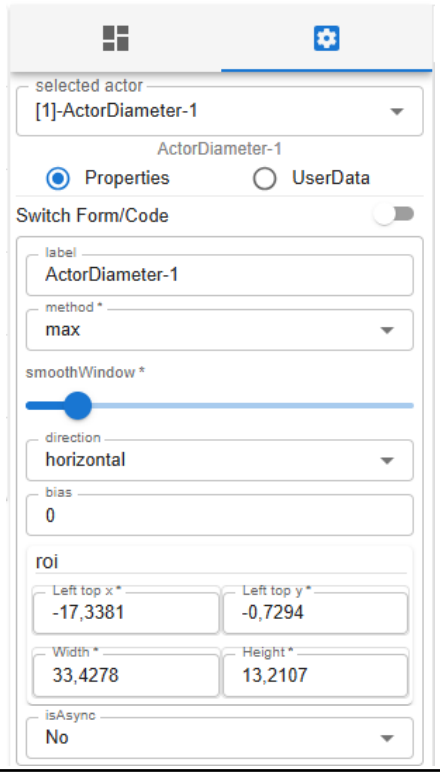
		
Smart block for working with the sensor	Smart block for finding an angle on a contour	Smart block for threshold processing

11.4.1.2. "Block Settings" tab

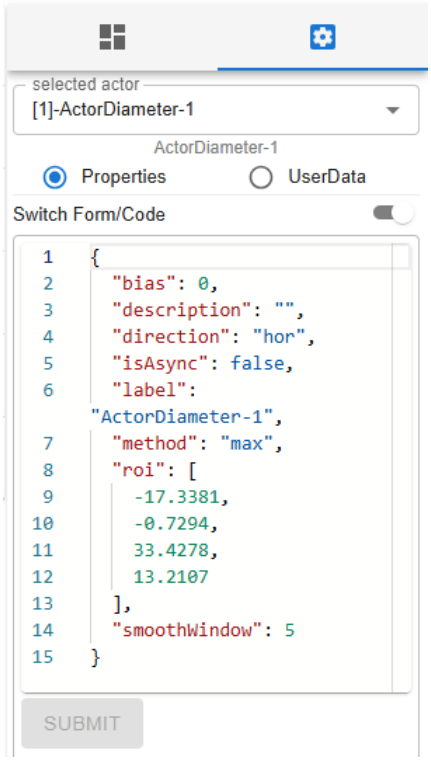
This tab provides access to the settings of the selected block.

You can select a block on the graph or from the "selected actor" drop-down list.

It is possible to edit the block parameters in **Form** mode and in **JSON Scheme** mode. Switching between modes is done with the **Switch Form/Code** toggle switch. Examples:



Smart block settings in Form mode



Smart block settings in JSON Scheme mode

11.4.2. Smart block sets

11.4.2.1. Data types

Each smart block works with a certain type (or several types) of data, which are measurement results, logic signals, etc. Byte order is LITTLE-ENDIAN (unless otherwise noted). Description of data types is given in the table:

Name	Icon	Type	Description
Common types	These types are used to transmit data to external (in relation to the sensor) devices and receive data from them. They are used in conjunction with special conversion blocks.		
Bool	TF	bool	Boolean value that has two mutually exclusive states "TRUE" and "FALSE". It corresponds to the uint8 type: 0 - “FALSE”, other - “TRUE”.
NumberInt8	i8	int8_t	Signed integer value (size - 1 byte).
NumberInt16	i16	int16_t	Signed integer value (size - 2 bytes).
NumberInt32	i32	int32_t	Signed integer value (size - 4 bytes).
NumberInt64	i64	int64_t	Signed integer value (size - 8 bytes).
NumberDouble	dbl	double	Double-precision floating-point value (size - 8 bytes).
Internal types	These types are used to transfer information within a graph. As a rule, they are composite (contain several fields) and should not be used for input and output of data from/to external systems (EthernetIP, UDP, etc.).		

Point2dDouble	●	Point2d<double>	Point. In the current revision, it has the following structure:
			{
			double x;
			double y;
			}

Name	Icon	Type	Description

Rect		Rect	Rectangle. In the current revision, it has the following structure:
			{
			Point2d<double> pointTl; // top-left
			double width;
			double height;
			}

SegmentLine		SegmentLine	Line segment. In the current revision, it has the following structure:
			{
			Point2d<double> point1;
			Point2d<double> point2;
			}

StraightLine		StraightLine	Straight line. In the current revision, it has the following structure:
			{
			double a;
			double b;
			double c;
			}

PolyLine		PolyLine	Polyline. It is specified by a set of points. In the current revision, it has the following structure:
			{
			uint64_t id;
			vector<Point2d<double>> polyline;
			}

Contour		Contour	Contour. It is specified by a set of points. Unlike a polyline, it is always closed. In the current revision, it has the following structure:
			{
			uint64_t id;
			deque<Point2d<double>> points;
			ContourType contourType;
			}
			where *ContourType* is the type of contour defined as outer or inner:
			enum ContourType
			{
			Outer = 0, // Outer contour
			Inner = 1 // Inner contour
			};

Profile		Profile	Profile. It is a set of contours and hierarchical links between them. Each outer contour of the profile may include a plurality of inner contours. It is the primary result of processing the shadow image by the micrometer. In the current revision, it has the following structure:
			{
			uint64_t id;
			uint64_t timestamp;
			vector<Contour> contours;
			vector<int> hierarchy;
			}
			hierarchy specifies the number of the outer contour within which the inner contour is located.

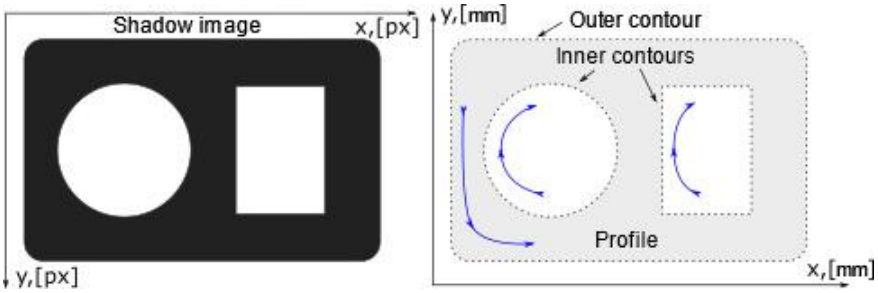
Frame		Frame	Frame. It is used to represent the shadow image obtained by the micrometer. In the current revision, it has the following structure: {

Name	Icon	Type	Description

			uint64_t id;
			uint64_t timestamp;
			uint32_t width;
			uint32_t height;
			PixelFormatType pixelFormat;
			vector<uint8_t> frame;
			}
			where *PixelFormatType* is the pixel format of the frame:
			enum PixelFormatType
			{
			Unknown = 0x00,
			// mono formats
			Mono8 = 0x01, // Monochrome, 8 bits (PFNC:Mono8)
			Mono10 = 0x02, // Monochrome, 10 bits in 16 bits (PFNC:Mono10)
			Mono10p = 0x03, // Monochrome, 10 bits in 16 bits (PFNC:Mono10p)
			Mono12 = 0x04, // Monochrome, 12 bits in 16 bits (PFNC:Mono12)
			Mono12Packed = 0x05, // Monochrome, 2x12 bits in 24 bits (GEV:Mono12Packed)
			Mono12p = 0x06, // Monochrome, 2x12 bits in 24 bits (PFNC:MonoPacked)
			Mono14 = 0x07, // Monochrome, 14 bits in 16 bits (PFNC:Mono14)
			Mono16 = 0x08, // Monochrome, 16 bits (PFNC:Mono16)
			};

Description	D	Description	JSON description of measurement results. Description of smart blocks may vary.

The initial information received from the optical sensor of the micrometer is a shadow image (**Frame**). The result of processing this shadow image by the micrometer is a profile. The profile is a composite data type, it is a collection of contours and hierarchical links between them. Each contour is represented by an ordered sequence of points (**Point2dDouble**). Contours can be outer and inner. Each outer contour of the profile may hierarchically include a plurality of inner contours. The contour points are ordered in such a way that when moving from point to point in forward order, the measured object is to the left of the direction of movement. I.e., for outer contours, the order of points is counterclockwise, and for inner contours, it is clockwise.



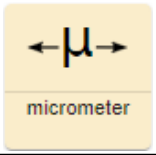
11.4.2.2. Sections

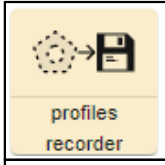
Smart blocks are grouped into the following sections:

1. **Data source/sink** - Smart blocks designed to enter information from sensors and external systems into the graph, as well as to output measurement results.
2. **Position Correction** - Smart blocks designed to transform the profile coordinate system (rotate and transfer the coordinate system).

3. **Measurement** - Smart blocks designed to perform measurements, as well as find primitives on the profile (points, lines, angles, etc.).
4. **Math functions** - Smart blocks that perform mathematical operations on primitives, including filtering and monitoring whether measured values are within tolerances.
5. **Converters** - Smart blocks that perform transformations (data type conversion, composition and decomposition of primitives, etc.).

11.4.2.2.1. "Data source/sink" section

	"micrometer" - this smart block is designed to work with the optical sensor of the micrometer.		
Inputs:	AcqistionStartStop	Bool	Control signal to start/stop capturing frames from the sensor.
Outputs:	OutFrame	Frame	Original shadow image from the micrometer.
	OutProfile	Profile	Profile calculated from the shadow image and converted according to the calibration table (millimeter coordinate system).
	OutProfilePix	Profile	Profile calculated from the shadow image (pixel coordinate system).
Parameters:	micrometr Id	String	ID number of the micrometer.
	table	String	Path to the directory with calibration tables. By default, calibration tables are stored in /media/users/sumuser/tables/*

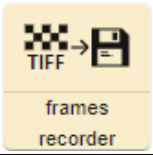
	"profiles recorder" - this smart block is designed to save profiles to files. Each profile is saved in csv format.		
Inputs:	InpProfile	Profile	Profiles to be saved in files.
Parameters:	dir	String, [tmp_dump/${id}]	Directory where profile files will be saved. There are two options: • dump/${id} - directory in the non-volatile memory of the controller. Full path /media/users/sumuser/dump • tmp_dump/${id} - directory in the volatile memory of the controller. Full path /tmp/sumdaemon/dump NOTE. ${id} is the unique ID of the smart block.
	namePrefix	String,["prof_"]	File name prefix for each profile.
	postfixType	String enum, ["daytime"]	Algorithm for generating a unique part of the name for each file. The following options are available: • counter • daytime (date and time according to "postfixDateFormat" parameter) • dataid (profile ID is used) • timestamp (profile timestamp is used)
	postfixDateFormat	String, ["%d-%m-%y_%H-%M-%S"]	Date/time format for the file name. The field is active only when "postfixType": "daytime". It is set in accordance with: std::put_time - cppreference.com

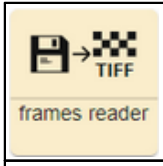
%a	Day name abbreviation	Thu
%A	Full day name	Thursday
%b	Month name abbreviation	Aug

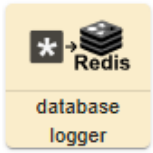
			%B	Full month name	August
			%C	First two digits of the year	20
			%d	Day of the month, with zeros (01-31)	23
			%e	Day of the month, with a space (1-31)	23
			%F	Date format YYYY-MM-DD is equivalent to %Y-%m-%d	2001-08-23
			%g	Year, last two digits (00-99)	01
			%G	Year	2001
			%h	Month name abbreviation (same as %b)	Aug
			%H	Hours in 24-hour clock (00-23)	14
			%I	Hours in 12-hour clock (01-12)	02
			%j	Day of the year (001-366)	235
			%m	Month in numeric format (01-12)	08
			%M	Minutes (00-59)	55
			%p	AM or PM	PM
			%S	Seconds (00-60)	02
			%u	Day of the week in numeric format according to ISO 8601, counting Monday as the first day of the week (1-7)	4
			%U	Week number, counting Sunday as the first day of the week (00-53)	33
			%V	Week number according to ISO 8601 (00-53)	34
			%w	Day of the week in numeric format, counting Sunday as 0 (0-6)	4
			%W	Week number, counting Monday as the first day of the week (00-53)	34
			%y	Year, last two digits (00-99)	01
			%Y	Year	2001
	isAsync	bool,[true]	Flag. It indicates whether processing is asynchronous.		
	queueSize	uint16_t, 0...65535, [255]	Queue of asynchronously executing tasks. If set to 0, the queue size is not limited. The field is only active when `"isAsync": true.`		

 profiles reader	"profiles reader" - this smart block is designed to read profiles saved in files. Files matching the “filesMask” mask are sequentially selected from the given directory. Each next file is read at the specified time interval "minLoopTimeMks". After reading all files from the directory, the reading cycle is repeated if the “isCyclic” flag is set, otherwise the block stops reading the profiles.		
Outputs:	OutProfile	Profile	Profile.
Parameters:	dir	String, [tmp_dump/${id}]	Directory from which files matching the "filesMask" mask will be sequentially selected. The form automatically offers all directories from: • dump/* - directories in the non-volatile memory of the controller. Full path /media/users/sumuser/dump • tmp_dump/* - directories in the volatile memory of the controller. Full path /tmp/sumdaemon/dump
	filesMask	String,["*.csv"]	Mask for filtering the names of the files being read.
	isCyclic	bool,[true]	File replay flag. If the flag is set to "true", then after reading all the files from the directory, playback is

			repeated in a cyclic mode.
	minLoopTimeMks	uint32_t, 0...2^{32}, [10000]	The minimum delay before reading the next file.

	"frames recorder" - this smart block is designed to save frames to files. Each frame is saved in tiff format.		
Inputs:	OutProfile	Profile	Profile.
Parameters:	dir	String, [tmp_dump/${id}]	Directory where the frame files will be saved. The form automatically offers two options: • dump/${id} - directory in the non-volatile memory of the controller. Full path /media/users/sumuser/dump • tmp_dump/${id} - directory in the volatile memory of the controller. Full path /tmp/sumdaemon/dump NOTE. ${id} is the unique ID of the smart block.
	namePrefix	String,["prof_"]	File name prefix for each frame.
	postfixType	String enum, ["daytime"]	Algorithm for generating a unique part of the name for each file. The options are similar to the “Profiles recorder” block.
	postfixDateFormat	String, ["%d-%m-%y_%H-%M-%S"]	Date/time format for the file name. The field is only active when `"postfixType": "daytime"`. The format is the same as for the “Profiles recorder” block.
	isAsync	bool,[true]	Flag. It indicates whether processing is asynchronous.
	queueSize	uint16_t, 0...65535, [255]	Queue of asynchronously executing tasks. If set to 0, the queue size is not limited. The field is only active when `"isAsync": true.`

	"frames reader" - this smart block is designed to read frames saved in tiff files. Files with the tiff” extension are sequentially selected from the specified directory. Each subsequent file is read at a specified time interval "minLoopTimeMks". After reading all files from the directory, the reading cycle is repeated if the “isCyclic” flag is set, otherwise the block stops reading profiles.		
Outputs:	OutFrame	Frame	Frame.
Parameters:	dir	String, [tmp_dump/${id}]	The directory from which files corresponding to the “filesMask” mask will be sequentially selected. The form automatically offers all directory options from: • dump/* - directory in the non-volatile memory of the controller. Path: /media/users/sumuser/dump • tmp_dump/* - directory in the volatile memory of the controller. Path: /tmp/sumdaemon/dump
	filesMask	String,["*.tiff"]	Mask for filtering file names.
	isCyclic	bool,[true]	Flag for repeating the reading cycle. If this flag is set to true, then after reading all the files from the directory, playback is repeated in cyclic mode.
	minLoopTimeMks	uint32_t, 0...2^{32}, [10000]	Minimum delay before reading the next file.

	"database logger" - block for saving profiles and calculation results to the built-in Redis database. Data are stored in the "timeSeries" structure under the specified key (label) with timestamps attached.		
Inputs:	Input	All	Input data for saving to the database.

Parameters:	label	String,[${id}]	Block name. This name is also used as the key for storing all records in the database. When the block is created, the name is assigned automatically and, by default, corresponds to the value ${id}, where ${id} is the unique identifier of the smart block.
	maxLen	int32,[5000]	Maximum number of records per key. When this limit is exceeded, new records are added cyclically: as new entries are inserted, the oldest ones are automatically deleted to keep the total number within the defined limit.
	isAsync	bool,[true]	Flag. Indicates whether the processing is asynchronous.

	"python loop performer" - this smart block is designed to execute custom scripts written in Python. Custom scripts provide ample opportunities for customizing and expanding the functionality of the system. They can be used to implement custom measurement processing algorithms, proprietary information exchange protocols, control the measurement process, and others tasks. Custom scripts allow you to adapt the system to specific needs and implement additional control logic. Rules for writing scripts are described in par. "Custom scripts".		
Inputs:	Created dynamically. The description of each input is represented by an element of the "ports" array.		
Outputs:	Created dynamically. The description of each output is represented by an element of the "ports" array.		
Parameters:	path	string	The path to the script to be executed. Typically, scripts are located here: /media/users/sumuser/scripts. Next to the editable field there is a button for calling the integrated script editor.
	minLoopTimeMks	uint64_t, 0..., [30000]	Minimum time for calling the Process function handler.
	ports:[{},{},..]		
	id	string	Unique ID for the port.
	type	string	Port type: • Input - for the input port that receives data from the scheme. • Output - for the output port that sends data to the scheme.
	messageTypes	[string,...]	Lists the allowed message types from/to the port. The following values are possible: Void=0 ,Bool=1, NumberInt8=10, NumberInt16=11, NumberInt32=12, NumberInt64=13, NumberDouble=14, Point2dDouble=50, Rect=100, SegmentLine=101, StraightLine=102, PolyLine=103, Contour=104, Profile=105, Frame=1000, Json=5000, Description = 5001.
	traceLevel	string enum, ["Info"]	The type of information messages sent for debugging to the integrated Code Editor. The following options are available: • Trace - trace messages, as well as all of the following message types. • Info - information messages, as well as all of the following types. • Warning - messages containing warnings about incorrect execution of scripts, as well as all the messages listed below. • Error - error messages when executing the script. • NoTrace - messages are not transmitted.

	"Riftek step motor" - this smart block is designed to control the proprietary Riftek stepper motor driver.		
Inputs:	CycleStartStop	Bool	Control signal to start/stop capturing frames from the sensor.
	MoveTo	int8_t, int16_t, int32_t, int64_t, double	Sending a command to the driver to turn the engine for N steps. The N sign determines the direction, cw or ccw. All values are converted to int32. The double value is mathematically rounded to an integer value.
	MoveToZero	All	Sending a command to the driver to move to the starting position. The command will be executed for any type of incoming message, regardless of its content.
	RequestPosition	All	Requesting the current position from the driver. The command will be executed for any type of incoming message, regardless of its content.
	RequestState	All	Requesting the current status of the driver. The command will be executed for any type of incoming message, regardless of its content.
	SetSpeed	int8_t, int16_t, int32_t, int64_t, double	Sending a command to the driver to change the speed of the stepper motor. All values are converted to int32. The double value is mathematically rounded to an integer value.
	Stop	All	Sending a command to stop the motor. The command will be executed for any type of incoming message, regardless of its content.
Outputs:	State	int8_t	Status of the driver and stepper motor. Possible values: • 97 (0x61) - reached Hall1. • 160 (0xA0) - reached Hall2. • 192 (0xC0) - moves towards Hall2 from Hall1. • 193 (0xC1) - moves towards Hall1 from Hall2. • 224 (0xE0) - stopped before reaching the Hall2 sensor. • 225 (0xE1) - stopped before reaching the Hall1 sensor. • 255 (0xFF) - the state is undefined.
	Position	int32_t	Current position from the motor driver.
Parameters:	portName	string, ["/dev/ttyS3"]	The file name that the operating system associates with the serial device, for example /dev/ttyS3.
	motorAddress	uint8_t, 0...256, [10]	Logical address of the stepper motor. Determined when flashing the stepper motor driver.
	baundRate	uint32,[115200]	Port baud rate: 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200.
	isAsync	bool,[true]	Flag. It indicates whether the process is asynchronous.
	queueSize	uint16_t, 0...65535, [255]	A queue of asynchronously executing tasks. When set to 0, the queue size is not limited. The field is active only when `"isAsync": true`.

	"CiA 402 Motor" - block for controlling the stepper motor driver in accordance with the CiA 402 protocol and EtherCAT interface. It provides the exchange of control commands and status information through standard objects and states defined by the CiA 402 specification. This block supports the following operating modes:

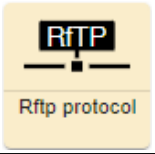
<table>
<tr><td></td><td colspan="3">• Profile Position Mode - a mode in which the drive moves to a specified position following a defined motion profile (considering the set parameters of speed, acceleration, and deceleration). Enables smooth and controlled positioning.
• Velocity Mode - a mode in which the drive maintains a constant preset rotational or linear velocity.
• Profile Velocity Mode - a mode in which the drive changes its speed according to a predefined profile (for example, with gradual acceleration and deceleration), helping to avoid abrupt speed changes.
• Homing - automatic search and setting of the motor’s zero (reference) position.
The block provides reading and writing of parameters, as well as position and speed control of the motor according to the selected operating mode.</td></tr>
<tr><td rowspan="20">Inputs:</td><td>RequestPosition</td><td>All</td><td>Request the current motor position. The command is executed for any type of incoming message, regardless of its content.</td></tr>
<tr><td>RequestState</td><td>All</td><td>Request the current driver state. The command is executed for any type of incoming message, regardless of its content.</td></tr>
<tr><td>Stop</td><td>All</td><td>Send a motor stop command. The command is executed for any type of incoming message, regardless of its content.</td></tr>
<tr><td>ReadStatusWord</td><td>All</td><td>Request to read the StatusWord. The result is sent to the StatusWord output.</td></tr>
<tr><td>ReadWord</td><td>Json</td><td>Request to read an arbitrary word. The input message is a JSON object:
<pre>{
  "index": uint16,
  "subIndex": uint8,
  "size": uint8
}</pre>The result is sent to the RadWord output.</td></tr>
<tr><td>RestartLink</td><td>All</td><td>Request to re-establish the EtherCAT connection with the driver.</td></tr>
<tr><td>WriteControllWord</td><td>int16</td><td>Write a value to ControlWord.</td></tr>
<tr><td>WriteWord</td><td>Json</td><td>Write an arbitrary word. The input message is a JSON object:
<pre>{
  "index": uint16,
  "subIndex": uint8,
  "size": uint8,
  "value": int64
}</pre>The result is sent to the RadWord output.</td></tr>
<tr><td>SwitchOperationMode</td><td>int8_t</td><td>Attempt to switch the controller to the specified operating mode:
1 – Profile Position Mode
2 – Velocity Mode
3 – Profile Velocity Mode
6 – Homing</td></tr>
<tr><td colspan="3">Inputs for "Profile Position Mode":</td></tr>
<tr><td>MovePpmAbs</td><td>int32_t</td><td>Absolute position for movement (absolute target).</td></tr>
<tr><td>MovePpmRel</td><td>int32_t</td><td>Relative position for movement (relative target).</td></tr>
<tr><td colspan="3">Inputs for "Velocity Mode"</td></tr>
<tr><td>MoveVM</td><td>int32_t</td><td>Set the desired movement speed in "Velocity Mode" (VM).</td></tr>
<tr><td colspan="3">Inputs for "Profile Velocity Mode"</td></tr>
<tr><td>MovePVM</td><td>int32_t</td><td>Set the desired movement speed in "Profile Velocity Mode" (PVM).</td></tr>
<tr><td colspan="3">Inputs for "Homing"</td></tr>
<tr><td>Homing</td><td>Bool</td><td>Activate the "Homing" mode.</td></tr>
</table>

Outputs:	Position	int32_t	Current motor position.
	State	int8_t	Status of the driver and stepper motor. Possible values are defined by the CiA 402 specification.
	portOutTargetReached	bool	Target reached flag.
	OperationMode	int8_t	Current controller operating mode. A value of 255 indicates an error.
	StatusWord	int16	StatusWord value. It is updated upon request through the ReadStatusWord or RequestState inputs.
	RadWord	int64_t	Result of executing ReadWord and WriteWord requests.
Parameters:	label	string,[${id}]	Block name. It is assigned automatically when the block is created and, by default, corresponds to the value ${id}, where ${id} is the unique identifier of the smart block.
	portName	string,[enp1s0]	Name of the network interface used to connect to the EtherCAT network (for example, enp1s0).
	motorIndex	int8_t	Device number (address) in the EtherCAT network. Used to identify a specific drive among other devices on the bus.
	addPortReadStatusWord	bool	Flag for adding an input to request reading of StatusWord.
	addPortReadWord	bool	Flag for adding an input to read an arbitrary word by a specified index and subindex.
	addPortRestartLink	bool	Flag for adding an input to re-establish connection with the driver.
	addPortWriteControlWord	bool	Flag for adding an input to write a value to ControlWord.
	addPortWriteWord	bool	Flag for adding an input to write an arbitrary word by a specified index and subindex.
	addPortSwitchOperationMode	bool	Flag for adding an input to switch the drive's operating mode.
	isAsync	bool,[true]	Flag. It indicates whether the processing is asynchronous.
	queueSize	uint16_t, 0...65535, [255]	Queue size for asynchronously executed tasks. When set to 0, the queue size is unlimited. This field is active only when `"isAsync": true`.
	Parameters for "Profile Position Mode"		
	usePropfilePositionMode	bool	Enable the Profile Position Mode (PPM).
	profileVelocity	uint32_t	Movement speed in Profile Position Mode (units depend on driver settings, typically pulses per second).
	endVelocity	uint32_t	Final speed at the end of movement.
	profileAcceleration	uint32_t	Acceleration in Profile Position Mode.
	profileDeceleration	uint32_t	Deceleration in Profile Position Mode.
	quickStopDeceleration	uint32_t	Deceleration during emergency stop (Quick Stop).
	useAbsoluteTarget	bool	Use absolute positioning (target is set as an absolute position).
	useRelativeTarget	bool	Use relative positioning (target is set as an offset from the current position).
	Parameters for "Velocity Mode"		
	useVelocityMode	bool	Enable the Velocity Mode.
	targetVelocity	int16_t	Target motor rotation speed.

	velocityAcceleration	uint32_t	Acceleration when reaching the target speed.
	velocityDeceleration	uint32_t	Deceleration when reducing speed.
	velocityMinAmount	uint32_t	Minimum allowable speed.
	velocityMaxAmount	uint32_t	Maximum allowable speed.
	Parameters for "Profile Velocity Mode"		
	useProfileVelocityMode	bool	Enable the Profile Velocity Mode.
	targetVelocity	int32_t	Target velocity in Profile Velocity Mode.
	profileVelocityAcceleration	uint32_t	Acceleration in Profile Velocity Mode.
	profileVelocityDeceleration	uint32_t	Deceleration in Profile Velocity Mode.
	quickStopDeceleration	uint32_t	Deceleration during emergency stop (Quick Stop).
	Parameters for "Homing Mode"		
	useHomingMode	bool	Enable the Homing Mode.
	homeOffset	int32_t	Offset applied after completing the homing procedure.
	homingAcceleration	uint32_t	Acceleration during the homing procedure.
	homingMethod	int8_t	Homing method (defined by the CiA 402 specification, e.g., search by sensor, by limit switch, etc.).
	homingSpeedSearchForSwitch	uint32_t	Switch search speed during the homing procedure.
	homingSpeedSearchForZero	uint32_t	Zero position search speed during the homing procedure.
	An array of objects defining the parameters for writing to the SDO. Each object describes a single write operation to the SDO.		
	useSdoWriteParams	bool	Flag enabling the use of a parameter array for writing to SDO. When this flag is activated, the sdoWrite array becomes available.
	index	uint16_t	Index of the object in the Object Dictionary of the EtherCAT/CiA 402 device where the value will be written.
	subIndex	uint8_t	Subindex of the object in the Object Dictionary specifying the particular field or array element within the selected index.
	value	uint32_t	Value to be written to the specified SDO object.
	size	uint8_t	Data size for writing (in bytes). Defines how many bytes of the value will be written to the object (for example, 1, 2, or 4 bytes).

Modbus TCP Modbus protocol	"Modbus protocol" - this smart block is designed to transmit and receive data via the Modbus protocol (both TCP and RTU). The block implements the interface of the slave device (server - in Modbus terminology). Each input and output of the block is associated with the address space of Modbus registers, while the inputs of the block are associated with the Input Registers, and the outputs of the block are associated with the Holding Registers. All data received at the inputs of the block from other blocks of the scheme are written to the Input Registers at the specified address ("address") for the corresponding input. On subsequent polling, the data from the registers will be provided to the Modbus client controller. Input Registers are 65536 (addressing 0 to 65535) 16-bit registers. All data written by the Modbus client to the Holding Registers will be transferred to other blocks connected to the corresponding output of the block. Holding Registers are 65536 (addressing from 0 to 65535) 16-bit registers. Each message is allocated in 16-bit registers based on the type of this message. The order of writing different types of messages to registers is given in "Annex 3. Modbus data types".

	The inputs and outputs of the block are created dynamically based on the entries of the "ports" array in the block parameters.			
Inputs:	Created dynamically. The description of each input is represented by an element of the "ports" array.			
Outputs:	Created dynamically. The description of each output is represented by an element of the "ports" array.			
Parameters:	minLoopTimeMks		uint32_t, 0...232, [10000]	Minimum delay before reprocessing connection requests from new clients and processing incoming requests from connected clients.
	channel:{}			
	backend		String enum, ["TCP"]	Modbus protocol type. Possible options: • "TCP" - Modbus TCP protocol for TCP/IP networks, • "RTU" - Modbus RTU protocol for data transmission via serial communication lines RS-485, RS-422, RS-232.
	backend=TCP	ip	string["192.168.2.130"]	Server IP address. Must match the IP address of the network interface being used.
		port	uint16[502]	Server TCP port number.
	backend=RTU	port file	string	File name associated with the serial device, for example /dev/ttyS0.
		baund rate	uint32	Port baud rate: 9600, 19200, 57600, 115200.
	ports:[{},{},..]			
	id		string	Unique ID for the port.
	type		string	Port type: • PortInput - for the input port that receives data from the scheme. • PortOutput - for the output port that sends data to the scheme.
	message type		string	Message type. Possible values: Bool, NumberInt8, NumberInt16, NumberInt32, NumberInt64, NumberDouble, Point2dDouble, Rect, SegmentLine, StraightLine.
	address		uint16_t	Address of data location in registers: Input Registers - for block inputs, and Holding Registers - for block outputs.
	mode		string	This parameter is defined only for the ports of the "PortOutput" type. It sets the following modes for sending data from Holding Registers to the measurement scheme: • SendNever - no data is sent. • SendWhenChanged - data is sent as a message only if the value in the registers has been changed. • SendEverytime - data is sent as a message on each loop defined by minLoopTimeMks. • SendWhenChangedToTrue - data is sent as a message only if the value in the registers has been changed from false to true. Only for message type == Bool. • SendWhenChangedToFalse - data is sent as a message only if the value in the registers has been changed from true to false. Only for message type == Bool.

	"RFTP protocol" - this smart block is designed to transmit data using a proprietary protocol based on UDP or UART.		
Inputs:	InpData	* (all types supported in schemes)	Input data.
Parameters:	“isAsync”	bool,[true]	Flag. It indicates whether the process is asynchronous.
	“queueSize”	uint16_t, 0...65535, [255]	Queue of asynchronously executing tasks. If set to 0, the queue size is not limited. The field is only active when `"isAsync": true`.

	"Web Hmi" - this smart block is design to interact with the panel integrated into the HMI web interface. It provides data transmission from the circuit outputs to the HMI and reception of data from the HMI via a web socket. Configuring the "Web HMI" block and interacting with this block is described in par. "Hmi Adjustment".		
Inputs:	Created dynamically. The description of each input is represented by an element of the "ports" array.		
Outputs:	Created dynamically. The description of each output is represented by an element of the "ports" array.		
Parameters:	minLoopTimeMks	uint32_t, 0...232, [10000]	Minimum processing time for new data from the web HMI.
	ports:[{},{},..]		
	id	string	Unique ID for the port.
	type	string	Port type: • Input - for the input port that receives data from the scheme. • Output - for the output port that sends data to the scheme.
	messageTypes	[string,...]	Lists the allowed message types from/to the port. The following values are possible: Void=0, Bool=1, NumberInt8=10, NumberInt16=11, NumberInt32=12, NumberInt64=13, NumberDouble=14, Point2dDouble=50, Rect=100, SegmentLine=101, StraightLine=102, PolyLine=103, Contour=104, Profile=105, Frame=1000, Json=5000, Description = 5001.

	"OPC UA" - this smart block implements the server-side functionality of the industrial OPC UA protocol for bidirectional data exchange between external client systems (SCADA, MES) and the internal computation blocks of the scheme. Data from the block’s inputs are transmitted to clients via the OPC UA protocol, while values received from clients are sent to the block’s outputs and then to other blocks in the scheme.		
Inputs:	Created dynamically. The description of each input is represented by an element of the "ports" array.		
Outputs:	Created dynamically. The description of each output is represented by an element of the "ports" array.		
Parameters:	label	String,[${id}]	Block name. This name is also used as the unique node identifier (NodeId) in the OPC UA server’s address space. When the block is created, the name is assigned automatically and, by default, corresponds to the value ${id}, where ${id} is the unique identifier of the smart block.
	port	uint16[4840]	TCP network port for client connections.

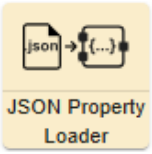
	namespaceIndex	uint16[1]	Namespace index in the OPC UA NodeId. Defines the identifier context: • 0 - reserved for standard OPC Foundation namespaces; • 1 - typically used for user-defined device objects; • >1 - custom namespaces.
	ports:[{},{},..]		
	id	string	Unique ID for the port.
	type	string	Port type: • Input - for the input port that receives data from the scheme. • Output - for the output port that sends data to the scheme.
	messageTypes	[string,...]	Lists the allowed message types from/to the port. The following values are possible: Void=0, Bool=1, NumberInt8=10, NumberInt16=11, NumberInt32=12, NumberInt64=13, NumberDouble=14, Point2dDouble=50.

	"S7 client" - this smart block implements the client-side functionality of the industrial S7 protocol for bidirectional data exchange between external Siemens controllers (S7-300/400/1200/1500) and the internal computation scheme. Data is read from specified memory areas of the Siemens controller (according to the defined *dbNumber* and *address* parameters for each port) and sent to the outputs of this block for transmission to other elements of the scheme. Values received at the inputs are written to the corresponding memory areas of the controller, providing feedback between the computation scheme and the external device.		
Inputs:	Created dynamically. The description of each input is represented by an element of the "ports" array.		
Outputs:	Created dynamically. The description of each output is represented by an element of the "ports" array.		
	isConnected	bool	Information about the presence or absence of a connection with the Siemens controller.
Parameters:	label	string,[${id}]	Block name. It is assigned automatically when the block is created and, by default, corresponds to the value ${id}, where ${id} is the unique identifier of the smart block.
	ipAddress	string	IP address of the Siemens S7 controller to which the connection is established.
	rack	uint8 [0]	Rack number of the Siemens S7 controller where the CPU module is installed. In most cases, for S7 controllers, this value is 0.
	slot	uint8 [1]	Slot number in the rack where the required module (CPU or another module) is installed and with which the connection is established.
	addIsConnectedPort	bool[false]	Adds an output displaying the current connection status with the Siemens controller. When this parameter is enabled, the module generates an additional output showing whether the connection with the controller is active or not.
	ports:[{},{},..]		
	id	string	Unique ID for the port.
	type	string	Port type: • Input - for the input port that receives data from the scheme. • Output - for the output port that sends data to the scheme.
	dbNumber	uint16	Data Block (DB) number in the Siemens S7 controller memory associated with this input or

			output. Defines which data block is accessed for reading or writing information.
	address	uint16	Offset (address) within the selected data block (DB) from which data reading or writing begins. Specified in bytes and determines the exact location of the variable or memory area within the given data block.
	messageTypes	string[Bool]	Valid message types for data transmission through the port. Each type defines the data format and size used when exchanging information with the Siemens S7 controller. Possible values: • Void = 0 — 0 bytes (no data; not used in Siemens S7). • Bool = 1 — 1 byte (logical type; corresponds to S7 BOOL). • NumberInt8 — 1 byte (8-bit integer; corresponds to S7 SINT). • NumberInt16 — 2 bytes (16-bit integer, Big Endian; corresponds to S7 INT). • NumberInt32 — 4 bytes (32-bit integer, Big Endian; corresponds to S7 DINT). • NumberInt64 — 8 bytes (64-bit integer, Big Endian; corresponds to S7 LINT). • NumberDouble — 4 bytes (floating-point number, Big Endian; corresponds to S7 REAL). • Point2dDouble — 8 bytes (coordinates x and y, 4 bytes each, Big Endian; both correspond to S7 REAL). • Rect = 100 — 16 bytes (sequentially: tl.x, tl.y, width, height — 4 bytes each, Big Endian; all correspond to S7 REAL). • SegmentLine — 12 bytes (sequentially: a, b, c — 4 bytes each, Big Endian; all correspond to S7 REAL). • StraightLine — 16 bytes (sequentially: p1.x, p1.y, p2.x, p2.y — 4 bytes each, Big Endian; all correspond to S7 REAL).
	messageMode	string[OnlyData]	Defines the data exchange format with the controller's memory areas for each input and output of the block. Possible values: • OnlyData - only the data are written to or read from the memory area according to the type defined by the *messageTypes* parameter. • Full - the memory area starting from the specified address contains, in sequence: ○ id (uint64) - message identifier ("address" offset), ○ timestamp (uint64) - time stamp ("address+8" offset), ○ data - the data themselves according to the type defined in *messageTypes* ("address+16" offset).

	"Riftek Gpio" - the smart block for communication with the proprietary GPO controller, designed to control the discrete outputs. The controller is connected via one of the serial interfaces: RS-232, RS-422, or RS-485.		
Inputs:	GpioMulti	int16	Writing the required state of all discrete outputs as a single word (each bit of the word corresponds to a separate output).

	Request	All	Request to obtain the current state of all outputs. When activated, it triggers an update of the GpioMultiStates and gpoLvl0-gpoLvl7 output signals.
	gpo0-gpo7	bool	Writing the required state for each individual discrete output (gpo0 - first output, gpo7 - eighth output).
Outputs:	GpioMultiStates	int16	State of all discrete outputs as a single word (each bit of the word corresponds to the state of a separate output).
	gpoLvl0-gpoLvl7	bool	Current state (logic level) of each individual discrete output.
Parameters:	label	string,[${id}]	Block name. It is assigned automatically when the block is created and, by default, corresponds to the value ${id}, where ${id} is the unique identifier of the smart block.
	portName	string, ["/dev/ttyS3"]	Device file name associated with the serial port in the operating system, e.g., /dev/ttyS3.
	controllerAddress	uint8_t, 0...256, [10]	Logical address of the GPO controller. Assigned during controller firmware update.
	baundRate	uint32,[115200]	Data transmission rate over the serial port. Possible values: 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200.
	addRequestPort	bool[false]	Flag to add the "Request" input to the block.
	addGpioMultiWrite	bool[false]	Flag to add the "GpioMulti" input to the block.
	addGpioMultiStates	bool[false]	Flag to add the "GpioMultiStates" output to the block.
	checkStateOnWrite	bool[false]	When activated, after writing a new value to the block inputs, the current output state is automatically requested, and the "GpioMultiStates" and "gpoLvl0-gpoLvl7" signals are updated.
	portsGpio:[{addGpoStatePort: bool, useGpo: bool}, ...]		
	addGpoStatePort	bool[false]	Flag to add the "gpoLvl0-gpoLvl7" output to the block.
	useGpo	bool[false]	Flag to add the "gpoLvl0-gpoLvl7" input to the block.

	"JSON Property Loader" - the smart block for dynamically setting parameter values of other blocks in the scheme based on data defined in the JSON file. The file is regularly monitored for changes, and when modifications are detected, the corresponding parameters of linked blocks are automatically updated according to the current data from the file. JSONPath syntax is used to search and match parameters.		
Parameters:	filePath	string	File name containing the defined parameters. By default, the following directory is used: /media/users/sumuser/scripts.
	minLoopTimeMks	uint32_t, 0...232, [10000]	Minimum period (in microseconds) between data updates from the JSON file.
	jsonProperties:[{jsonPathInFile:string,actorId:string, jsonPathInActorConfig:string },{},..]		

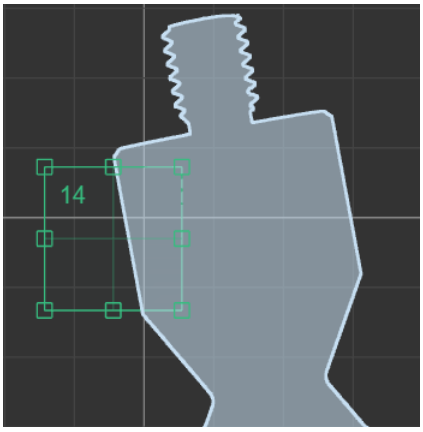
	jsonPathInFile	string	Path to the parameter in the JSON file (using JSONPath). For example, jsonPathInFile:"ActorD1.bias" for JSON:
			`{`
			`"ActorD1": {`
			`"bias": 0.005`
			`}`
			`}` will allow you to access the *bias* parameter inside the ActorD1 object.

	actorId	string	The ID of the block for which the parameter needs to be updated.

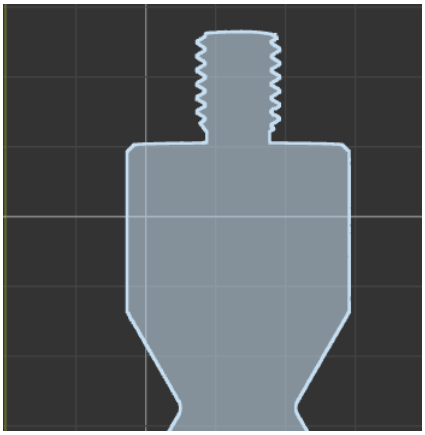
	jsonPathInActorConfig	string	The path to the parameter within the specified block's *properties* object (using JSONPath). For example, to update the value of the block's bias parameter, specify jsonPathInActorConfig: "bias".

11.4.2.2.2. "Position correction" section

"align compensate" - this smart block is designed to align the profile along a given edge detected within the ROI. The ROI must contain a set of points that include only one edge of the profile to be aligned. If there is more than one edge in the detection area, the first edge detected is selected for aligning. It is also possible to align to the center line. In this case, the ROI should include two symmetrical edges of the figure. The angle of rotation of the resulting profile is defined as the minimum angle from the edge or center line to the horizontal line and to the vertical line.

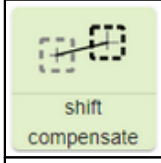


Profile before tilt compensation



Vertically aligned profile

Inputs:	InpProfile	Profile	Input profile.
	InpRoi	Rect	Search area. When the input is disabled, the default value from the "roi" parameter is used for calculation.
Outputs:	OutProfile	Profile,Contour	A profile aligned to a horizontal or vertical line.
Parameters:	alignLine	String enum, ["Side"]	Type of line defining the profile alignment angle: • Side - edge. • Center - center line.
	roi	Rect	Measurement area - ROI. Set by the following parameters: left top x, left top y, width, height.



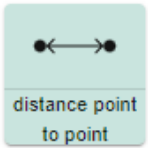
"shift compensate" is a parallel shift of the coordinate system relative to a given position.

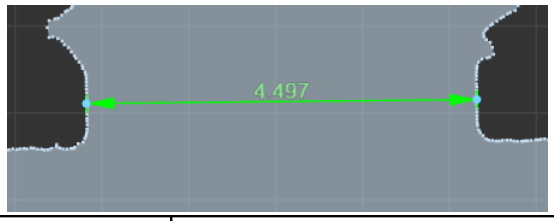
Inputs:	InpProfile	Profile,Contour	Input profile or contour.
	InpRoi	Rect	Search area. When the input is disabled, the default value from the "roi" parameter is used for calculation.
Outputs:	OutProfile	Profile,Contour	Profile with a transformed coordinate system.
Parameters:	horizontalAlign	String enum, ["No"]	The side on which the origin of the new coordinate system will be determined along the X coordinate. The following options are available: • No - the coordinate system is not transferred. • Left - the origin of the coordinate system along the X coordinate is determined by the profile point that falls within the ROI with the minimum X value. The X value of this point becomes the origin of the coordinate system. • Center - the origin of the coordinate system along the X coordinate is determined by the point equidistant from the profile point with the minimum X value and

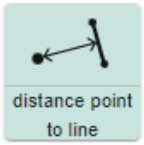
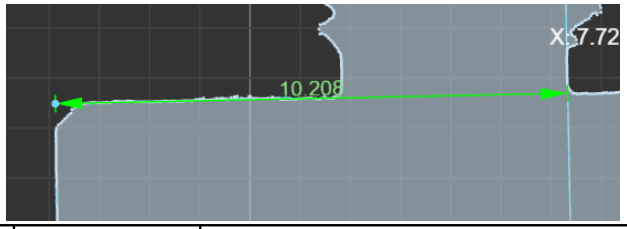
			from the profile point with the maximum X value. The X value of this point becomes the origin of the coordinate system. • Right - the origin of the coordinate system along the X coordinate is determined by the profile point that falls within the ROI with the maximum X value. The X value of this point becomes the origin of the coordinate system.
	verticalAlign	String enum, ["No"]	The side on which the origin of the new coordinate system will be determined along the Y coordinate. The following options are available: • No - the coordinate system is not transformed. • Top - the origin of the coordinate system along the Y coordinate is determined by the profile point that falls within the ROI with the maximum Y value. The Y value of this point becomes the origin of the coordinate system. • Middle - the origin of the coordinate system along the Y coordinate is determined by the point equidistant from the profile point with the minimum Y value and from the profile point with the maximum Y value. The Y value of this point becomes the origin of the coordinate system. • Bottom - the origin of the coordinate system along the Y coordinate is determined by the profile point that falls within the ROI with the minimum Y value. The Y value of this point becomes the origin of the coordinate system.
	roi	Rect	Measurement area - ROI. Set by the following parameters: left top x, left top y, width, height.

	"tilt and shift correction" - rotation around a given point and parallel shift of the profile coordinate system.		
Inputs:	Angle	double	The angle by which the profile needs to be rotated.
	InpProfile	Profile,Contour	Input profile or contour.
	RotationCenterPoint	Point2dDouble	The point that defines the center of the coordinate system being rotated.
	Shift	Point2dDouble	The point that defines the new center of the coordinate system (after parallel shift).
Outputs:	OutProfile	Profile,Contour	Profile with a transformed coordinate system.
Parameters:	angle	double	The angle by which the profile needs to be rotated.
	rotationCentrePoint	Point2dDouble	The point that defines the center of the coordinate system being rotated.
	shift	Point2dDouble	The point that defines the new center of the coordinate system (after parallel shift).
	invertAngle	bool	Indicates whether the input angle value needs to be multiplied by -1.
	invertShift	bool	Indicates whether the coordinate value should be taken with a negative sign.

11.4.2.2.3. "Measurement" section

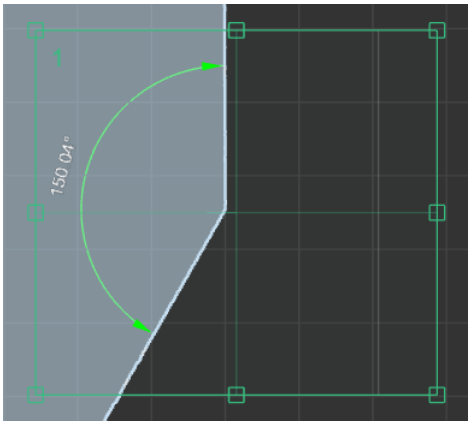
	“distance point to point” - this smart block is designed to calculate the distance between two points arriving at the block inputs. It is possible to calculate both the Euclidean distance and the distance at a given coordinate.

			
Inputs:	Point1	Point2dDouble	First input point.
	Point2	Point2dDouble	Second input point.
Outputs:	Distance	double	The resulting Euclidean distance between points.
	ResultDescription	Description	The result of the check with descriptive semantics. The result is represented as a json object with the following fields: • "type" - DistancePointToPoint. • "D" - the resulting Euclidean distance between points. • "Point1" - the first point of the distance. Point object {"x", "y"}; • "Point2" - the second point of the distance. Point object {"x", "y"}. • "Valid" - indicates whether the result is correct, true or false. False indicates an error in the calculation. The output is intended to be displayed in the results area.
Parameters:	measureType	string enum, ["Distance"]	The type of calculated distance between points. The following options are available: • Distance - Euclidean distance. • Horizontal - distance along the X coordinate. • Vertical - distance along the Y coordinate.
	syncMode	string enum, ["SameId"]	Synchronization of calculations is carried out based on the arrival of input points. The following options are available: • NoSync - calculations are made upon the arrival of each point. • SameId - calculations are made only after the arrival of both points with the same Id.
	bias	double	Offset added to the measured result. Used to adjust (calibrate) the measurement result, taking into account systematic errors or individual characteristics.

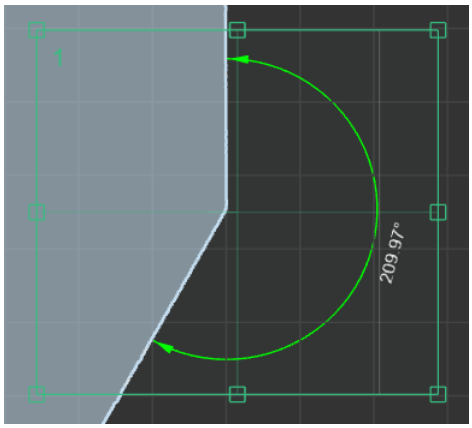
	“distance point to line” - this smart block is designed to calculate the distance between a point and a segment (line) entering the block inputs. It is determined as the length of the perpendicular drawn from a point to a line. 		
Inputs:	Line	StraightLine, SegmentLine	Input line/segment.
	Point	Point2dDouble	Input point.
Outputs:	Distance	double	The resulting distance between the point and the line.
	ResultDescription	Description	The result of the check with descriptive semantics. The result is represented as a json object with the following fields: • "type" - DistancePointToPoint.

			• "D" - the resulting Euclidean distance between points. • "Point1" - the first point of the distance. Point object {"x", "y"}. • "Point2" - the second point of the distance. Point object {"x", "y"}; • "Valid" - indicates whether the result is correct, true or false. False indicates an error in the calculation. The output is intended to be displayed in the results area.
Parameters:	syncMode	string enum, ["SameId"]	Synchronization of calculations is carried out based on the arrival of the input point and line. The following options are available: • NoSync - calculations are made upon the arrival of both a point and a line. • SameId - calculations are made only after the arrival of both a point and a line with the same Id.
	bias	double	Offset added to the measured result. Used to adjust (calibrate) the measurement result, taking into account systematic errors or individual characteristics.

“angle” - this smart block is designed to calculate the angle between two adjacent profile edges. The search and calculation of the angle is done within the measurement area defined by the ROI. If there are more than two edges in the measurement area, the angle is calculated either between the first two detected segments on the profile or between the two longest segments, depending on the specified parameters. When searching for the required line segments, the profile is first divided into a polyline.The resulting angle can be external or internal based on the given "angleType" parameter:



internal

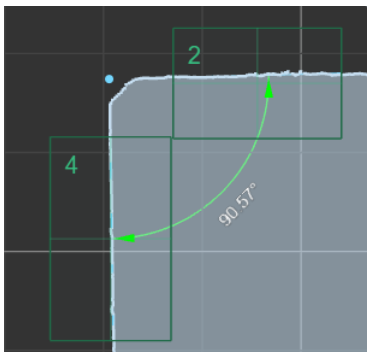


external

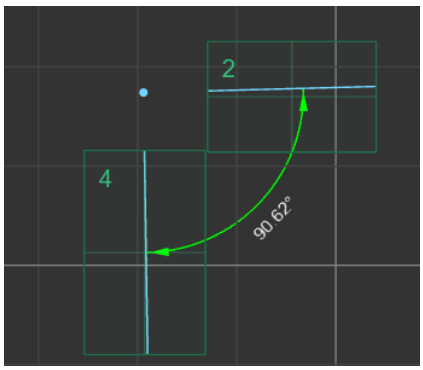
Inputs:	InpProfile	Profile, Contour	Input profile or contour.
	InpRoi	Rect	Measurement area. When the input is disabled, the calculation uses the default value from the "roi" parameter.
Outputs:	Angle	double	The resulting value of the angle between adjacent edges. In case of calculation error, there is no output value.
	ResultDescription	Description	The result of the check with descriptive semantics. The result is represented as a json object with the following fields: • "type" - the value is always "Angle". • "Angle" - the resulting value of the angle. • "angleType" - angle type "Internal" or "External" according to the "angleType" parameter. • "Segment1" - the first segment approximating the first detected edge. The object contains {"x1", "y1", "x2", "y2"}.

			• "Segment2" - the second segment approximating the second detected edge. The object contains {"x1", "y1", "x2", "y2"}. • "Valid" - indicates whether the result is correct, true or false. False indicates an error in the calculation. The output is intended to be displayed in the results area.
Parameters:	"angleType"	string	The type of angle to be detected. Possible values: • "Internal" - the internal angle of the object. When moving from the first to the second edge, the angle is on the left. • "External" - the external angle of the object. When moving from the first to the second edge, the angle is on the right.
	angleUnit	string enum, ["Degrees"]	The unit of measurement of the angle in which the result will be displayed: Degrees - in degrees, Radians - in radians.
	lineSelector	string enum, ["FirstTwo"]	Тип обнаруживаемого угла. Возможные значения: • "FirstTwo" - the internal angle of the object. When moving from the first to the second edge, the angle is on the left. • "Biggest" - the external angle of the object. When moving from the first to the second edge, the angle is on the right.
	maxHalfWidthMm	double,[0.3]	Threshold value (mm) that determines the maximum offset of a point from the polyline line at which a new polyline segment begins.
	bias	double	Offset added to the measured result. Used to adjust (calibrate) the measurement result, taking into account systematic errors or individual characteristics.
	roi	Rect	Measurement area - ROI. Defined by the following parameters: left top x, left top y, width, height.

“angle lines” - this smart block is designed to calculate the center of intersection of lines/segments and the angle between them.
If both input lines are represented by SegmentLine segments, the algorithm determines the line type "Internal" or "External" according to the "angleSegmentType" parameter.
Examples of calculations in the case of two input segments:



Result with profile

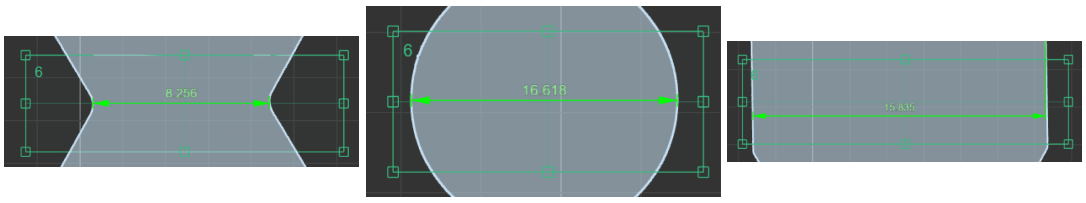


Result with segments

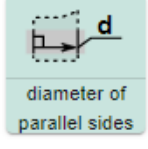
Inputs:	Line1	StraightLine, SegmentLine	First line/segment.
	Line2	StraightLine, SegmentLine	Second line/segment.
Outputs:	Angle	double	The resulting value of the angle between adjacent faces. In case of a calculation error, there is no output value.

	Intersection	Point2dDouble	Intersection point of segments/lines. In case of a calculation error, there is no output value.
	ResultDescription	Description	The result of the check with descriptive semantics. The result is represented as a json object with the following fields: • "type" - the value is always "Angle". • "Angle" - the resulting value of the angle. • "Valid" - indicates whether the result is correct, true or false. False indicates an error in the calculation. Parameters that must be specified, provided that both input lines are represented by SegmentLine segments: • "angleType" - angle type "Internal" or "External" according to the "angleSegmentType" parameter. • "Segment1" - the first segment received from the Line1 input. The object contains {"x1", "y1", "x2", "y2"}. • "Segment2" - the second segment received from the Line2 input. The object contains {"x1", "y1", "x2", "y2"}. Parameters that must be specified if at least one of the input lines is represented by a StraightLine: • "angleType" - angle type: Default, Exp, Sup, SupExp. It corresponds to the "angleStraightType" parameter. • "Straight1"- the first straight line received from the Line1 input. The object contains {"a", "b", "c"}. • "Straight2"- the second straight line received from the Line2 input. The object contains {"a", "b", "c"}. The output is intended to be displayed in the results area.
Parameters:	angleSegmentType	string	Angle type, provided that both input lines are represented by SegmentLine segments. Possible values: • "Internal" - the internal angle of the object. When moving from the first to the second segment, the angle is on the left. • "External" - the external angle of the object. When moving from the first to the second segment, the angle is on the right.
	angleStraightType	string	Angle type, provided that at least one of the input lines is represented by a StraightLine. Possible values: • Default • Exp • Sup • SupExp
	angleUnit	string enum, ["Degrees"]	The unit of measurement of the angle in which the result will be displayed: Degrees - in degrees, Radians - in radians.
	bias	double	Offset added to the measured result. Used to adjust (calibrate) the measurement result, taking into account systematic errors or individual characteristics.
	syncMode	string enum, ["SameId"]	Synchronization of calculations is carried out based on the arrival of input lines. The following options are available: • NoSync - calculations are made upon the arrival of each line. • SameId - calculations are made only after the arrival of both lines with the same Id.

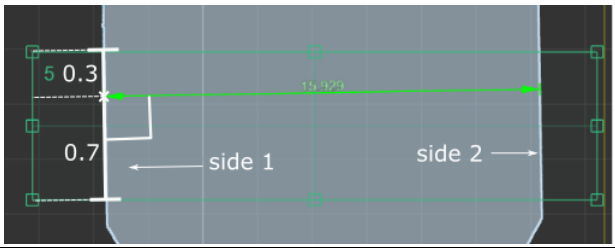
<table>
<tr><td> 

diameter</td><td colspan="3">“diameter” - this smart block is designed to calculate the object diameter within the measurement area specified by the ROI. The slope of the input profile must first be eliminated. The ROI must contain two sets of points that correspond to two edges of the object. The calculation is performed in the horizontal or vertical direction, depending on the "direction" parameter. There are three methods for determining the distance, which are specified by the "method" parameter:
 

min max avg</td></tr>
<tr><td rowspan="2">Inputs:</td><td>InpProfile</td><td>Profile</td><td>Input profile.</td></tr>
<tr><td>InpRoi</td><td>Rect</td><td>Measurement area. When the input is disabled, the calculation uses the default value from the "roi" parameter.</td></tr>
<tr><td rowspan="2">Outputs:</td><td>Diameter</td><td>double</td><td>The resulting value of the diameter. In case of calculation error, there is no output value.</td></tr>
<tr><td>ResultDescription</td><td>Description</td><td>The result of the check with descriptive semantics. The result is represented as a json object with the following fields:
• "type" - "Width" or "Height" value depending on the "direction" parameter.
• "D" - the resulting value of the diameter of the object.
• "Point1" - the first point of the diameter, point object {"x", "y"}.
• "Point2" - the second point of the diameter, point object {"x", "y"}.
• "Valid" - indicates whether the result is correct, true or false. False indicates an error in the calculation.
The output is intended to be displayed in the results area.</td></tr>
<tr><td rowspan="4">Parameters:</td><td>method</td><td>string</td><td>Methods for determining the distance. Possible values:
• min - the minimum distance between all pairs of points along the corresponding coordinate.
• max - the maximum distance between all pairs of points along the corresponding coordinate.
• avg - the average distance between all pairs of points along the corresponding coordinate.</td></tr>
<tr><td>direction</td><td>string</td><td>Direction:
• horizontal (hor) - distance along the X coordinate.
• vertical (ver) - distance along the Y coordinate.</td></tr>
<tr><td>bias</td><td>double</td><td>Offset added to the measured result. Used to adjust (calibrate) the measurement result, taking into account systematic errors or individual characteristics.</td></tr>
<tr><td>roi</td><td>Rect</td><td>Measurement area - ROI. Defined by the following parameters: left top x, left top y, width, height.</td></tr>
</table>

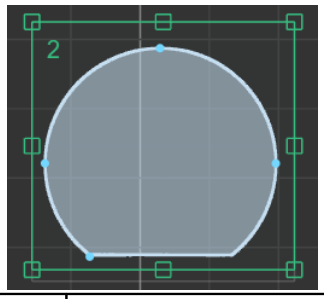
<table>
<tr><td> 

diameter of parallel sides</td><td>“diameter of parallel sides” - this smart block is designed to calculate the diameter of an object within the measurement area specified by the ROI. The ROI must contain two sets of points that correspond to two edges of the object.
Each set of points is approximated by line segments. Next, a perpendicular is drawn to the line specified by the "fromSide" parameter at the point specified by the "pointRatio" parameter, and its intersection with the second line is determined. The length of this perpendicular is the calculated diameter.</td></tr>
</table>

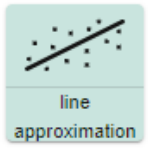
	An example of calculating the diameter for the parameter "fromSide" = 1 (perpendicular to side 1) and "pointRatio" = 0.3: 		
Inputs:	InpProfile	Profile	Input profile.
	InpRoi	Rect	Measurement area. When the input is disabled, the calculation uses the default value from the "roi" parameter.
Outputs:	Diameter	double	The resulting value of the diameter. In case of calculation error, there is no output value.
	ResultDescription	Description	The result of the check with descriptive semantics. The result is represented as a json object with the following fields: • "type" - "Width" or "Height" value depending on the "direction" parameter. • "D" - the resulting value of the diameter of the object. • "Point1" - the first point of the diameter, point object {"x", "y"}. • "Point2" - the second point of the diameter, point object {"x", "y"}. • "Valid" - indicates whether the result is correct, true or false. False indicates an error in the calculation. The output is intended to be displayed in the results area.
Parameters:	fromSide	int [1]	The number of the edge to which the perpendicular is drawn. Possible values are 1 or 2.
	pointRatio	double [0.5]	The proportion that determines the ratio of the sizes of the segments when determining the point on the edge, at which the perpendicular to the face is drawn.
	bias	double	Offset added to the measured result. Used to adjust (calibrate) the measurement result, taking into account systematic errors or individual characteristics.
	roi	Rect	Measurement area - ROI. Defined by the following parameters: left top x, left top y, width, height.

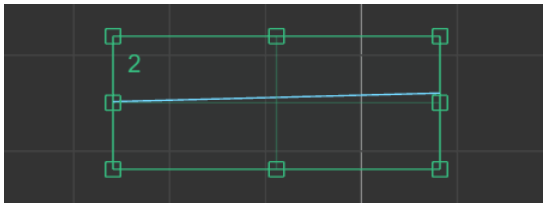
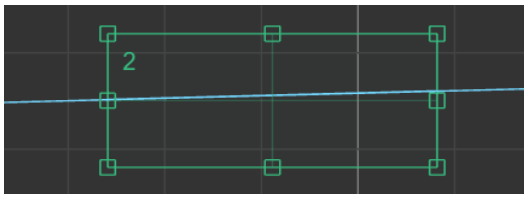
	"extreme coordinates" - this smart block is designed to find the extreme coordinates of an object within the measurement area specified by the ROI.		
Inputs:	InpProfile	Profile	Input profile.
	InpRoi	Rect	Measurement area. When the input is disabled, the calculation uses the default value from the "roi" parameter.
Outputs:	MaxX	double	Maximum X value for profile points.
	MaxY	double	Maximum Y value for profile points.
	MinX	double	Minimum X value for profile points.
	MinY	double	Minimum Y value for profile points.
Parameters:	smoothWindow	uint8_t,[5]	The width of the smoothing (averaging) window for the coordinates of profile points. This parameter determines the size of the window used by the

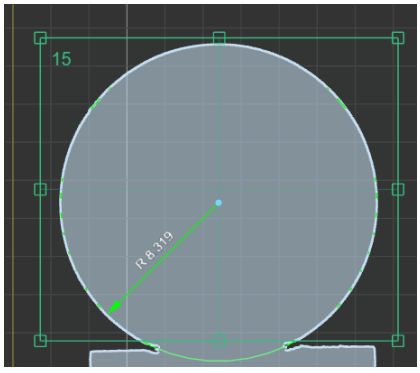
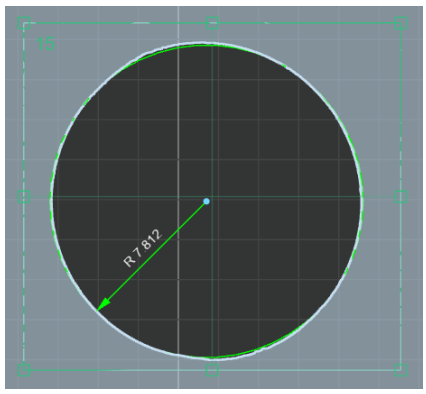
			algorithm to smooth (average) the coordinates of profile points.
	roi	Rect	Measurement area - ROI. Defined by the following parameters: left top x, left top y, width, height.

	"extreme points" - this smart block is designed to search for extreme points of the profile within the measurement area specified by the ROI. 		
Inputs:	InpProfile	Profile	Input profile.
	InpRoi	Rect	Measurement area. When the input is disabled, the calculation uses the default value from the "roi" parameter.
Outputs:	PointMaxX	double	The point at which the X coordinate reaches its maximum value in the profile.
	PointMaxY	double	The point at which the Y coordinate reaches its maximum value in the profile.
	PointMinX	double	The point at which the X coordinate reaches its minimum value in the profile.
	PointMinY	double	The point at which the Y coordinate reaches its minimum value in the profile.
Parameters:	smoothWindow	uint8_t,[5]	The width of the smoothing (averaging) window for the coordinates of profile points. This parameter determines the size of the window used by the algorithm to smooth (average) the coordinates of profile points.
	roi	Rect	Measurement area - ROI. Defined by the following parameters: left top x, left top y, width, height.

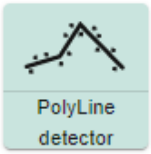
	"mean" - this smart block is designed to find the central point as the center of mass of all profile points within the measurement area specified by the ROI.		
Inputs:	InpProfile	Profile	Input profile.
	InpRoi	Rect	Measurement area. When the input is disabled, the calculation uses the default value from the "roi" parameter.
Outputs:	OutPoint	Point2dDouble	The resulting center point.
Parameters:	roi	Rect	Measurement area - ROI. Defined by the following parameters: left top x, left top y, width, height.

	"line approximation" - this smart block is designed to approximate the input profile by a straight line. The approximation of points is based on the least squares method or its outlier-resistant modification. The approximation takes into account all the profile points that are within the measurement area specified by the ROI. The result can be represented as a straight line or a straight line segment bounded at both ends by the ROI:

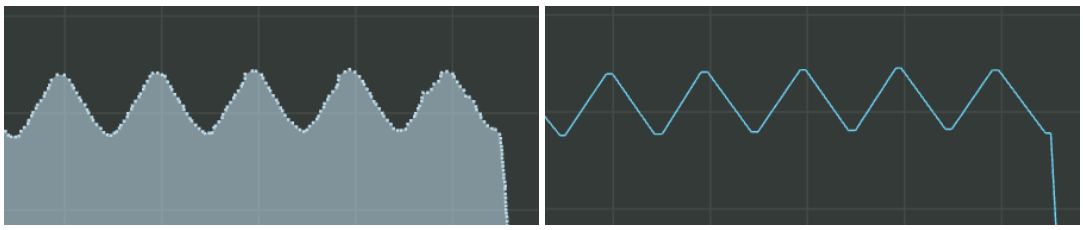
	 Approximation by a segment  Approximation by a line		
Inputs:	InpProfile	Profile	Input profile.
	InpRoi	Rect	Measurement area. When the input is disabled, the calculation uses the default value from the "roi" parameter.
Outputs:	Line	StraightLine, SegmentLine	The result of approximation by a straight line or a segment based on the "lineType" parameter. In case of a calculation error, there is no value at the output.
Parameters:	lineFittingMethod	string enum, ["Stable"]	Method for approximating a straight line using measured points: Stable - a method that provides increased reliability in the presence of outliers; LeastSquares - a classic least-squares method.
	lineType	string	Line type. Possible values: • Segment - straight line segment. • Straight - straight line.
	roi	Rect	Measurement area - ROI. Defined by the following parameters: left top x, left top y, width, height.

 circle approximation	"circle approximation" - this smart block is designed to approximate the input profile by a circle and find its center and radius. The approximation of points by a circle is based on the least squares method. When approximating, all profile points located within the measurement area specified by the ROI parameters are taken into account. It is possible to approximate both external and internal contours:  Outer circle  Inner circle		
Inputs:	InpProfile	Profile	Input profile.
	InpRoi	Rect	Measurement area. When the input is disabled, the calculation uses the default value from the "roi" parameter.
Outputs:	OutCenter	Point2dDouble	The center of the approximated circle. In case of calculation error, there is no output value.
	OutRadius	double	The radius of the approximated circle. In case of calculation error, there is no output value.
	ResultDescription	Description	The result of the check with descriptive semantics. The result is represented as a json object with the following fields: • "type" - the value is always "Circle". • "R" - the resulting value of the circle radius.

			• "Center" - the center point of the circle, the point object {"x", "y"}. • "Valid" - indicates whether the result is correct, true or false. False indicates an error in the calculation. The output is intended to be displayed in the results area.
Parameters:	contourType	string enum, ["Outer"]	Type of analyzed contours of the input profile. Possible values: • Outer - approximation is performed using outer contour points that are located within the region of interest (ROI). • Inner - approximation is performed using inner contour points that are located within the region of interest (ROI).
	roi	Rect	Measurement area - ROI. Defined by the following parameters: left top x, left top y, width, height.

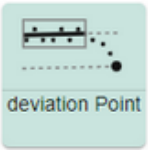


"PolyLine detector" - approximation of the profile by a piecewise linear continuous function. Approximation example:

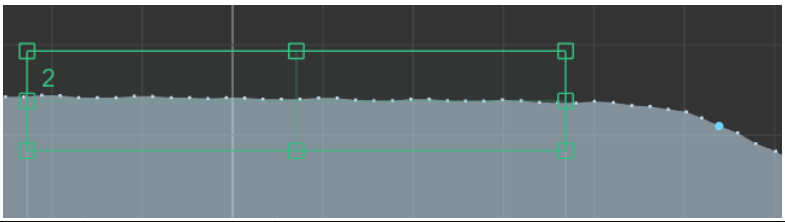


Input profile | Approximation result

Inputs:	InpProfile	Profile, Contour	Input profile or contour.
	InpRoi	Rect	Measurement area. When the input is disabled, the calculation uses the default value from the "roi" parameter.
Outputs:	PolyLine	PolyLine	The resulting polyline.
Parameters:	roi	Rect	Measurement area - ROI. Defined by the following parameters: left top x, left top y, width, height.

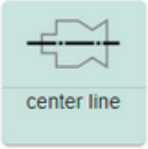


"deviation point" - search for the first point that deviates from a straight line by a distance greater than a specified threshold value. The search for a point is carried out in a given direction relative to the ROI. The desired point is located at a distance exceeding the specified threshold level from the line approximated within the ROI.

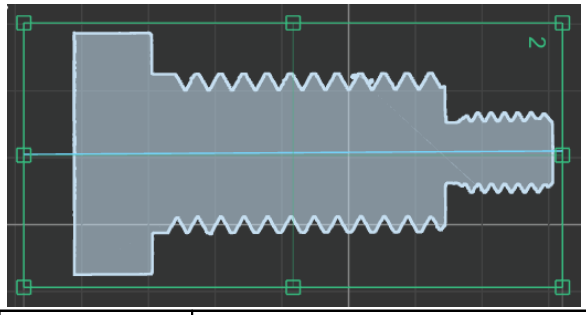


Inputs:	InpProfile	Profile	Input profile.
	InpRoi	Rect	The area where profile points are approximated by a line. When the input is disabled, the calculation uses the default value from the "roi" parameter.
Outputs:	DeviationPoint	Point2dDouble	A profile point located at a specified distance (distanceThreshold) from the line.
Parameters:	distanceTreshold	double,[0.03]	Threshold value of the distance (in millimeters) from the line to the profile points being checked. If this value is exceeded, the required point will be detected.
	isForward	bool,[true]	The direction of searching for a point relative to a specified ROI.

	smoothWindow	uint8_t,[5]	The width of the smoothing (averaging) window for the coordinates of profile points. This parameter determines the size of the window used by the algorithm to smooth (average) the coordinates of profile points.
	roi	Rect	Measurement area - ROI. Defined by the following parameters: left top x, left top y, width, height.

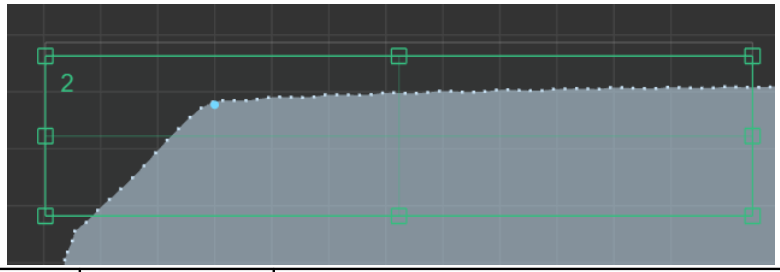


"center line" - search for the center line (line of symmetry) for a given profile.



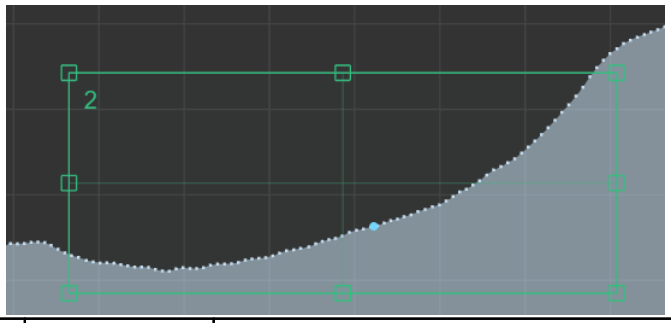
Inputs:	InpProfile	Profile	Input profile.
	InpRoi	Rect	Measurement area. When the input is disabled, the calculation uses the default value from the "roi" parameter.
Outputs:	Line	StraightLine, SegmentLine	The resulting center line.
Parameters:	lineType	string	Line type. Possible values: • Segment - straight line segment. • Straight - straight line.
	roi	Rect	Measurement area - ROI. Defined by the following parameters: left top x, left top y, width, height.

"inflection point" - search for the point of maximum inflection of the line on the profile, limited by the ROI.

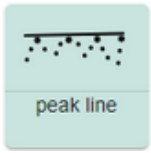


Inputs:	InpProfile	Profile	Input profile.
	InpRoi	Rect	Inflection point search area. When the input is disabled, the calculation uses the default value from the "roi" parameter.
Outputs:	Inflection	Point2dDouble	Maximum inflection point.
Parameters:	smoothWindow	uint8_t,[5]	The width of the smoothing (averaging) window for the coordinates of profile points. This parameter determines the size of the window used by the algorithm to smooth (average) the coordinates of profile points.
	roi	Rect	Inflection point search area - ROI. Defined by the following parameters: left top x, left top y, width, height.

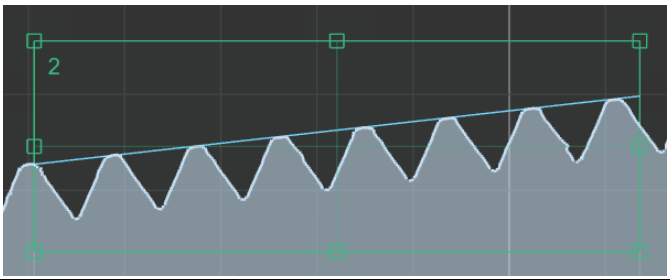
"peak point" - search for a local extremum on the profile along the principal component, which is determined using the principal component method. The principal component is a line in two-dimensional space that indicates the direction of the largest data length.



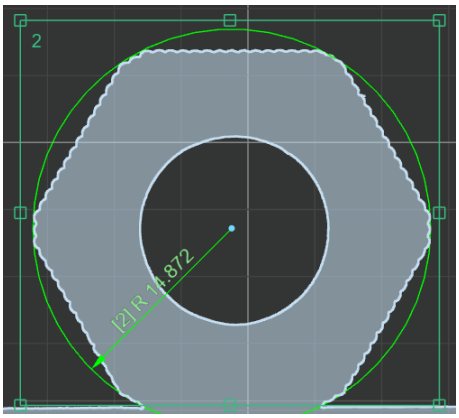
Inputs:	InpProfile	Profile	Input profile.
	InpRoi	Rect	Extremum point search area. When the input is disabled, the calculation uses the default value from the "roi" parameter.
Outputs:	Inflection	Point2dDouble	Extremum point.
Parameters:	smoothWindow	uint8_t,[5]	The width of the smoothing (averaging) window for the coordinates of profile points. This parameter determines the size of the window used by the algorithm to smooth (average) the coordinates of profile points.
	roi	Rect	Extremum point search area - ROI. Defined by the following parameters: left top x, left top y, width, height.

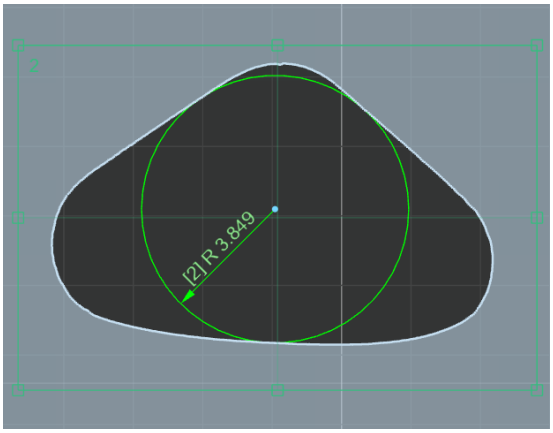


"peak line" - search for an adjacent straight line. An adjacent straight line is defined as a straight line in contact with the profile outside the material of the part and located in relation to the profile so that the distance from its most distant point to the adjacent straight line is the smallest.



Inputs:	InpProfile	Profile	Input profile.
	InpRoi	Rect	Measurement area. When the input is disabled, the calculation uses the default value from the "roi" parameter.
Outputs:	Line	StraightLine, SegmentLine	The resulting adjacent line.
Parameters:	lineType	string	Line type. Possible values: • Segment - straight line segment. • Straight - straight line.
	smoothWindow	uint8_t,[5]	The width of the smoothing (averaging) window for the coordinates of profile points. This parameter determines the size of the window used by the algorithm to smooth (average) the coordinates of profile points.
	roi	Rect	Measurement area - ROI. Defined by the following parameters: left top x, left top y, width, height.

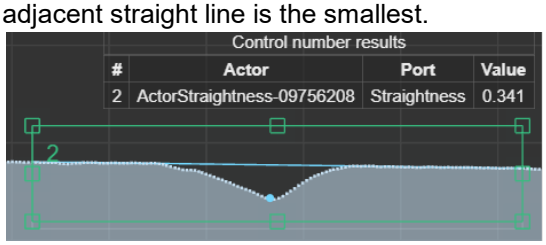
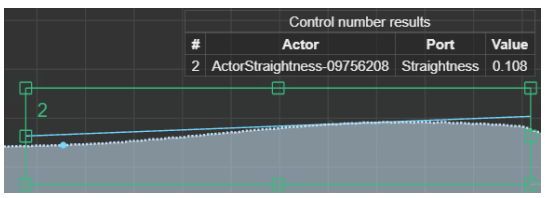
<table>
<tr><td> 
</td><td colspan="3">"peak circle" - search for an adjacent circle. The search takes into account all contour points of a given type (outer or inner) located within the measurement area specified by the ROI. It is possible to approximate both outer and inner contours. For the outer contour, the search is made for a circle located around the contour points and having the smallest diameter. For the inner contour, the search is made for a circle located inside the contour and having the largest diameter.
 

Outer circle
 

Inner circle</td></tr>
<tr><td rowspan="2">Inputs:</td><td>InpProfile</td><td>Profile</td><td>Input profile.</td></tr>
<tr><td>InpRoi</td><td>Rect</td><td>Measurement area. When the input is disabled, the calculation uses the default value from the "roi" parameter.</td></tr>
<tr><td rowspan="3">Outputs:</td><td>OutCenter</td><td>Point2dDouble</td><td>The center of the adjacent circle. In case of calculation error, there is no output value.</td></tr>
<tr><td>OutRadius</td><td>double</td><td>The radius of the adjacent circle. In case of calculation error, there is no output value.</td></tr>
<tr><td>ResultDescription</td><td>Description</td><td>The result of the check with descriptive semantics. The result is represented as a json object with the following fields:
• "type" - the value is always "Circle".
• "R" - the resulting value of the circle radius.
• "Center" - the center point of the circle, the point object {"x", "y"}.
• "Valid" - indicates whether the result is correct, true or false. False indicates an error in the calculation.
The output is intended to be displayed in the results area.</td></tr>
<tr><td rowspan="3">Parameters:</td><td>contourType</td><td>string enum, ["Outer"]</td><td>Type of analyzed contours of the input profile. Possible values:
• Outer - the search is performed using outer contour points that are located within the region of interest (ROI).
• Inner - approximation is performed using inner contour points that are located within the region of interest (ROI).</td></tr>
<tr><td>smoothWindow</td><td>uint8_t,[5]</td><td>The width of the smoothing (averaging) window for the coordinates of profile points. This parameter determines the size of the window used by the algorithm to smooth (average) the coordinates of profile points.</td></tr>
<tr><td>roi</td><td>Rect</td><td>Measurement area - ROI. Defined by the following parameters: left top x, left top y, width, height.</td></tr>
</table>

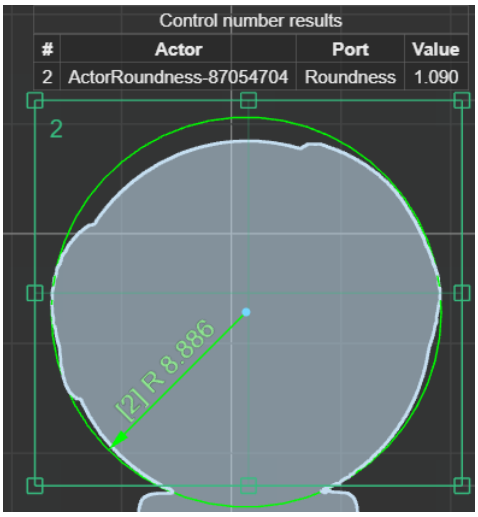
	"tilt" - this smart block is designed to calculate the tilt angle of the profile along a given edge detected within the ROI. The ROI must contain a set of points that includes only one edge of the input profile. If more than one edge falls into the detection area, the first detected edge is used in the calculation. The tilt angle of the profile is defined as the minimum angle from the edge to the horizontal line and to the vertical line.		
Inputs:	InpProfile	Profile	Input profile.
	InpRoi	Rect	Search area. When the input is disabled, the calculation uses the default value from the "roi" parameter.
Outputs:	Tilt	double	Tilt angle value.
	ResultDescription	Description	The result of the check with descriptive semantics. The result is represented as a json object with the following fields: • "type" - the value is always "Angle". • "Angle" - the resulting value of the angle. • "Segment1" - a segment on the slope line, limited by the ROI. The object contains {"x1", "y1", "x2", "y2"}. • "Segment2" - a segment of a horizontal (or vertical) line, limited by the ROI. The object contains {"x1", "y1", "x2", "y2"}. • "Valid" - indicates whether the result is correct, true or false. False indicates an error in the calculation. The output is intended to be displayed in the results area.
Parameters:	roi	Rect	Measurement area - ROI. Defined by the following parameters: left top x, left top y, width, height.
	bias	double	Offset added to the measured result. Used to adjust (calibrate) the measurement result, taking into account systematic errors or individual characteristics.

	"area" - this smart block is designed to calculate the profile area within the ROI. The resulting value includes the sum of the areas of all outer contours minus the area of all inner contours located within the ROI ($S = \Sigma S_{ext} - \Sigma S_{int}$).		
Inputs:	InpProfile	Profile	Input profile.
	InpRoi	Rect	Measurement area. When the input is disabled, the calculation uses the default value from the "roi" parameter.
Outputs:	Area	double	Profile area value.
	OutProfile	Profile	Output profile located within the ROI.
Parameters:	roi	Rect	Measurement area - ROI. Defined by the following parameters: left top x, left top y, width, height.
	bias	double	Offset added to the measured result. Used to adjust (calibrate) the measurement result, taking into account systematic errors or individual characteristics.

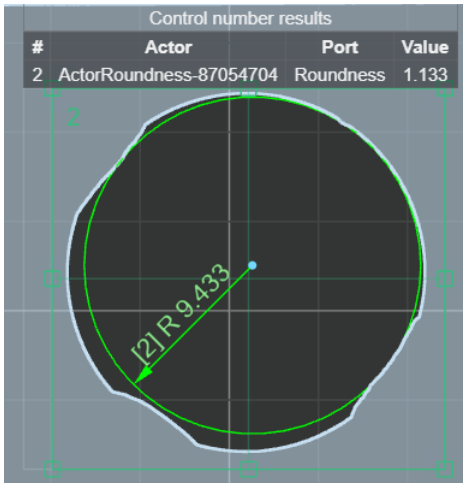
	"straightness" - assessment of straightness within the ROI. Straightness is defined as the greatest distance from the profile points to the adjacent straight line. An adjacent straight line is a straight line that is in contact with the profile outside the material of the part and is located in relation to the profile so that the distance from its most distant point to the adjacent straight line is the smallest.  		
Inputs:	InpProfile	Profile	Input profile.

	InpRoi	Rect	Measurement area. When the input is disabled, the calculation uses the default value from the "roi" parameter.
Outputs:	SuperimposedLine	SegmentLine	The resulting adjacent line.
	Straightness	double	Straightness value.
	FarthestPoint	Point2dDouble	The farthest point from the adjacent line.
Parameters:	lineType	string	Line type. Possible values: • Segment - straight line segment. • Straight - straight line.
	smoothWindow	uint8_t,[5]	The width of the smoothing (averaging) window for the coordinates of profile points. This parameter determines the size of the window used by the algorithm to smooth (average) the coordinates of profile points.
	roi	Rect	Measurement area - ROI. Defined by the following parameters: left top x, left top y, width, height.

"roundness" - assessment of roundness within the ROI. It is defined as the greatest distance from the profile points to the adjacent circle. The adjacent circle is defined differently for the outer and inner contours. For the outer contour, the search is made for a circle located around the contour points and having the smallest diameter. For the inner contour, the search is made for a circle located inside the contour and having the largest diameter.



Outer circle

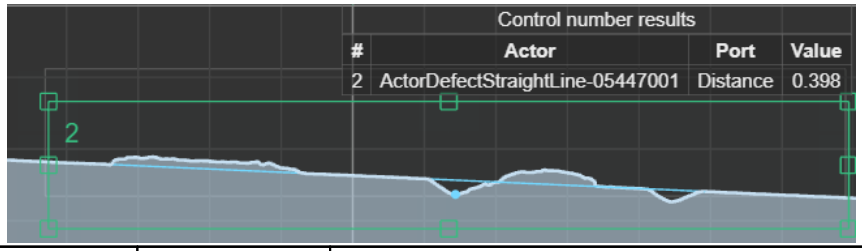


Inner circle

Inputs:	InpProfile	Profile	Input profile.
	InpRoi	Rect	Measurement area. When the input is disabled, the calculation uses the default value from the "roi" parameter.
Outputs:	OutCenter	Point2dDouble	The center of the adjacent circle. In case of calculation error, there is no output value.
	OutRadius	double	The radius of the adjacent circle. In case of calculation error, there is no output value.
	ResultDescription	Description	The result of the check with descriptive semantics. The result is represented as a json object with the following fields: • "type" - the value is always "Circle". • "R" - the resulting value of the circle radius. • "Center" - the center point of the circle, the point object {"x", "y"}. • "Valid" - indicates whether the result is correct, true or false. False indicates an error in the calculation. The output is intended to be displayed in the results area.

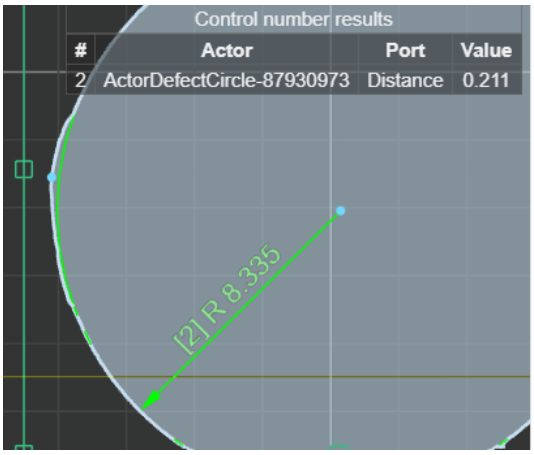
	Roundness	double	Roundness value.
Parameters:	contourType	string enum, ["Outer"]	Type of analyzed contours of the input profile. Possible values: • Outer - the search is performed using outer contour points that are located within the region of interest (ROI). • Inner - approximation is performed using inner contour points that are located within the region of interest (ROI).
	smoothWindow	uint8_t,[5]	The width of the smoothing (averaging) window for the coordinates of profile points. This parameter determines the size of the window used by the algorithm to smooth (average) the coordinates of profile points.
	roi	Rect	Measurement area - ROI. Defined by the following parameters: left top x, left top y, width, height.

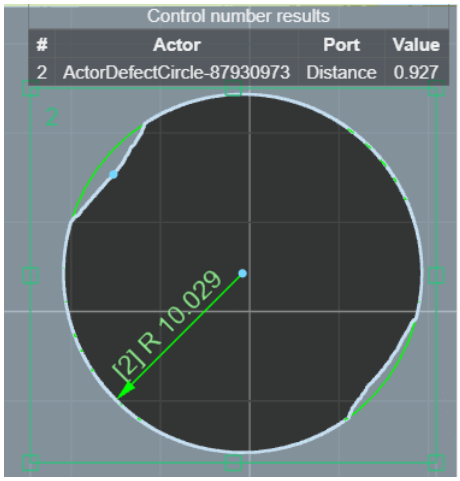
"defect straight line" - search for a defect relative to an approximated straight line within the ROI. The approximation of points by a line is based on the least squares method with the exclusion of defective points.



Inputs:	InpProfile	Profile	Input profile.
	InpRoi	Rect	Measurement area. When the input is disabled, the calculation uses the default value from the "roi" parameter.
Outputs:	Line	SegmentLine	Approximated line.
	Distance	double	The greatest distance from the profile points to the approximated circle.
	FarthestPoint	Point2dDouble	The farthest point of the profile from the approximated line.
Parameters:	distanceThreshold	double,[0.05]	The minimum threshold distance at which a defect will be detected.
	smoothWindow	uint8_t,[5]	The width of the smoothing (averaging) window for the coordinates of profile points. This parameter determines the size of the window used by the algorithm to smooth (average) the coordinates of profile points.
	roi	Rect	Measurement area - ROI. Defined by the following parameters: left top x, left top y, width, height.

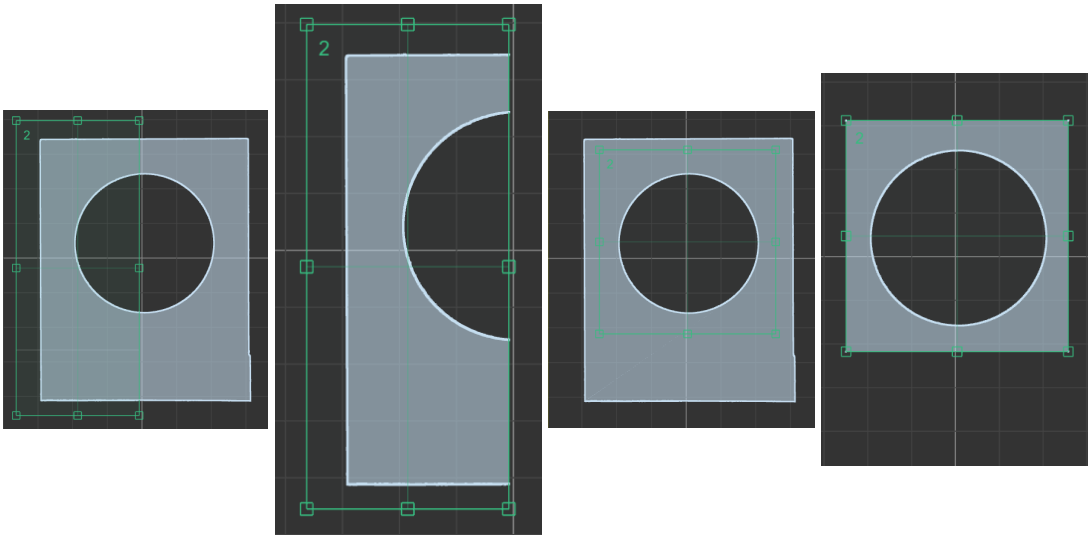
"defect circle" - search for a defect on a circle within the ROI. It is defined as the greatest distance from the profile points to the approximated circle. The approximation of points by a circle is based on the least squares method with the exclusion of defective points.

<table>
<tr><td></td><td colspan="3"> 

Outer circle


Inner circle</td></tr>
<tr><td rowspan="2">Inputs:</td><td>InpProfile</td><td>Profile</td><td>Input profile.</td></tr>
<tr><td>InpRoi</td><td>Rect</td><td>Measurement area. When the input is disabled, the calculation uses the default value from the "roi" parameter.</td></tr>
<tr><td rowspan="6">Outputs:</td><td>Distance</td><td>double</td><td>The greatest distance from the circle to the contour point.</td></tr>
<tr><td>FarthestPoint</td><td>Point2dDouble</td><td>The farthest point from the approximated circle.</td></tr>
<tr><td>OutCenter</td><td>Point2dDouble</td><td>The center of the approximated circle. In case of calculation error, there is no output value.</td></tr>
<tr><td>OutRadius</td><td>double</td><td>The radius of the approximated circle. In case of calculation error, there is no output value.</td></tr>
<tr><td>ResultDescription</td><td>Description</td><td>The result of the check with descriptive semantics. The result is represented as a json object with the following fields:
• "type" - the value is always "Circle".
• "R" - the resulting value of the circle radius.
• "Center" - the center point of the circle, the point object {"x", "y"}.
• "Valid" - indicates whether the result is correct, true or false. False indicates an error in the calculation.
The output is intended to be displayed in the results area.</td></tr>
<tr><td>Roundness</td><td>double</td><td>Roundness value.</td></tr>
<tr><td rowspan="4">Parameters:</td><td>distanceThreshold</td><td>double,[0.05]</td><td>The minimum threshold distance at which a defect will be detected.</td></tr>
<tr><td>contourType</td><td>string enum, ["Outer"]</td><td>Type of analyzed contours of the input profile. Possible values:
• Outer - the search is performed using outer contour points that are located within the region of interest (ROI).
• Inner - approximation is performed using inner contour points that are located within the region of interest (ROI).</td></tr>
<tr><td>smoothWindow</td><td>uint8_t,[5]</td><td>The width of the smoothing (averaging) window for the coordinates of profile points. This parameter determines the size of the window used by the algorithm to smooth (average) the coordinates of profile points.</td></tr>
<tr><td>roi</td><td>Rect</td><td>Measurement area - ROI. Defined by the following parameters: left top x, left top y, width, height.</td></tr>
</table>

"select in ROI" - this smart block is designed to select a part of the profile/contour within the area of interest specified by the ROI.

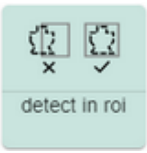


ROI includes outer and inner contours ROI includes inner contour

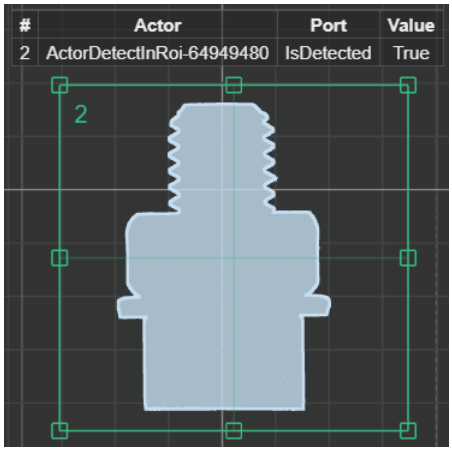
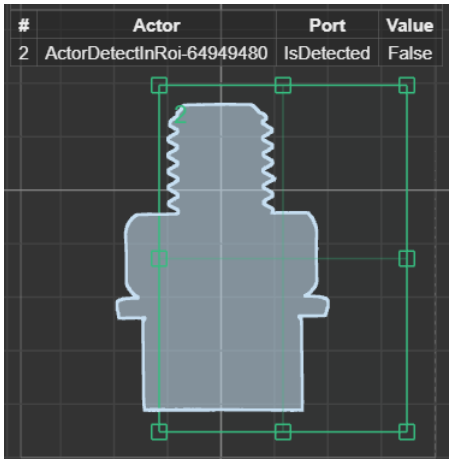
Inputs:	InpProfile	Profile,Contour	Input profile/contour.
	InpRoi	Rect	Measurement area. When the input is disabled, the calculation uses the default value from the "roi" parameter.
Outputs:	OutProfile	Profile,Contour	Output profile/contour.
Parameters:	roi	Rect	Measurement area - ROI. Defined by the following parameters: left top x, left top y, width, height.

"union" - this smart block is designed to unite profiles.

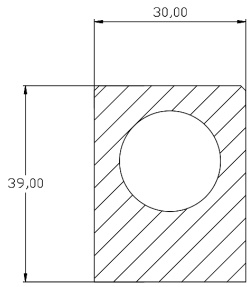
Inputs:	InpProfile	Profile,Contour	Input profile/contour. Must be shifted relative to the previous one.
	Shift	Point2dDouble	The point that defines the new center of the coordinate system (after a parallel shift).
	Reset	* (all types supported in schemes)	Resetting the profile accumulated from previous iterations.
Outputs:	OutProfile	Profile,Contour	Profile with a transformed coordinate system.
Parameters:	isAsync	bool,[true]	Flag. It indicates whether processing is asynchronous.

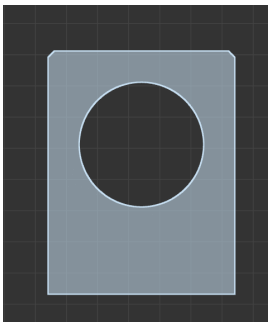
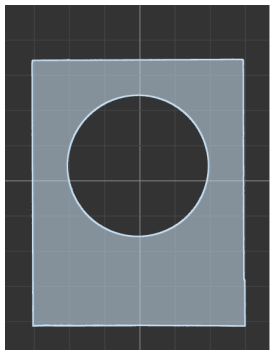
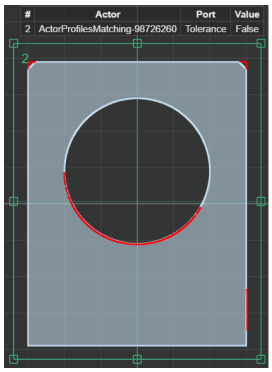


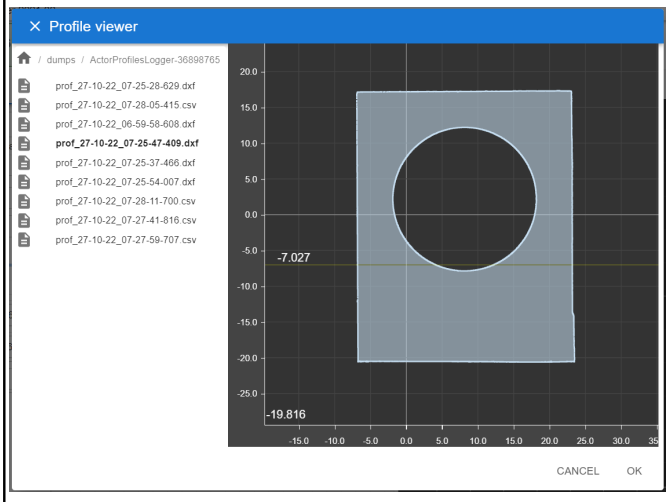
"detect in roi" - this smart block is designed to check whether a profile is within the ROI. If the profile is within the ROI, it is duplicated to the OutProfile output. It is possible to exclude any sides of the ROI from checking for intersection with the profile.

	 Profile within ROI		 Profile outside ROI
Inputs:	InpProfile	Profile	Input profile/contour.
	InpRoi	Rect	Check area. When the input is disabled, the calculation uses the default value from the "roi" parameter.
Outputs:	IsDetected	bool	Signal that the profile is within the ROI.
	OutProfile	Profile	Output profile. It only appears if the profile is within the ROI.
Parameters:	includeLeftSide	bool,[true]	Indicates the inclusion of the ROI side in the check. When excluding a side from the check, the fact that the profile intersects the specified side is ignored, and the profile is considered to be within the ROI, provided that all sides for which the value is True are within the ROI.
	includeRightSide		
	includeTopSide		
	includeBottomSide		
	OutProfileOnlyPartInRoi	bool,[true]	A flag indicating that only the part of the profile that is within the ROI is extracted. When set to True, the *OutProfile* output only receives the part of the profile located within the ROI and only when the detection condition is met. This only makes sense if any side is excluded from the check.
	minArea	double	Minimum allowable profile area for it to be considered within the ROI. If the profile area is smaller than the specified value, detection does not occur.
	profileSequenceSelectionMode	string enum, ["All"]	Profile selection mode for output to *OutProfile* from all profiles that passed the ROI inclusion check: • All - all profiles that passed the check are sent to the output. • First - only the first profile that passed the check is sent to the output. • Last - only the last profile that passed the check is sent to the output. • Middle - the middle profile in order is sent to the output. • ByIndex - the profile with the specified index (*desiredIndex*) is sent to the output.
	desiredIndex	uint8_t	Profile index to select when profileSequenceSelectionMode = ByIndex.
	roi	Rect	Check area - ROI. Defined by the following parameters: left top x, left top y, width, height.

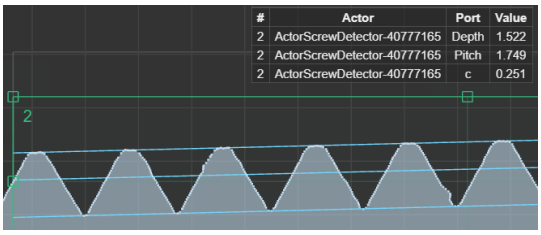
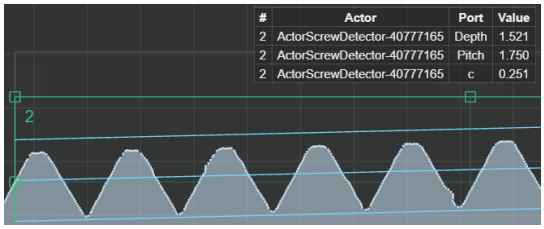
	"matching" - this smart block is designed to compare the measured profile with the reference one. The reference profile is a previously saved profile obtained from a sensor, or a hatch drawing from a DXF file. First, the input profile is aligned relative to the reference one. A comparison is made between the aligned contour points and the reference profile points within the region of interest (ROI). When aligning, all points of the input and reference

<table>
<tr><td></td><td colspan="3">profiles are taking into account. When matching, only points located in the ROI are taking into account.
 

DXF reference drawing (CAD) DXF reference drawing (ProfileViewer) Input profile Matching result ReferenceProfile+ResultDescription</td></tr>
<tr><td>Inputs:</td><td>InpProfile</td><td>Profile</td><td>Input profile.</td></tr>
<tr><td rowspan="4">Outputs:</td><td>AllignedProfile</td><td>Profile</td><td>Profile aligned relative to the reference one.</td></tr>
<tr><td>ReferenceProfile</td><td>Profile</td><td>Reference profile.</td></tr>
<tr><td>ResultDescription</td><td>Description</td><td>The result of the check with descriptive semantics. The result is represented as a json object with the following fields:
• "type" - the value is always "Matching".
• "R" - the resulting value of the circle radius.
• "Polylines" - an array containing groups of consecutive points that deviate from the reference profile by a distance greater than or equal to distanceThreshold. Each point in the sequence is represented as a json object with the following fields: {"x":float, "y":float, "dist":float}.
• "Valid" - indicates whether the result is correct, true or false. False indicates an error in the calculation.
The output is intended to be displayed in the results area.</td></tr>
<tr><td>Tolerance</td><td>bool</td><td>Compliance of the input profile with the reference one. If at least one section (group of consecutive points) deviates from the reference, the resulting value will be False.</td></tr>
<tr><td>Parameters:</td><td>pathEthalonProfile</td><td>string</td><td>Path to the reference profile. Csv, dxf and svg files are available and located in the dumps and tmp_dumps directories. To select the required file, use Profile Viewer:
 
</td></tr>
</table>

			To open Profile Viewer, click the "SelectProfile" button located to the right of the pathEthalonProfile field.
	distanceThreshold	double,[0.02]	Permissible threshold for deviation of the measured profile from the reference one.
	minPoints	int,[4]	The minimum number of points located sequentially, the deviation of which from the reference will lead to the detection of a defect.
	roi	Rect	Check area - ROI. Defines the area on the reference profile involved in comparison with the aligned input profile. However, it should be noted that this area does not define the section of the reference profile that is used to align the input profile relative to the reference profile. Set by the following parameters: left top x, left top y, width, height.

	"screw detector" - measurement of thread parameters.  RootC-Center-CrestC  RootH-Center-CrestH		
Inputs:	InpProfile	Profile,Contour	Input profile/contour.
	InpRoi	Rect	Measurement area. When the input is disabled, the calculation uses the default value from the "roi" parameter.
Outputs:	Center	SegmentLine	Central line. It is located equidistant from the RootH and CrestH lines.
	CrestC	SegmentLine	A line approximated by all real points on the thread crests.
	CrestH	SegmentLine	A line approximated by all imaginary points on the thread crests. Each imaginary point is formed by the intersection of pairwise adjacent sides of the thread profile and is located on its outer side.
	RootC	SegmentLine	A line approximated by all real points on the thread roots.
	RootH	SegmentLine	A line approximated by the imaginary bases of all original thread triangles within the ROI. Each imaginary point of the triangle base is formed by the intersection of pairwise adjacent sides of the thread profile and is located along its inner side.
	Depth	double	The height of the original thread triangle. It is defined as the distance between the apex and base of the original thread triangle in the direction perpendicular to the thread axis.
	Pitch	double	Thread pitch. It is defined as the distance along a line parallel to the thread axis between the midpoints of the nearest sides of the thread profile, lying on the same axial plane on one side of the thread axis.
	c	double	Thread cut. It is defined as the distance perpendicular to the thread axis from the imaginary point of intersection of two adjacent sides of the thread profile to the nearest point of its top or bottom.
	ResultDescription	Description	The result of the check with descriptive semantics. The result is represented as a json object with the following fields:

			• "type" - the value is always "Screw". • "Valid" - indicates whether the result is correct, true or false. False indicates an error in the calculation. The output is intended to be displayed in the results area.
Parameters:	roi	Rect	Measurement area - ROI. Defined by the following parameters: left top x, left top y, width, height.

	"classifier" - this smart block is designed to automatically determine the part type based on the input profile or contour. Classification is performed based on parameters specified in the JSON file.		
Inputs:	InpProfile	Profile,Contour	Input profile or contour of the part for classification.
	InpRoi	Rect	Measurement area. If the input is disabled, the default value from the "roi" parameter is used.
Outputs:	ClassNumber	int8_t	Number of the part class. If no class is detected, the output value is set to -1.
	ResultDescription		Structure containing detailed classification result information, including the following fields: • type:string - block type, always contains the value "Classifier". • classNumber:int8_t - number of the identified part class. • className:string - name of the identified part class. • features:array - array of objects in the form {"name": <feature_name>, "value": <value>} containing the set of feature values used for classification. • classProbability:array - array of probabilities for belonging to each possible class. Each element of the array corresponds to the probability for a specific class. • Valid:bool - flag indicating the validity and reliability of the classification result.
	Profile (for each class)	Profile	Input profile that is output to the corresponding port if it has been assigned to this class.
Parameters:	path	string	Path to the JSON file containing classifier parameters and the list of classes. All algorithm settings and class descriptions are defined in this file.
	roi	Rect	Measurement area - ROI. Defined by the following parameters: left top x, left top y, width, height.

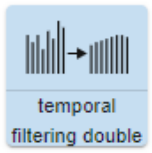
	"gauge line" - this smart block is designed to automatically determine the position of an imaginary line perpendicular to a given direction and passing through a specified diameter on a conical or other profile surface. The line is searched for within a specified measurement area (ROI) and can be performed along various directions, including horizontal, vertical, axial, or any user-defined direction.		
Inputs:	InpProfile	Profile,Contour	Input profile or contour for analysis.
	InpRoi	Rect	Measurement area. If the input is disabled, the default value from the "roi" parameter is used.
	LineScan	SegmentLine, StraightLine	Reference line along which the search is performed. This port is active when the search direction scanDirection = SpecifiedLineScan is selected.
Outputs:	SegmentLine	SegmentLine	Detected line of the specified diameter, perpendicular to the chosen search direction.

Parameters:	scanDirection	string enum, ["HorizontalScan"]	Search direction along which the target line is determined: HorizontalScan - search along a horizontal line; VerticalScan - search along a vertical line; CentralLineScan - preliminary detection of the profile's central axis, followed by search along it; SpecifiedLineScan - search along a line defined through the LineScan input.
	width	double	Line length that the search result must correspond to.
	approximateEdgesAsLines	bool	If true, the profile elements within the search zone are approximated by straight lines, which increases accuracy when analyzing conical or complex surfaces.
	syncMode	string enum, ["SameId"]	Computation synchronization mode when using scanDirection = SpecifiedLineScan: • NoSync - computation is performed upon receiving data at any input; • SameId - computation is performed only after both the profile and line with the same identifier (Id) are received.
	roi	Rect	Measurement area - ROI. Defined by the following parameters: left top x, left top y, width, height.

	"Runout & Concentricity" - this smart block is designed for automatic measurement of radial runout, total radial runout, and concentricity of a part profile relative to a specified reference surface. It is used to control the geometric parameters of rotating and symmetrical parts. For correct operation, each input profile (InpProfile) must correspond to the rotation angle at which it was captured. This angle can be provided directly through the InpAngle input (individually for each profile), or, if no values are received at this input, the angles are calculated automatically using a fixed step defined by the angleStepDeg parameter.		
Inputs:	InpProfile	Profile,Contour	Input profile or contour.
	InpAngle	double	Rotation angle of each profile received at the InpProfile input.
	Request	All	Request to perform the calculation.
	Reset	All	Resetting the block state before starting a new measurement of the part.
Outputs:	Slices	[Polyline,...]	Array of polylines by sections used to determine the reference axis and the sections for runout and concentricity measurement.
	ResultDescription	Description	Verification result with descriptive semantics. The result is presented as a JSON object.
	{mes-id}	double	Individual output for each measured parameter from the Measurements array. The output name corresponds to the specified identifier (id) of the parameter.
Parameters:	angleStepDeg	double	Angular rotation step between each input profile. Used if no real rotation values are provided at the InpAngle input.
	addSlicesPort	bool	Flag for adding the Slices output.
	dumpSlices	bool	Flag for saving slices in the dump directory.
	Parameters for defining the base axis *datumAxisProperties*		
	axisDirection	string enum, ["Horizontal"]	Base axis direction: Horizontal or Vertical.
	startPos endPos	double	Start and end coordinates along x (for axisDirection = Horizontal) or along y (for axisDirection = Vertical) within which the base axis will be determined.
	plans	uint8_t[2]	Number of sections used to determine the base axis. For each section, one point is defined in 3D space. The value must be at least 2. Increasing the number of

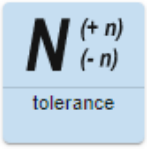
			sections improves accuracy but increases calculation time.
	halsWidth	double	Half of the ROI width along the specified axis when averaging points.
	Measurements array		
	id	string	Unique identifier of the measured quantity.
	measurementType	string enum, ["Runout"]	Type of measured quantity: Runout, TotalRunout, Concentricity.
	position	double	Coordinate along the specified axis where the measurement is performed (for Runout and Concentricity).
	halsWidth	double	Half of the ROI width along the specified axis when averaging points.
	startPos endPos	double	Start and end coordinates along x (for axisDirection = Horizontal) or along y (for axisDirection = Vertical) for determining TotalRunout.
	plans	uint8_t[2]	Number of sections used to determine TotalRunout.
	halsWidth	double	Half of the ROI width along the specified axis when averaging points.

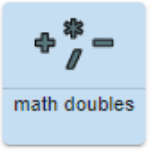
11.4.2.2.4. "Math functions" section

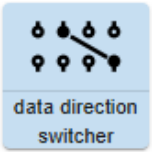
	“scalar filtering” - filtering incoming scalar values. Pre-filtering is performed by the median filter set by the “Median filter” parameter. The smoothing of the values is done by simple averaging.		
Parameters:	“medianSize”	uint16_t, 0...65535, [7]	Sample size for median filtering. If medianSize<2, no filtering is performed.
	“smoothSize”	uint16_t, 0...65535, [7]	Sample size for averaging the result. If smoothSize<2, no averaging is performed.
Inputs:	“InpNum”	double	Input value for filtering.
	"Reset"	* ()	Filter reset signal.
Outputs:	“OutNum”	double	Output filtered value.

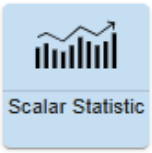
	"spatial filtering" - smoothing contour (profile) points using various spatial filtering methods. It reduces noise and improves profile analysis quality by applying filters to the input data.		
Parameters:	Filter parameters for the minimum number of points (FilterContourMinPointsProperties):		
	“useFilterContourMinPoints”	bool, [true]	Enabling a filter that removes contours with fewer points than the specified value.
	“minPoints”	uint32_t, [32]	The minimum number of points allowed in a contour. Contours with fewer points will be filtered out.
	Simple moving average filter parameters (FilterSimpleMovingAverageProperties):		
	“useFilterSimpleMovingAverage”	bool,[false]	Enabling the Simple Moving Average filter to smooth the profile.
	windowSize	uint8_t, 0...255 [5]	Window size for the moving average filter.
	Savitzky-Golay filter parameters (FilterSavitzkyGolayProperties):		
	useFilterSavitzkyGolay	bool,[false]	Enabling the Savitzky-Golay filter for more precise profile smoothing.
	windowSize	uint8_t, 0...255 [5]	Window size for the Savitzky-Golay filter.

	polynomialDegree	uint8_t, 0...255 [3]	Degree of polynomial for the Savitzky-Golay filter.
Inputs:	InpProfile	Profile	Input profile (contour) for processing.
Outputs:	OutProfile	Profile	Smoothed (filtered) profile.

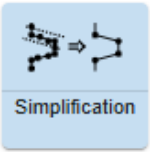
	“tolerance” - checking the input value for falling into the specified range.		
Parameters:	label	string [label]	Alias for the value being checked, included in the "ResultDescription" result.
	minValue	double [0]	The lower limit of the range.
	maxValue	double [100]	The upper limit of the range.
Inputs:	“Number”	double	The value to check.
Outputs:	“Tolerance”	bool	Result.
	“ResultDescription”	Description	The result of the check with descriptive semantics. The result is represented as a json object with the following fields: • "type" - always "Tolerance". • "label" - alias. • "tolerance" - true or false - result. • "value" - checked value received at the “Number” input. • "minValue" - value from the input parameter of the same name. • "maxValue" - value from the input parameter of the same name. • "Valid" - true or false - indicates whether the result is correct. False indicates an error. The output is intended to be displayed in the results area.

	“math” - mathematical operations with two operands.		
Parameters:	num1	double	Default value for the Num1 input.
	num2	double	Default value for the Num2 input.
	operation	string["add"]	Mathematical operation. Possible values: • add (+) - addition. Num = Num1+ Num2 • sub (-) - subtraction. Num = Num1 - Num2 • div (/) - division. Num = Num1 / Num2 • mult (*) - multiplication. Num = Num1 * Num2 • min (minimum) - minimum value. Num = min(Num1 , Num2) • max (maximum) - maximum value. Num = max(Num1 , Num2) • avg (average) - average value. Num = (Num1 + Num2)*0.5
Inputs:	Num1	double	Operand 1. When the input is disabled, the calculation uses the default value (parameter num1).
	Num2	double	Operand 2. When the input is disabled, the calculation uses the default value (parameter num2).
Outputs:	Num	double	The X coordinate of the left point in the 3D coordinate system of the external device.

 data direction switcher	"data direction switcher" - redirecting information from the i-th input to the j-th output of the block. Allows for switching between different parts of the scheme involved in the measurement.		
Inputs:	ActiveInput	NumberInt8	Active input number (Inp). Takes values from 1 to N, where N is the specified number of inputs (countInputs).
	ActiveOutput	NumberInt8	Active output number (Out). Takes values from 1 to N, where N is the specified number of outputs (countOutputs).
	Inp1 ... InpN	All	N inputs. Automatically created based on the countInputs parameter.
Outputs:	Out1 ... OutN	All	N outputs. Automatically created based on the countOutputs parameter.
Parameters:	activeInput	uint8_t, 0...255, [1]	Active input number (Inp). This value is used until another number arrives at the ActiveInput port. Possible values: from 1 to N, where N is the specified number of inputs (countInputs).
	activeOutput	uint8_t, 0...255, [1]	Active output number (Out). This value is used until another number arrives at the ActiveOutput port. Possible values: from 1 to N, where N is the specified number of outputs (countOutputs).
	countInputs	uint8_t, 0...255, [1]	Number of automatically generated input ports (Inp1…InpN).
	countOutputs	uint8_t, 0...255, [1]	Number of automatically generated output ports (Out1…OutN).
	isAsync	bool,[true]	Flag indicating whether the output process is asynchronous.

 Scalar Statistic	"Scalar statistic" - calculating statistical parameters from input scalar values. It allows computation of various sample characteristics, including mean values, extrema, mode, range, standard deviation, and variance, with the option to exclude outliers from the calculation. The resulting values are generated and updated at the corresponding outputs each time a new value is received at the Scalar input. If the *notifyOnlyRequest* parameter is enabled, results are updated only upon request (when a signal is received at the Request input). Outputs with corresponding statistical results (e.g., Mean, Median, Range, etc.) are created and activated only if the corresponding calculation option is enabled in the block parameters. If the option is disabled, there is no corresponding output.		
Parameters:	calculateMax	bool,[true]	Calculate the maximum value.
	calculateMaxExcludingOutliers	bool,[fasle]	Calculate the maximum value excluding outliers.
	calculateMean	bool,[true]	Calculate the arithmetic mean.
	calculateMeanExcludingOutliers	bool,[false]	Calculate the arithmetic mean excluding outliers.
	calculateMedian	bool,[false]	Calculate the median.
	calculateMin	bool,[true]	Calculate the minimum value.
	calculateMinExcludingOutliers	bool,[false]	Calculate the minimum value excluding outliers.
	calculateModa	bool,[false]	Calculate the mode (most frequently occurring value).
	calculateRange	bool,[false]	Calculate the range (difference between maximum and minimum values).
	calculateRangeExcludingOutliers	bool,[false]	Calculate the range excluding outliers.
	calculateStandardDeviation	bool,[false]	Calculate the standard deviation.

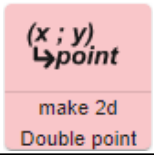
	calculateVariance	bool,[false]	Calculate the variance.
	calculateVarianceExcludingOutliers	bool,[false]	Calculate the variance excluding outliers.
	notifyOnlyRequest	bool,[false]	If enabled, calculation results are updated only upon request (when a signal is received at the Request input).
Inputs:	“Scalar”	double	Input scalar value for statistical processing.
	"Reset"	* ()	Filter reset signal. Clears the accumulated sample.
	"Request"	* ()	Signal to force result update (relevant when the notifyOnlyRequest parameter is enabled).
Outputs:	Max	double	Maximum value.
	MaxFiltered	double	Maximum value with outlier exclusion.
	Mean	double	Arithmetic mean.
	MeanFiltered	double	Arithmetic mean with outlier exclusion.
	Median	double	Median.
	Min	double	Minimum value.
	MinFiltered	double	Minimum value with outlier exclusion.
	Moda	double	Mode.
	Range	double	Range.
	RangeFiltered	double	Range with outlier exclusion.
	StandardDeviation	double	Standard deviation.
	Variance	double	Variance.
	VarianceFiltered	double	Variance with outlier exclusion.

	"simplification" - contour (profile) simplification using various point reduction algorithms. This smart block makes it possible to reduce the number of points in a profile while preserving its basic shape, facilitating subsequent processing and analysis.		
Parameters:	“method”	string enum, ["Opheim"]	Contour simplification algorithm: • NthPoint - selects every n-th point; • RadialDistance - removes points located closer than the specified radius; • PerpendicularDistance - removes points based on perpendicular deviation; • ReumannWitkam - rectangular window method; • Opheim - double-threshold method; • Lang - threshold and step-based method; • DouglasPeucker - classic Douglas-Peucker algorithm; • DouglasPeuckerVariant - modified Douglas-Peucker algorithm.
	roi	Rect	Measurement area - ROI. Defined by the following parameters: left top x, left top y, width, height.
	Parameters of the NthPoint method		
	n	uint32_t,[3]	Point selection step. Every n-th point of the original profile is retained in the resulting contour, and the rest are deleted.
	RadialDistance parameters		
	tolerance	double	Points closer to each other than the specified value are considered redundant and are removed.
	PerpendicularDistance parameters		
	tolerance	double	The maximum permissible perpendicular deviation of a point from the line between adjacent reference points. Points with smaller deviations are removed.

	n	uint32_t,[3]	The minimum number of points that must remain in the contour after simplification.
	ReumannWitkam parameters		
	tolerance	double	The width of the rectangular window. All points within the window are considered collinear and can be removed.
	Opheim parameters		
	toleranceMin	double	The minimum distance threshold between points, below which points are considered redundant.
	toleranceMax	double	The maximum distance threshold, above which points are necessarily retained in the contour.
	Lang parameters		
	tolerance	double	The maximum permissible deviation of a point from the approximating line. Points with smaller deviations are removed.
	n	uint32_t,[3]	The minimum number of points that must remain in the contour.
	DouglasPeucker parameters		
	tolerance	double	The maximum distance from a point to the approximating line. Points deviating less than this value are removed.
	DouglasPeuckerVariant parameters		
	n	uint32_t,[3]	The minimum number of points that must remain in the contour after simplification.
Inputs:	“InpProfile”	Profile	Input profile (contour) to be processed.
	InpRoi	Rect	Measurement area. When the input is disabled, the default value from the "roi" parameter is used in the calculation.
Outputs:	“OutProfile”	double	Simplified (reduced) profile.

	"Delay" - delaying the transmission of an input message for a specified period of time.		
Parameters:	“delayMks”	uint64_t, [5000]	The delay time in microseconds between the arrival of an input message and its transmission to the output.
Inputs:	“Inp”	* ()	Input message.
Outputs:	“Out”	* ()	The message transmitted to the output with a delay specified by the delayMks parameter.

11.4.2.2.5. "Converters" section

	“make 2d double point” - making a 2D point based on its X and Y coordinates.		
Inputs:	“X”	double	The X coordinate of the point.
	“Y”	double	The Y coordinate of the point.
Outputs:	“Point”	Point2dDouble	The resulting point on the plane.

point ↳(x ; y) split point	“split point” - splitting a 2D point into its X and Y coordinates.		
Inputs:	“Point”	Point2dDouble	Point.
Outputs:	“X”	double	The X coordinate of the point.
	“Y”	double	The Y coordinate of the point.

p1,p2 ↳line line from 2 points	“line from 2 points” - creating a straight line or segment based on two 2D points.		
Inputs:	Point1	Point2dDouble	The first point of a line on a plane.
	Point2	Point2dDouble	The second point of a line on a plane.
Outputs:	Line	StraightLine, SegmentLine	The resulting line/segment.
Parameters:	lineType	string,[Straight]	Line type. Possible values: • Segment - straight line segment. • Straight - straight line.

pnt,ang ↳line line through point	“line through point” - creating a straight line or segment based on a 2D point and angle of inclination.		
Inputs:	Point	Point2dDouble	A point belonging to a line on a plane.
	Angle	double	The angle in radians that defines the inclination of the line to the X axis.
Outputs:	Line	StraightLine,	The resulting line.
Parameters:	angle	double,[0]	The original angle, in radians, used to create the line. It is used if no new angle values have been received at the Angle input.

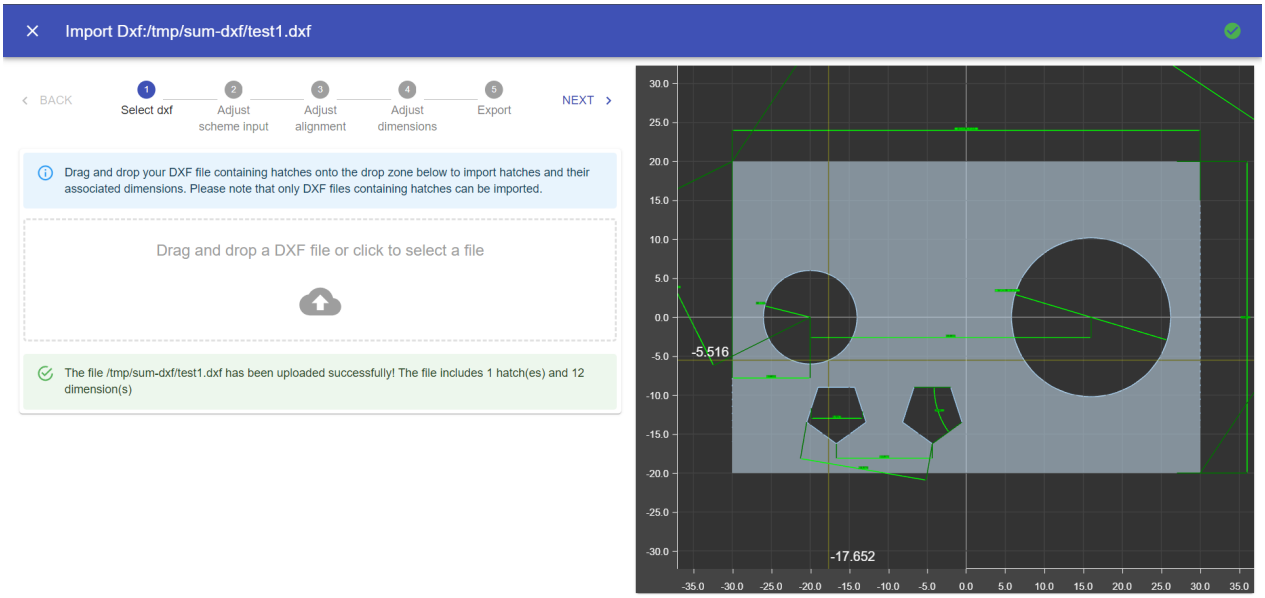
line ↳point point on line	“point on line” - finding a point on a line with a given X or Y coordinate.		
Inputs	Line	StraightLine, SegmentLine	Input line.
Outputs:	Point	Point2dDouble	The resulting point belonging to a line on a plane.
Parameters:	coordinateType	string enum, ["x"]	The axis of the coordinate system, the value on which will be specified: • x - X axis. • y - Y axis.
	coordinateValue	double,[0]	The value of the coordinate from which it is necessary to get a point belonging to the input line.

seg ↳p1,p2 split segment line	“split segment line” - obtaining points lying at the ends of a segment.		
Inputs:	Line	SegmentLine	Input segment.

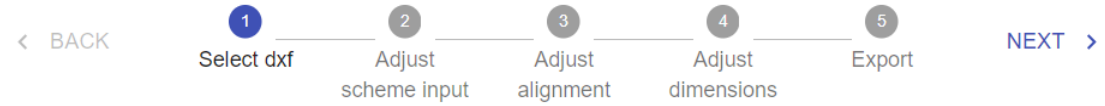
Outputs:	Point1	Point2dDouble	The first point of a segment on a plane.
	Point1	Point2dDouble	The second point of a segment on a plane.

11.4.3. DXF scheme builder

The measurement scheme can be created automatically based on the DXF file of the measured object. For this purpose, the web interface has a special builder. To open it, click the **DXF** button  located at the top of the scheme building area. The DXF scheme builder is displayed in a dialog box on top of the main page of the web application.



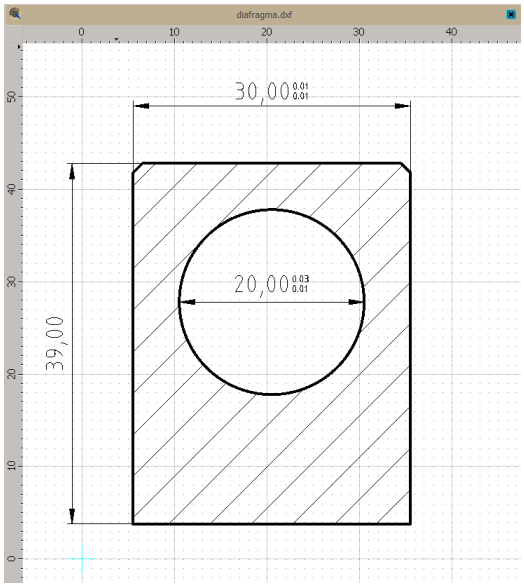
The procedure includes the following steps:

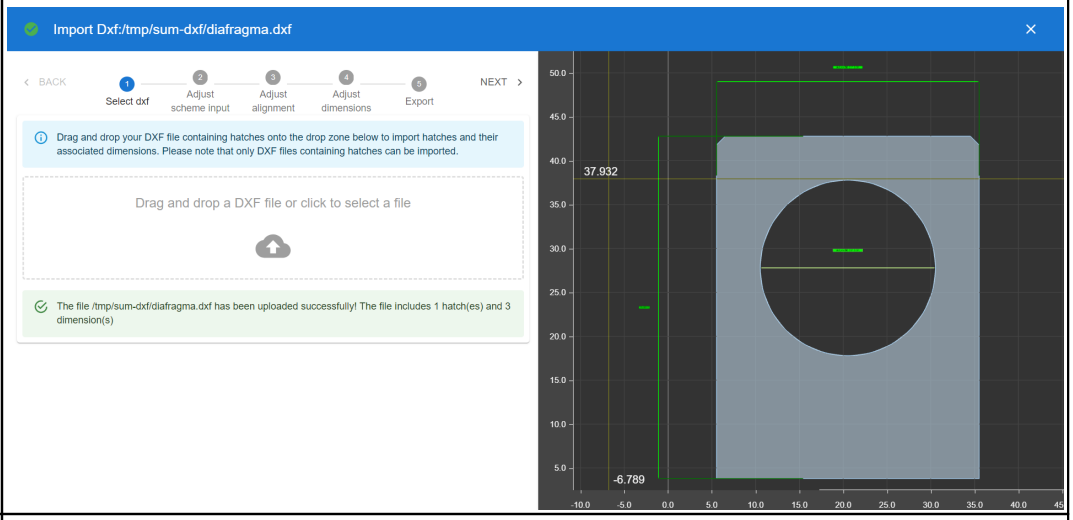
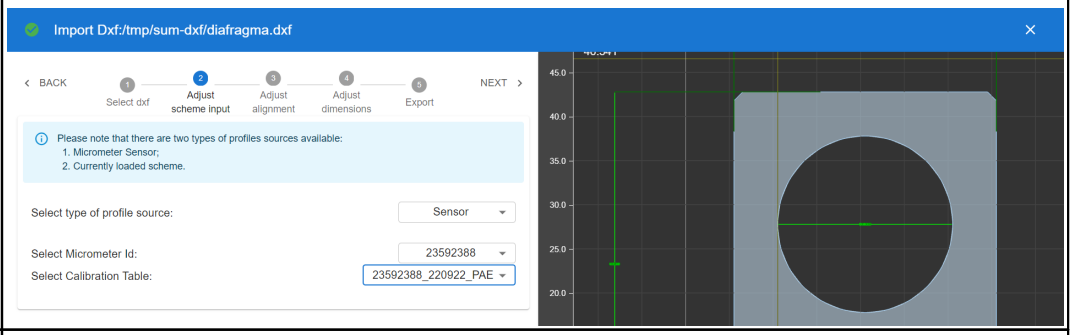
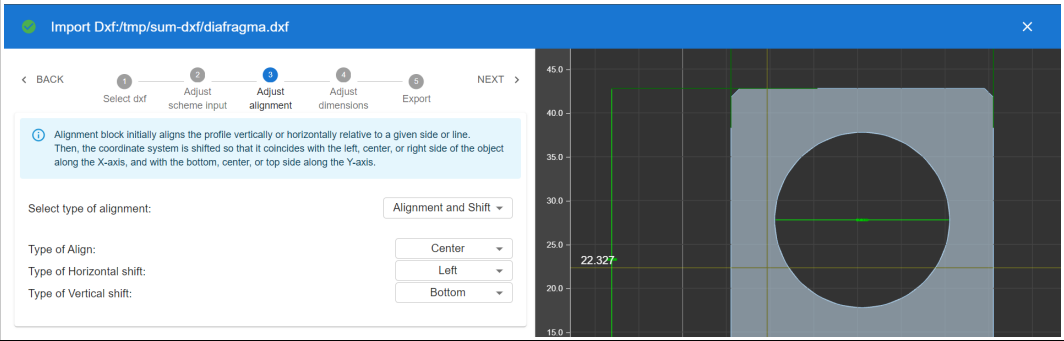
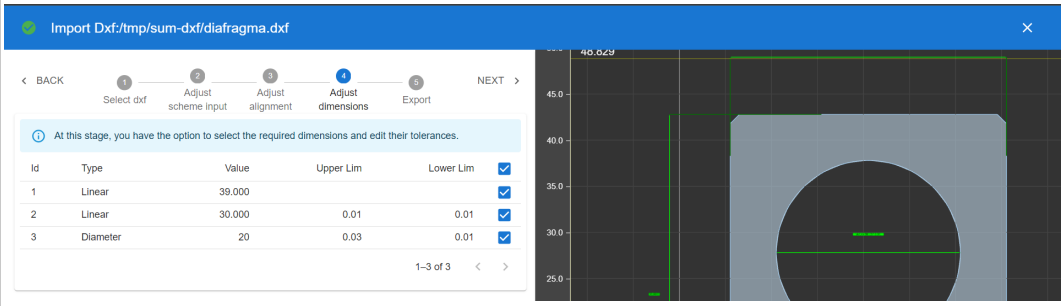
Step	Description
	
Select dxf	Uploading a CAD file to the scheme builder. Uploading is available in drag and drop mode, or using a dialog box. First you need to prepare the DXF file. The file must contain a drawing of the part being measured, as well as the necessary dimensions (dimension lines) and tolerances. It is important that the part drawing is presented as a hatched figure (Hatch). After uploading the DXF file, a message will be displayed about the number of hatch objects found and their dimensions, for example: "The file /tmp/sum-dxf/test1.dxf has been uploaded successfully! The file includes 1 hatch(es) and 12 dimension(s)".
Adjust scheme input	Configuring the source of profiles for the scheme. There are two types of sources available: Sensor and Template Scheme. The Sensors type is used if a sensor must be specified as a profile source. In this case, any connected sensor is available for selection. The Template Scheme type is used when you have some kind of basic scheme and you need to use any of the outputs of this scheme as a source. This scheme must be uploaded as current at the time the builder is launched, and the block and its output port must be selected by their Id as the profile source.
Adjust alignment	Configuring the binding of measured profiles to the reference one. Correction of tilt and transfer of coordinate system.
Adjust dimensions	Selecting and setting the required dimensions and tolerances. To set dimensions and tolerances, a table is provided, each row of which corresponds to one dimension in the drawing. The table contains the following fields:

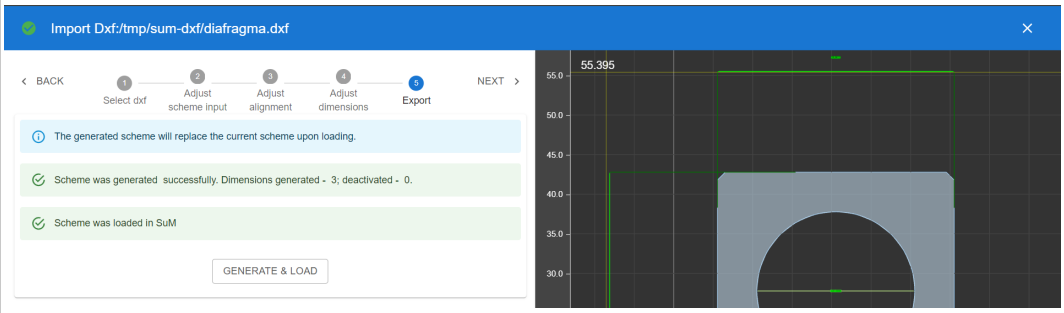
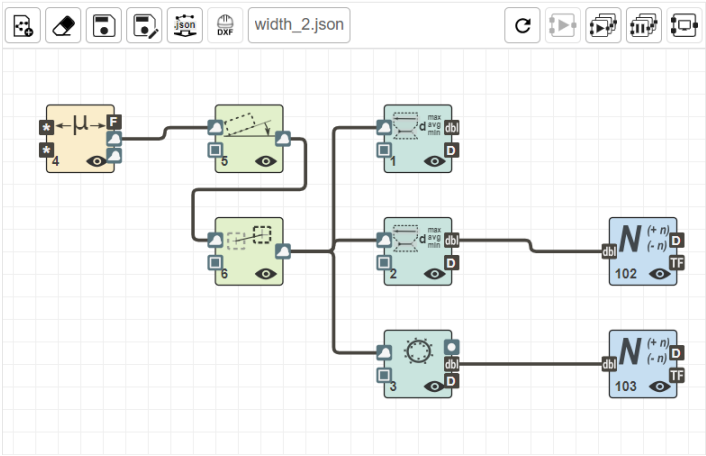
Step	Description
	• *ID* (sequence number): each row of the table has a unique sequence number to identify the dimension. • *Type* (dimension type): the type of parameter being measured, such as length, width, diameter, etc. • *Value* (nominal value): the specified or required value of the dimension. • *Upper Lim* (upper deviation from the nominal value): the maximum permissible upper deviation of the dimension from the nominal value. • *Lower Lim* (lower deviation from the nominal value): the maximum permissible lower deviation of the dimension from the nominal value. • *Checkbox* (flag): allows you to include or exclude the dimension from the measurement scheme. When you select a row in the table, the corresponding dimension is highlighted in yellow. The dimension can also be selected directly from the drawing. Upper Lim and Lower Lim values can be added, deleted or edited in the table. If a dimension is excluded from the scheme, it will be displayed in gray in the drawing.
Export	Generating the measurement scheme and exporting it to the current measurement scheme. To generate a scheme and upload it later, click the **Generate & Load** button. In this case, all blocks of the previously uploaded scheme will be deleted, and blocks of the new scheme will be displayed instead. After generating the scheme, three types of messages are possible: 1. Success. Scheme was generated successfully. Dimensions generated - ${countGenerated}; deactivated - ${countDeactivated}. 2. Warning. Scheme was generated with errors. Dimensions generated - ${countGenerated}; Skipped - ${countSkipped} dimensions. 3. Error. Scheme wasn`t generated. ${MakeSchemeErrorMessage}. After loading the scheme, three types of messages are possible: 1. Success. Scheme was loaded in SuM. 2. Error. Scheme wasn`t loaded in SuM. ${LoadSchemeErrorMessage}.

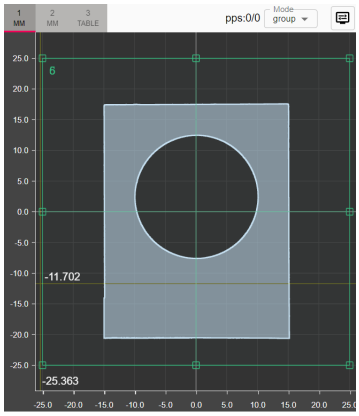
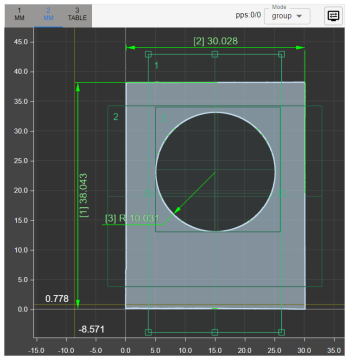
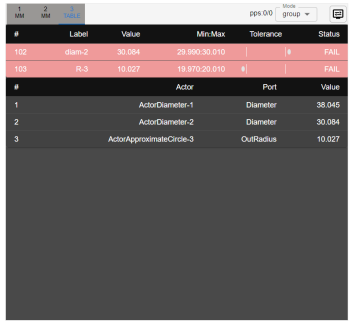
Id	Type	Value	Upper Lim	Lower Lim	
1	Linear	60	0.1	0.3	☑
2	Linear	40			☑
3	Diameter	20.396	0.05	0.1	☑

An example of creating the measurement scheme:

Step	Description
Select dxf	CAD file. This file contains a hatched figure and three dimensions. 

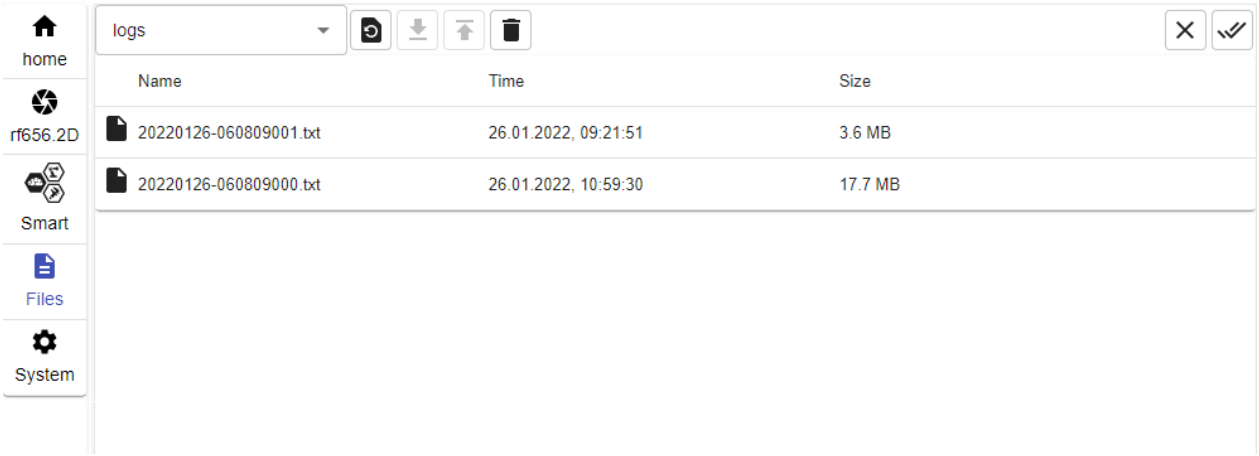
Step	Description
	Once the DXF file is uploaded, a message appears indicating that one hatch object and three associated dimensions have been uploaded successfully. The uploaded drawing will be displayed. 
Adjust scheme input	The sensor with id: 23592388 and calibration table 23592388_220922_PAE was selected as a source of profiles. 
Adjust alignment	The alignment of the slope of the measured profiles along the center line (Type of Align: Center). The origin of the coordinate system is bound to the measured profiles as follows: x - determined by the leftmost point of the profile (Type of Horizontal shift: Left); y - determined by the lowest point of the profile (Type of Vertical shift: Bottom). 
Adjust dimensions	Selecting and setting the required dimensions and tolerances. To set dimensions and tolerances, a table is provided in which each row corresponds to one dimension in the drawing. The table contains the following fields: 

Step	Description
	When you select a row in the table, the corresponding dimension is highlighted in yellow. The dimension can also be selected directly from the drawing. Upper Lim and Lower Lim values can be added, deleted or edited in the table. If a dimension is excluded from the scheme, it will be displayed in gray in the drawing.
Export	After generating the scheme, two messages were received: 1. "Scheme was generated successfully. Dimensions generated - 3; deactivated - 0". This means that the required measurement blocks have been selected for all dimensions. 2. "Scheme was loaded in SuM". This means that the blocks of the current scheme have been replaced with newly generated ones.  The resulting scheme is created automatically and looks like this:  Blocks 1,2,3 correspond to the dimension numbers in the table at the **Adjust dimensions** step. Regions of interest (ROI) are automatically set for each block. Blocks 102 and 103 are "tolerance" blocks and are designed to check values for compliance with a given range. In this example, tolerances were specified for dimensions 2 and 3. Block 4 - the “Micrometer” block for working with an optical micrometer sensor. It is already configured to work with a specific sensor 23592388 and the corresponding calibration table. Block 5 - the "align compensate" block is designed to eliminate the tilt of the profile along a given edge or center line, as in this case. Block 6 - the "shift compensate" block allows you to make a parallel shift of the coordinate system relative to a given position. This example uses the leftmost and bottom points of the profile to determine this position. The generated scheme is automatically configured to show three displays: 1. The first display shows the original profile from the sensor, as well as regions of interest (ROI) from blocks 5 and 6, which are responsible for transforming the coordinate system. 2. The second display shows the aligned and shifted profile from the output of block 6, as well as all dimension lines (ResultDescription) and regions of interest (ROI) from blocks 1, 2 and 3. 3. The third display shows a table with the tolerance check results from blocks 102 and 103, as well as the numeric values from blocks 1, 2 and 3.

Step	Description
	 Display 1  Display 2  Display 3

11.5. "Files" tab

This tab provides a simplified file browser interface for manipulating dump, log, and calibration table files.



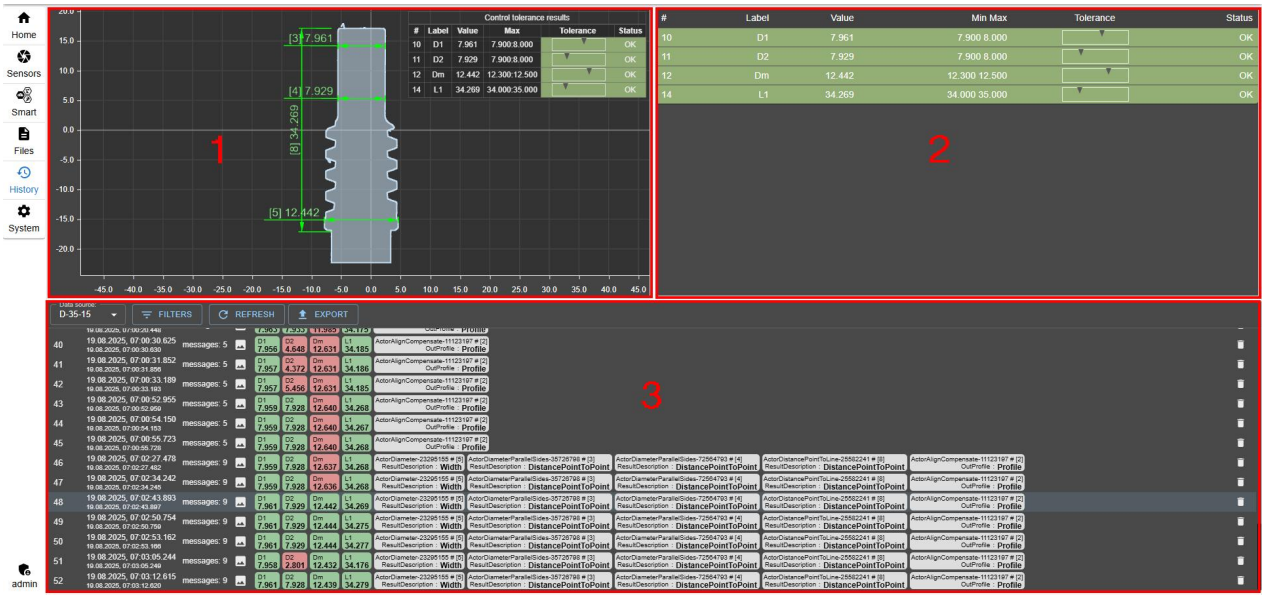
To create, save, load calculation schemes and perform other actions, use the corresponding buttons located in the upper part of the tab:

Element	Description
schemes	Displayed directory. The following directories are available in the drop-down list: • logs - log files, • dumps - dump files, • tmp_dumps - temporary dump files, • schemes - calculation schemes, • tables - calibration tables.
	Refreshing the list of displayed files and directories.
	Downloading selected files/directories from the controller and saving them on the user's computer.
	Uploading files from the computer to the micrometer controller.
	Deleting selected files/directories on the controller.

Element	Description
	Deselecting all files.
	Selecting all files/directories available in the directory.

11.6. "History" tab

This tab is designed for working with measurement results stored in the **Redis** database using the **"database logger"** block. It provides tools for viewing and visualizing data, deleting selected records, as well as generating and exporting reports.



The **History** tab contains three main areas:

1. Profile and measurement visualization area. Displays a graphical representation of the profile and its associated measurement results. The visualization is similar to the "2D mm" display.
2. Scalar values visualization area (table view). Displays scalar measurement results and verification results showing whether these values are within the specified range. The visualization is similar to the "Table" display.
3. Measurement history visualization area. Each row corresponds to a unique profile identifier (id) and aggregates all measurement results related to that profile. Thus, all records with the same id are shown in a single row. The visualization is similar to the "Monitoring" display.

Selecting a row in the history area (area 3) displays detailed information for the selected profile in areas 1 and 2, provided the corresponding data are available in the database.

For convenient work with large datasets, the history area includes a filtering system.

In area 3, a set of filters is implemented, allowing the displayed records to be limited according to various criteria:

- "created on" – filtering by the record creation date and time ("is after", "is before").
- By data source name (smart block/port):
  - contains [value] – finds all records where the value associated with the selected data source (smart block/port) partially matches the specified value;

- does not contain [value] – excludes all records where the value associated with the selected data source partially matches the specified value;
- equals [value] – finds only the records where the value associated with the selected data source exactly matches the specified value;
- does not equal [value] – excludes records where the value associated with the selected data source exactly matches the specified value;
- more than [value] – finds all records where the numeric value associated with the selected data source is greater than the specified value;
- less than [value] – finds all records where the numeric value associated with the selected data source is less than the specified value;
- included – finds all records that contain a value from the selected data source;
- excluded – finds all records that do not contain a value from the selected data source.

Filters can be combined using the "and" and "or" operators to create complex data selection conditions.

11.6.1. Generating reports

The results displayed in the measurement history visualization area can be exported to XLS and PDF formats.

For XLS export, two modes are available: "simple" and "detail".

	A	B	C	D	E	F	G
1	Id	Write time	Sender	Value	leftB	rightB	Tolerance
2	1753083552589106	21.07.2025, 07:39:12.731	D1-tol	97,447	98,3	98,7	NG
3	1753083552589106	21.07.2025, 07:39:12.731	D2-tol	92,89	93,7	94,1	NG
4	1753083552589106	21.07.2025, 07:39:12.802	H1(A)-tol	20,059	19,9	20,1	OK
5	1753083552589106	21.07.2025, 07:39:12.782	H2(A)-tol	10,012	9,9	10,1	OK
6	1753083552589106	21.07.2025, 07:39:12.732	H3-tol	10,192	10	10,4	OK
7							
8	1753083556031668	21.07.2025, 07:39:16.147	D1-tol	97,447	98,3	98,7	NG
9	1753083556031668	21.07.2025, 07:39:16.147	D2-tol	92,89	93,7	94,1	NG
10	1753083556031668	21.07.2025, 07:39:16.218	H1(A)-tol	20,06	19,9	20,1	OK
11	1753083556031668	21.07.2025, 07:39:16.198	H2(A)-tol	10,013	9,9	10,1	OK
12	1753083556031668	21.07.2025, 07:39:16.148	H3-tol	10,192	10	10,4	OK
13							
14	1753083559653851	21.07.2025, 07:39:19.781	D1-tol	97,447	98,3	98,7	NG
15	1753083559653851	21.07.2025, 07:39:19.781	D2-tol	92,89	93,7	94,1	NG
16	1753083559653851	21.07.2025, 07:39:19.852	H1(A)-tol	20,059	19,9	20,1	OK
17	1753083559653851	21.07.2025, 07:39:19.832	H2(A)-tol	10,013	9,9	10,1	OK
18	1753083559653851	21.07.2025, 07:39:19.781	H3-tol	10,192	10	10,4	OK
19							
20	1753083809617556	21.07.2025, 07:43:29.743	D1-tol	97,447	98,3	98,7	NG
21	1753083809617556	21.07.2025, 07:43:29.743	D2-tol	92,89	93,7	94,1	NG
22	1753083809617556	21.07.2025, 07:43:29.813	H1(A)-tol	20,059	19,9	20,1	OK
23	1753083809617556	21.07.2025, 07:43:29.793	H2(A)-tol	10,012	9,9	10,1	OK
24	1753083809617556	21.07.2025, 07:43:29.743	H3-tol	10,193	10	10,4	OK
25							
26	1753083850994450	21.07.2025, 07:44:11.116	D1-tol	97,447	98,3	98,7	NG
27	1753083850994450	21.07.2025, 07:44:11.116	D2-tol	92,89	93,7	94,1	NG
28	1753083850994450	21.07.2025, 07:44:11.187	H1(A)-tol	20,059	19,9	20,1	OK
29	1753083850994450	21.07.2025, 07:44:11.167	H2(A)-tol	10,012	9,9	10,1	OK
30	1753083850994450	21.07.2025, 07:44:11.117	H3-tol	10,193	10	10,4	OK

XLS > simple

In "simple" mode, a report is generated containing all measurements displayed in the visualization area. All measurement results related to one profile are written sequentially in rows. After the profile changes, a blank line is added, followed by the measurements of the next profile.

	A	B	C	D	E	F	G	H	I	J	K	L
1	protocolName	N-123										
2	objectName	Disk										
3	count	#1 - #34										
4	checkingProjectName	-										
5	checkProjectNumber	-										
6	operator	Petrov										
7	companyName	MIA										
8												
9	names	D1-tol		D2-tol		H1(A)-tol		H2(A)-tol		H3-tol		
10	nominal	99		94		20		10		10		
11	maxUpperDifference	-0,3		0,1		0,1		0,1		0,4		
12	maxLowerDifference	-0,7		-0,3		-0,1		-0,1		0		
13	maxUpperValue	98,7		94,1		20,1		10,1		10,4		
14	maxLowerValue	98,3		93,7		19,9		9,9		10		
15	maxMeasuredValue	98,46		93,846		20,061		10,048		10,193		
16	minMeasuredValue	97,425		92,854		20,058		10,012		10,124		
17	range	1,035		0,992		0,003		0,036		0,068		
18	arithmeticMean	98,007		93,417		8,85		4,422		10,166		
19	#1	97,447	-1,553	92,89	-1,11	20,059	0,059	10,012	0,012	10,192	0,192	NG
20	#2	97,447	-1,553	92,89	-1,11	20,06	0,06	10,013	0,013	10,192	0,192	NG
21	#3	97,447	-1,553	92,89	-1,11	20,059	0,059	10,013	0,013	10,192	0,192	NG
22	#4	97,447	-1,553	92,89	-1,11	20,059	0,059	10,012	0,012	10,193	0,193	NG
23	#5	97,447	-1,553	92,89	-1,11	20,059	0,059	10,012	0,012	10,193	0,193	NG
24	#6	97,44	-1,56	92,875	-1,125	20,058	0,058	10,023	0,023	10,19	0,19	NG
25	#7	97,444	-1,556	92,878	-1,122	20,059	0,059	10,024	0,024	10,189	0,189	NG
26	#8	97,444	-1,556	92,877	-1,123	20,059	0,059	10,024	0,024	10,19	0,19	NG
27	#9	97,444	-1,556	92,878	-1,122	20,059	0,059	10,048	0,048	10,192	0,192	NG
28	#10	97,445	-1,555	92,877	-1,123	20,059	0,059	10,031	0,031	10,19	0,19	NG
29	#11	97,445	-1,555	92,889	-1,111	20,061	0,061	10,031	0,031	10,19	0,19	NG
30	#12	97,435	-1,565	92,88	-1,12	20,059	0,059	10,024	0,024	10,19	0,19	NG
31	#13	97,435	-1,565	92,878	-1,122	20,059	0,059	10,024	0,024	10,191	0,191	NG
32	#14	97,435	-1,565	92,88	-1,12	20,06	0,06	10,024	0,024	10,191	0,191	NG
33	#15	97,425	-1,575	92,854	-1,146	20,061	0,061	10,024	0,024	10,188	0,188	NG
34	#16	98,452	-0,548	93,837	-0,163	-	-	-	-	10,148	0,148	OK
35	#17	98,452	-0,548	93,836	-0,164	-	-	-	-	10,149	0,149	OK

XLS > detail

In "detail" mode, a summarized report is created where each measured parameter is represented as a separate column, and each new measurement as a separate row. The report also includes summary characteristics such as maxMeasuredValue, minMeasuredValue, range, arithmeticMean, as well as parameters from the "tolerance" block: maxUpperDifference, maxLowerDifference, maxUpperValue, and maxLowerValue.

For PDF export, two modes are available: "all currently visible frames" and "selected frame".

RIFTEK
SENSORS & INSTRUMENTS

operator: Petrov

Time	BlockId	Source	Result
21.07.2025, 07:39:12.731	D1-tol	NG(D1-tol)	97.447
21.07.2025, 07:39:12.731	D2-tol	NG(D2-tol)	92.890
21.07.2025, 07:39:12.802	H1(A)-tol	OK(H1(A)-tol)	20.059
21.07.2025, 07:39:12.782	H2(A)-tol	OK(H2(A)-tol)	10.012
21.07.2025, 07:39:12.732	H3-tol	OK(H3-tol)	10.192

Time	BlockId	Source	Result
21.07.2025, 07:39:16.147	D1-tol	NG(D1-tol)	97.447
21.07.2025, 07:39:16.147	D2-tol	NG(D2-tol)	92.890
21.07.2025, 07:39:16.218	H1(A)-tol	OK(H1(A)-tol)	20.060
21.07.2025, 07:39:16.198	H2(A)-tol	OK(H2(A)-tol)	10.013
21.07.2025, 07:39:16.148	H3-tol	OK(H3-tol)	10.192

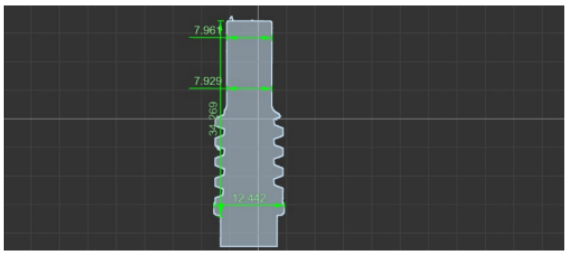
Time	BlockId	Source	Result
21.07.2025, 07:39:19.781	D1-tol	NG(D1-tol)	97.447
21.07.2025, 07:39:19.781	D2-tol	NG(D2-tol)	92.890
21.07.2025, 07:39:19.852	H1(A)-tol	OK(H1(A)-tol)	20.059
21.07.2025, 07:39:19.832	H2(A)-tol	OK(H2(A)-tol)	10.013
21.07.2025, 07:39:19.781	H3-tol	OK(H3-tol)	10.192

Time	BlockId	Source	Result
21.07.2025, 07:43:29.743	D1-tol	NG(D1-tol)	97.447
21.07.2025, 07:43:29.743	D2-tol	NG(D2-tol)	92.890
21.07.2025, 07:43:29.813	H1(A)-tol	OK(H1(A)-tol)	20.059
21.07.2025, 07:43:29.793	H2(A)-tol	OK(H2(A)-tol)	10.012
21.07.2025, 07:43:29.743	H3-tol	OK(H3-tol)	10.193

PDF > all currently visible frames
In "all currently visible frames" mode, a report is generated with all measurements displayed in the visualization area. All measurement results related to one profile are written sequentially in rows. After the profile changes, a blank line is added, followed by the measurements of the next profile.

RIFTEK
SENSORS & INSTRUMENTS

operator: Petrov



Time	BlockId	Source	Result
19.08.2025, 07:02:43.893	OutProfile	ActorAlignCompensate-11123197	Profile
19.08.2025, 07:02:43.894	D2	OK(D2)	7.929
19.08.2025, 07:02:43.894	ResultDescription	ActorDiameterParallelSides-72564793	7.929
19.08.2025, 07:02:43.895	D1	OK(D1)	7.961
19.08.2025, 07:02:43.895	ResultDescription	ActorDiameterParallelSides-35726798	7.961
19.08.2025, 07:02:43.897	Dm	OK(Dm)	12.442
19.08.2025, 07:02:43.897	ResultDescription	ActorDistancePointToLine-25582241	34.269
19.08.2025, 07:02:43.897	L1	OK(L1)	34.269
19.08.2025, 07:02:43.897	ResultDescription	ActorDiameter-23295155	12.442

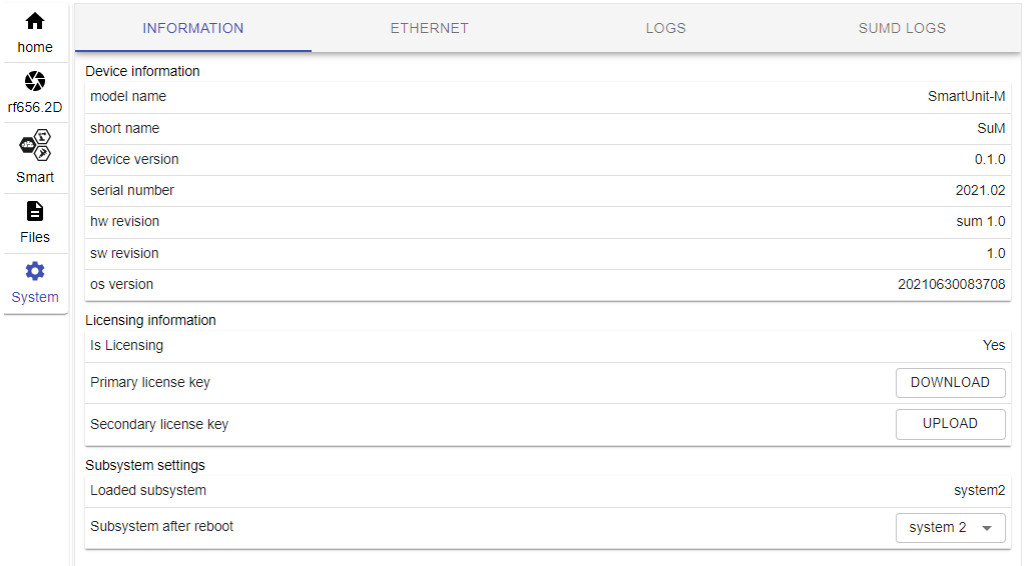
PDF > selected frame
In "selected frame" mode, a report is created for one selected record from the measurement visualization area. The report also includes a screenshot from the profile visualization area along with its measurement results. Before generating the report, the user can zoom or adjust the view of the part as needed.

11.7. "System" tab

11.7.1. "Information" section

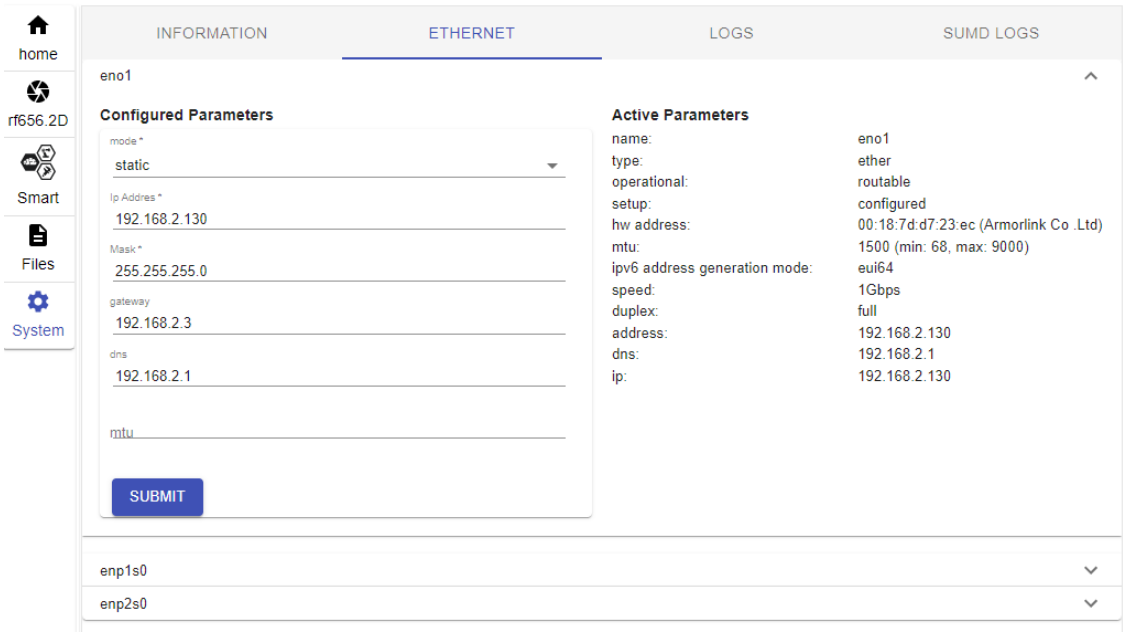
The **Information** section is intended for:

- providing general information about the controller,
- displaying the status and managing the license,
- managing the loaded system.



11.7.2. "Ethernet" section

The **Ethernet** section is designed to display the status and configure the network interfaces of the controller.



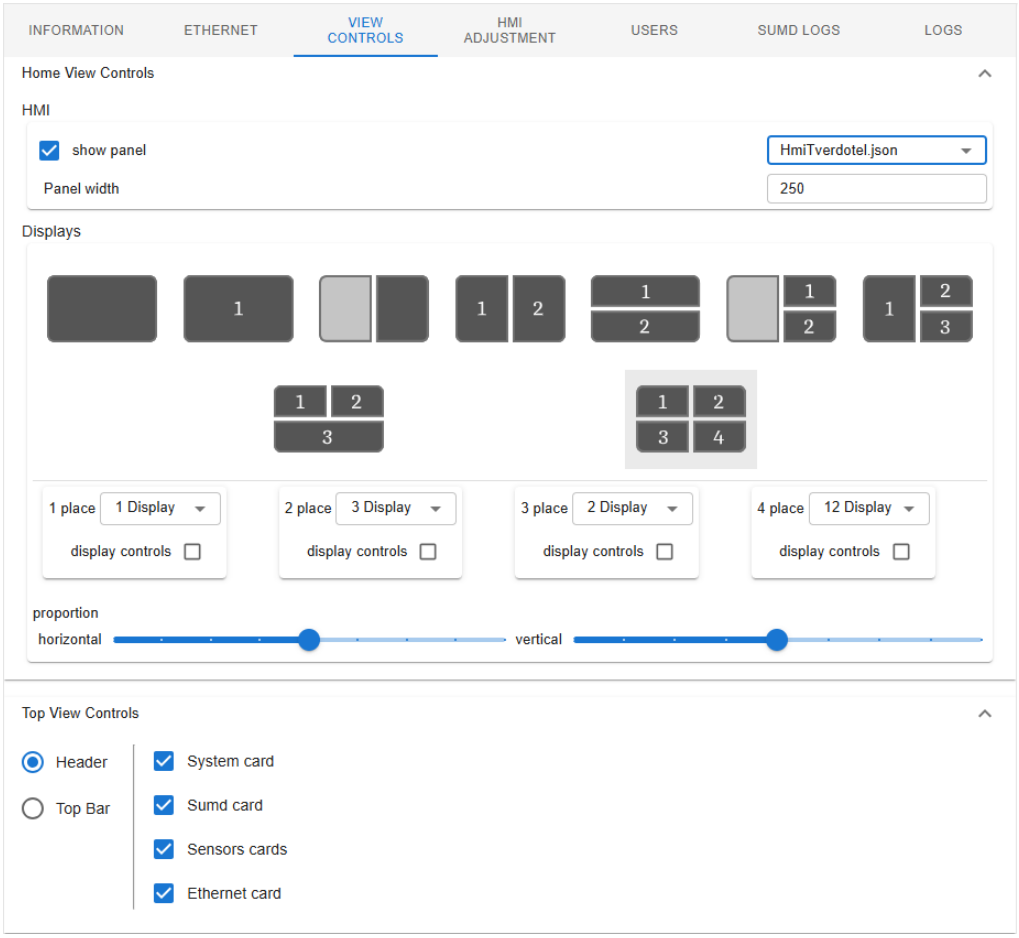
Configurable parameters for each network interface:

Parameter	Default value		Description
mode*	static		static - a static address specified manually is assigned. dhcp - an IPv4 or IPv6 address is dynamically assigned if there is a dhcp server on the network. dhcp.ipv4 - an IPv4 address is dynamically assigned if there is a DHCPv4 server on the network. dhcp.ipv6 - an IPv6 address is dynamically assigned if there is a DHCPv6 server on the network.
IP Address*	eno1	192.168.2.130	Controller IP address. Only for mode:static.
	enp1s0	192.168.1.130	
	enp2s0	192.168.3.130	
mask*	255.255.255.0		Subnet mask. Only for mode:static.
gateway	eno1	192.168.2.1	Gateway network address. Only for mode:static. The parameter is optional.
	enp1s0	192.168.1.1	
	enp2s0	192.168.3.1	
dns	eno1	192.168.2.1	DNS network address. Only for mode:static. The parameter is optional.
	enp1s0	192.168.1.1	
	enp2s0	192.168.3.1	
mtu	-		The maximum packet size that can be sent over the network without fragmentation.

For the changes to take effect, it is necessary to click the **Apply** button.

11.7.3. "View Controls" section

The **View Controls** section is designed to configure information display modes in the **Home** tab, as well as in the area of status indicators.

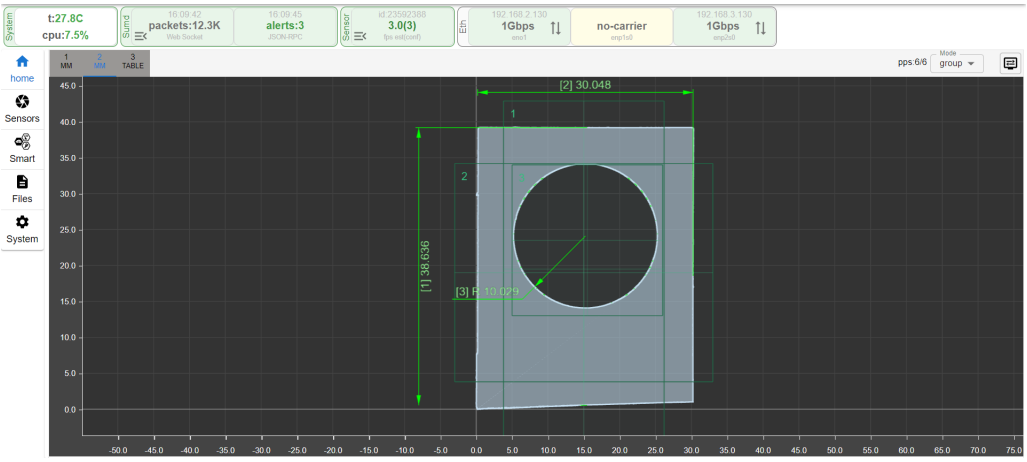
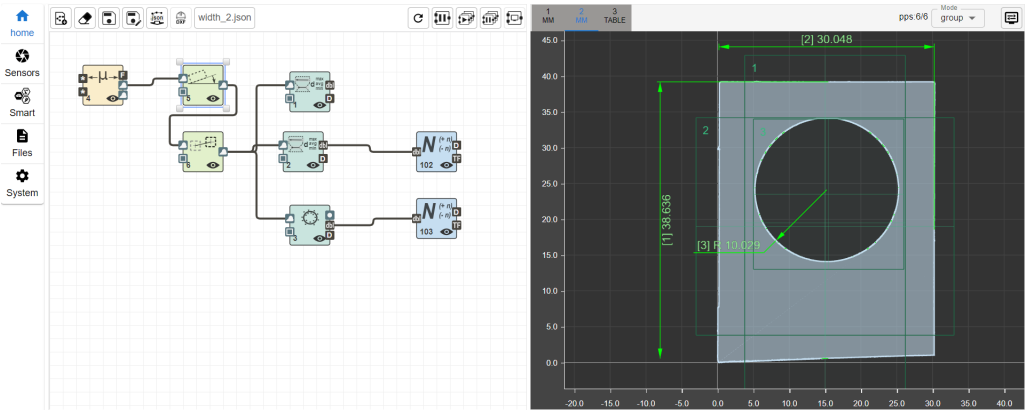
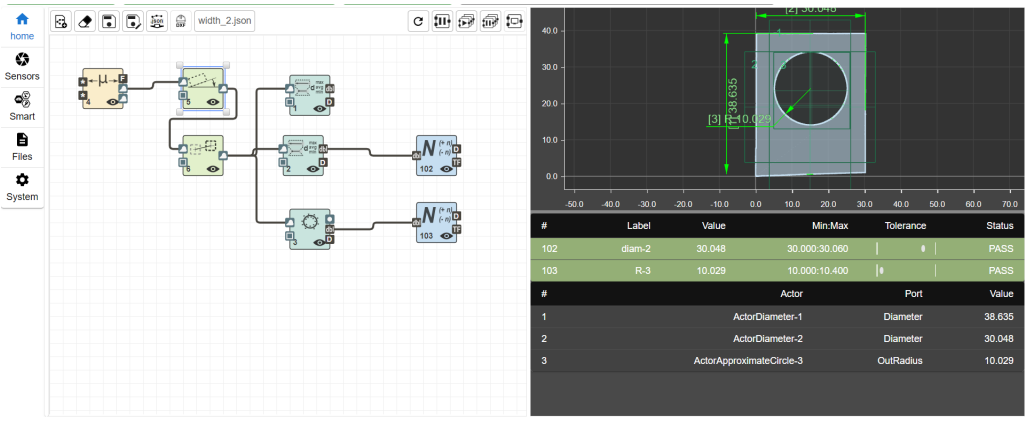


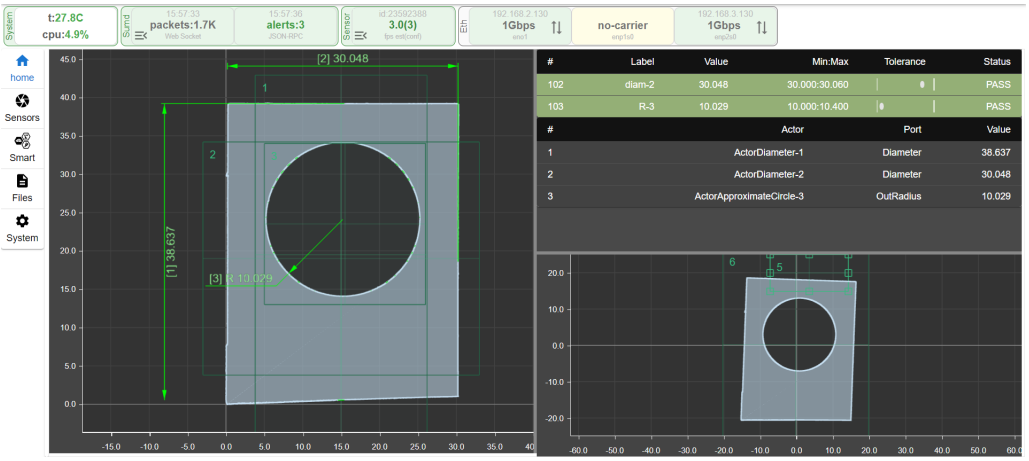
The section contains two groups of parameters:

1. The **Home View Controls** group is responsible for configuring information display modes on the **Home** tab. This section includes two main areas: **HMI** and **Displays**.

- The **HMI** area is intended for managing the display of the **Web HMI** panel: the "show panel" checkbox enables or disables the panel display, and when activated, a list of all available HMI panels becomes accessible. Adding and editing HMI panels is performed on the **HMI Adjustment** tab. The panel width is set by the **Panel width** parameter, in pixels.
- The **Displays** area is intended for configuring mnemonic diagrams (layouts) displayed on the **Home** tab. Here, the user selects the desired layout from the available list and defines its parameters. The following layout options are available:

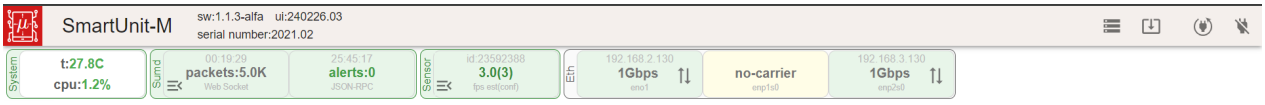
Mnemonic diagram (layout)	Description
	Shows all configured virtual displays as tabs. Example:

Mnemonic diagram (layout)	Description
	
	The display area is divided into two equal parts in accordance with the mnemonic diagram. The left part shows the measurement scheme, and the right part shows all configured virtual displays as tabs. Example: 
	The display area is divided into three parts in accordance with the mnemonic diagram. The left part shows the measurement scheme. The right part is divided into two parts, where two specified virtual displays are shown. Only virtual displays configured in the measurement scheme are available for selection. Example: 

Mnemonic diagram (layout)	Description
	The display area is divided into parts corresponding to the mnemonic diagram. Each part shows a specified virtual display. Only virtual displays configured in the measurement scheme are available for selection. View of the "Home" tab when setting up three displays: 

2. The **Top View Controls** group is responsible for customizing the area of status indicators. You can configure the list of displayed indicators, such as "System card", "Sumd card", "Sensors cards" and "Ethernet card", and where these indicators must be displayed. There are two options for the location of the indicator panel:

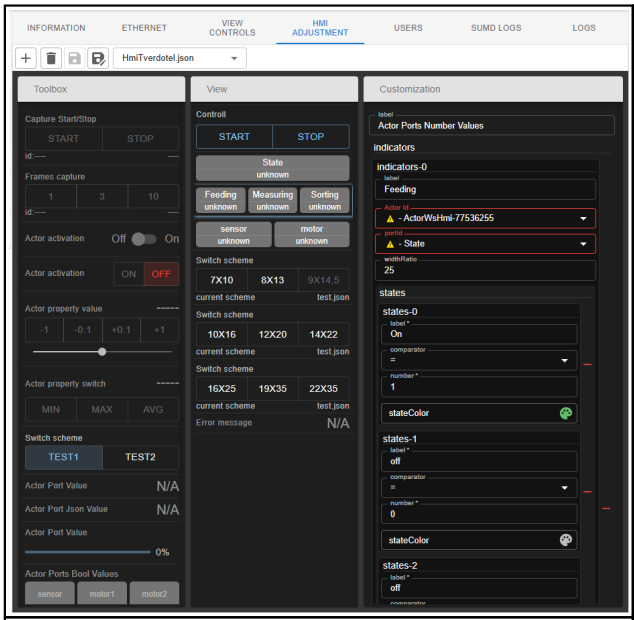
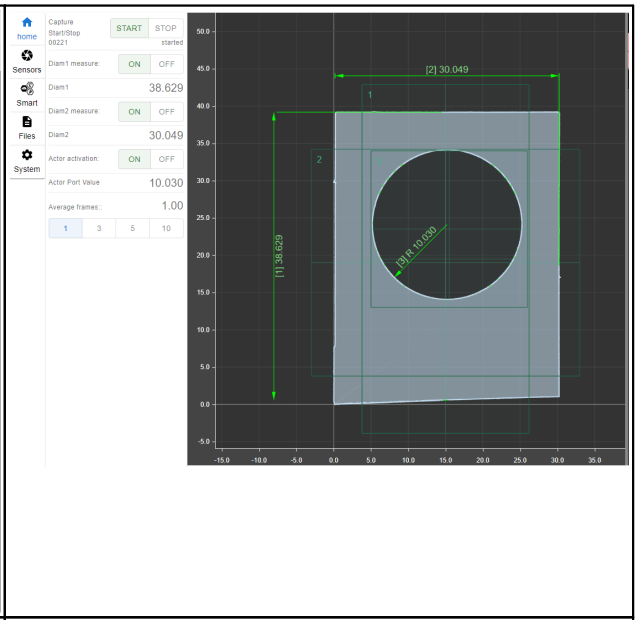
- "Top Bar" - indicators are displayed in full on a special panel at the top of the page.



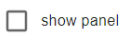
- "Header" - indicators are displayed in abbreviated form in the page header.

11.7.4. "HMI Adjustment" section

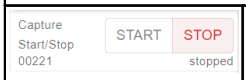
The **HMI Adjustment** section is intended for creating, deleting, loading and editing web HMI panels. The web HMI panel provides the ability to create controls and display information for the operator in the **Home** tab. This section allows the user to control sensors, smart blocks, active measurement schemes, as well as organize interaction with the input and output ports of the scheme.

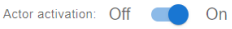
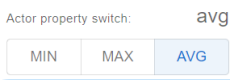
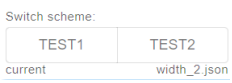
	
Web HMI setup mode	Example of displaying the web HMI in the Home tab

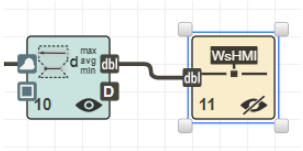
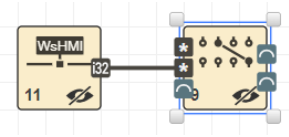
The buttons located at the top of the section are intended for creating, saving and loading HMI panels:

	Creating a new HMI panel. When creating a new panel, it is necessary to specify its name. In accordance with this name, a file will be created in the non-volatile memory of the controller.
	Deleting the current HMI panel. The corresponding file will be deleted from the non-volatile memory of the controller.
	Saving all changes made to the current HMI panel to non-volatile memory. Before clicking this button, all changes are stored in volatile memory and may be lost. The button is activated when there are unsaved changes.
	Saving the current HMI panel to non-volatile memory under a new name.
	Drop-down list for selecting the panel to be edited.
	Checkbox to show/hide the current panel in the "Home" tab. When this checkbox is selected, the panel automatically appears in the Home tab.

The following widgets are available:

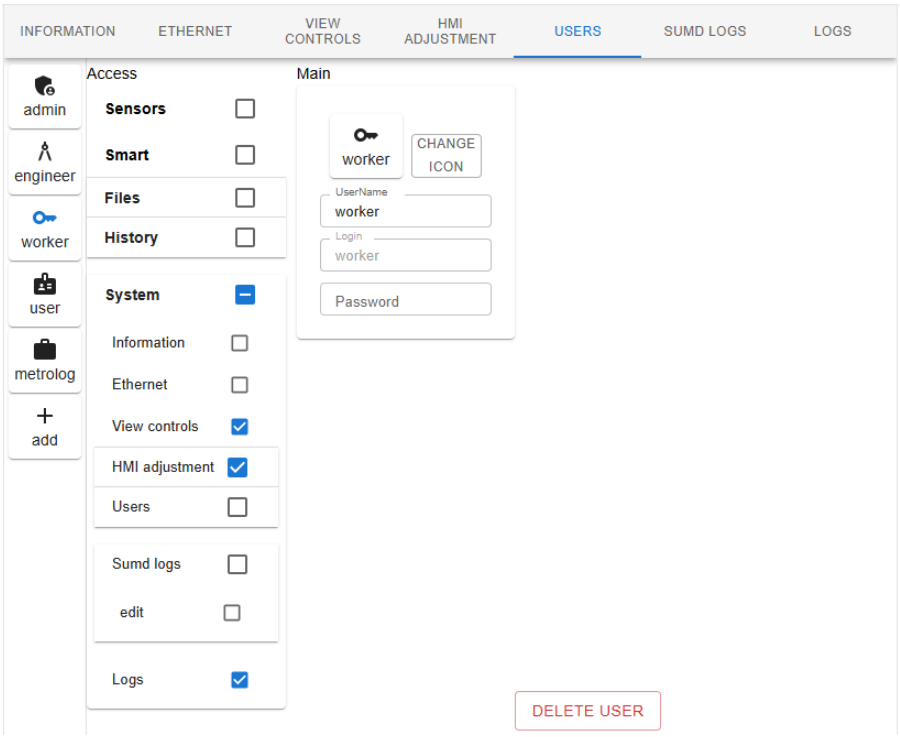
View	Description	Settings
Sensor control		
	Starting/stopping continuous frame capture from the sensor.	1. **label** field - the displayed label (by default - *Capture Start/Stop*). 2. **sensorId** drop-down list - a list of available sensors.
	Capturing the required number of frames from the sensor, and starting/stopping continuous frame capture from the sensor.	1. **label** field - the displayed label (by default - *Frames capture*). 2. **sensorId** drop-down list - a list of available sensors. 3. **buttons** group - a set of buttons for capturing a specified number of frames. Up to three buttons can be added, and the required number of frames can be set for each button. 4. **continuous Start/Stop button** checkbox - adding a button to the widget to start continuous frame capture.

View	Description	Settings
Control of the state and properties of smart blocks		
option 1:  option 2: 	Activating or deactivating the specified smart block.	1. **label** field - the displayed label (by default - *Actor activation*). 2. **actorId** drop-down list - a list of available smart blocks on the current scheme.
	Changing the numeric (integer or real) property of a smart block. The widget displays the current value of the property and also contains a slider and a group of buttons for changing this value. Both the slider and the group of buttons can be removed from the widget. The group of buttons contains four buttons, two of which provide for changing the value in small steps and two for changing in large steps.	1. **label** field - the displayed label (by default - *Actor property value*). 2. **actorId** drop-down list - a list of available smart blocks on the current scheme. 3. **propertyName** drop-down list - a list of available numeric properties for the selected **actorId**. The value must be selected after **actorId** is selected. 4. **sliderParams** group - parameters that define the slider settings: - **visible** checkbox - if selected, the slider is displayed on the widget. - **min** and **max** fields - set the range of acceptable values for the slider; - **step** field - minimum step for the slider. - **marks** checkbox - if selected, the minimum step marks are visible on the slider. 5. **buttonParams** group - parameters that define the settings of a group of buttons: - **visible** checkbox - if selected, the group of buttons is displayed on the widget. - **smallStep** field - sets the value for the small step (two central buttons). - **bigStep** field - sets the value for the large step (two side buttons).
	Toggling the string, numeric, or boolean property of a smart block. The widget displays the current property value and can contain up to 4 buttons with predefined property values.	1. **label** field - the displayed label (by default - *Actor property switch*). 2. **valuesType** dropdown list - the type of the property being toggled. Available values: string, number, integer, boolean. It is selected based on the type of smart block property to be changed. 3. **actorId** drop-down list - a list of available smart blocks on the current scheme. 4. **propertyName** drop-down list - a list of available numeric properties for the selected **actorId**. The value must be selected after the **actorId** is selected. 5. **buttonsString**, **buttonsNumber**, **buttonsInteger** or **buttonsBool** group depending on the selected **valuesType**. Allows the user to customize buttons and their fixed values.
Scheme control		
	Switching the current scheme. The widget displays the active scheme and contains a group of buttons with specified scheme names.	1. **label** field - the displayed label (by default - *Switch scheme*). 2. **buttons** group - parameters that define button settings: - **schemeFile** drop-down list - a list of available schemes to switch. - **schemeLabel** field - the value that will be displayed on the button instead of the scheme file name.

View	Description	Settings
Interaction with the input and output ports of the scheme (carried out through the “Web Hmi” smart block)		
	Widget for displaying a string, numeric or Boolean value from the output of the smart block. To obtain the value, you need to add the "Web Hmi" smart block to the scheme. For the added block, configure the input of the appropriate data type and connect the output in the scheme to the “Web Hmi” input. For example:  In this example, it is necessary to obtain the Diameter value of the ActorDiameter smart block. To do this, the Web Hmi block was added, its input of the double type was configured and connected to the Diameter output of the ActorDiameter smart block. After this, the following must be set for the widget: **actorId** (that corresponds to the WebHmi smart block) and **portId** (its input port).	1. **label** field - the displayed label (by default - *Actor Port Value*). 2. **actorId** drop-down list - a list of available Web Hmi smart blocks on the current scheme. 3. **portId** drop-down list - a list of available input ports for the Web Hmi smart block. The value should be selected after the **actorId** is selected. 4. **precision** group - sets the specified number of decimal places when displaying a floating-point number.
	Widget for transmitting a string, numeric or Boolean value to the input of the smart block. To transfer a value, you need to add the "Web Hmi" smart block to the scheme. For the added block, configure the output of the appropriate data type and connect this output to the required input on the scheme. For example:  In this example, it is necessary to transfer a given number to the ActiveOutput input of the ActorSwitcher block. To do this, the Web Hmi block was added, the output of the int32t type was configured and connected to the ActiveOutput input of the ActorSwitcher smart block. After this, the following must be set for the widget: **actorId** (that corresponds to the WebHmi smart block) and **portId** (its output port).	1. **label** field - the displayed label (by default - *Actor Port Value*). 2. **valuesType** dropdown list - the type of the property being switched. Available values: string, number, integer, boolean. Selected depending on the type of smart block property to be changed. 3. **actorId** drop-down list - a list of available Web Hmi smart blocks on the current scheme. 4. **portId** drop-down list - a list of available output ports for the Web Hmi smart block. The value should be selected after the **actorId** is selected. 5. **buttonsString**, **buttonsNumber**, **buttonsInteger** or **buttonsBool** group depending on the selected **valuesType**. Allows the user to customize buttons and their fixed values.

11.7.5. "Users" section

The **Users** section is designed for managing users and their access rights.



This section includes three main areas:

1. User selection.

This area displays a list of all registered users. A user can be selected for subsequent editing of their data and access permissions.

2. Access.

This area is used to manage the access rights of the selected user. Here, permissions can be added or removed for performing operations on various tabs and interface sections, as well as for making modifications. The structure of available permissions includes the following categories and actions:

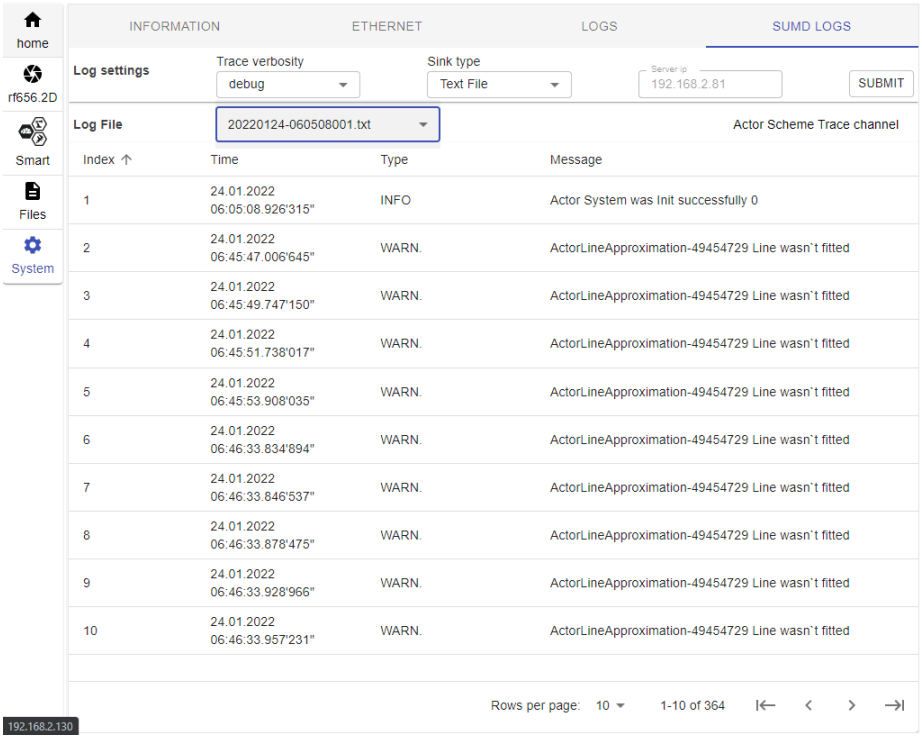
- Sensors
- Smart
- Files: logs, dumps, tmp_dumps, schemes, scripts, tables, panels.
- History: delete, download, reports.
- System:
  - Information
  - Ethernet
  - View controls
  - HMI adjustment: add,edit,delete
  - Users:add, edit, delete
  - Sumd logs
  - Logs

3. Main.

This area defines the user's basic data – name, login, password, icon.

11.7.6. "Sumd Logs" section

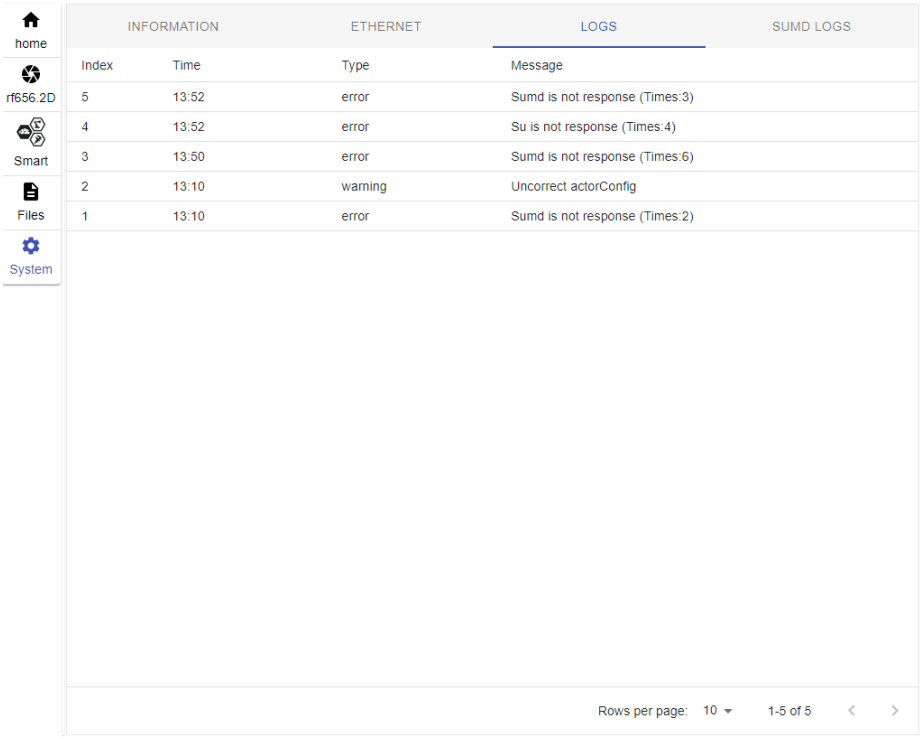
This section is intended for viewing information about the controller operation in order to identify possible errors.



At the bottom of the section is the pagination panel, which can be used to divide a large amount of log data into separate pages for easy viewing.

11.7.7. "Logs" section

This section is intended for viewing information about errors in the operation of the web interface of the controller.

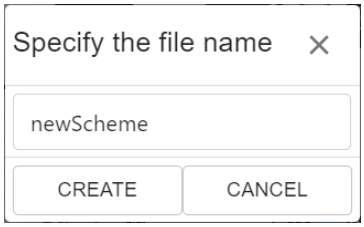


At the bottom of the section is the pagination panel, which can be used to divide a large amount of log data into separate pages for easy viewing.

11.8. Creating measurement schemes

11.8.1. Building a scheme

To create a new scheme, click the button on the top panel of the scheme area. In the dialog box that appears, specify a name for the new scheme and click the **CREATE** button:



A new json file will be created with the specified name. On the top panel of the scheme area, the name of the current scheme will be changed to the new one.

Next, you need to place the selected block on the scheme by dragging it from the **Smart Blocks** area to the graph construction area.

To create a connection between blocks, drag the output of one block to the input of another block (or several blocks) using the mouse. For convenience, the block inputs to which you can create a connection will be increased.

In the **Block Settings** tab, the user can change the block parameters. To do this, select the required block on the scheme or use the drop-down menu.

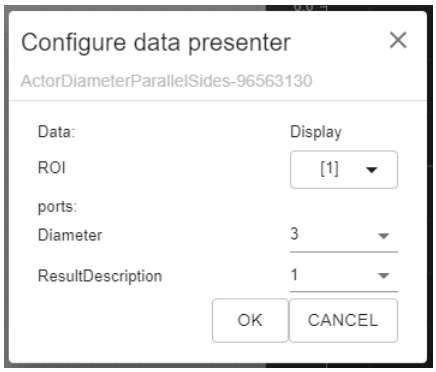
If the block has a search area (roi), then when the block is placed in the scheme, a search area appears in the measurement results display area. The search area is intended to specify the area in which the selected block operates. The search area can be moved and resized using the mouse.

Each block of the graph has a unique (within the graph) identifier (number) displayed in the lower left corner of the block, which makes it possible to quickly match the block and the search area in which it operates.

After making changes to the measurement scheme, it is possible to save them. To save changes to the current file, click the **Save** button on the top panel of the scheme area. To save changes to another file, click the **Save As** button.

11.8.2. Setting up displays to show data from a scheme

It is possible to customize how information from the block outputs should be displayed. To do this, click on the 'eye' symbol in the lower right corner of the block. In the dialog box that appears, you need to specify the number of the display on which you want to display the result.



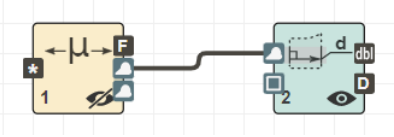
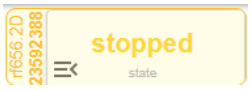
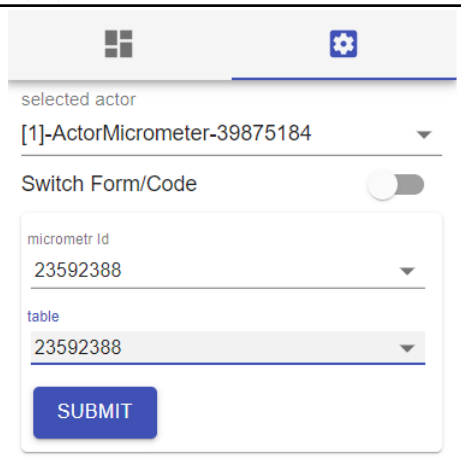
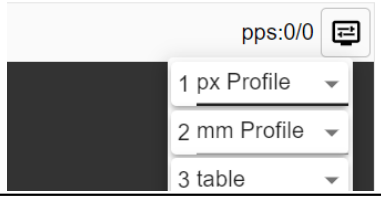
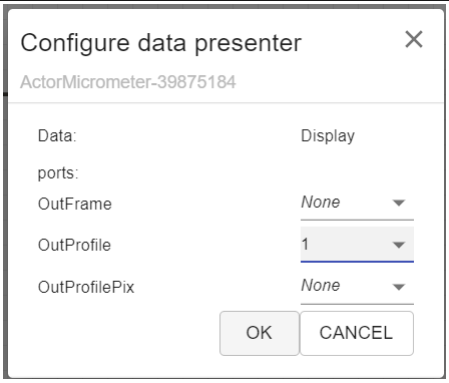
If the block has a search area (roi), the dialog box allows the user to specify the numbers of displays (any number from 1 to 12) on which it is necessary to display the search area.

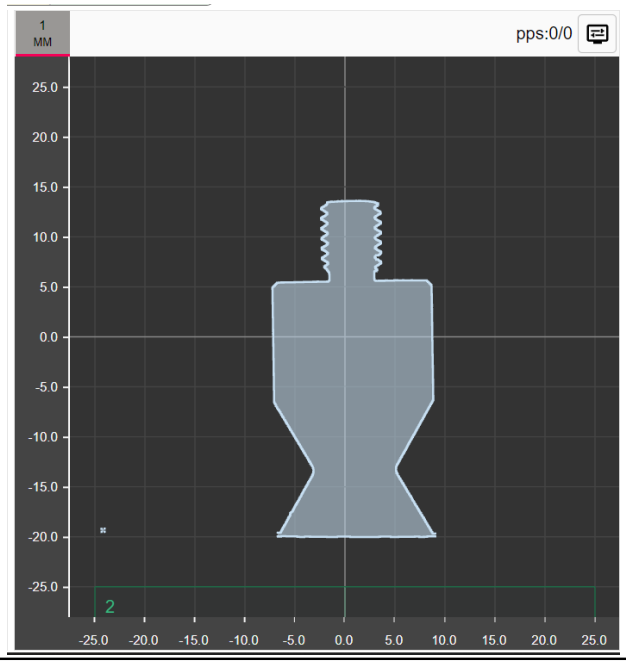
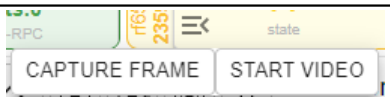
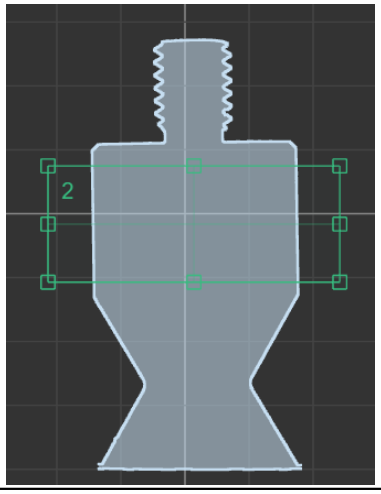
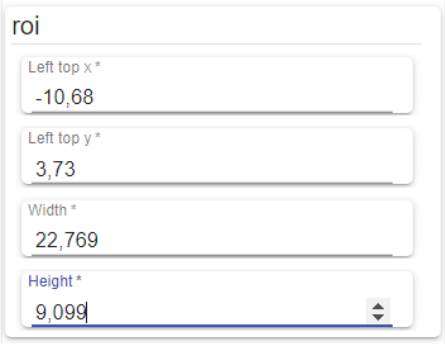
Next, for each display configured in the scheme, it is necessary to specify its type. This is done using the panel called by clicking the **Display settings** button in the measurement results display area. The display type must match the type of data to be displayed (see Measurement results display). This means that if the display type does not match the type of data to be displayed, no data will be displayed.

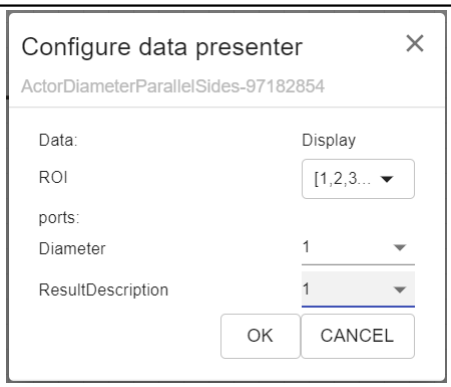
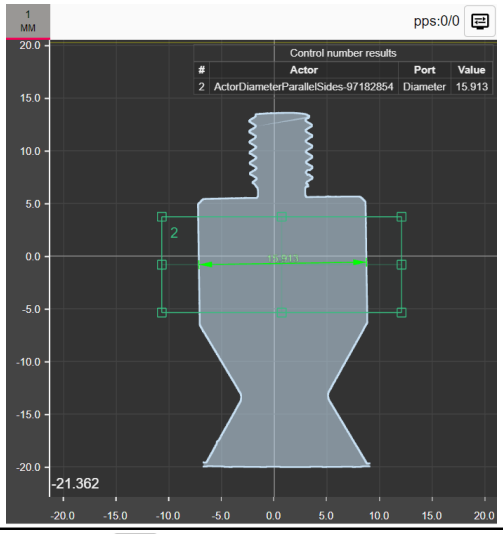
The display settings are saved with the measurement schemes, so after configuring the displayed data and/or displays, it is necessary to save the changes by clicking the **Save** button.

11.8.3. Example 1: Creating a scheme for measuring the diameter

As an illustration of the graph construction process, let's find the diameter of the cylindrical part of the measured product and present the result on the **2D mm** display.

1.	Create a new scheme named "cylinder". To do this, click the button on the top panel of the scheme area. In the dialog box that appears, enter the name ("cylinder") and click CREATE.	
2.	Add "micrometer" and "diameter of parallel sides" blocks to the scheme. Next, connect the "OutProfile" output of the "micrometer" block to the "inpProfile" input of the "diameter of parallel sides" block. 	
3.	For the "micrometer" block, specify the ID of the micrometer, the information from which should be sent to the scheme, as well as the directory with the calibration table. To change block settings, select the block on the scheme, go to the "Smart" tab, and then go to the "Block Settings" tab. The "micrometer Id" and "table" parameters are set by selecting the required value from the drop-down list. After setting the parameters, click SUBMIT. The Sensor indicator for the micrometer will change state from "accessible" to "stopped":  	
4.	To display the output profile ("OutProfile" output) of the "micrometer" block on display #1, do the following: 1. Click on the "eye" symbol in the lower right corner of the "micrometer" block. 2. In the dialog box that appears, select "1" for the OutProfile parameter and click OK. After this action, display "1" will appear in the result display area. 3. Using the "Display settings" panel, make sure that the type of display #1 is set to "px Profile": 	

5.	Place the object within the field of view of the micrometer. Capture a test frame from the micrometer to be displayed on display #1. To start capturing, you can use the "Sensor" panel or the "Sensors Settings" section of the "Sensors" tab. After performing this operation, the profile of the object will be displayed on display #1: 	
6.	For the "diameter of parallel sides" block, set the required ROI size. This operation can be performed in two ways: • On display #1, by moving the area with the number that corresponds to the unique block number on the scheme. • In the "Block Settings" tab, after selecting the block on the scheme and highlighting the required work area on the display. 	

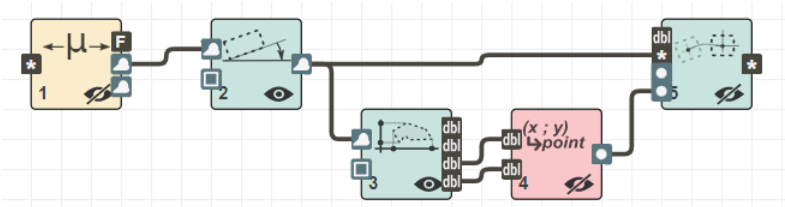
7.	Set up the display of calculation results on display #1. To do this: 1. Click on the "eye" symbol in the lower right corner of the "diameter of parallel sides" block. 2. In the dialog box that appears: • Select "1" for the Diameter parameter. The calculated diameter will be shown in the table located in the upper right corner of the display. • Select "1" for the ResultDescription parameter. The calculated diameter will be shown on the profile with a dimension line. 3. Click OK. After this action, display "1" will appear in the result display area.	
8.	Start capturing frames from the micrometer. After starting, display #1 will show: 1. Profile of the measured object ("OutProfile" output of the "micrometer" block). 2. Dimension line with the measured diameter value (ResultDescription output of the "diameter of parallel sides" block). 3. Table with the diameter value (Diameter output of the "diameter of parallel sides" block). 	
9.	Save the changes by clicking the Save button on the top panel of the scheme area.	

11.8.4. Example 2: Creating a scheme with coordinate system transformation

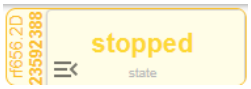
Often there is a need to align the inclination of the measured object relative to the sensor coordinate system. There is also the problem of binding the coordinate system of the measuring system to the measured sample. This operation makes it possible to set the measurement areas regardless of the position of the measured sample in the field of view of the sensor.

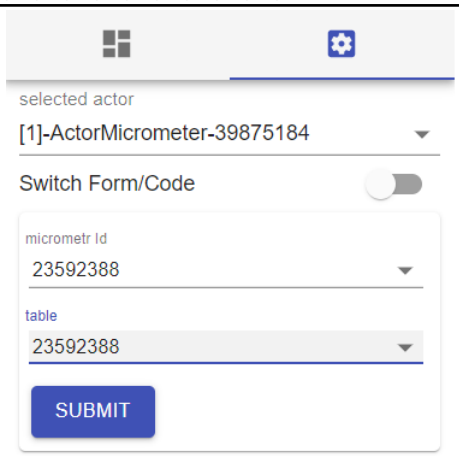
1.	Create a new scheme named "transformation". To do this, click the button on the top panel of the scheme area. In the dialog box that appears, enter the name ("transformation") and click CREATE.
2.	Make a scheme. To do this, add the following blocks to the scheme: • "micrometer"; • "align compensate" - to align the inclination to the specified edge; • "extreme coordinates" - to determine the required origin of the coordinate system; • "make 2d point" - to form a point; • "tilt & shift correction" - to move the coordinate system. Make connections: • connect the "OutProfile" output of the "micrometer" block to the "inpProfile" input of the "align compensate" block; • connect the "OutProfile" output of the "align compensate" block to the "inpProfile" input of the "extreme coordinates" block;

- connect the "OutProfile" output of the "align compensate" block to the "inpProfile" input of the "tilt & shift correction" block;
- connect the "minX" output of the "extreme coordinates" block to the "X" input of the "make 2d point" block;
- connect the "minY" output of the "extreme coordinates" block to the "Y" input of the "make 2d point" block;
- connect the "Point" output of the "make 2d point" block to the "Shift" input of the "tilt & shift correction" block.

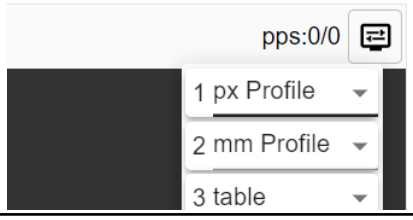


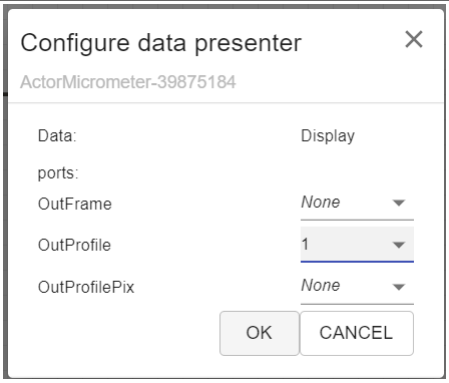
3. For the "micrometer" block, specify the ID of the micrometer, the information from which should be sent to the scheme, as well as the directory with the calibration table.
To change block settings, select the block on the scheme, go to the "Smart" tab, and then go to the "Block Settings" tab.
The "micrometer Id" and "table" parameters are set by selecting the required value from the drop-down list.
After setting the parameters, click SUBMIT. The Sensor indicator for the micrometer will change state from "accessible" to "stopped":



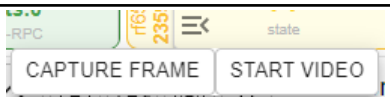


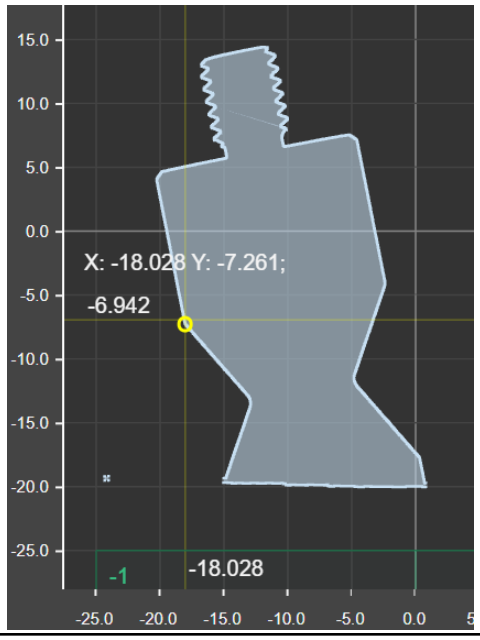
4. To display the output profile ("OutProfile" output) of the "micrometer" block on display #1, do the following:
1. Click on the "eye" symbol in the lower right corner of the "micrometer" block.
2. In the dialog box that appears, select "1" for the OutProfile parameter and click OK. After this action, display "1" will appear in the result display area.
3. Using the "Display settings" panel, make sure that the type of display #1 is set to "px Profile":



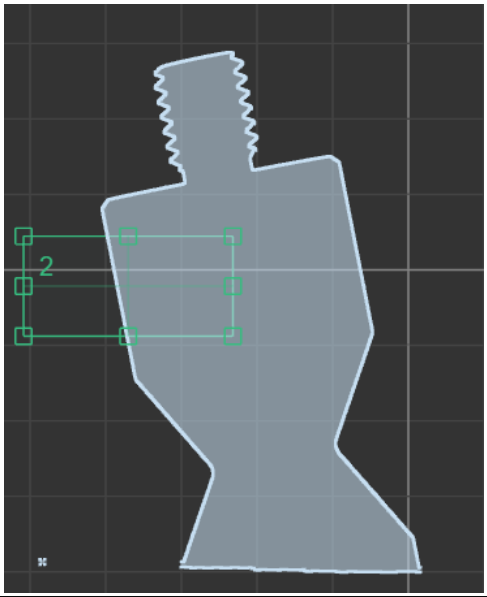


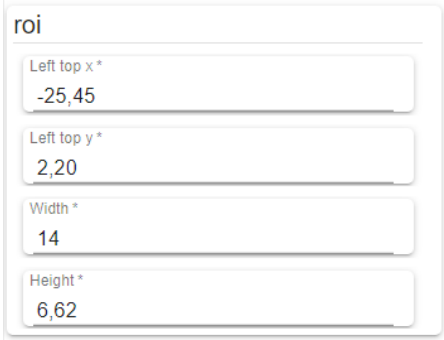
5. Capture a test frame from the micrometer to be displayed on display #1. To start capturing, you can use the "Sensor" panel or the "Sensors Settings" section of the "Sensors" tab. After performing this operation, the profile of the object will be displayed on display #1:





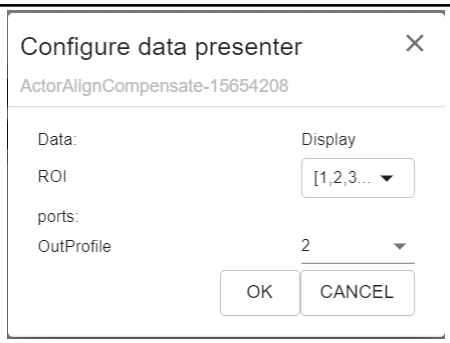
6. For the "align compensate" block, set the required ROI size. The ROI must be defined so that it covers only the face to which the slope is to be aligned. This operation can be performed in two ways:
   - On display #1, by moving the area with the number that corresponds to the unique block number on the scheme.
   - In the "Block Settings" tab, after selecting the block on the scheme and highlighting the required work area on the display.

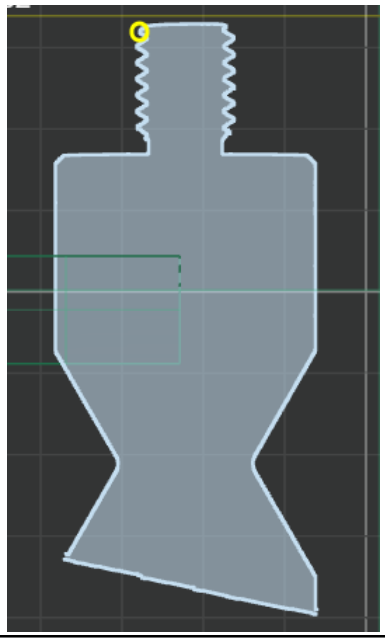




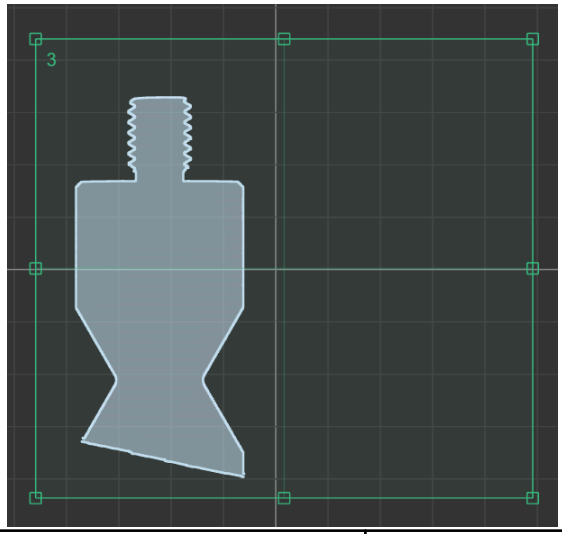
7. Set up the display of the profile rotation result on display #2. To do this:
   1. Click on the "eye" symbol in the lower right corner of the "align compensate" block.
   2. In the dialog box that appears, select "2" for the OutProfile parameter and click OK. After this action, display "2" will appear in the result display area.

   Start capturing frames from the micrometer. After starting, display #2 will show the profile aligned vertically (horizontally).

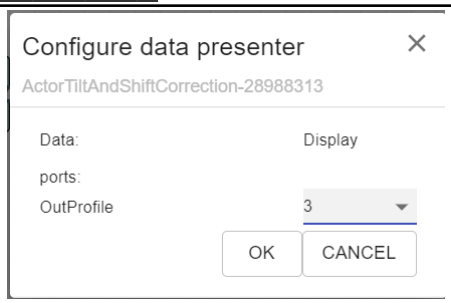


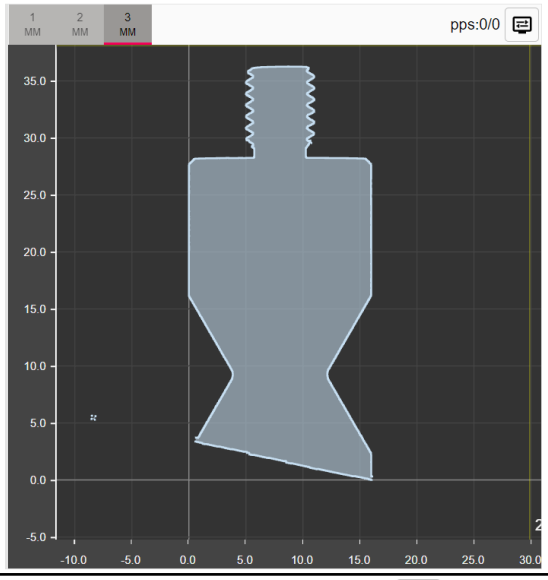


8. For the "extreme coordinates" block, set the required ROI size. The ROI must be defined so that it covers the entire object, and also takes into account the possible initial displacement of the object.



9. Set up the display of the profile shift result on display #3. To do this:
1. Click on the "eye" symbol in the lower right corner of the "tilt & shift correction" block.
2. In the dialog box that appears, select "3" for the OutProfile parameter and click OK. After this action, display "3" will appear in the result display area.
Start capturing frames from the micrometer. After starting, display #3 will show a profile aligned vertically and the coordinate system will be shifted relative to the lower left corner of the profile.



		
10.	Save the changes by clicking the Save button on the top panel of the scheme area.	

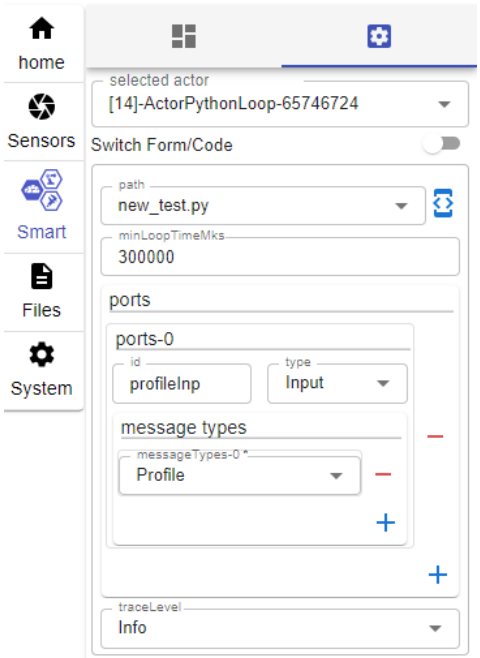
11.9. Custom scripts. "Python script" smart block

The “Python script” smart block is designed to execute custom scripts written in Python. They provide flexibility and the ability to expand the functionality of the system in accordance with user requirements and can be used for:

- implementation of custom measurement processing algorithms,
- implementation of proprietary information exchange protocols,
- measurement process control, including control of the logic for grouping results,
- implementation of stepper motor control logic,
- and so on.

Interaction with other smart blocks is carried out using input and output ports, which are created dynamically. Script functions can be executed both at a specified frequency and upon arrival of messages to the input ports.

Setting the script call time, editing ports, and editing the script are carried out in the **Block Settings** tab of the **Smart** tab:



11.9.1. Script structure

To implement actor functions, the script must define the following functions:

- **OnActivate**(*reference*) - called every time the actor is activated (including immediately after creating the actor).

- **OnDeactivate**(*reference*) - called every time the actor is deactivated (including immediately before deleting the actor).

- **OnInputReceive**(*reference, message, portId*) - called when messages are received from the actor inputs. Method parameter: *message* - an instance of the *Message.portId* class (identifier of the port to which the message arrived).

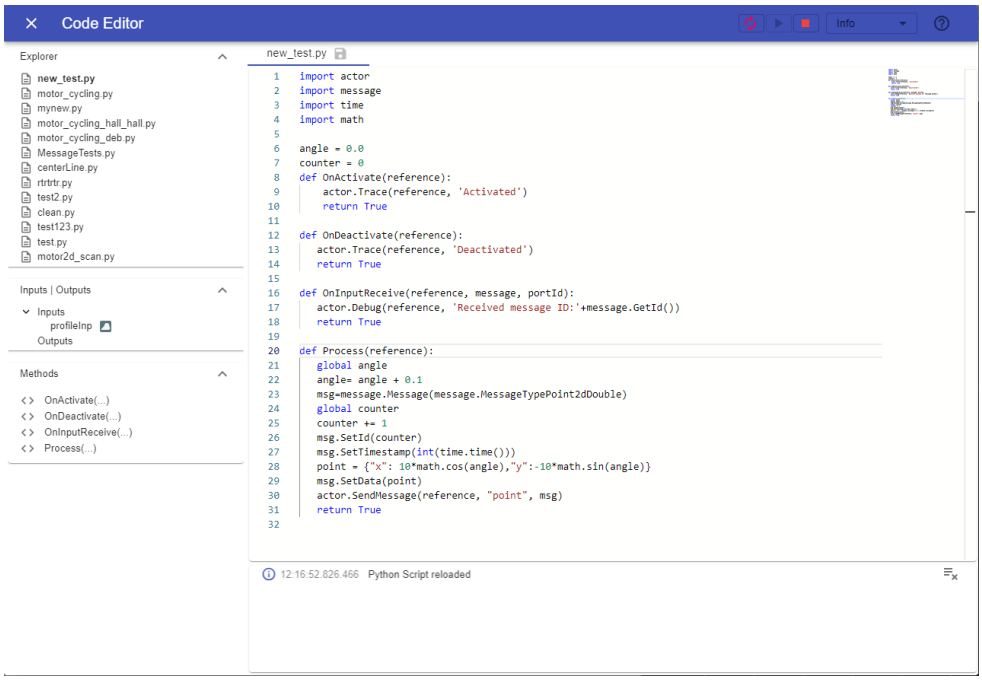
- **Process**(*reference*) - called in a loop with a period no more than *minLoopTimeMks*. The loop period is set in the actor configuration.

Script example:

```
import actor
import message
angle = 0.0

def OnActivate(reference):
actor.Trace(reference, 'Activated')
return True

def OnDeactivate(reference):
actor.Trace(reference, 'Deactivated')
return True

def OnInputReceive(reference, message, portId):
actor.Debug(reference, 'Received message ID:'+message.GetId())
return True

def Process(reference):
global angle
angle= angle + 0.1
msg=message.Message(message.MessageTypePoint2dDouble)
msg.SetId(total);
msg.SetTimestamp(int(time.time()))
point = {"x": math.cos(angle),"y":-math.sin(angle)}
msg.SetData(point)
actor.SendMessage(actorRef, "outint", msg)
return True
```

11.9.2. Script editor

For creating and editing the scripts, a special editor is provided:



The editor window is divided into the following areas:

1. Lists of smart block inputs and outputs, showing the input or output data type and its name. The user can change the name, taking into account that only ASCII characters are allowed and the length of the name should not exceed 60 characters.
2. List of data types supported by the script.
3. List of special methods provided for quick search and insertion. When you click on a method, its prototype will be inserted into the script editor.
4. Script loading and execution area.
5. Script editing area.
6. Console for displaying errors and messages.

11.9.3. Debugging a script in VS Code over the network

When writing a script, you can take advantage of the powerful debugging tool integrated into Visual Studio Code (VS Code). The **debugpy** library is used for this purpose. This library allows you to create a debug server for your Python code and connect to it from VS Code. Initialization of the **debugpy** server is possible **ONLY INSIDE** the **OnInputReceive**(*reference, message*) or **Process**(*reference*) functions. To initialize the server, use the *debugpy.listen(("0.0.0.0",5678))* function. To set breakpoints in a script, use the *debugpy.breakpoint()* function. Breakpoints are only activated after the VS Code debugging client connects to the **debugpy** server.

Script example:

```
import debugpy

# Debug server initialization function with protection against re-initialization
def CheckDebugger():
    if not debugpy.is_client_connected():
        if getattr(CheckDebugger, 'listenPort', -1) != 5678 :
            debugpy.configure(subProcess=True)
            host,CheckDebugger.listenPort = debugpy.listen(("0.0.0.0",5678))
    return True

def OnActivate(actorRef):
    # On the first activation, the breakpoint will not hit
    # because the debug server is not initialized
    debugpy.breakpoint()
    actor.Trace(actorRef, "On Activate")
    return True

def OnDeactivate(actorRef):
    debugpy.breakpoint()
    return True

def Process(actorRef):
    CheckDebugger()
    debugpy.breakpoint()
    return True

def OnInputReceive(actorRef,mess):
    CheckDebugger()
    debugpy.breakpoint()
    return True
```

In VS code, to connect to the debugger, you need to create the **launch.json** file with the following content:

```
{
    // Use IntelliSense to learn about possible attributes.
    "version": "0.2.0",
    "configurations": [
        {
            "name": "Python Attach On Remote",
            "type": "python",
            "request": "attach",
            "connect": {
                "host": "192.168.2.130",
                "port": 5678
              },
            "pathMappings": [
```

```
                {
                  "localRoot": "${workspaceFolder}",
                  "remoteRoot": "."
                }
              ]
          }
      ]
}
```

11.9.4. "message" module

The module is designed to generate new messages and process messages received from other smart blocks.

So, in the basic script function **OnInputReceive**, its message argument is an instance of the **Message** class of the **message** module. An instance of the class is created as follows: *msg=message.Message(message.MessageTypePoint2dDouble)*, where the type of message to be created must be specified in the constructor.

Message class methods:

GetId(): Returns the message identifier.

GetTimestamp(): Returns the timestamp of the message.

GetType(): Returns the message type.

GetIdSender(): Returns the message sender ID.

GetIdPortSender(): Returns the port ID of the message sender.

GetData(): Returns message data.

SetId(): Sets the message ID.

SetTimestamp(): Sets the timestamp of the message.

SetData(): Sets the message data.

Message types:

Symbolic names	Values
MessageTypeBool	1
MessageTypeNumberInt8	10
MessageTypeNumberInt16	11
MessageTypeNumberInt32	12
MessageTypeNumberInt64	13
MessageTypeNumberDouble	14
MessageTypePoint2dDouble	50
MessageTypeRect	100
MessageTypeSegmentLine	101
MessageTypeStraightLine	102
MessageTypePolyLine	103
MessageTypeContour	104
MessageTypeProfile	105
MessageTypeFrame	1000
MessageTypeJson	5000
MessageTypeDescription	5005

Data structure for composite types:

MessageTypePoint2dDouble: dictionary {x:(float),y:(float)}. Example:

```
point = {'x' : 0.5, 'y' : 3.1}
```

MessageTypeRect: dictionary {x:(float),y:(float),w:(float),h:(float)}. Example:

```
rect = {'x' : -1.5, 'y' : 2.0, 'w' : 3.0, 'h' : 4.0}
```

MessageTypeSegmentLine: dictionary {x:[x1(float),x2(float)],y:[y1(float),y2(float)]}. Example:

```
segment = {'x' : [1.0, 2.0], 'y' :[1.5, 2.5]}
```

MessageTypeStraightLine: dictionary {A:(float), B: (float), C: (float)}. Example:

```
segment = {'A' : 1.0, 'B' :1.5, 'C' :1.5}
```

MessageTypePolyLine : dictionary {id:(long), x:[(float)...],y:[(float)...]}. Example:

```
polyline = {'id':10001, 'x' : [1.0, 2.0, 3.0, 4.0], 'y' :[1.5, 2.5, 2.5, 1.5]}
```

MessageTypeContour: dictionary {id:(long),type:(0,1) x:[(float)...],y:[(float)...]}. The contour points are ordered in such a way that if you move from point to point in direct order, the measured object is to the left of the direction of movement. For outer contours (type=0), the order of points is counterclockwise, and for inner contours (type=1) - clockwise. Example:

```
contour_ext = {'id':10010, type:0, 'x' : [1.0, 1.0, 5.0, 5.0], 'y' :[5.0, 1.0, 1.0, 5.0]}
contour_int = {'id':10011, type:1, 'x' : [2.0, 3.0, 4.0, 3.0], 'y' :[3.0, 4.0, 3.0, 2.0]}
```


MessageTypeProfile: dictionary {id:(long),timestamp:(long), contours:[{type:(0,1) x:[],y:[]}], hierarchy:[]". A profile is a composite data type and is a collection of contours (contours:[]) and hierarchical connections between them (hierarchy:[]). Each outer contour of a profile can hierarchically include multiple inner contours. Each outer contour must contain -1 in the corresponding hierarchy element, and each inner contour contains the index of the outer contour in the hierarchy element. In this case, the contour index is its serial number in the contour array, starting from 0. Example:

```
contour_ext = {'id':10011, 'type':0, 'x' : [1.0, 1.0, 5.0, 5.0], 'y' :[5.0, 1.0, 1.0, 5.0]}
contour_int = {'id':10011, 'type':1, 'x' : [2.0, 3.0, 4.0, 3.0], 'y' :[3.0, 4.0, 3.0, 2.0]}
profile = {'id':10010, 'timestamp':000, 'contours':[contour_ext,contour_int], 'hierarchy':[-1,0] }
```

MessageTypeFrame: dictionary {id:(long),timestamp:(long), width:(long), height: (long), pixelFormat:(1-8) , data:(bytes)}. For a frame in Y800(Monochrome, 8 bits) format, pixelFormat=1. Example:

```
frame = {'id':10001,  'timestamp':000,  'width':5, 'height':3, 'pixelFormat':1, 'data':bytes(np.
random.bytes(width * height))}
```

11.9.5. "actor" module

The module is designed to interact with the "Python script" smart block, namely sending messages from the outputs of the smart block, as well as sending messages to the logging/tracing system (see "Sumd Logs").

To connect the module, you need to add the *import actor* line to the script.

All functions of the actor module have *reference* as their first parameter. This parameter is used as a link to a specific "Python script" smart block of the measurement scheme.

Module functions:

SendMessage(reference, outputName, message) - sending the *message* object to the *outputName* output of the actor.

Trace(reference, message); **Debug**(reference, message); **Info**(reference, message); **Warning**(reference, message); **Error**(reference,message); **Critical**(self,message) - sending messages to the actor logging system.

11.9.6. Script examples

11.9.6.1. Finding the center line of the profile

```
1   import actor
2   import message
3   import time
4   import numpy as np


5   def OnInputReceive(reference, message, portId):
6       messType = message.GetType()
7       if portId=="profile" and messType == 105:
8           profile = message.GetData()
9           contours = profile['contours']
10          hierarchy = profile['hierarchy']
11          if len(contours)>0:
```

```
12              point1,point2 = calculate_symmetry_axis(contours[0])
13              sendLine(reference, message.GetId(), point1, point2 )
14      return True
15
16  def calculate_symmetry_axis(contour):
17      # Create a list of contour points
18      contourMy = np.column_stack((contour['x'],contour['y']))
19      #print(f"{contourMy=}")
20      # Calculate the covariance matrix of the contour
21      covariance = np.cov(contourMy.T)
22
23      # Perform eigen decomposition of the covariance matrix
24      eigenvalues, eigenvectors = np.linalg.eig(covariance)
25
26      # Find the eigenvector corresponding to the largest eigenvalue
27      largest_eigenvalue_index = np.argmax(eigenvalues)
28      major_axis = eigenvectors[:, largest_eigenvalue_index]
29
30      # Find the perpendicular vector
31      perpendicular_axis = np.array([-major_axis[1], major_axis[0]])
32
33      # Calculate the centroid of the contour
34      centroid = np.mean(contourMy, axis=0)
35
36      # Define two points on the major axis line
37      scale = 50
38      point1 = centroid - scale * major_axis
39      point2 = centroid + scale * major_axis
40      point1t = centroid - scale * perpendicular_axis
41      point2t = centroid + scale * perpendicular_axis
42      return point1,point2
43
44  def sendLine(actorReference,id,point1, point2):
45      msg=message.Message(message.MessageTypeSegmentLine)
46      msg.SetId(id); #Have to initialize counter variable like a global
47      msg.SetTimestamp(int(time.time())) # Have to include time
48      segment = {"x": [point1[0],point2[0]],"y":[point1[1],point2[1]]}
49      msg.SetData(segment)
50      actor.SendMessage(actorReference, "centerLine", msg)
51      return True
52
53  def Process(reference):
54      return True
55
56  def OnActivate(reference):
57      return True
58
59  def OnDeactivate(reference):
60      return True
```

This script for finding the center line of the profile is based on the Principal Component Analysis (PCA). PCA is a statistical technique used to reduce the dimensionality of data by transforming it into a new space of variables called principal components.

For the script to function, the PythonLoop smart block must be configured with one input port and one output port:

1. Input port - profile. Parameters: {id:profile, type:Input; messageTypes: [Profile]}. The port must be connected to any profile source.

2. Output port - centerLine. Parameters: {id:centerLine, type:Output; messageTypes:[SegmentLine]}.

This script calculates the central line immediately after a message of type 105 (MessageTypeProfile) arrives at the input of the smart block. The OnInputReceive function checks the message type and selects the first profile contour to calculate the center line. The actual calculation of the center line is done in the *calculate_symmetry_axis(contour)* function. The Numpy library is used to calculate PCA.

After finding the center line, the *sendLine(actorReference,id,point1, point2)* function is called. This function generates a message containing information about the

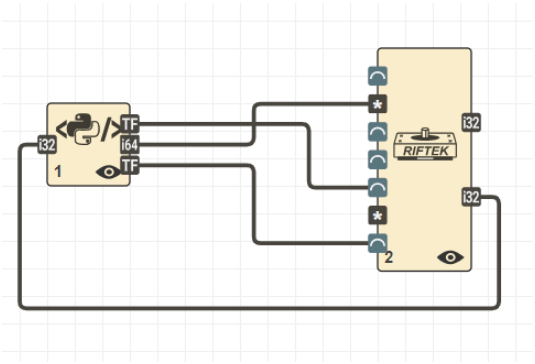
center line, represented as a segment, and sends it to the centerLine output port of the smart block.

11.9.6.2. Controlling the system motion in cyclic mode

Controlling the system motion in cyclic mode from the point determined by limit switch 1 to the point determined by limit switch 2:

```
import actor
import time
import message

recived = 0
id = 0
prevState=0

def Process(actorRef):
    #Request for motor state
    msg=message.Message(message.MessageTypeBool)
    global id
    id=id+1
    msg.SetId(id);
    msg.SetData(True)
    actor.SendMessage(actorRef, "reqest", msg)
    return True

def OnInputReceive(actorRef,mess, portId):
    global recived
    recived+=1
    steps = 3000000
    data = mess.GetData()
    if data not in [97,160,224,225]:
        return True
    msg=message.Message(message.MessageTypeNumberInt32)
    msg.SetId(id);
    if data==224 : # Stop Before Hall 2
        msg.SetData(-steps)
    if data==225 : # Stop Before Hall 1
        msg.SetData(steps)
    if data==97 : # AchiveHall1
        msg.SetData(-steps)
    if data==160 : # 255 - AchiveHall2
        msg.SetData(steps)
    actor.SendMessage(actorRef, "steps", msg)
    return True

def OnDeactivate(actorRef):
    # Stop Motor
    msg=message.Message(message.MessageTypeBool)
    global id
    id=id+1
    msg.SetId(id);
    msg.SetData(True)
    actor.SendMessage(actorRef, "stop", msg)
    return True
```

For the script to function, you need to create a scheme of two blocks: “Python loop” and “Riftek step motor”:



The PythonLoop smart block is configured with one input port and three output ports:

1. Input port - state. Parameters: {id:state, type:Input; messageTypes: [NumberInt32]}. This port is connected to the State output of the ActorRfMotor smart block.

2. Output port - reqest. Parameters: {id:reqest, type:Output; messageTypes: [Bool]}. This port is connected to the RequestState input of the ActorRfMotor smart block.

3. Output port - steps. Parameters: {id:steps, type:Output; messageTypes: [NumberInt32]}. This port is connected to the MoveTo input of the ActorRfMotor smart block.

4. Output port - stop. Parameters: {id:stop, type:Output; messageTypes:[Bool]}. This port is connected to the Stop input of the ActorRfMotor smart block.

The script works as follows:

1. The *Process(actorRef)* function is called cyclically at the specified frequency. This frequency is determined by the minLoopTimeMks parameter of the "Python loop" block. This function implements sending a request about the state of the motor and limit switches. As a result of this request, a message will be sent to the state input.

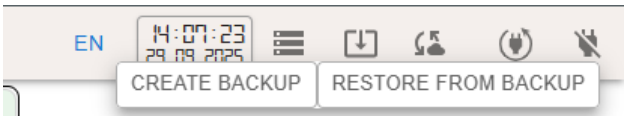
2. The *OnInputReceive(actorRef,mess, portId)* function analyzes the message about the state of the motor and limit switches. Based on the current state, a message is generated to the steps output about the required number of steps for the motor and the direction of rotation.

3. When the smart block is deactivated or paused, a message is sent to the stop output.

11.10. Backing up system files and settings

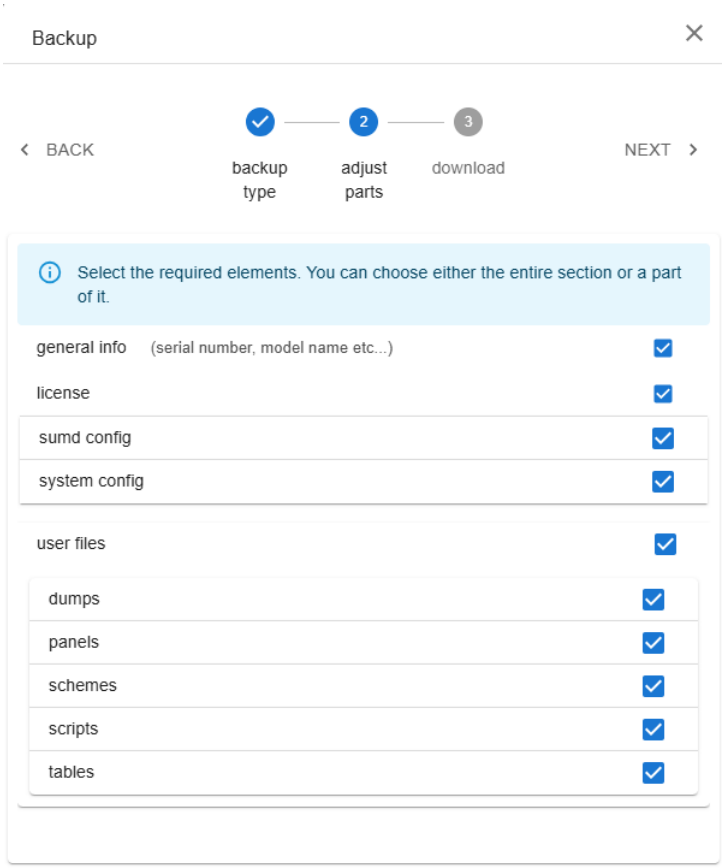
To ensure data integrity and enable quick system recovery in case of failures or hardware migration, functions for creating and restoring system file and settings backups are provided. This section describes the procedures for creating a backup and restoring the system from a backup file.

Backup creation and system restoration are performed through dedicated dialog windows. To open them, click the **System Backup** button in the auxiliary control panel. After that, a panel will appear with two buttons: **Create Backup** and **Restore From Backup**.

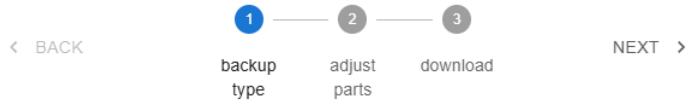


11.10.1. Creating a backup

Backup creation is performed using a special dialog box:



The backup process consists of three stages:

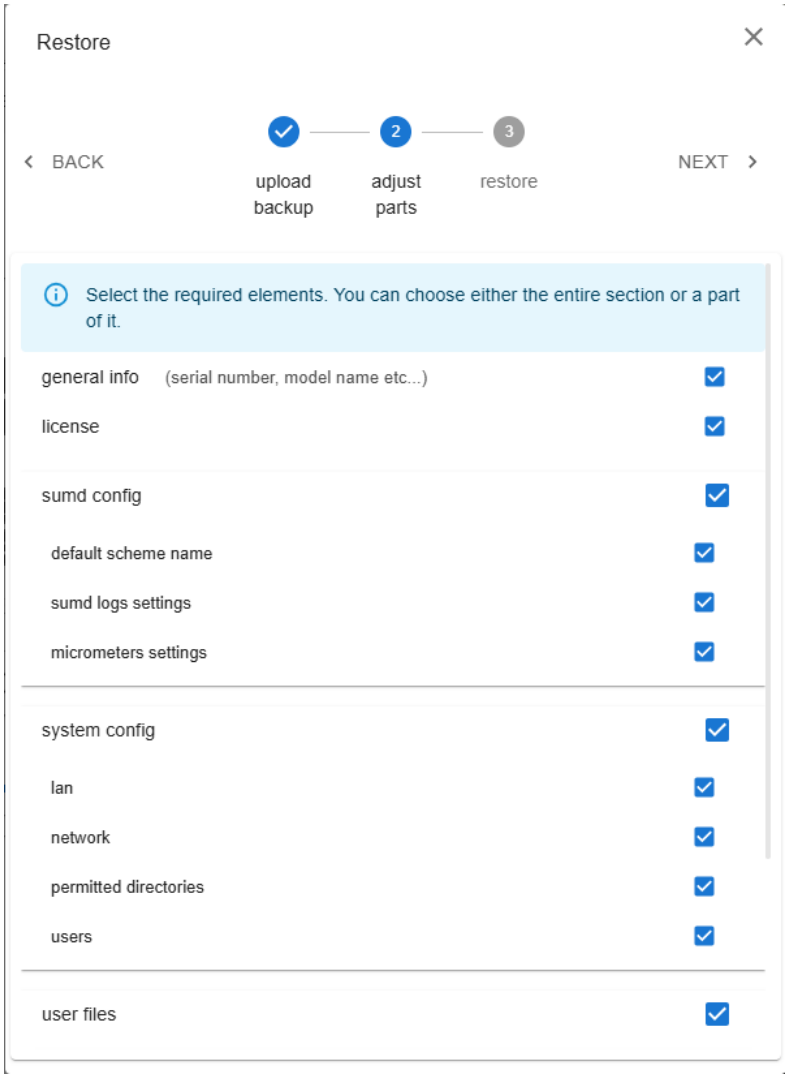
Stage	Description
	
Backup type	Selecting the backup type. Two options are available: • Full — saves all necessary files and settings for a complete system recovery. • Partly — selective mode, allowing the user to specify which data to save.

Adjust parts	Selecting the data to be saved (active only when "Partly" mode is selected).
	In "Full" mode, this step is skipped, and the dialog proceeds directly to the next stage.
	At this step, the user can choose which files and settings to include in the backup:
	• general info — general information (serial number, model, etc.);
	• license — license file linked to the current controller;
	• sumd config:
	  ○ default scheme name — scheme loaded at device startup;
	  ○ sumd logs settings — logging settings ("System" tab > Sumd Logs);
	  ○ micrometer settings — micrometer settings ("Sensors Settings" tab);
	• system config:
	  ○ network — network interface settings ("System" tab > Ethernet);
	  ○ users — user settings ("System" tab > Users);
	• user files:
	  ○ dumps — dump files;
	  ○ panels — HMI panel configurations;

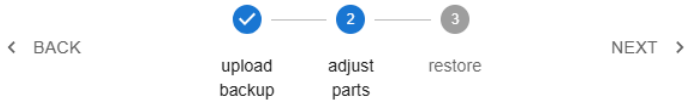
Stage	Description
	○ schemes — computational scheme files; ○ scripts — user script files; ○ tables — calibration table files and directories.
Download	Backup file generation and download to the local computer. At this stage, the Sumd service will be temporarily stopped. For this reason, it is necessary to save all modifications made to the measurement scheme beforehand.

11.10.2. Restoring from a backup

Restoring the system from a backup is carried out using a special dialog box:



The backup recovery process consists of three stages:

Stage	Description
	
Upload Backup	Uploading the backup file. At this stage, the user must select and upload a previously saved backup file from the local computer. Two options are available: • Full — restores all files and settings saved in the backup file. • Partly — selective mode, allowing the user to specify which data to restore.
Adjust parts	Selecting the data to restore.

Stage	Description
	In "Full" mode, this step is skipped, and the dialog proceeds directly to the next stage. At this step, the user can choose which files and settings to restore: • general info — general information (serial number, model, etc.); • license — license file linked to the current controller; • sumd config: ○ default scheme name — scheme loaded at device startup; ○ sumd logs settings — logging settings ("System" tab > Sumd Logs); ○ micrometer settings — micrometer settings ("Sensors Settings" tab); • system config: ○ network — network interface settings ("System" tab > Ethernet); ○ users — user settings ("System" tab > Users); • user files: ○ dumps — dump files; ○ panels — HMI panel configurations; ○ schemes — computational scheme files; ○ scripts — user script files; ○ tables — calibration table files and directories.
Restore	Starting the restoration process. After confirming the selected data, the system restoration process begins. Upon completion, the system will notify the user of the result and, if necessary, prompt to restart the device to apply the changes.

12. Maintenance

2D optical micrometers are virtually maintenance free. As these are optical systems, they are sensitive to dust and sputter on the front windows. Cleaning is best done with a soft cloth. Do not use scratching cleaners or other aggressive media.

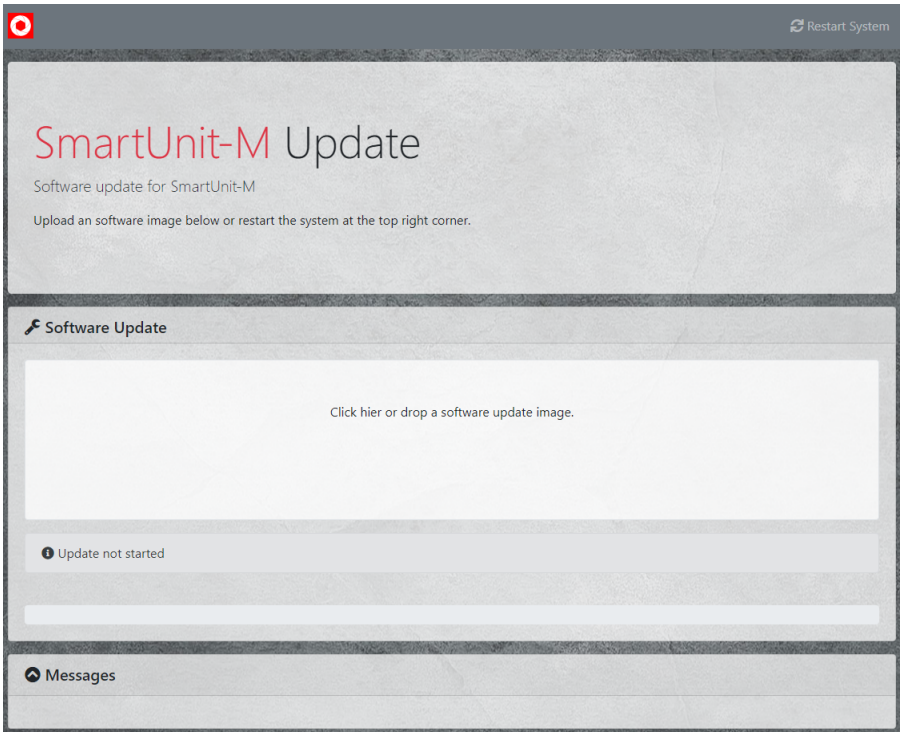
Make sure that there are no fingerprints on the surface of the windows, as they significantly degrade the accuracy of the measurement.

In order to remove fingerprints or grease, clean the windows with 20% alcohol and soft paper.

13. Software update

The device contains two identical instances of the operating system. After turning on the power, the operating system boots from one of them. This is implemented to provide the ability to update the operating system and software of the device. From the active instance of the operating system, the user can update the second instance, and then, after booting from the second instance, the user can update the first. The operating system can be updated only through the web interface of the device. To access the web interface, turn on the device and enter its IP address and port number in the address bar of the web browser: "192.168.2.130:8080". **ATTENTION:** The device must be turned on and connected to the computer from which this device is configured via the network (Ethernet interface).

The web page for updating the device software is shown below. To update the device software, you need to upload the update file provided by the manufacturer. To do this, the user must select the update file in the dialog box (after clicking the left mouse button in the upload area) or drag it to the **Software Update** section of the page. The update status is displayed at the bottom of the **Software Update** section. When the progress bar reaches "100%", a message will appear indicating the success or failure of the update. The **Messages** section displays service information about the current update operation in progress. After a successful software update, the device will automatically reboot. The operating system will boot from the updated instance of the operating system.



ATTENTION! To update both instances of the operating system, the user must first update one instance, and after the device automatically reboots from the updated instance, update the second instance.

ATTENTION! If you cannot boot from one of the system instances, you must change the system type.

14. Warranty policy

Warranty assurance for 2D Optical Micrometers RF656.2D Series – 24 months from the date of shipping; warranty shelf-life – 12 months.

Warranty repair is not provided in the following cases:

- mechanical damage caused by impacts or falling from height,
- damage caused by opening the housing, incorrect connection, or absence of grounding.

15. Technical support

Technical support related to the use of 2D optical micrometers is provided free of charge and includes technical assistance related to incorrect operation of 2D optical micrometers and problems with settings, development and research of use cases for 2D optical micrometers, training in working with software tools and libraries.

Technical support for software developed by the customer is provided on a paid basis and includes the possibility of adding new features to the software.

Technical support contacts:

- E-mail: info@riftek.com

16. Revisions

Date	Revision	Description
04.02.2022	1.0.0	Starting document.
12.03.2024	1.0.1	1. Changed the names of smart block groups, par. 11.4.2.2. 2. Added description of new smart blocks, par. 11.4.2.2.1 - 11.4.2.2.3, 11.4.2.2.5. 3. Added the DXF scheme builder, par. 11.4.3. 4. Added settings for displaying information in the Home tab, par. 11.6.3. 5. Added the web HMI panel, par. 11.6.4. 6. Added the ability to use custom scripts in schemes, par. 11.8. 7. Changed the pin assignment of the optical sensor connector, Annex 1. 8. Added a description of how to configure the controller's response to power supply, Annex 4.
30.09.2025	1.0.2	1. Added description of the HW3 controller, par. 6.3. 2. Added authorization step upon first power-on, par. 10.2. 3. Added interface elements, par. 11, 11.1, 11.1.1, 11.2.1, 11.7.3, 11.7.4. 4. Added description of new displays of measurement results, par. 11.2.5, 11.2.6. 5. Added description of new sensor parameters, par. 11.3.1. 6. Added description of new smart blocks, par. 11.4.2.2.1, 11.4.2.2.3, 11.4.2.2.4. 7. Added description of the History tab and the report generation procedure, par. 11.6. 8. Added management of users and their access rights to interface elements, par. 11.7.5. 9. Added description of the procedure for creating backups and restoring the system, par. 11.10.

17. Annex 1. Electrical characteristics

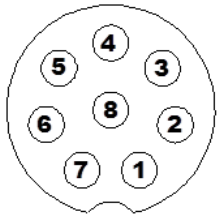
The micrometer comes with three cables:
1. Cable for connecting the scanner to the Ethernet network.
2. Optical sensor power cable with synchronization and output lines.
3. Controller power cable.

ATTENTION!
This User's Manual describes the cables that come with standard scanners.
Documentation for the cables is always included in the delivery.

17.1. Pinout of optical sensor connectors

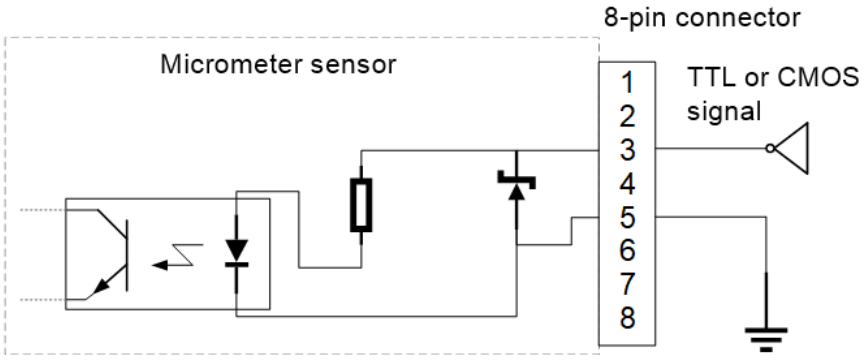
The optical sensor has two connectors:
1. Gigabit Ethernet, RJ-45 connector.
2. Multifunctional 8-pin connector (Binder 712 Series, #09-0428-30-08).

The pin assignment of the multifunctional connector, as well as the corresponding wire colors, are shown in the table.

Pin number	Wire color	Assignment	View
1	White	Backlight control output (Exposure Active). GPIO output.	
2	Brown	0V power supply, GND for GPIO.	
3	Green	Frame capture gating Line-1. Optoisolated input (OptoCoupled IN).	
4	Yellow	-	
5	Grey	OptoCoupled GND.	
6	Pink	UserDefined Output - Optoisolated output (OptoCoupled Out).	
7	Blue	Frame capture gating Line-0. Optoisolated input (OptoCoupled IN).	
8	Red	24V power supply.	

17.2. Electrical characteristics of the signal inputs and outputs of the sensor

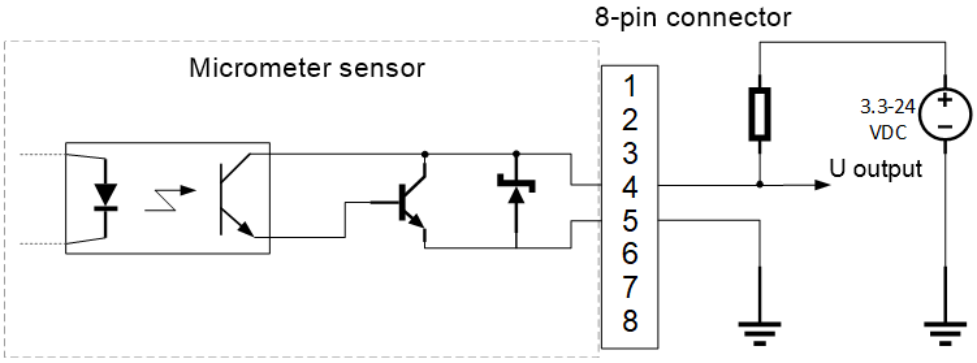
1. Optoisolated input.
Input signal connection diagram:



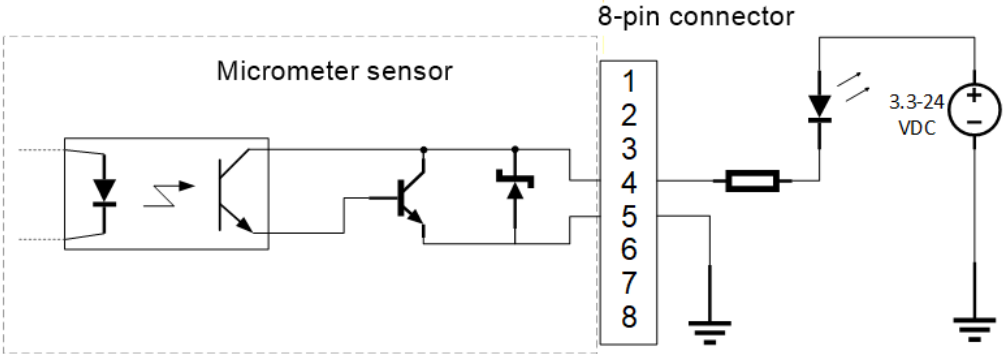
Parameter	Value
Maximum allowable input voltage (exceeding this voltage may damage the micrometer sensor and void the warranty)	30V DC
Safe voltage level	0–24V DC
Logic '0' voltage level (signal inversion disabled)	0–1.4V DC
Logic '1' voltage level (signal inversion disabled)	>2.2V DC
Consumption current	5–15 mA

2. Optoisolated output.

A typical variant of connecting the load to the optoisolated output of the sensor:

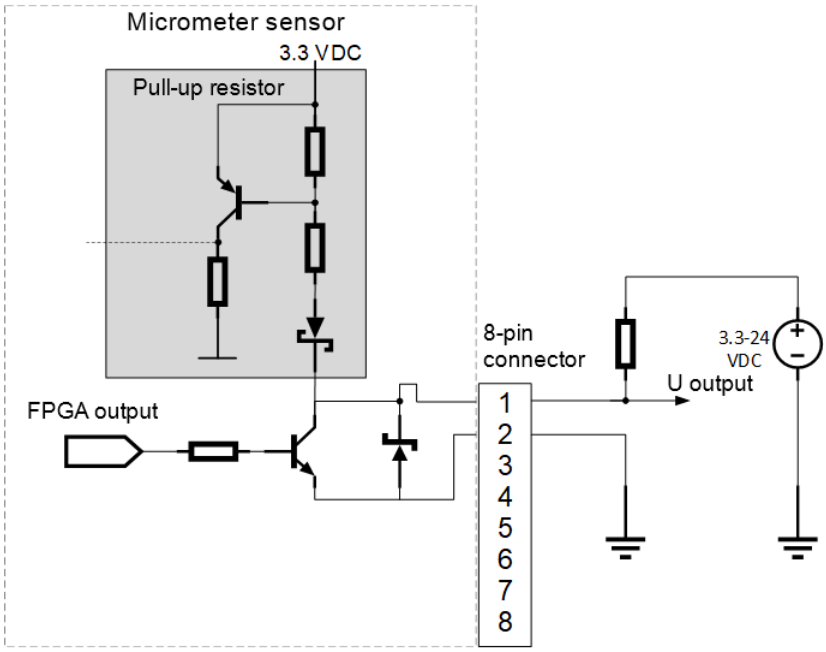


A typical variant of signal monitoring using the LED on the optoisolated output of the sensor:



Parameter	Value
Maximum allowable input voltage (exceeding this voltage may damage the micrometer sensor and void the warranty)	30V DC
Safe voltage level	3.3–24V DC
Leakage current	<60 μA
Maximum load current	50 mA

3. GPIO output

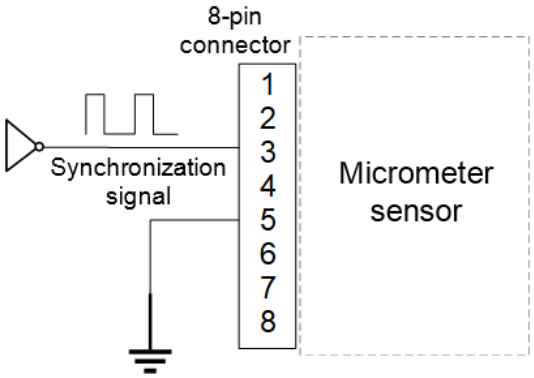


Parameter	Value
Maximum allowable input voltage (exceeding this voltage may damage the micrometer sensor and void the warranty)	30V DC
Safe voltage level	3.3–24 VDC
Internal pull-up resistor in an open collector circuit	≈2 kΩ
Leakage current	<60 μA
Maximum load current	50 mA

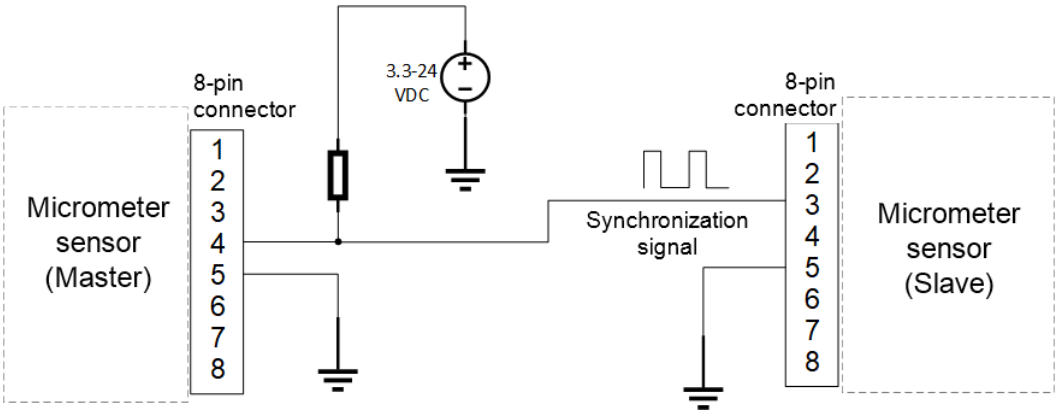
17.3. Wiring diagrams for synchronization signals

Optical sensors provide the ability to connect external synchronization signals.

External synchronization is connected to the optoisolated input of the sensor. Wiring diagram:



When synchronizing several sensors, one of them can be used as a synchronization source. Wiring diagram:

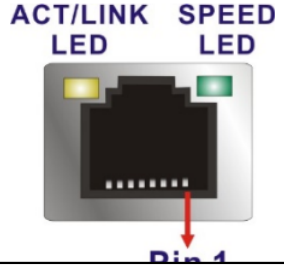


17.4. Pinout of controller connectors

1. Ethernet connector.
2. Multifunction connector.

Connector pins are shown in the tables below.

Gigabit Ethernet with RJ-45 connectors:

Pin number	Assignment	View
1	LAN_MDI0P	
2	LAN_MDI0N	
3	LAN_MDI1P	
4	LAN_MDI1N	
5	LAN_MDI2P	
6	LAN_MDI2N	
7	LAN_MDI3P	

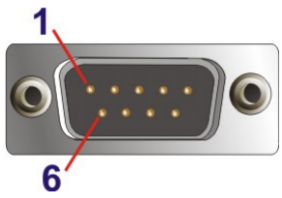
Pin number	Assignment	View
8	LAN_MDI3N	

The RJ-45 Ethernet connector has two LEDs, green/orange and yellow. Possible indicator signals:

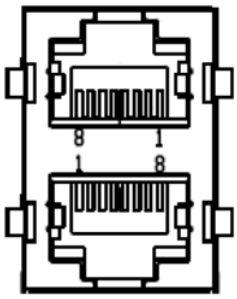
Yellow - Activity/Link LED		Green/Orange - Speed LED	
Status	**Description**	**Status**	**Description**
Off	No connection	Off	10 Mbps connection
Lit	Connected	Lit Green	100 Mbps connection
Blink	TX/RX activity	Lit Orange	1 Gbps connection

RS-232/422/485 serial ports with DB-9 connectors.

Interface type can be changed in BIOS (see Annex 2. Configuring the protocol for controller serial interfaces). The default interface type is RS-232.

Pin number	RS-232	RS-422	RS-485	View
1	DCD	TX-	TX-	
2	RX	TX+	TX+	
3	TX	RX+		
4	DTR	RX-		
5	GND			
6	DSR			
7	RTS			
8	CTS			
9	RI			

RS-232 serial ports with RJ-45 connectors:

Pin number	RS-232	View
1	RI	
2	DTR	
3	CTS	
4	TXD	
5	RTS	
6	RXD	
7	DSR	
8	DCD	

17.5. Cables

Optical sensor power cable, free leads:

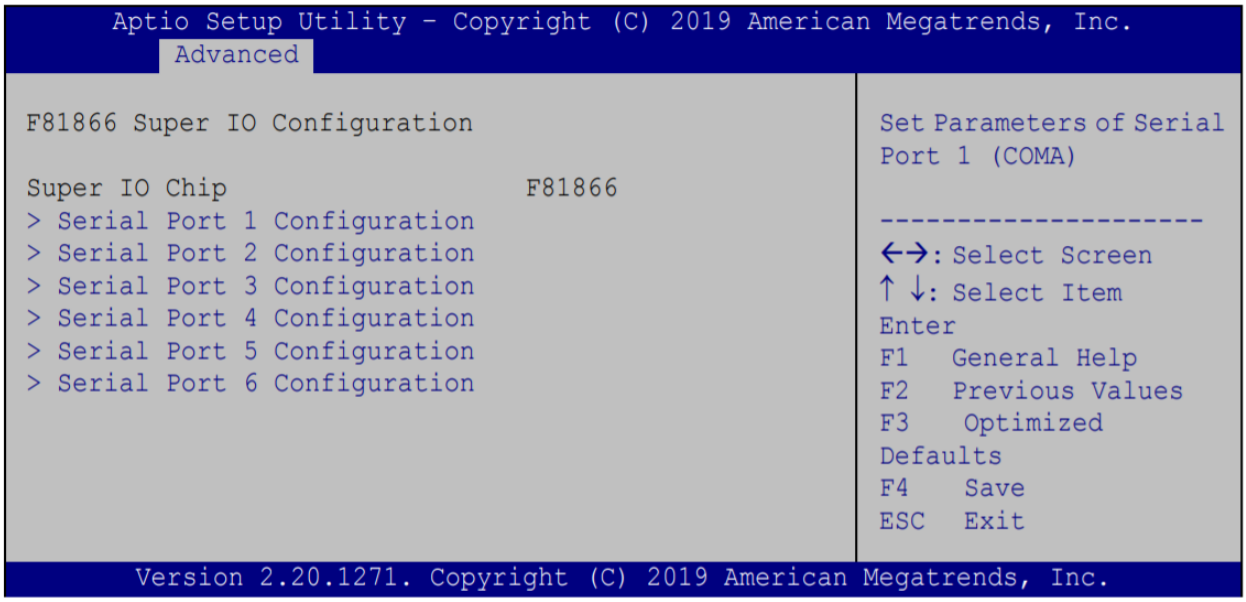
Wire color	Pin/connector number	Assignment
White	1	GPIO output
Brown	2	Power supply 0V, GND for GPIO
Green	3	OptoCoupled IN
Yellow	4	OptoCoupled Out
Grey	5	OptoCoupled GND
Pink	6	-
Blue	7	-
Red	8	Power supply 24V

18. Annex 2. Configuring the protocol for controller serial interfaces

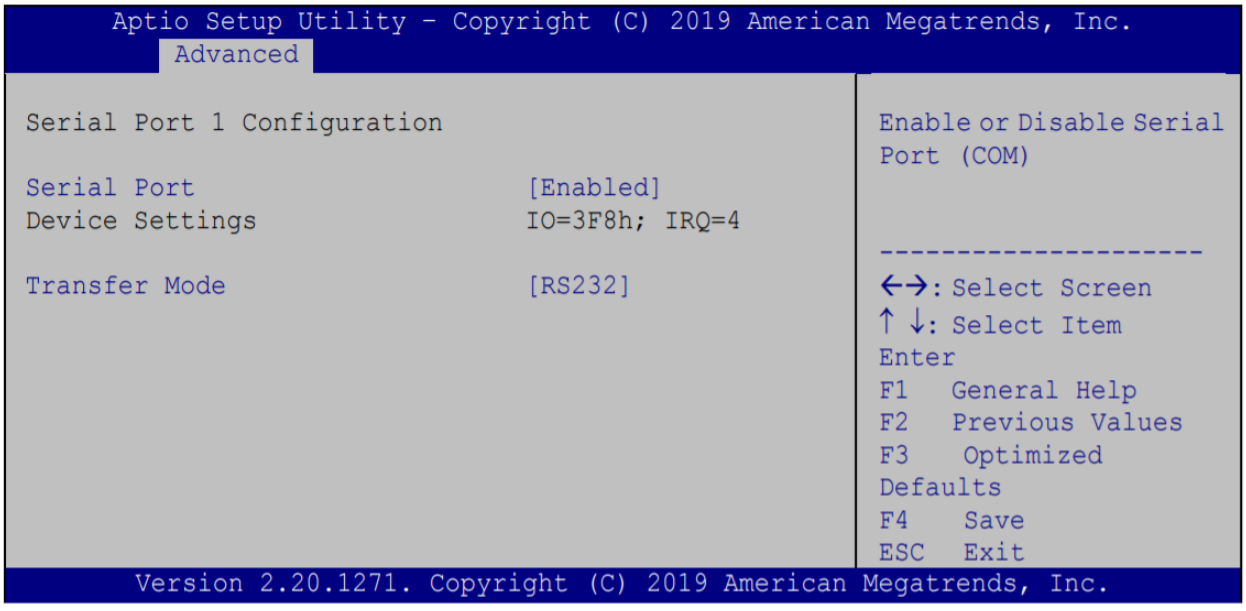
Changing the port type is only possible in the UEFI BIOS of the controller. To do this, it is necessary to connect a display and a keyboard to the controller.

To enter the BIOS menu, press the **DEL** key or the **F2** key immediately after turning on the controller.

The serial ports are configured on the **Advanced** tab in the **F81866 Super IO Configuration** menu:



Select the required menu item to configure the port.

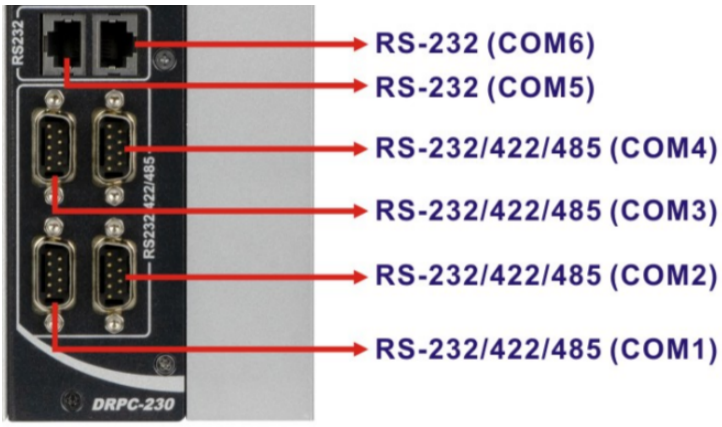


The following parameters are available for each serial port:

1. **Serial Port** - to enable/disable the port. Possible values:
   - Disabled
   - Enabled

2. **Transfer Mode** - to change the interface type. Possible values:
   - RS422
   - RS232
   - RS485

Ports are numbered as follows:



19. Annex 3. Modbus data types

The structure of composite data types transferred from the computational scheme (or to the scheme) via the Modbus protocol.

1. MessageBool - length 9 registers * 16 bit

Registers	+ 0	+ 1	+ 2	+ 3	+ 4	+5	+6	+7
address	id - int64				timestamp - int64			
address+8	value-bool(int16)							

2. MessageNumberInt8 - length 9 registers * 16 bit

Registers	+ 0	+ 1	+ 2	+ 3	+ 4	+5	+6	+7
start	id - int64				timestamp - int64			
address+8	value-int16							

3. MessageNumberInt16 - length 9 registers * 16 bit

Registers	+ 0	+ 1	+ 2	+ 3	+ 4	+5	+6	+7
start+	id - int64				timestamp - int64			
address+8	value-int16							

4. MessageNumberInt32 - length 10 registers * 16 bit

Registers	+ 0	+ 1	+ 2	+ 3	+ 4	+5	+6	+7
start+	id - int64				timestamp - int64			
address+8	value-int32							

5. MessageNumberInt64 - length 12 registers * 16 bit

Registers	+ 0	+ 1	+ 2	+ 3	+ 4	+5	+6	+7
start+	id - int64				timestamp - int64			
address+8	value-int64							

6. MessageDouble - length 10 registers * 16 bit

Registers	+ 0	+ 1	+ 2	+ 3	+ 4	+5	+6	+7
start+	id - int64				timestamp - int64			
address+8	value-float							

7. MessagePoint2dDouble - length 12 registers * 16 bit

Registers	+ 0	+ 1	+ 2	+ 3	+ 4	+5	+6	+7
start+	id - int64				timestamp - int64			
address+8	x-float		y-float					

8. MessageRect - length 16 registers * 16 bit

Registers	+ 0	+ 1	+ 2	+ 3	+ 4	+5	+6	+7
start+	id - int64				timestamp - int64			
address+8	x-float		y-float		width-float		height-float	

9. MessageSegmentLine - length 16 registers * 16 bit

Registers	+ 0	+ 1	+ 2	+ 3	+ 4	+5	+6	+7
start+	id - int64				timestamp - int64			
+8	point1.x-float		point1.y-float		point2.x-float		point2.y-float	

10. MessageStraightLine - length 14 registers * 16 bit

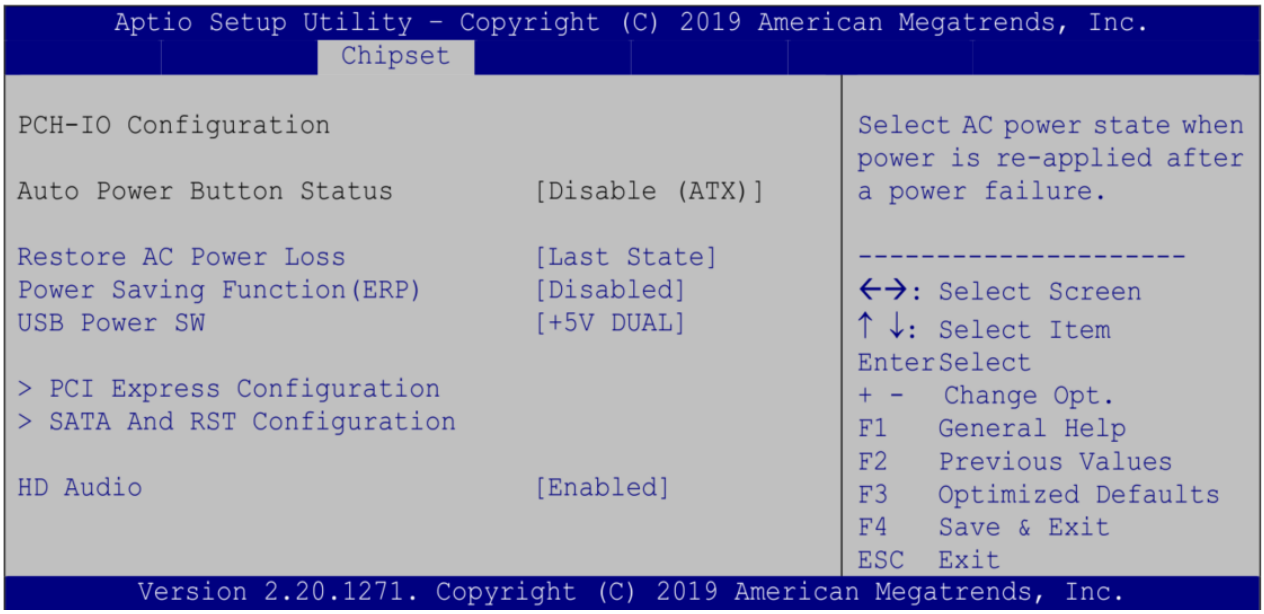
Registers	+ 0	+ 1	+ 2	+ 3	+ 4	+5	+6	+7
start+	id - int64				timestamp - int64			
+8	A-float		B-float		C-float			

20. Annex 4. Setting the controller's response to power supply. Automatic switching on

To change the controller's response to power supply, including turning it on automatically, it is necessary to use the UEFI BIOS. To do this, you need to connect the display and keyboard to the controller.

To enter the BIOS menu, press the **DEL** or **F2** keys immediately after turning on the controller.

Change the **Restore AC Power Loss** parameter on the **Chipset** tab in the **PCH-IO Configuration** menu:



The following options are available for the **Restore AC Power Loss** parameter:

- "Power Off" - The system remains turned off after power is applied.
- "Power On" - The controller automatically turns on after power is applied, regardless of the previous state.
- "Last State" - The controller turns on after power is applied only if it was on at the time the power was removed.

After changing the parameter, save the changes by selecting the appropriate item in the **Save & Exit** tab.

21. Distributors

AUSTRALIA

Applied Measurement Australia Pty Ltd
RAILWAY INSTRUMENTS ONLY
Thornton Plaza, Unit 5,
27 Thornton Crescent, Mitcham
VIC 3132, Australia
Tel: +61 39874 5777
Fax: +61 39874 5888
sales@appliedmeasurement.com.au
www.appliedmeasurement.com.au

BELGIUM

Althen Sensors & Controls BV
Verrijn Stuartlaan 40, 2288 EL,
Rijswijk, Leidschendam
The Netherlands
Tel: +31 0 70 392 4421
Tel: +31 0 61 396 7830
Tel: +31 0 64 323 8393
sales@althen.nl
info@althen.nl
www.althensensors.com

BOSNIA AND HERZEGOVINA

ASCO RAIL sp. z o.o.
EXCLUSIVE REPRESENTATIVE FOR RAILWAY EQUIPMENT
ul. Wielowiejska 53, 44-120
Pyskowice, Poland
Tel: +48 32 230 45 70
Fax: + 48 32 233 21 34
biuro@ascorail.pl
export@ascorail.pl
www.ascorail.pl

BRAZIL

CAPI Controle e Automacao Ltda
Rua Itororo, 121, CEP 13466-240
Americana-SP, Brazil
Tel: +55 19 36047068
Fax: +55 19 34681791
capi@capicontrole.com.br
www.capicontrole.com.br

BULGARIA

ASCO RAIL sp. z o.o.
EXCLUSIVE REPRESENTATIVE FOR RAILWAY EQUIPMENT
ul. Wielowiejska 53, 44-120
Pyskowice, Poland
Tel: +48 32 230 45 70
Fax: + 48 32 233 21 34
biuro@ascorail.pl
export@ascorail.pl
www.ascorail.pl

CHILE

MOL INGENIERIA LTDA
EXCLUSIVE REPRESENTATIVE FOR RAILWAY EQUIPMENT
Republica de Honduras 11936
Las Condes, Santiago de Chile
Tel: +56 9 59200362
hconcha@molingenieria.com
www.molingenieria.com

CHILE

Verne SpA
Apoquindo 2818, oficina 31
Las Condes, Santiago, Chile
Tel: +56 2 228858633
info@verne.cl
jsaavedra@verne.cl
www.verne.cl

CHINA

Beijing Haiwei Lutong Technology Co., Ltd
Yard 1, Tianxing Street, Fangshan
District, Beijing, China
Tel: +86 10 8366 1866
Fax: +86 10 8366 1866
info@haiwlt.com
www.haiwlt.com

CHINA

Chongqing Wolf Industrial Technology Co., Ltd
Room 2307 / 2308, Light of City
international business building,
No. 19 Jiangnan Avenue, Nan'an
District, Chongqing, China
Tel: 023 62832618
Fax: 023 62832113
info@wolf-hk.com
www.wolf-hk.com

CHINA

Beijing Gemston Mechanical & Electrical Equipment Co., Ltd
RAILWAY INSTRUMENTS ONLY
Room 613, Anfu Mansion, Fengtai
District, Beijing, China
Tel: +86 10 6765 0516
Fax: +86 10 6765 6966
Mobile: +86 137 1755 1423
dh0526@163.com
www.baoft.cn

CHINA

Xi'an Win-Success Automation Technology Co.,Ltd
Room 3-1-1039, Iduhui Building,
No.11 Tangyan South Road
High-Tech Zone, Xi'an
Shaanxi PRC, China
Tel: +86 29 81106280
Fax: +86 29 81106285
Mob: +86 133 19271405
info@maxsenor.com
www.maxsensor.com

CHINA

Micron-Metrology co., Ltd
No.2, Kecheng Rd., Industrial Park
District, Suzhou,
Jiangsu Province., China
Tel: 0512 65589760
Mob: +86 189 1806 9807
sales@micron-metrology.cn
www.micron-metrology.cn

CHINA

Zhenshangyou Technologies Co., Ltd
Rm 2205-2210, Zhongyou Hotel
1110 Nanshan Road, Nanshan
District 518054 Shenzhen, China
Tel: +86 755-26528100/8011/8012
Fax: +86 755-26528210/26435640
info@51sensors.com
www.51sensors.com

CROATIA

ASCO RAIL sp. z o.o.
EXCLUSIVE REPRESENTATIVE FOR RAILWAY EQUIPMENT
ul. Wielowiejska 53, 44-120
Pyskowice, Poland
Tel: +48 32 230 45 70
Fax: + 48 32 233 21 34
biuro@ascorail.pl
export@ascorail.pl
www.ascorail.pl

CZECH REPUBLIC

ASCO RAIL sp. z o.o.
EXCLUSIVE REPRESENTATIVE FOR RAILWAY EQUIPMENT
ul. Wielowiejska 53, 44-120
Pyskowice, Poland
Tel: +48 32 230 45 70
Fax: + 48 32 233 21 34
biuro@ascorail.pl
export@ascorail.pl
www.ascorail.pl

DENMARK

BLConsult
Ryssbalt 294
95 291 Kalix, Sweden
Tel: +46 70 663 19 25
info@blconsult.se
www.blconsult.se

ESTONIA

FoodLab OU
Haabersti linnaosa, Astangu tn 52
13519 Eesti, Tallinn, Estonia
Tel: +372 56 363110
foodlab.ee@gmail.com

FINLAND

Kvalitest Industrial AB
EXCEPT FOR RAILWAY INSTRUMENTS
Ekbacksvagen 28,
16869 Bromma, Sweden
Tel: +46 0 76 525 5000
sales@kvalitest.com
www.kvalitest.com
www.kvalitest.se

FINLAND

TERASPYORA-STEELWHEEL OY
RAILWAY INSTRUMENTS ONLY
Juvan teollisuuskatu 28
FI-02920 ESPOO, Finland
Tel: +358 400 422 900
Fax: +358 9 2511 5510
steelwheel@steelwheel.fi
www.teraspyora.fi

FRANCE

BLET Measurement Group S.A.S.
1 avenue du President Georges
Pompidou, 92500 Rueil
Malmaison, France
Tel: + 33 0 1 80 88 57 85
Fax: +33 0 1 80 88 57 93
technique@blet-mesure.fr
www.blet-mesure.fr

GERMANY

Disynet GmbH
Breyeller Str. 2, 41379
Brueggen, Germany
Tel: +49 2157 8799 0
Fax: +49 2157 8799 22
disynet@sensoren.de
www.sensoren.de

GERMANY

Finger GmbH & Co. KG
OPTICAL MICROMETERS ONLY
Sapelloh 172, 31606
Warmsen, Germany
Tel: +49 5767 96020
Fax: +49 5767 93004
finger@finger-kg.de
www.finger-kg.de

GERMANY

ALTHEN GmbH MeЯ- und Sensortechnik
Dieselstrasse 2, 65779
Kelkheim, Germany
Tel: +49 0 6195 7 00 60
info@althen.de
www.althensensors.com/de/

HUNGARY

ASCO RAIL sp. z o.o.
EXCLUSIVE REPRESENTATIVE FOR RAILWAY EQUIPMENT
ul. Wielowiejska 53, 44-120
Pyskowice, Poland
Tel: +48 32 230 45 70
Fax: + 48 32 233 21 34
biuro@ascorail.pl
export@ascorail.pl
www.ascorail.pl

INDIA

Influx Big Data Solutions Pvt Ltd
No:2, Krishvi, Ground Floor,
Old Airport Road, Domlur,
Bangalore - 560071, India
Tel: +91 73 37748490
Tel: +91 94 48492380
milan@influxtechnology.com
support_india@influxtechnology.com
www.influxtechnology.com

INDIA

Paragon Instrumentation Engineers Pvt. Ltd.
RAILWAY INSTRUMENTS ONLY
200, Station Road,
Roorkee, 247 667, India
Tel: +91 1332 272394
tanuj@paragoninstruments.com
www.paragoninstruments.com

INDONESIA

PT. DHAYA BASWARA SANIYASA
Botanic Junction Blok H-9 NO. 7
Mega Kebon Jeruk, Joglo
Jakarta,11640, Indonesia
Tel: +62 21 2932 5859
management@ptdbs.co.id

ISRAEL

Nisso Dekalo Import Export LTD
1 David Hamelech Street
Herzlia 46661 Israel
Tel: +972 99577888
Fax: +972 99568860
nissodekaloltd@outlook.com
www.fly-supply.net
www.aircraft-partsupply.com

ITALY

FAE s.r.l.
Via Tertulliano, 41
20137 Milano, Italy
Tel: +39 02 55187133
Fax: +39 02 55187399
fae@fae.it
www.fae.it

JAPAN

Tokyo Instruments, Inc.
6-18-14 Nishikasai, Edogawa-ku,
Tokyo, 134-0088 Japan
Tel: +81 3 3686 4711
Fax: +81 3 3686 0831
f_kuribayashi@tokyoinst.co.jp
www.tokyoinst.co.jp

LATVIA

FoodLab OU
Haabersti linnaosa, Astangu tn 52
13519 Eesti, Tallinn, Estonia
Tel: +372 56363110
foodlab.ee@gmail.com

LUXEMBOURG

Althen Sensors & Controls BV
Verrijn Stuartlaan 40, 2288 EL,
Rijswijk, Leidschendam
The Netherlands
Tel: +31 0 70 392 4421
Tel: +31 0 61 396 7830
Tel: +31 0 64 323 8393
sales@althen.nl
info@althen.nl
www.althensensors.com

MALAYSIA

OptoCom InstruVentures
H-49-2, Jalan 5, Cosmoplex
Industrial Park, Bandar Baru
Salak Tinggi, Sepang, Malaysia
Tel: 603 8706 6806
Fax: 603 8706 6809
optocom@tm.net.my
www.optocom.com.my

MONTENEGRO

ASCO RAIL sp. z o.o.
EXCLUSIVE REPRESENTATIVE FOR RAILWAY EQUIPMENT
ul. Wielowiejska 53, 44-120
Pyskowice, Poland
Tel: +48 32 230 45 70
Fax: + 48 32 233 21 34
biuro@ascorail.pl
export@ascorail.pl
www.ascorail.pl

NETHERLANDS

Althen Sensors & Controls BV
Verrijn Stuartlaan 40, 2288 EL,
Rijswijk, Leidschendam
The Netherlands
Tel: +31 0 70 392 4421
Tel: +31 0 61 396 7830
Tel: +31 0 64 323 8393
sales@althen.nl
info@althen.nl
www.althensensors.com

NORWAY

BLConsult
Ryssbalt 294,
95 291 Kalix, Sweden
Tel: +46 70 663 19 25
info@blconsult.se
www.blconsult.se

NORWAY

Salitec AS
PB 468, N-1327
Lysaker, Norway
Tel: +47 23 891015
Fax: +47 92101005
mail@salitec.no
www.salitec.no

PERU

Verne Peru S.A.C.
Las Codornices 104,
Surquillo, Lima, Peru
Tel/fax: +51 992436734
info@verne.cl
www.verne.cl

POLAND

ASCO RAIL sp. z o.o.
EXCLUSIVE REPRESENTATIVE FOR RAILWAY EQUIPMENT
ul. Wielowiejska 53, 44-120
Pyskowice, Poland
Tel: +48 32 230 45 70
Fax: + 48 32 233 21 34
biuro@ascorail.pl
export@ascorail.pl
www.ascorail.pl

POLAND

RIFTEK EUROPE sp. z o.o.
ul. Domaniewska 17/19, 02-672
Warsaw, Poland
info@riftek.com
www.riftek.com

PORTUGAL

Campal Inovacoes Ferroviarias Lda.
Lagoas Park, Edificio 7, 1° Piso
Sul, 2740-244 Porto Salvo, Oeiras,
Portugal
Tel: +351 21 584 4348
campal@campal.pt
www.campal.pt

SERBIA

ASCO RAIL sp. z o.o.
EXCLUSIVE REPRESENTATIVE FOR RAILWAY EQUIPMENT
ul. Wielowiejska 53, 44-120
Pyskowice, Poland
Tel: +48 32 230 45 70
Fax: + 48 32 233 21 34
biuro@ascorail.pl
export@ascorail.pl
www.ascorail.pl

SLOVAKIA

ASCO RAIL sp. z o.o.
EXCLUSIVE REPRESENTATIVE FOR RAILWAY EQUIPMENT
ul. Wielowiejska 53, 44-120
Pyskowice, Poland
Tel: +48 32 230 45 70
Fax: + 48 32 233 21 34
biuro@ascorail.pl
export@ascorail.pl
www.ascorail.pl

SLOVENIA

ASCO RAIL sp. z o.o.
EXCLUSIVE REPRESENTATIVE FOR RAILWAY EQUIPMENT
ul. Wielowiejska 53, 44-120
Pyskowice, Poland
Tel: +48 32 230 45 70
Fax: + 48 32 233 21 34
biuro@ascorail.pl
export@ascorail.pl
www.ascorail.pl

SOUTH KOREA

BS Holdings
B-201,Wonpogongwon 1ro,
59 Danwon-gu, Ansan-si,
Gyeonggi-do 15455, Republic of Korea
Tel: +82 31 411 5011
Fax: +82 31 411 5015
bsh5011@hanmail.net
www.lasersolution.co.kr

SOUTH KOREA

PROSEN. CO., LTD
M-1001, Songdo techno park IT center, 32, Songdogwahak-ro,
Yeonsu-gu, Incheon, 21984,
Republic of Korea
Tel: +82 32 811 3457
Fax: +82 32 232 7458
trade@prosen.co.kr
www.prosen.co.kr

SPAIN

IBERFLUID Instruments S.A.
C/ Botanica, 122, 08908
L'Hospitalet de Llobregat
Barcelona
Tel: +34 93 447 10 65
Fax: +34 93 334 05 24
myct@iberfluid.com
www.iberfluid.com

SWEDEN

BLConsult
Ryssbalt 294,
95 291 Kalix, Sweden
Tel: +46 70 663 19 25
info@blconsult.se
www.blconsult.se

SWEDEN

Kvalitest Industrial AB
EXCEPT FOR RAILWAY INSTRUMENTS
Ekbacksvagen 28,
16869 Bromma, Sweden
Tel: +46 0 76 525 5000
sales@kvalitest.com
www.kvalitest.com
www.kvalitest.se

SWITZERLAND

ID&T GmbH
Gewerbestrasse 12/a
8132 Egg (Zurich), Switzerland
Tel: + 41 44 994 92 32
Fax: + 41 44 994 92 34
info@idtlaser.com
www.idtlaser.com

THAILAND

Advantech Solution Co., Ltd.
20/170 Motorway Rd.,
Kwang Pravet, Khet Pravet,
Bangkok, Thailand 10250
Tel: +662 1848705
Fax: +662 1848708
sales@advantechsolution.com
www.advantechsolution.com

TURKEY

MAK Elektronik Malzeme Analiz ve Kalite Kontrol Cihazlari Dis Tic. Ltd. Sti.
Cenap Sahabettin Sokak, No:39,
34718 Kosuyolu - Kadikoy /
Istanbul - TURKEY
Tel: +90 216 402 10 34
Fax: +90 216 402 10 35
ulastac@metalografi.net
www.makelektronik.com.tr

TURKEY

TEKMA Muhendislik A.S.
Cevizli Mh. M. Kemal Cd.,
Hukukcular Towers,
A-Blok, No: 66-A/39
Kartal - Istanbul
Tel: +90 216 970 1318
Tel: +90 850 840 2334
info@tekma.eu
www.tekma.eu

UKRAINE

KODA
Frunze st. 22, 61002,
Harkov, Ukraine
Tel/Fax: +38 057 714 26 54
mail@koda.com.ua
www.koda.com.ua

UNITED KINGDOM, IRELAND

Althen UK
Northamptonshire
United Kingdom
Tel: +44 0 7823 921427
t.stoyles@althen.co.uk
www.althensensors.com
www.althencontrols.com

USA

Althen Sensors & Controls
2531 Bradley St., Oceanside, CA, 92056, USA
Tel: 858 633 3572
r.ream@althensensors.com

USA, CANADA, MEXICO

Acuity Products of Schmitt Industries, Inc.
2765 NW Nicolai Street
Portland, OR, 97210, USA
Tel: +1 503 227 7908
Fax: +1 503 223 1258
sales@acuitylaser.com
www.acuitylaser.com

USA, CANADA, MEXICO

International Electronic Machines Corporation
RAILWAY INSTRUMENTS ONLY
850 River Street, Troy,
New York, USA
Tel: +1 518 268-1636
Fax: +1 518 268-1639
marketing@iem.net
www.iem.net