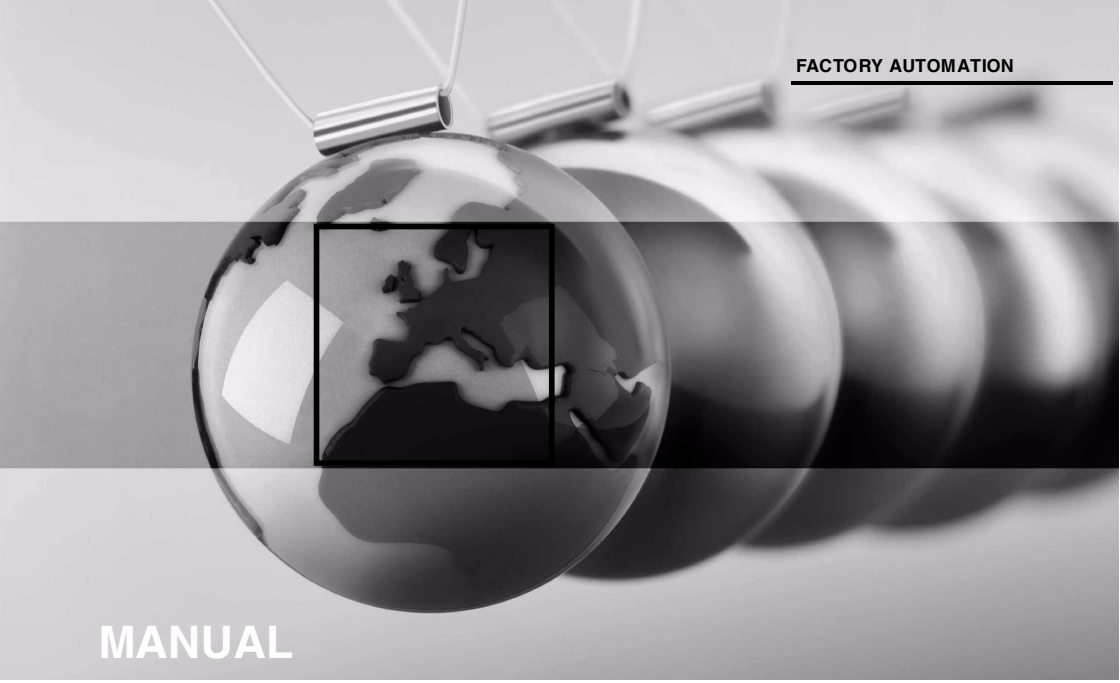
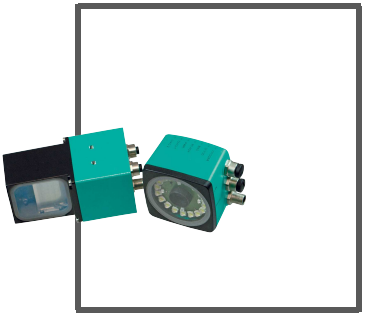
FACTORY AUTOMATION



MANUAL

OPC / OPD / OPE
Optical Print Inspector



CE

PEPPERL+FUCHS
SENSING YOUR NEEDS

With regard to the supply of products, the current issue of the following document is applicable: The General Terms of Delivery for Products and Services of the Electrical Industry, published by the Central Association of the Electrical Industry (Zentralverband Elektrotechnik und Elektroindustrie (ZVEI) e.V.) in its most recent version as well as the supplementary clause: "Expanded reservation of proprietorship"

2015-07

2015-07

1 Introduction

Informative Symbols

Note!

This symbol brings important information to your attention.

Action

This symbol indicates a paragraph with instructions.

Contact

If you have any questions about the device, its functions, or accessories, please contact us at:

Pepperl+Fuchs GmbH
Lilienthalstraße 200
68307 Mannheim
Telephone: +49 621 776-4411
Fax: +49 621 776-274411
E-Mail: fa-info@pepperl-fuchs.com

2015-07

2 Declaration of Conformity

This product was developed and manufactured under observance of the applicable European standards and guidelines.

Note!

A Declaration of Conformity can be requested from the manufacturer.

The product manufacturer, Pepperl+Fuchs GmbH, D-68307 Mannheim, has a certified quality assurance system that conforms to ISO 9001.

ISO9001

3 Safety

3.1 Symbols Used

This document contains information that you must observe for your own personal safety and to prevent property damage. Warning messages are shown in descending order according to the risk level, as follows:

Safety-Relevant Symbols

Danger!
This symbol indicates an imminent danger.

Non-observance will result in personal injury or death.

Warning!
This symbol indicates a possible fault or danger.

Non-observance may cause personal injury or serious property damage.

Caution!
This symbol indicates a possible fault.

Non-observance could interrupt the device and any connected systems and plants, or result in their complete failure.

3.2 Intended Use

The stationary read devices OPC-*, OPD-*, and OPE-* are used to decode 1-D and 2-D codes.

The device is only approved for appropriate and intended use. Ignoring these instructions will void any warranty and absolve the manufacturer from any liability.

Protection of the personnel and the plant is not ensured if the device is not being used according to its intended use.

4 Product Description

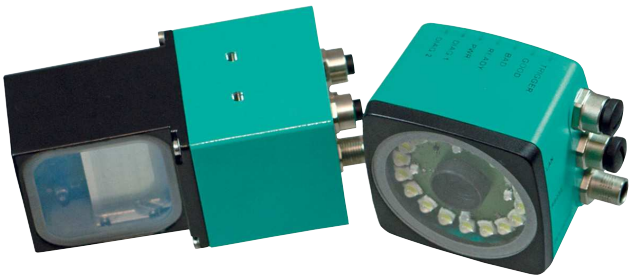
4.1 Use and Application

The optical print inspector is available in three versions and reads both 1-D and 2-D codes. For high speed, great depth of focus, long barcodes, or detection tasks; with the optical print inspector you always have the right vision sensor at hand. Use the innovative image processing technology from Pepperl+Fuchs for your application.

You can configure the stationary readers quickly and easily over a standard Ethernet interface, using the software provided. The reader has an integrated fault pattern memory.

Highlights:

- Reads 1-D and 2-D codes at extremely high speeds of up to 10 m/s at 30 readings/s
- Detects long barcodes of up to 200 mm long using multiple image capture
- Reads up to four different codes at the same time
- Logo comparison
- Position of the reading field can be adjusted flexibly via rotary encoder input (OPC and OPD)
- Captures codes and logos at different distances and of different sizes with one setting
- Side lighting for reliable reading with highly reflective surfaces



4.2 Indicators and Operating Controls

OPC reader and OPD reader

The illumination unit has 7 LED indicators that provide information on the various device statuses.

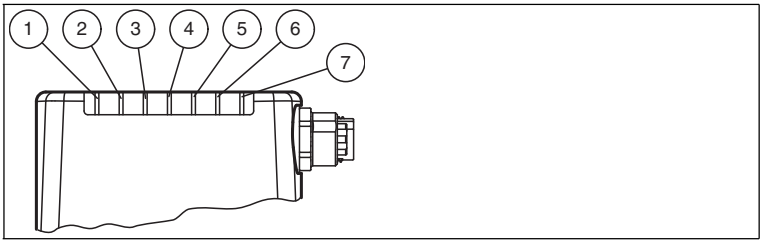


Figure 4.1 Indicators and Operating Controls

1. **DIAG 2**
   Yellow LED. Generates different flashing sequences to signal diagnostic messages.
2. **DIAG 1**
   Yellow LED. Generates different flashing sequences to signal diagnostic messages.
3. **Power (PWR)**
   Lights up green when the sensor is ready for operation.
4. **Ready (READY)**
   Lights up yellow when the sensor is ready.
5. **Reading process triggered (BAD)**
   Lights up yellow if the reading was unsuccessful.
6. **Reading process triggered (GOOD)**
   Lights up yellow if the reading was successful.
7. **Trigger sensor (TRG)**
   Lights up yellow when a connected trigger sensor is activated.

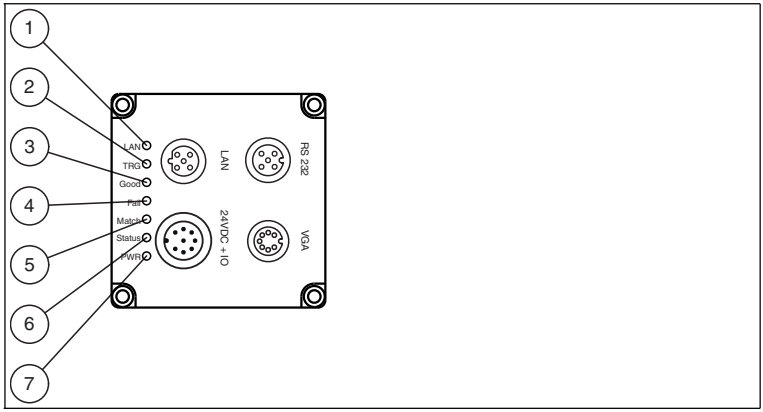
Caution!
Software update

The **READY** LED flashes while the sensor is being programmed. During this time, the sensor must **not** be switched off.

Definition of the LEDs and Outputs for Individual Statuses

	GOOD LED	**BAD** LED	Match code operating mode	Status	Good output	Bad output	Match code output
Good read	ON	OFF	OFF	80	ON	OFF	OFF (Not used)
Bad read	OFF	ON	OFF	81	OFF	ON	OFF (Not used)
Good read and match code OK	ON	OFF	ON	82	ON	OFF	ON
Good read and match code not OK	OFF	ON	ON	83	OFF	ON	OFF
Bad read	OFF	ON	ON	81	OFF	ON	OFF
Decoder timeout	OFF	ON		84	OFF	ON	OFF

OPE reader



1. **Network (LAN):**
   Lights up yellow as soon as a physical connection is established.
2. **Trigger (TRG):**
   Lights up yellow when a connected trigger sensor is activated.
3. **Good (Good):**
   Lights up green if the reading is successful.
4. **Error (Fail):**
   Lights up red if the reading is unsuccessful.
5. **Result (Match):**
   Flashes green if the match code matches the read code.
   Flashes red if the match code does not match the read code.
6. **Status (Status):**
   Briefly flashes yellow when the device is switched on.
   Flashes yellow during a firmware update.
   Lights up green when the device is ready.
   Lights up red in the event of a device error.
7. **Power (PWR):**
   Lights up green when the sensor is connected to a power supply.

4.3 Interfaces and Connections

OPC reader and OPD reader

The device includes the following connections:

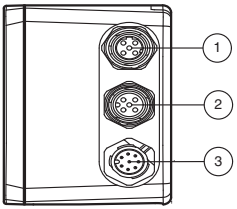


Figure 4.2 Device Connections

1. Network (4-pin M12 socket)
2. Input IO (5-pin M12 socket)
3. Power supply, inputs, and outputs (8-pin M12 plug)

Power Supply

There is an 8-pin M12 plug on the side of the housing for connecting the power supply, and the inputs and outputs. The following diagram shows the pinning:

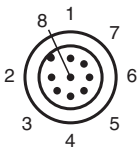


Figure 4.3 Connection layout for operating voltage, and inputs and outputs

1. Trigger IN
2. + UB
3. OUT Good
4. OUT Bad
5. IN 1
6. OUT 1
7. GND
8. OUT match code

2015-07

RS-232 Interface

There is a 5-pin M12 socket on the side of the sensor housing. The following diagram shows the pinning:

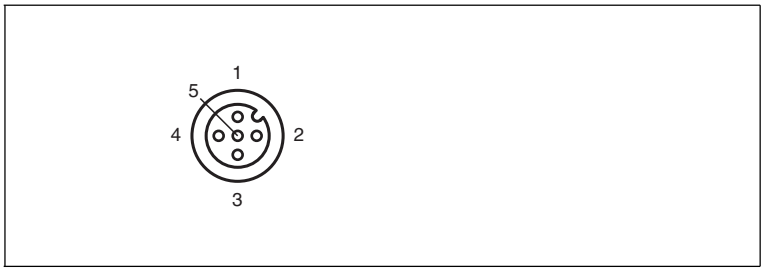


Figure 4.4 RS-232 input connection layout

1. + UB
2. TX RS-232
3. GND
4. RX RS-232
5. NC

Network

There is a 4-pin M12 socket on the side of the housing for connecting to the network. The following diagram shows the pinning:

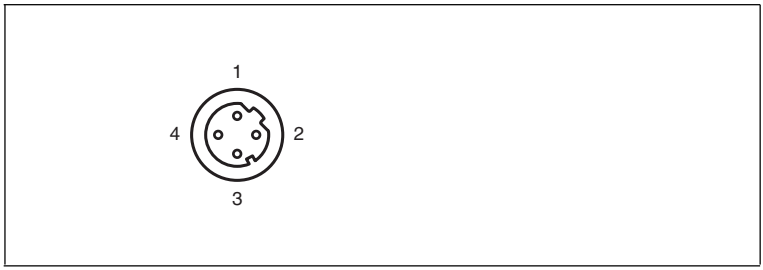
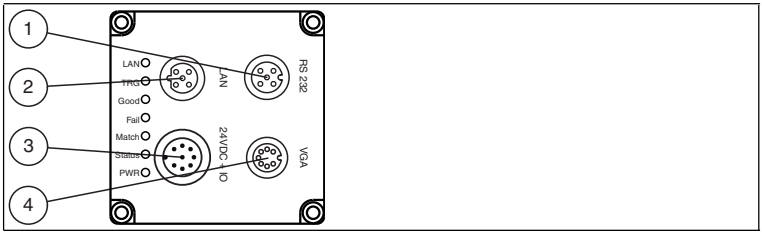


Figure 4.5 Network connection layout

1. TX+ Ethernet
2. RX+ Ethernet
3. TX- Ethernet
4. RX- Ethernet

OPE reader

The device includes the following connections:



1. RS-232 interface (5-pin M12 socket)
2. Network (4-pin M12 socket)
3. Power supply, inputs, and outputs (8-pin M12 plug)
4. VGA output (7-pin M12 socket)

Power Supply

There is an 8-pin M12 plug on the back of the sensor housing to connect the power supply, and to the inputs and outputs. The following diagram shows the pinning:

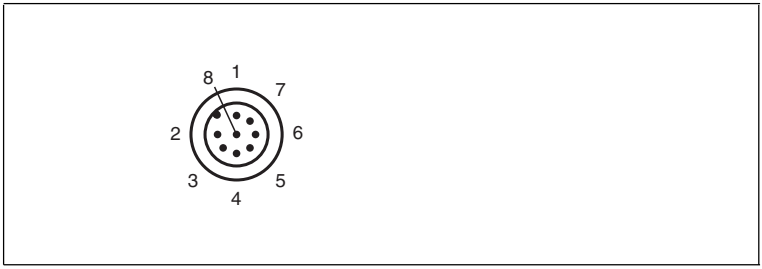


Figure 4.6 Connection layout for operating voltage, and inputs and outputs

1. IN TRG / OUT 1
2. +UB
3. OUT Good / IN 1
4. OUT Fail / IN 2
5. IN 3
6. IN 4 / OUT 2
7. GND
8. OUT Match

Pin 1, pin 3, pin 4, and pin 6 have dual assignments, but these dual assignments are not currently supported by the software: OUT 1, IN 1, IN 2, IN 3, IN 4 are intended for future applications.

Network

There is a 4-pin M12 socket on the back of the sensor housing to connect to the network. The following diagram shows the pinning:

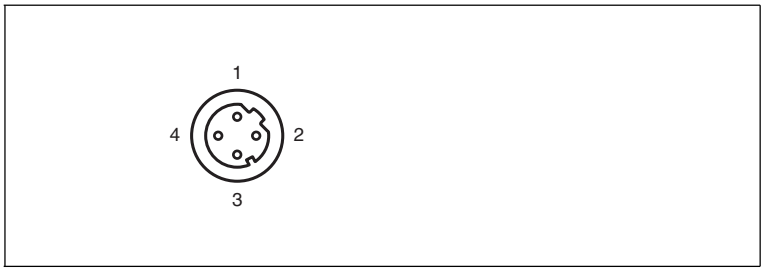


Figure 4.7 Network connection layout

1. TX+ Ethernet
2. RX+ Ethernet
3. TX- Ethernet
4. RX- Ethernet

RS-232 Interface

There is a 5-pin M12 socket on the back of the sensor housing for connecting the RS-232 interface or external lighting. When using as the RS-232 interface, do not connect any cables to pin 1 or pin 5. The following diagram shows the pinning:

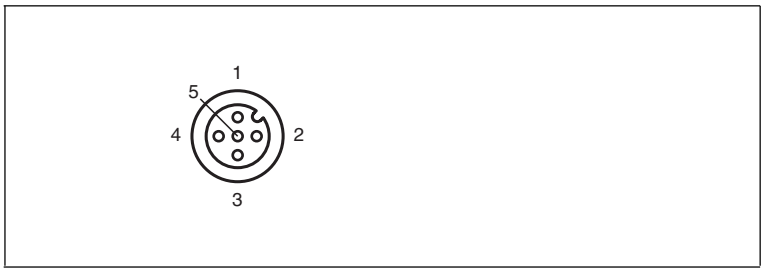
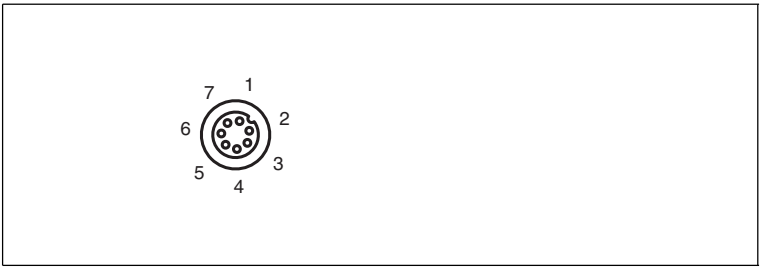


Figure 4.8 RS-232 connection layout

1. +UB
2. TX RS-232
3. GND
4. RX RS-232
5. IN 5 / OUT 3

VGA Output

There is a 7-pin M9 socket on the back of the sensor housing to connect the VGA plug. The following diagram shows the pinning:



1. OUT VSYNC
2. GND
3. OUT R
4. OUT G
5. GND
6. OUT B
7. OUT HSYNC

4.4 Scope of Delivery

- Optical Print Inspector
- Quick start guide

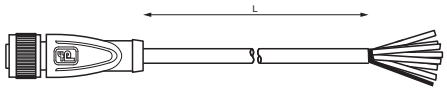
4.5 Accessories

Various accessories are available.

4.5.1 Power Supply

Use the following double-ended cordset to connect the power supply, inputs, and outputs to the sensor.

Female field connector

Model number			
V19-G-2M-PUR-ABG	8-pin M12 socket, straight	L = 2 m	Open cable end with multistranded conductors
V19-G-5M-PUR-ABG	8-pin M12 socket, straight	L = 5 m	Open cable end with multistranded conductors
V19-G-10M-PUR-ABG	8-pin M12 socket, straight	L = 10 m	Open cable end with multistranded conductors

2015-07

Other lengths on request.

Field-attachable M12 socket

Model number	
V19-G-ABG-PG9	■ 8-pin M12 socket, straight ■ Screw terminals for max. 0.75 mm^2 PG9 cable gland ■ Cable diameter: 5 mm ... 8 mm

4.5.2 Network Cable

The sensor is connected to the network using an M12 plug.

Designation	Description
V45-G	RJ45 network plug, field attachable
V1S-G	M12 plug, 4-pin, field attachable
V1SD-G-2M-PUR-ABG-V45X-G	Double-ended cordset, RJ45 network plug with M12 plug, crossed, 4-pin
V1SD-G-2M-PUR-ABG-V45-G	Double-ended cordset, RJ45 network plug with M12 plug, 4-pin

4.5.3 RS-232 Interface

The sensor's RS-232 interface is connected via an M12 plug.

Designation	Description
V15S-G-5M-PUR-ABG	Male single-ended cordset, M12, 5-pin, PUR cable, shielded cap nut
V15S-G-5M-PUR-ABG-SUBD9	Connection cable, M12 plug, 5-pin, 9-pin to D-Sub housing

4.5.4 VGA Output

The sensor is connected to a monitor using an M9 plug.

Designation	Description
ODZ-MAC-CAB-VIDEO	Video connection cable, round plug, 7-pin on D-Sub socket, 15-pin VGA, length: 2 m

5 Installation

5.1 Preparation

Unpacking the unit

1. Check that all package contents are present and undamaged.

   ↳ If anything is damaged, inform the shipper and contact the supplier.

2. Check that all items are present and correct based on your order and the shipping documents.

   ↳ If you have any questions, please contact Pepperl+Fuchs.

3. Keep the original packing material in case you need to store or ship the unit at a later time.

5.2 Mounting the Device

Note!

Mounting an optical device

- Do not aim the sensor at the sun.
- Protect the sensor from direct long-term exposure to sun.
- Prevent condensation from forming by not exposing the sensor to any major fluctuations in temperature.
- Do not expose the sensor to the effects of any aggressive chemicals.
- Keep the lenses and reflector of the device clean. Clean with a soft cloth, using standard commercial glass cleaner if necessary.

We recommend to clean the optical surface and to check screw fittings and electrical connections at regular intervals.

Note!

Preventing reflection and glare

Reflection and glare from reflective surfaces can impair the captured image and therefore lead to incorrect readings. To prevent reflection and glare, install the stationary reading device at a slight angle.

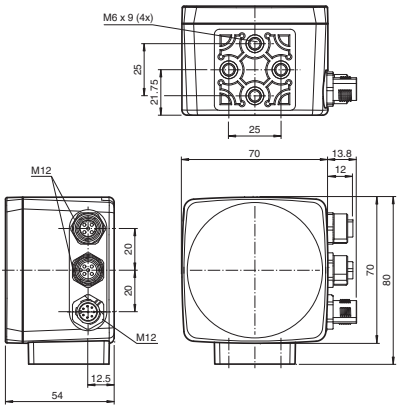
The surface must be level to prevent the housing from becoming distorted when the fittings are tightened. After mounting the sensor, ensure that there is still enough space to connect the connection cable to the sensor.

OPC and OPD mounting

The reader has four symmetrically positioned M6 threads on the base of the housing to allow easy mounting of the reader in your plant.

The read distance differs according to the reader. You can find the correct read distance in the technical data for the reader to be installed.

The following illustration shows all the relevant housing dimensions in mm:



OPE mounting

The device has two M5 threads positioned on the base of the housing to allow easy mounting of the reader in your plant.

The read distance differs according to the reader. You can find the correct read distance in the technical data for the reader to be installed.

The following illustration shows all the relevant housing dimensions in mm:

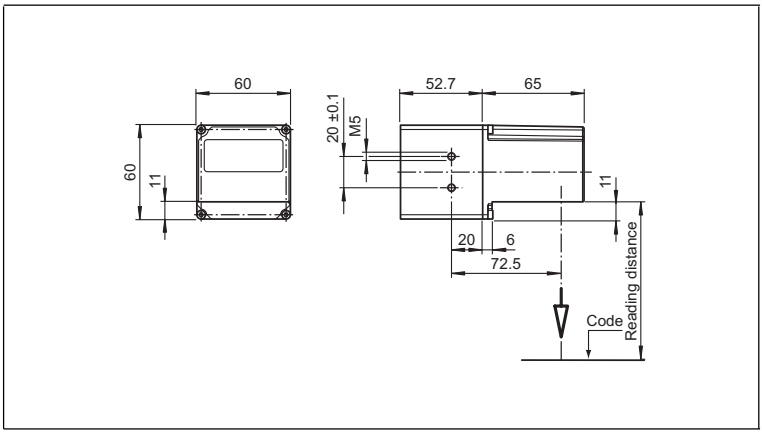


Figure 5.1 Dimensions of the **angle** housing

Note!

Connection to ground

When installing the device, ensure that it is has a ground connection.

5.3 Setting up the Power Supply

Connecting the Supply Voltage

1. Insert the 8-pin M12 socket into the plug provided on the side of the housing.
2. Screw the cap nut onto the connector as far as it will go.

   ↳ This ensures that the power cable cannot be inadvertently pulled out.

5.4 Setting Up a Network Connection

Note!

Network Cabling

Use a crossover network cable to connect the sensor directly to a computer. If you are operating the sensor within a network, use a twisted-pair network cable.

Connecting the Network Cable

1. Use a network cable that has an RJ45 network connector on one side and a 4-pin M12 plug on the other. Insert the M12 plug into the **LAN** socket on the sensor.
2. The sensor is supplied with the IP address `192.168.002.003`. To connect it to a PC, set up an IP address on the PC where the first three segments are identical to the sensor's IP address, e.g., `192.168.002.090`. The last segment must be different from the one in the sensor's IP address.

Note!

Documenting the Network Configuration

If you change the sensor's IP address, make a note of this change so that you can connect the sensor to a PC.

Resetting the IP Address

If you do not know the IP address of the sensor, you can reset the IP address to the factory default setting.

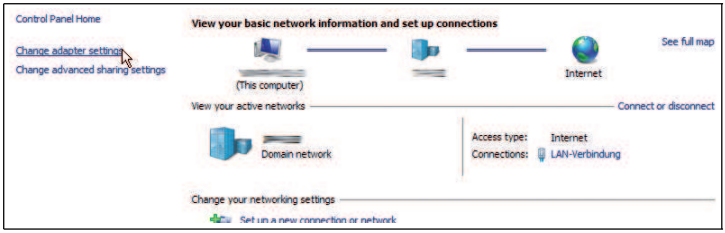
1. Connect the sensor to the power supply.

   ↳ The sensor powers up.
2. Wait until the LEDs flash.
3. Hold down buttons **1** and **2** on the back of the sensor simultaneously for approx. two seconds.

   ↳ The LED flash sequence changes.
   The sensor powers up again and now uses the factory default IP address.

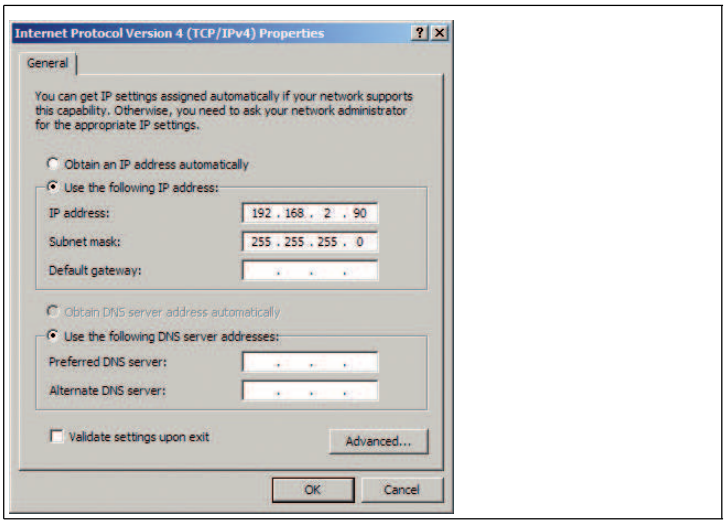
Setting the IP Address of the PC (Windows 7)

1. Select **Start > Control Panel**.
2. Select **Network and Sharing Center**.
3. Select **Change Adapter Settings**.



4. Right-click the required connection and select **Properties**.
5. Select **Internet Protocol Version 4 (TCP/IPv4)** and click **Properties**.

   ↳ The **Properties** window for the TCP/IP protocol opens.



6. Select **Use the Following IP Address**.
7. Enter an IP address where the first three segments are identical to those of the IP address for the sensor, e.g., `192.168.002.090`. The last segment must be different to the IP address of the sensor.
8. Enter `255.255.255.0` as the subnet mask.
9. Click **OK** and **Close**.

   ↳ This completes the network configuration and the sensor can be used.

2015-07

5.5 Storage and transport

For storage and transport purposes, package the unit using shockproof packaging material and protect it against moisture. The best method of protection is to package the unit using the original packaging. Furthermore, ensure that the ambient conditions are within allowable range.

6 Commissioning

6.1 Connecting the Stationary Read Device

The reader can be controlled via the Vision Configurator software. You have the ability to make adjustments on the stationary reader directly using the Vision Configurator software.

Aligning the Stationary Reader

Use the image display in the Vision Configurator software to optimally align the reader.

1. Power the reader via the 24 VDC + IO socket on the device.
2. Align the device such that the contrast between the code and the background in the image display is as sharp as possible.

   ↳ This sets the ideal reading distance between the stationary reader and the code to be read.

7 Vision Configurator Software

The sensor is commissioned and operated using the Vision Configurator software.

The Vision Configurator software makes it easy to operate the sensor with its user-friendly interface. Standard functions include making connections to the sensor, specifying the operating parameters, saving data sets, and displaying data and error diagnostics.

Note!

The following user roles are predefined with different authorizations in the Vision Configurator.

User Rights and Password

User rights	Description	Password
Default	View all information Create users at same or lower level	A password is not required
User	View all information Sensor configuration Create users at same or lower level	User
Admin **PFAdmin**	View all information Sensor configuration Create and delete users	Request the admin password from Pepperl+Fuchs

Establishing a Network Connection

To establish a network connection with the sensor, proceed as follows:

1. Supply the sensor with power.
2. Start the Vision Configurator software.
3. Select the connected sensor.
4. Check that the correct IP address has been entered.
5. Enter your user name and password.

   ↳ A connection to the sensor is established.

Note!

Documenting the Network Configuration

The sensor communicates with the connected machine control system using the TCP/IP protocol. To ensure proper communication, you must record all the changes made to the network configuration.

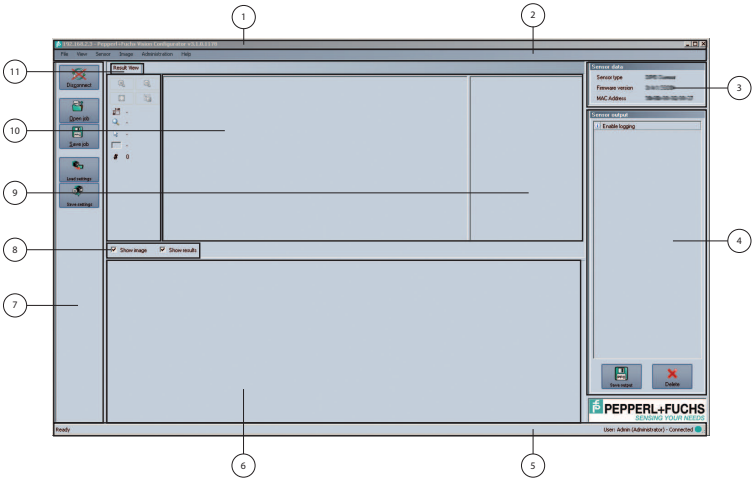
An up-to-date description of the Vision Configurator software can be found at http://www.pepperl-fuchs.com.

7.1 Application Window Structure

The application screen opens after you log in.

Note!

The individual functions depend on the type of sensor connected and the current authorization level, so they are not always all visible.



The software is designed to be similar to most Windows applications.

1	Title bar	■ Shows the IP address, the software name, and the version number ■ Contains the **Minimize/Maximize/Close** buttons
2	Menu bar	■ Displays all the menus in the program ■ Provides an overview and helps with navigation
3	**Sensor data** screen	■ Displays data for the connected sensor
4	**Sensor output** screen	■ Shows the log display
5	Status bar	■ Displays status information about the application
6	Configuration window	■ Contains the sensor-specific parameters that you can set
7	Toolbar	■ Contains icon buttons as an extension to the menu
8	Check boxes	■ **Show images**: Enables or disables the image display ■ **Show results**: Enables or disables the results area

9	Results area	■ Displays results from the sensor ■ A varying number of tabs can be displayed depending on which sensor is connected ■ This field can be enabled or disabled via **Show results**
10	Image display	■ Displays the images captured or stored in the error memory ■ This field can be enabled or disabled via **Show images**
11	Tab	Displays information about the current image and the pixel under the mouse pointer. The following items are displayed: ■ Image size ■ Zoom level ■ Mouse position in image coordinates ■ Current grayscale value ■ Image number

7.2 Menu Bar

The menu bar contains a list of menu items. The functionality depends on the type of sensor that is connected and the permissions of the user logged in.

Figure 7.1 Menu Bar

7.2.1 *File* Menu

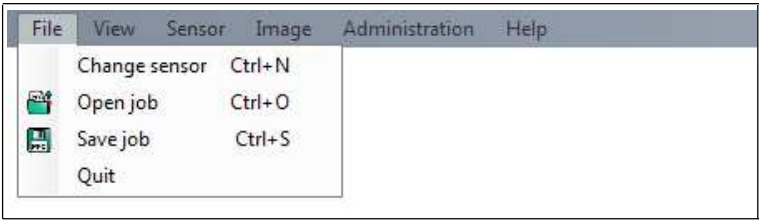


Figure 7.2 **File** menu

Change sensor	Disconnects the sensor and returns to the Login dialog.
Open job	Loads a sensor configuration stored on the PC.
Save job	Saves the current sensor configuration on the PC.
Quit	Terminates the program.

Table 7.1 **File** Menu

7.2.2 *View* Menu

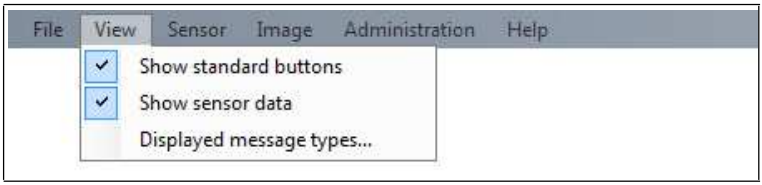


Figure 7.3 **View** menu

Show standard buttons	Toggles the display of the buttons in the bar on the left on and off.
Show sensor data	Hides the output of the sensor data in the top right of the screen.
Displayed message types…	This item enables you to specify which types of message are to be output by the sensor. Output messages of the selected types are displayed in the **Sensor output** column on the right. **Info**: Information is displayed **Warning**: Warnings are displayed **Error**: Errors are output **Critical**: Serious errors are output **Assert**: Internal errors are displayed

Table 7.2 **View** menu

7.2.3 *Sensor* Menu

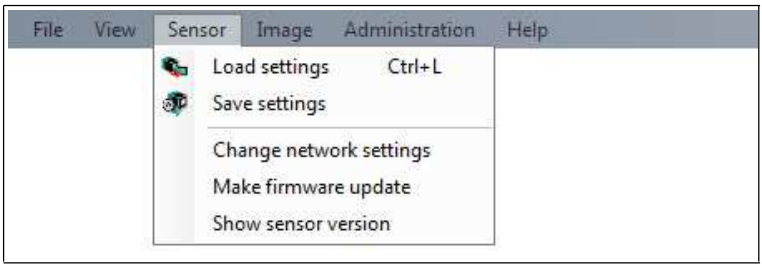


Figure 7.4 **Sensor** menu

Load settings...	Loads the saved settings from the sensor
Save settings...	Saves the settings to the sensor
Change network settings...	If the sensor is connected to the PC via Ethernet, this command enables you to change some of the connection settings on the sensor. Once you have changed them, the connection between Vision Configurator and the sensor is automatically disconnected. If required, change the network address to the newly allocated IP and reconnect Vision Configurator to the sensor. Press the **Connect** button to open the Login screen again.
Make firmware update...	Performs firmware updates. This command should be used by experienced users only.

2015-07

Show sensor version...	Displays the sensor version number.

Table 7.3 **Sensor** menu

Note!

Firmware Update

Once you have upgraded the firmware and **Update complete** is displayed, restart the sensor.

7.2.4 *Image* Menu

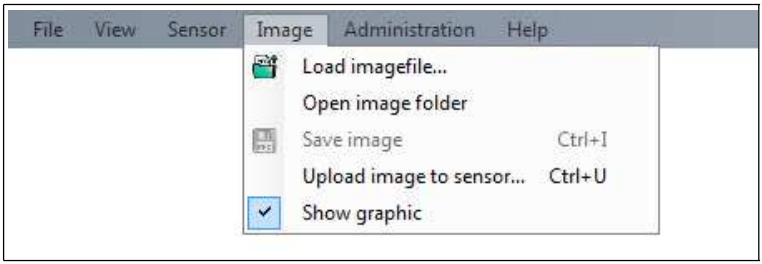


Figure 7.5 **Image** menu

Load imagefile	Opens an image file and displays the image in the image display.
Open image folder	Opens the folder in which images are currently saved.
Save image	Saves the image currently displayed on the PC.
Upload image to sensor	Uploads an image file from the PC to the sensor.
Show graphic	Turns display data sent from the sensor on and off in the image.

Table 7.4 **Image** Menu

7.2.5 *Administration* Menu

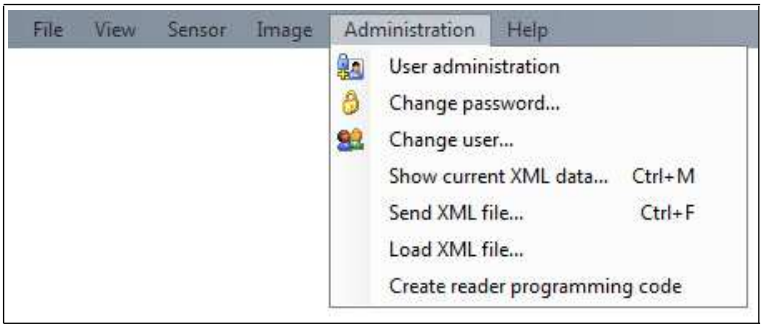


Figure 7.6 **Administration** menu

User administration	Opens a window which displays all the currently created users at the same authorization level or lower. New users with the same authorization level or lower can also be created and deleted here. In addition, a user password can be reset to the default password for the relevant user level.
Change password	Changes the current user's password.
Change user	The Login screen opens and a different user and/or sensor can be selected.
Show current XML data	Loads the current XML data from the sensor and displays it in a separate window.
Send XML file...	Saves the XML data on a computer.
Load XML file...	Loads XML data from a computer.
Create reader programming code	Enables the user to create control codes that can assign the sensor a device name and an IP address.

Table 7.5 **Administration** menu

7.2.6 *Help* Menu

Figure 7.7 **Help** menu

Info	Displays information about Vision Configurator.

Table 7.6 **Help** menu

7.3 Configuration window

Various parameters are specified in the configuration window. The individual parameters depend on the current authorization level and are, therefore, not always all visible. Some features are available in different variants only. Depending on the parameters set, some fields will be grayed out.

7.3.1 **System** Tab

System Tab, Interfaces Menu Item

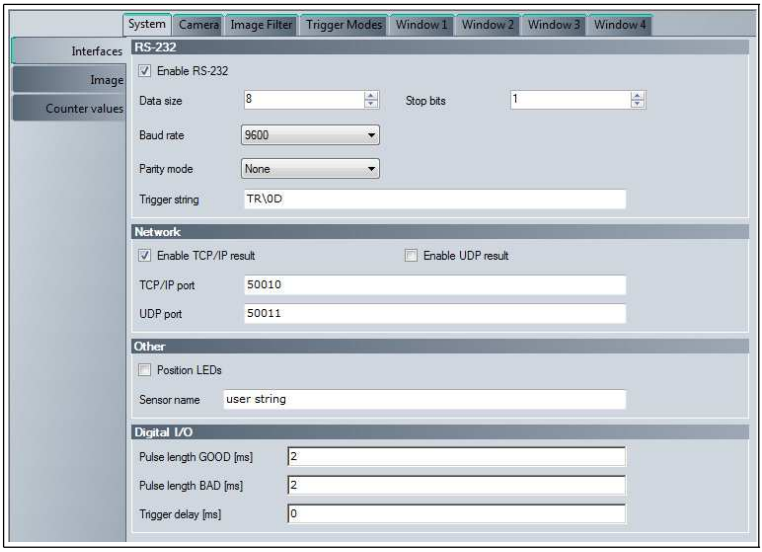


Figure 7.8 **System** tab, **Interfaces** menu item

RS-232

Enable RS-232	Enables or disables RS-232 transmission
Data size	Number of data bits. Adjustment range 5 ... 8
Stop bits	Number of stop bits. Adjustment range 1 ... 2
Baud rate	Selection of the baud rate ■ 9600 ■ 57600 ■ 38400 ■ 19200 ■ 115200

Parity mode	Setting for detection of transmission errors ■ None ■ Odd ■ Even Result output via TCP/IP
Trigger string	Definition of the trigger string that can be used to trigger a trigger command via the RS-232 interface

Network

Enable TCP/IP result	Result output via TCP/IP
Enable UDP result	Result output via UDP (User Datagram Protocol)
TCP/IP port	Entry of the corresponding port
UDP port	Entry of the corresponding port

Other

Position LEDs	Activation of the position LEDs
Sensor name	Enter a device name for the sensor. The device name can be used instead of the IP address to connect to the sensor

Digital I/O

Pulse length GOOD [ms]	Length of the output signal for a successful read Value range: 0 ... 30,000 ms 0 means that the output signal is not reset
Pulse length BAD [ms]	Length of the output signal for a failed read Value range: 0 ... 30,000 ms 0 means that the output signal is not reset
Trigger delay [ms]	Delay for a trigger Value range: 0 ... 30,000 ms

System Tab, Image Menu Item

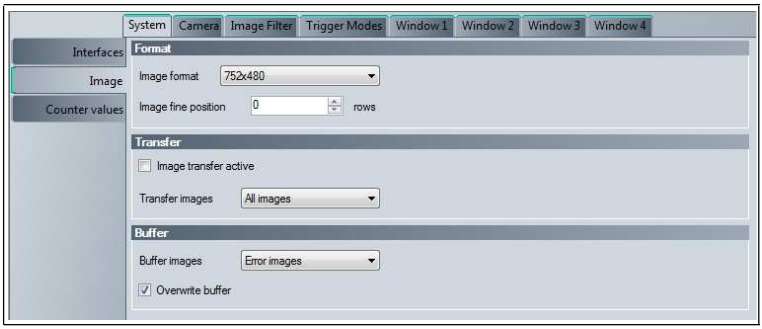


Figure 7.9 **System** tab, **Image** menu item

2015-07

PEPPERL+FUCHS

Format

Image format	Selects the image size ■ 752x480 (full screen) ■ 752x240 ■ 752x160 ■ 752x120 Selecting a smaller field of view can increase the reading speed.
Image fine position	Moves the field of view up or down (line by line) The field of view can be moved only if you select a field of view instead of the full screen under **Image format**.

Transfer

Image transfer active	Image transfer from the sensor to Vision Configurator
Transfer images	Determines which images are to be transferred to Vision Configurator ■ **All images**: All images ■ **Error images**: Images from failed reads ■ **Good images**: Images from successful reads

Buffer

Buffer images	Specifies which images are to be saved. The memory holds approx. ten images. ■ **All images**: All images ■ **Error images**: Images from failed reads ■ **Good images**: Images from successful reads
Overwrite buffer	If this is enabled, the image memory is continuously overwritten with new images. In other words, if this is enabled, the image memory contains the last ten images that correspond to the settings configured under **Buffer images**. If it is disabled, the image memory contains the first ten images that correspond to the settings configured under **Buffer images**.

System Tab, Counter Values Menu Item

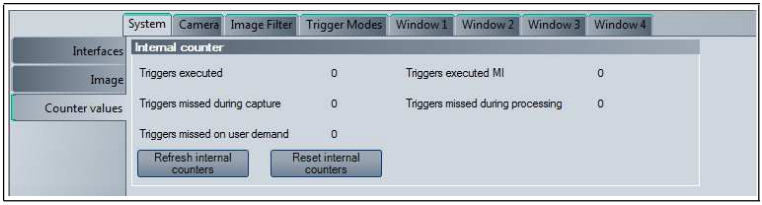


Figure 7.10 **System** tab, **Counter Values** menu item

Internal Counter

Triggers executed	Counter for executed trigger signal
Triggers executed MI	Counter for codes that extend over multiple image captures For example, if two images had to be evaluated when reading a long barcode, this counter increases the count by one.
Triggers missed during capture	Counter for triggers that were not executed when capturing an image
Triggers missed during processing	Counter for triggers that were not executed when evaluating an image
Triggers missed on user demand	Counter for triggers that were not executed as a result of user intervention
Refresh internal counters	Updates the counter values in Vision Configurator
Reset internal counters	Resets the counter values to 0

Note!

The counter values are not automatically increased in Vision Configurator. To retrieve the current counter values, click **Refresh internal counters**.

7.3.2 **Camera** Tab

Camera Tab, Common Menu Item

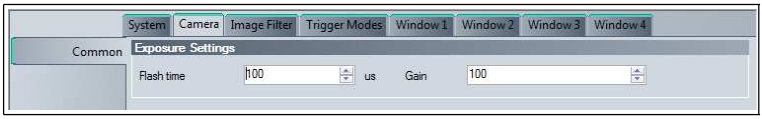


Figure 7.11 **Camera** tab, **Common** menu item

Exposure Settings

Flash time	Exposure time setting in µs
Gain	Gain setting

7.3.3 **Image Filter** Tab

Image Filter Tab, Expansion Filter Menu Item

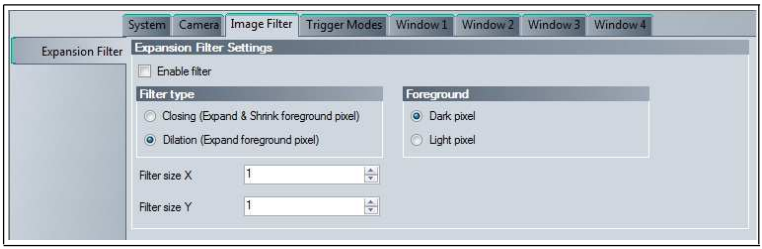


Figure 7.12 **Image Filter** tab, **Expansion Filter** menu item

2015-07

Expansion Filter Settings

Enable filter	Enables or disables expansion of the modules, i.e., the points used to generate the code. This allows modules of needlepunched codes to be expanded so that they can be read more reliably. However, using this filter increases the evaluation time.
Filter type	■ **Closing**: Fills gaps between light or dark areas depending on the setting ■ **Dilation**: Magnifies light or dark areas depending on the setting
Foreground	■ **Dark pixel**: The settings under **Filter type** relate to dark areas. For example, if **Dilation** is selected, dark areas are expanded, shrinking the surrounding light areas.  ■ **Light pixel**: The settings under **Filter type** relate to light areas. For example, if **Dilation** is selected, light areas are expanded, shrinking the surrounding dark areas. 
Filter size X	Select the expansion factor in the X direction.
Filter size Y	Select the expansion factor in the Y direction.

7.3.4 **Trigger Modes** Tab

Trigger Modes Tab, Setup Menu Item

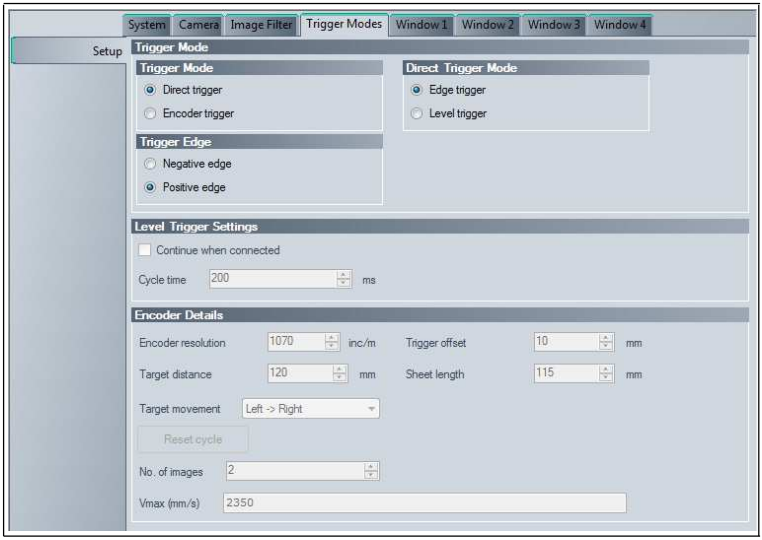
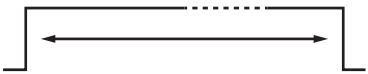


Figure 7.13 **Trigger Modes** tab, **Setup** menu item

Trigger Mode

Trigger mode	
Direct trigger	Trigger is evaluated directly
Encoder trigger	Trigger is generated by rotary encoder increments
Direct Trigger Mode	
Edge trigger	Trigger signal edge activates a trigger 
Level trigger	Signal level triggers a continuous trigger 
Trigger Edge	

Negative edge	The falling edge activates a trigger 
Positive edge	The rising edge activates a trigger 

Note!

Use the **Encoder trigger** trigger mode in conjunction with 1-D barcodes only.

The **Level Trigger Settings** menu item is active only if you have selected **Level trigger** under **Direct Trigger Mode**.

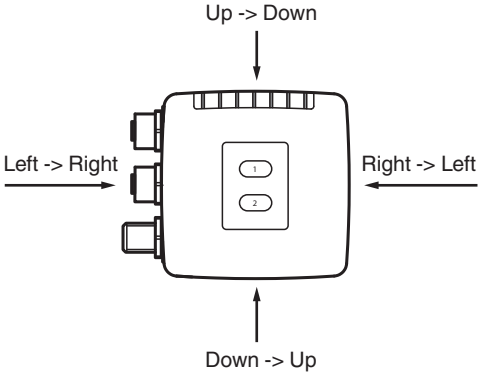
Level Trigger Settings

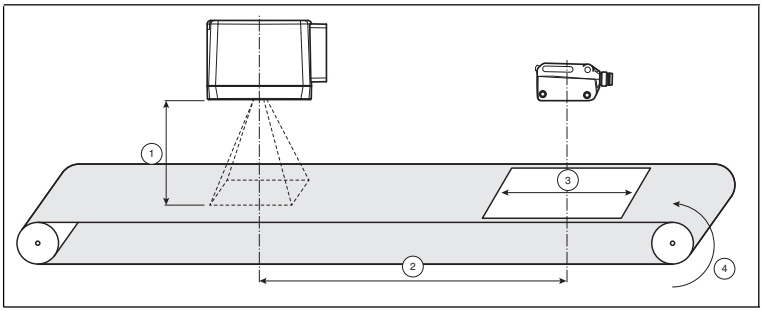
Continue when connected	Emergency switch for a continuous trigger. If this is enabled, the sensor operates continuously. Disable this option to interrupt the sensor's task. For example, if **Level trigger** is selected in conjunction with **Negative edge**, the sensor will operate continuously until the next rising edge occurs. Disable this option to interrupt this status.
Cycle time	Time between two reads in ms

The **Encoder Details** menu item is active only if you have selected **Encoder trigger** under **Trigger Mode**.

Encoder Details

Encoder resolution	Rotary encoder resolution in increments/m
Trigger offset	Distance in mm between trigger (e.g., light barrier) and optical axis of sensor
Target distance	Read distance in mm Distance from the sensor lens to the object on which the code is located (OPC120* only)
Sheet length	Object's length in mm Length of the object on which the code is located

Target movement	Code's direction of movement from the sensor's perspective 
Reset Cycle	Interruption of the cycle
No. of images	The sensor uses the **Encoder resolution**, **Trigger offset**, **Target distance**, and **Sheet length** values to automatically calculate the number of reads and the trigger times they require to capture the entire object. If the number of images is greater than the image memory available, the application is not feasible because the entire object cannot be captured.
Vmax (mm/s)	Maximum speed of the object



1. **Target distance**: Read distance in mm
2. **Trigger offset**: Distance in mm between trigger and sensor's optical axis
3. **Sheet length**: Object's length in mm
4. **Encoder resolution**: Rotary encoder resolution in increments/m

7.3.5 **Window** Tab

Four different tabs are available: **Window 1** ... **Window 4**. Each tab has the same layout.

Window Tab, Window Setup Menu Item

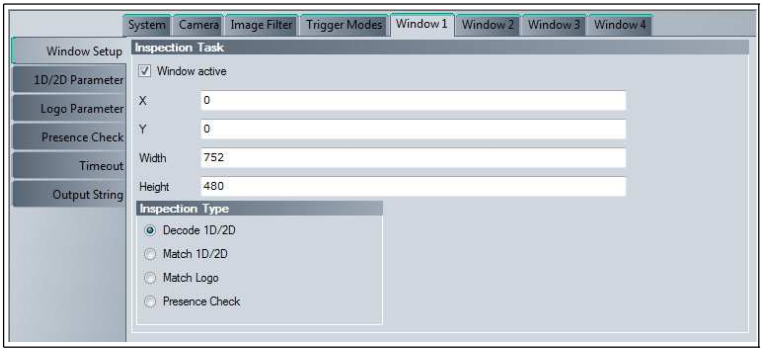


Figure 7.14 **Window** tab, **Window Setup** menu item

Inspection Task

Window active	Activation of the window
X	Top left position of the window in the X direction
Y	Top left position of the window in the Y direction
Width	Width of the window The width of the window can be read in the image display at the bottom left of the relevant window frame
Height	Height of the window The height of the window can be read in the image display at the bottom left of the relevant window frame
Inspection Type	Option to select whether a code is to be decoded or compared, whether a taught-in image area is to be retrieved, or whether presence monitoring is to be performed ■ Decode 1-D/2-D ■ Match 1-D/2-D ■ Match Logo ■ Presence Check

Changing the Window Size and Window Position

You can change the window size and window position either in the tab or in the image display itself.

1. Click on the frame of the window to be edited in the image display.

   ↳ The frame is now displayed in bold.

2. To change the position of the frame, move the cursor toward the center of the frame until it changes into a crosshair.

↳ Hold down the mouse button and drag the frame to the required position, and then release the mouse button.

3. To change the size of the frame, move the cursor to one of the corners of the frame until it changes into a double arrow.

   ↳ Hold down the mouse button and drag the frame to the required size, and then release the mouse button.

Window Tab, 1D/2D Parameter Menu Item

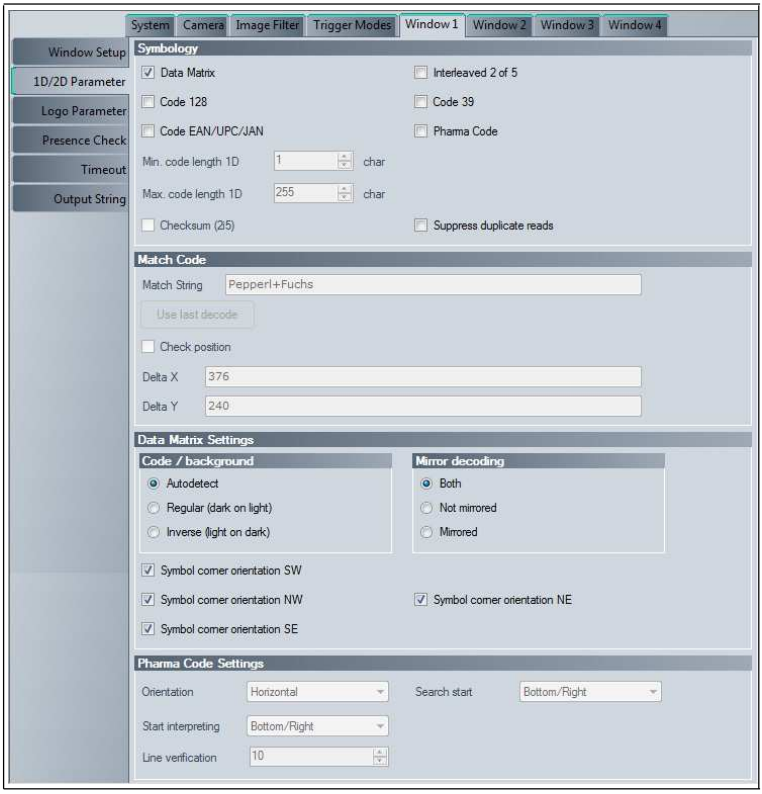


Figure 7.15 **Window** tab, **1D/2D Parameter** menu item

Symbology

Data Matrix	Activates or deactivates reading of Data Matrix codes
Interleaved 2 of 5	Activates or deactivates reading of interleaved 2 of 5 codes
Code 128	Activates or deactivates reading of code 128 codes
Code 39	Activates or deactivates reading of code 39 codes
Code EAN/UPC/JAN	Activates or deactivates reading of EAN/UPC/JAN codes

2015-07

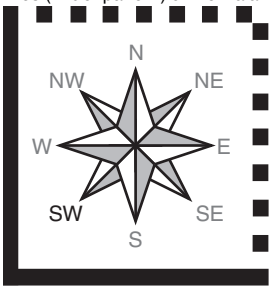
Pharma Code	Activates or deactivates reading of Pharmacodes
Min. code length 1D	Enters the minimum code length for 1D barcodes
Max. code length 1D	Enters the maximum code length for 1D barcodes
Checksum (2i5)	Evaluates the checksum for interleaved 2 of 5 codes
Suppress duplicate reads	Enables or disables duplicate suppression, which prevents the same code being read several times in succession.

The **Match Code** menu item is active only if you have selected **Match 1D/2D** under **Inspection Type** in **Window Setup**.

Match Code

Match string	Enters the comparison string
Use last decode	The last result string read is taken as the new comparison string
Check position	Checks the position of the read code
Delta X	Horizontal range (+/-) in which the decoding is evaluated as "good"
Delta Y	Vertical range (+/-) in which the decoding is evaluated as "good"

Data Matrix Settings

Code / background	■ **Autodetect**: Automatic detection of the code color and code background ■ **Regular (dark on light)**: Dark code on a light background ■ **Inverse (light on dark)**: Light code on a dark background
Mirror Decoding	■ **Both**: Reads mirrored and non-mirrored codes ■ **Not mirrored**: Reads non-mirrored codes ■ **Mirrored**: Reads mirrored codes
Symbol corner orientation SW Symbol corner orientation NW Symbol corner orientation NE Symbol corner orientation SE	Position of the corner formed from the continuous boundary lines (finder pattern) of the Data Matrix code 

The **Pharma Code Settings** menu item is active only if you have enabled reading of Pharmacodes in **Symbology**.

Pharma Code Settings

Orientation	Orientation of the Pharmacode ■ **Horizontal**: Horizontal ■ **Vertical**: Vertical
Search start	Direction from which a Pharmacode search is to start ■ **Bottom/Right**: From the bottom right ■ **Top/Left**: From the top left
Start interpreting	Direction from which the Pharmacode is to be interpreted ■ **Bottom/Right**: Starting from the bottom or from the right, depending on the orientation of the Pharmacode ■ **Top/Left**: Starting from the top or from the left, depending on the orientation of the Pharmacode
Line verification	Number of bars constituting the Pharmacode Specifying this prevents accidental interpretation of too many or too few lines.

Window Tab, Logo Parameter Menu Item

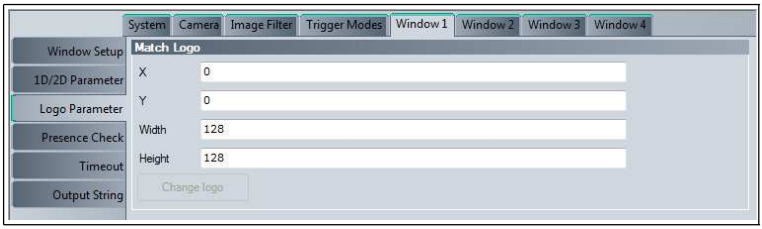


Figure 7.16 **Window** tab, **Logo Parameter** menu item

Match Logo

X	X position of the **New logo area** window
Y	Y position of the **New logo area** window
Width	Width of the **New logo area** window
Height	Height of the **New logo area** window
Change logo	Activates the **New logo area** window

You can change the height and width of the **New logo area** window in 64 steps only.

Window Tab, Presence Check Menu Item

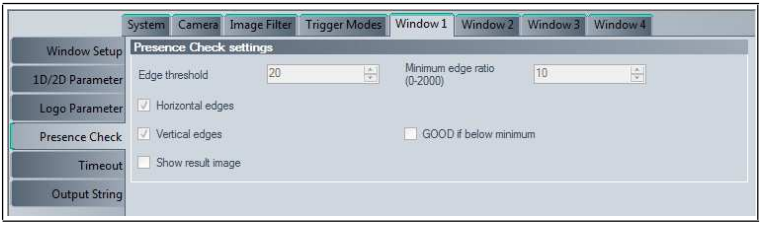


Figure 7.17 **Window** tab, **Presence Check** menu item

Presence Check Settings

Edge threshold	Measurement for the shape of the edge
Minimum edge ratio (0 - 2000)	Measurement for the number of edges in the image area, up to a maximum of 2000
Horizontal edges	Activates counting of horizontal edges
Vertical edges	Activates counting of vertical edges
GOOD if below minimum	The result is "good" if the number of edges in the image area is below the **Minimum edge ratio** value
Show result image	The edge image is plotted in the input image

Window Tab, Timeout Menu Item

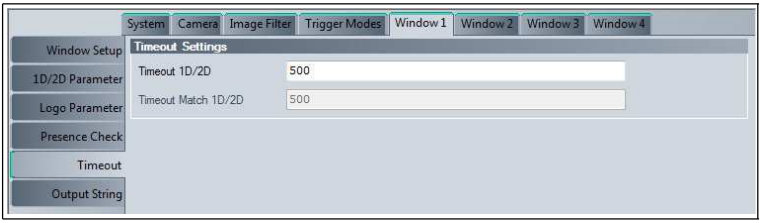


Figure 7.18 **Window** tab, **Timeout** menu item

Timeout Settings

Timeout 1-D/2-D	Maximum duration for the image capture and decoding of the code in milliseconds. If the read times out, it is considered to have failed. The image capture time is the time required to transfer the image to the memory. This time depends on the **Image format** image size that you have set under **System > Image > Format**. The decoding time depends on the symbology and code quality. The decoding time is displayed on the Vision Configurator image display under **Time**. See chapter 7.7
Timeout Match 1-D/2-D	Maximum duration for inspection type **Match 1D/2D** in milliseconds.

Only one input field is active at a time. Depending on the inspection type selected, the related input field is active.

Window Tab, Output String Menu Item

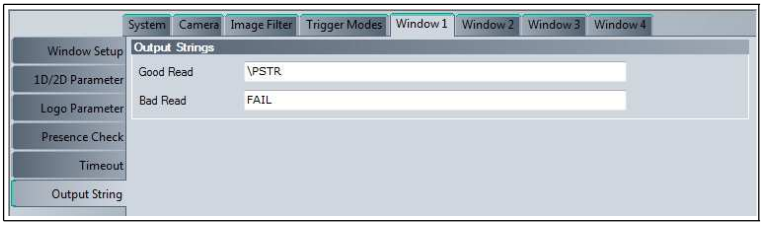


Figure 7.19 **Window** tab, **Output String** menu item

Output Strings

Good Read	This allows you to specify which data string is output for a successful read. You can enter both normal characters and predefined parameters that output a specific data string. You can also output several parameters in succession. Examples: ■ `\PSTR` outputs all content of the read code. ■ `\PISO` outputs various properties relating to the code quality. If, for example, the contrast deteriorates continuously, this may indicate decreasing ink levels in the code printer. If the axial nonuniformity of the code increases, this may indicate that the paper rolls are running too fast.
Bad Read	This allows you to specify which data string is output if the read was unsuccessful.

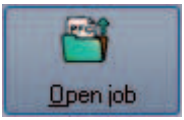
Output Strings

Parameters	Description
\PSTR	Returns all content of the code if the read was successful.
\PSTR(`pos,len`)	Outputs `len` characters from position `pos` of the code
\PSTR(E,`len`)	Outputs the last `len` characters of the code
\PLEN	Outputs the length of the read code (4-digit)
\PTIM	Outputs the time required to read the code in ms (4-digit)
\PTST	Outputs a timestamp in ms (10-digit)

Parameters	Description
\PISO	This parameter can be used in the **Good Read** field only, as the values can be output only if a code was successful. Outputs the code quality based on eight properties, in accordance with ISO/IEC 16022:2006. A mark between 0 (bad) and 4 (good) is output for each property. ■ **OG**: overall grade The overall quality is derived from the worst value for the other properties. ■ **SC**: symbol contrast Indicates the extent to which the code stands out against the background. ■ **ANU**: axial nonuniformity Indicates the uniformity of the code in the axial direction. ■ **GNU**: grid nonuniformity Indicates the uniformity of the points, i.e., the points constituting the code. ■ **MOD**: modulation grade Indicates how easily light points can be distinguished from dark ones. ■ **FPD**: fixed pattern damage Indicates whether defined areas are present, e.g., the surrounding quiet zone or the alignment pattern. ■ **UEC**: unused error correction Indicates the extent to which error correction was required to read the code. ■ **PG**: print growth Indicates the extent to which points have increased in size, e.g., due to the ink running.
\PCXL	Outputs the X coordinate of the code's central point.
\PCYL	Outputs the Y coordinate of the code's central point.
\cc	Control characters, cc (2-digit, hexadecimal), e.g., ■ 0D = CR (carriage return) ■ 0A = LF (line feed) ■ 5C = \ (backslash)

7.4 Toolbar

The toolbar contains various function icons.

	The connection between the PC and the sensor is disconnected.
	Opens a saved setting.
	Saves the settings made.
	Settings are read out from the sensor.
	All settings made are saved on the sensor.
	Initiates a trigger. Depending on the sensor operating mode, the sensor returns a single measured value or continuous measured values.
	Resends the last fault pattern.

7.5 Sensor Data

This area shows the type of sensor connected, the firmware version of the connected sensor, and the MAC address.

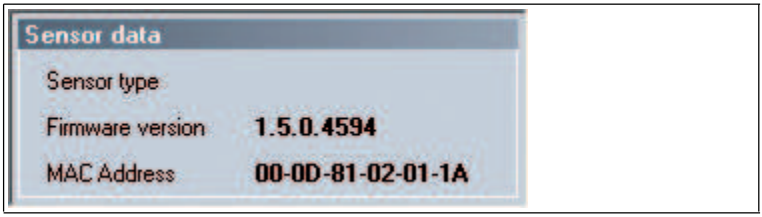


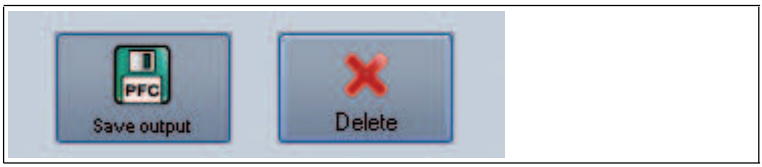
Figure 7.20 Sensor data

7.6 Sensor Output

This area shows the communication between the Vision Configurator and the connected sensor. To select which messages are to be displayed, select **View > Displayed message types**.

Figure 7.21 Sensor output

Two buttons are located in the lower area.



Save output	Saves the content of the window to a text file.
Delete	Deletes the contents of the window.

7.7 Image Display

The image display has different options for displaying the captured data. The **Show image** and **Show results** check boxes can be used to activate and deactivate the image display and result display.

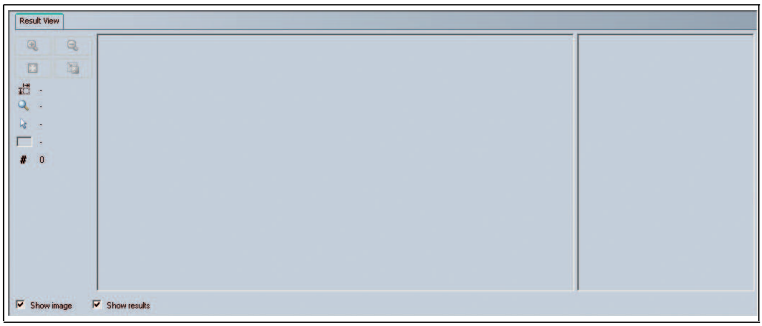


Figure 7.22 Result View screen

Figure 7.23 Result View screen with image display and result display

Different results are shown in the result display, depending on the type of inspection. In the example shown, the code read out, the decoding time required, the type of code read, and the position of the code read (quality) are displayed.

When you press the right mouse button or the context menu button, the following context menu appears:

Figure 7.24 Result View context menu screen

2015-07

PEPPERL+FUCHS

Context Menu

Load image file...	Loads a sensor image. You can select the sensor image
Open image folder	Opens the storage location
Save image	Saves the displayed sensor image

Figure 7.25 Image View toolbar screen

Magnifier +	Zooms in on sensor image
Magnifier -	Zooms out on sensor image
Fit to window	Fits the sensor image to the window
Original size	Sets original image size
Size details	Specifies the size of the sensor image
Zoom factor	Displays the zoom factor. Zoom factor 1 is the original size
Position details	Specifies the position of the cursor
Grayscale value details	Grayscale value details for the pixel indicated by the cursor
Image number	Specifies the image number

Table 7.7 **Result View toolbar** menu

8 Operation

You can select up to four different functions (inspection types).

If more than one inspection type (more than one window tab) is activated, all the read codes must be OK for the decoding in order to output a good read.

8.1 Reading a 1-D/2-D Code

You can process up to four different 1-D/2-D codes with one image capture using the sensor. To do this, enable tabs **Window 1** ... **Window 4**. If more than one tab is enabled, all read codes must be OK to output a good read.

Reading 1-D/2-D Codes

1. Select **Decode 1D/2D** in the **Window Setup** menu item in the **Window 1** configuration window. See chapter 7.3.5
2. Set the symbology to be read in the **1D/2D Parameter** menu item in the **Window 1** configuration window.
3. Make sure that the window is switched to active in the **Window Setup** menu item in the **Window 1** configuration window.
4. Place the first sheet with the code to be read under the sensor.
5. Click **Trigger**. The captured image will be shown in the image display.
6. Click the frame labeled **Window 1**.

7. Adjust the size of the frame if necessary.
8. Click **Trigger**. The captured image will be shown in the image display.

9. The result from the read code is displayed in the result window.

↳ The sensor is set up to read codes.

8.2 Reading Overlong 1-D Codes

In addition to standard 1-D/2-D codes, the sensor can read overlong barcodes that extend beyond the sensor's capture range. The sensor does this by capturing several images and evaluating them in succession to obtain a result.

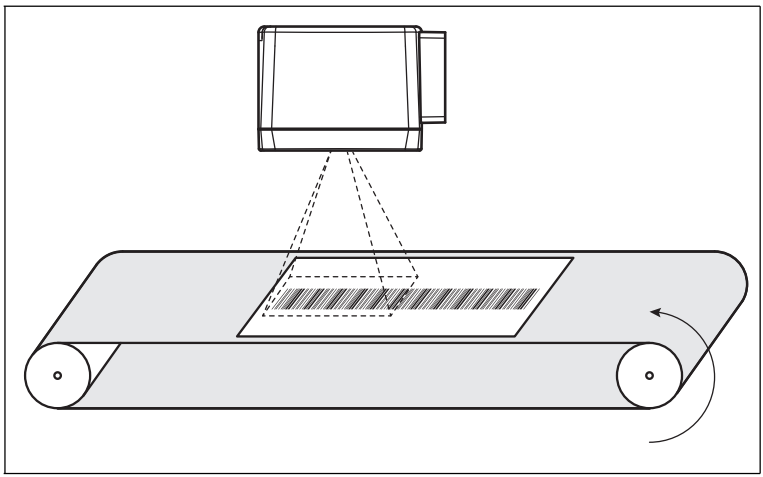


Figure 8.1 Reading Overlong 1-D Barcodes

Reading Overlong 1-D Codes

1. Set the symbology to be read in the **1D/2D Parameter** menu item in the **Window 1** configuration window. Note that the function for reading overlong codes can be used with 1-D codes only. See chapter 7.3.5
2. Enable the **Encoder trigger** trigger mode in the **Setup** menu item in the **Trigger Modes** configuration window.
3. Enter the values for the **Encoder resolution**, **Trigger offset**, **Target distance**, **Sheet length**, and **Target movement**. See chapter 7.3.4

   ↳ The sensor automatically calculates the number of images required to capture the whole object, including the overlong barcode. If you need to capture only the barcode rather than the whole object, you can enter the length of the barcode in the **Sheet length** field.

8.3 Comparing a 1-D/2-D Code

You can compare up to four different 1-D/2-D codes with one image capture using the sensor. To do this, enable tabs **Window 1** ... **Window 4**. If more than one tab is enabled, all compared codes must be OK to output a good read.

Comparing 1-D/2-D Codes

1. Select **Match 1D/2D** in the **Window Setup** menu item in the **Window 1** configuration window. See chapter 7.3.5
2. Set the symbology to be read in the **1D/2D Parameter** menu item in the **Window 1** configuration window.
3. Make sure that the window is switched to active in the **Window Setup** menu item in the **Window 1** configuration window.
4. Place the first sheet with the code to be compared under the sensor.
5. Click **Trigger**. The captured image will be shown in the image display.
6. Click the frame labeled **Window 1**.

7. Adjust the size of the frame if necessary.
8. Click **Trigger**. The captured image will be shown in the image display.

2015-07

9. Enter the code to be compared under **Match Code** in the **1D/2D Parameter** menu item of the **Window 1** configuration window. You can adopt a previously read code via the **Use last decode** button.

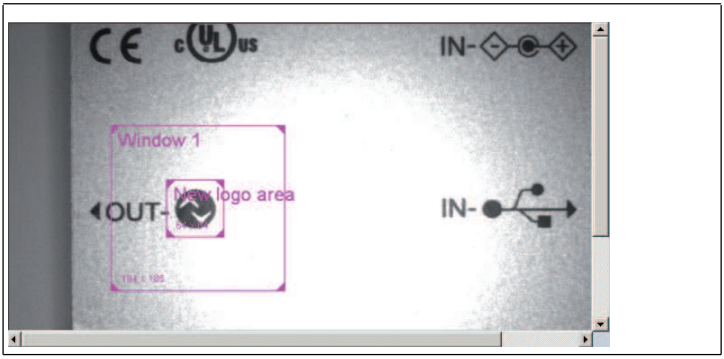
   ↳ The sensor is set up to compare codes.

8.4 Comparing a Logo

You can process up to four different logos with one image capture using the sensor. To do this, enable tabs **Window 1** ... **Window 4**. If more than one tab is enabled, all compared logos must be OK to output a good read.

Comparing Logos

1. Select **Match Logo** in the **Window Setup** menu item in the **Window 1** configuration window. See chapter 7.3.5
2. Make sure that the window is switched to active in the **Window Setup** menu item in the **Window 1** configuration window.
3. Place the first sheet with the reference logo under the sensor.
4. Click **Trigger**. The captured image will be shown in the image display.
5. Click the frame labeled **Window 1**.
6. Adjust the size of the frame if necessary.
7. Click **Change logo** in the **Logo Parameter** menu item in the **Window 1** configuration window.



↳ A new frame appears, labeled **New logo area**.

8. Move the **New logo area** frame over the reference logo.
9. Click **Trigger**.

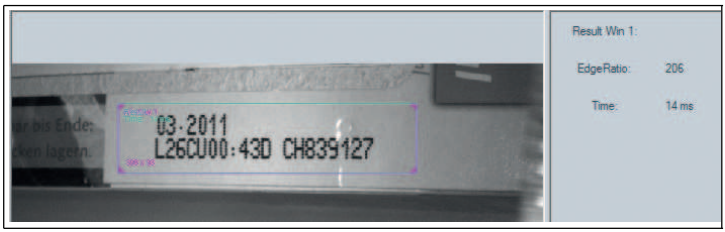
↳ The reference logo is taught in.

8.5 Checking the Presence of Materials with a Changeable Texture

You can check up to four different textures with one image capture using the sensor. To do this, enable tabs **Window 1** ... **Window 4**. If more than one tab is enabled, all read textures must be OK to output a good read.

Presence Check

1. Select **Presence Check** in the **Window Setup** menu item in the **Window 1** configuration window. See chapter 7.3.5
2. Make sure that the window is switched to active in the **Window Setup** menu item in the **Window 1** configuration window.
3. Place the first sheet with the field to be checked under the sensor.
4. Click **Trigger**. The captured image will be shown in the image display.
5. Click the frame labeled **Window 1**.
6. Adjust the size of the frame if necessary.
7. Click **Trigger**.

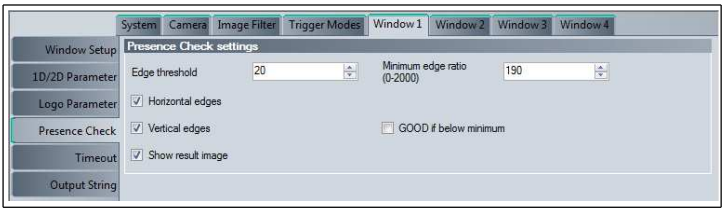


↳ The captured image will be shown in the image display together with additional information.

8. Enable **Show result image** under the **Presence Check** menu item in the **Window 1** configuration window, and click **Trigger** again.

↳ The resulting image is shown in the image display. Edge transitions are displayed in graphic form in the resulting image. The resulting value is shown in the image display under **Result Win 1**.

9. Set the edge detection to be checked under **Minimum edge ratio** in the **Presence Check** menu item in the **Window 1** configuration window.



10. If the value read in is greater than the value entered under **Minimum edge ratio**, **DECODE OK** is output. Conversely, if the result should be **DECODE OK** when the value read in is lower than the value entered, enable the **GOOD if below minimum** option.

↳ The presence check is configured.

9 Maintenance and Repair

9.1 Maintenance

To get the best possible performance out of your device, keep the optical unit on the device clean and clean it when necessary.

Observe the following instructions when cleaning:

- Do not touch the optical unit with your fingers.
- Do not immerse the device in water. Do not spray the device with water or other fluids.
- Do not use a scouring agent to clean the surface of the device.
- Use a cotton or paper cloth moistened with water or isopropyl alcohol (not soaked).
- Remove any residual alcohol using a cotton or paper cloth moistened with distilled water (not soaked).
- Wipe the surface of the device dry using a lint-free cloth.

9.2 Repair

The devices must not be repaired, changed or manipulated. If there is a defect, the product must always be replaced with an original device.

10 Troubleshooting

10.1 What to Do in the Event of an Error

Before requesting a service call, please check that the following actions have been taken:

- Equipment has been tested according to the following checklists,
- Telephone assistance obtained from the Service Center to isolate the problem.

Checklist

Fault	Cause	Remedy
"PWR" LED does not light up	The power supply is switched off.	Check whether there is a reason why the power supply is switched off (installation or maintenance work, etc.). Switch on the power supply if appropriate.
"PWR" LED does not light up	Wiring fault in the splitter or switch cabinet.	Check the wiring carefully and repair any wiring faults.
No connection to the device	Network cable not connected.	Connect the network cable.
No connection to the device	Wrong network cable used.	Direct connection between PC and device: use a crossover network cable. Connection via an existing network: use a twisted-pair network cable.

- If none of the above corrects the problem, please contact the Service Center. Please have the fault patterns and version number of the sensor available. The version number can be found at the bottom left of the user interface.

11 Appendix

11.1 Network Interface

The network interface is used to transfer data (parameters, images, software updates) and commands in both directions (host to sensor, and sensor to host).

XML strings are used to transfer parameters, commands, and results.

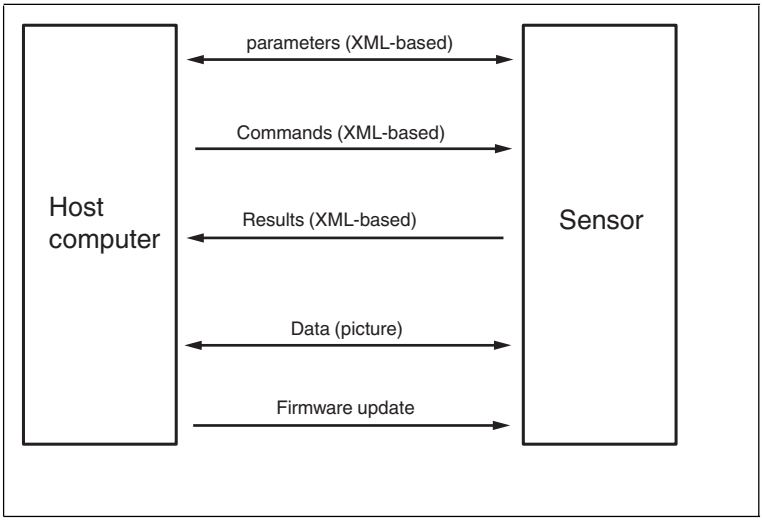


Figure 11.1 Network interface XML-Stings

Communication is via TCP/IP using port 50005. The default IP address is 192.168.2.3.
Each data packet consists of a preheader and the actual data:

Preheader (length: 12 bytes)	Data (length varies)

11.2 Software Interface

A **.NET 2.0-based** software interface is provided for easy integration with PC software. This software interface takes the form of DLLs and handles the communication with the sensor. Integrate the DLLs into the programming environment and run the programming lines indicated.

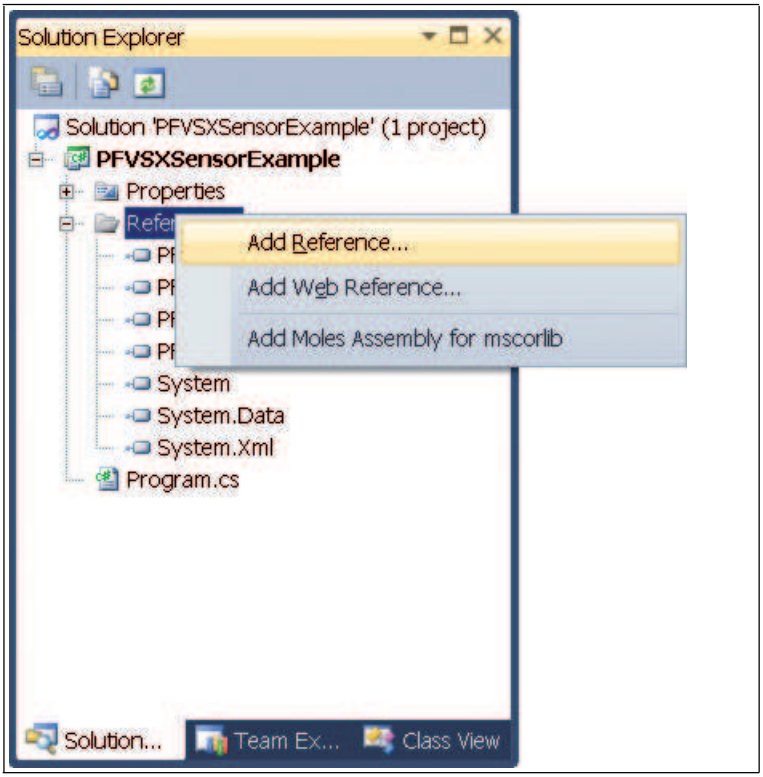


Figure 11.2 Integrating DLLs into the programming environment

All the examples relate to the Visual Studio 2010 programming environment and to the C# programming language.

Example!

Below is a sample program for integrating and triggering the sensor:

```
class Program
{
  static void Main(string[] args)
  {
    PF.Foundation.VsxFactory.PFVsxFactoryVCCustom sensor;
    sensor = new
PF.Foundation.VsxFactory.PFVsxFactoryVCCustom();
    sensor.Connect("192.168.2.3", 50005);
    sensor.SetSpecificSingleParameter(1, "Command",
"TriggerStart", "1");
    System.Threading.Thread.Sleep(1000);
    sensor.Disconnect();
  }
}
```

General

The library is used to support the creation of a graphic user interface for sensors that use the VSX protocol. The library connects to the sensor and handles the communication in accordance with the protocol. The user has functions for setting parameters on the sensor, retrieving parameter values from the sensor, and saving and loading whole parameter sets both locally and on the sensor. The user can also receive sensor images.

The library is implemented in C# and requires .NET 2.0 or higher as a minimum.

Make sure that the libraries supplied are in the project's execution folder.

Creating an Object

Create an object to access the library functions.

```
PFVsxFactoryVCCustom _vsxFactory =new
PFVsxFactoryVCCustom();
```

Retrieving Parameter Data

Received parameter data is stored as a list in the sensor. Individual items of parameter data from this list can be retrieved using the following function:

```
string GetSpecificSingleParameter(string version, string
configId, string parameterId)
```

Displaying Modified Data

If data is being received by the sensor, this is indicated by the event:

```
event ParameterDataReceived(DataModifier modifier)
```

Library Functions

```
bool Connect(string ip,int port)
```

`ip`: IP of the connected sensor
`port`: Port of the connected sensor
Response: False if the connection could not be created, otherwise true
Opens a connection to a sensor with the IP and port specified.

```
void Disconnect()
```

Disconnects the open connection

```
bool Connected {get;}
```

Response: Returns the connection status

```
void SaveSensorSettings()
```

Saves the current parameter set in the sensor's memory.
After a confirmation has been received from the sensor, a `SaveDataOnSensorReceived` event is triggered.

```
void LoadSensorSettings()
```

Requests a parameter set that has been saved with `SaveSensorSettings` from the sensor.
After the parameter set has been received from the sensor, a `ParameterDataReceived` event is triggered with `Modifier = LOAD_Data`. The data can then be retrieved using the `GetSingleParameter` method.

```
void GetNetworkSettings()
```

Requests the sensor's current connection settings from the sensor.
After this data has been received from the sensor, a GetNetworkReceived event is triggered. This provides the current IP address, network mask, and gateway for the sensor.

```
void GetLogMessages(bool on)
```

`on`: Switches log messages on
`off`: Switches log messages off
Defines whether or not the sensor sends log messages. If log messages are switched on, a `LogDataReceived` event is triggered each time a log message is received from the sensor.

```
void GetAllParametersFromSensor()
```

Requests the current parameter set from the sensor. After the parameter set has been received from the sensor, a `ParameterDataReceived` event is triggered with `Modifier = None`. This indicates that the internal list has been updated with the parameter set. You can then retrieve individual parameters via `GetSingleParameter`.

```
public bool ExistsParameter(string configId, string parameterId)
```

Used to query whether a particular parameter exists on the sensor
`configId`: Configuration ID of a parameter
`parameterId`: Parameter ID of a parameter
Response: True or false, depending on whether the parameter exists

```
string GetSpecificSingleParameter(ushort version, string
configId, string parameterId)
```

`configId`: Configuration ID of a parameter
`parameterId`: Parameter ID of a parameter
Response: The current value of the specified parameter
Retrieves the current value for a parameter from the sensor; details of the version, configId, and parameterId can be taken from the accompanying table. Returns the value of the parameter, or an empty string if the value was not retrieved.

```
void SetSpecificSingleParameter(ushort version, string configId,
string parameterId, string newValue)
```

configId: Configuration ID of a parameter
parameterId: Parameter ID of a parameter
newValue: New value for the parameter
Sets the value for a parameter to newValue; version, configId, and parameterId can be taken from the table in this case.

```
public bool ExistsResultParameter(ushort frameCounter, string
configId, string parameterId)
```

frameCounter: Number of the image associated with the result parameter
configId: Configuration ID of a parameter
parameterId: Parameter ID of a parameter
Response: True or false, depending on whether the parameter exists
Used to query whether a particular result parameter exists for a particular image.

```
string GetResultParameter(ushort frameCounter, string configId,
string parameterId)
```

configId: Configuration ID of a parameter
parameterId: Parameter ID of a parameter
Response: The current value of the specified parameter
Returns the value of an individual parameter defined by the configuration ID and parameter ID. The value is taken from an internal list and corresponds to the status last received from the sensor.
The receipt of result data for a particular image is indicated by the `SensorResultDataReceived(ushort frameCounter)` event.

```
Bitmap GetImage(ushort frameCounter)
```

frameCounter: Number of a received image
Response: The image associated with the image number
Returns an image previously received from the sensor. Receipt of an image with a particular number is indicated by the `ImageReceived(ushort frameCounter, Bitmap image)` event.

```
IList<ElementResult> GetResultList(ushort frameCounter)
```

frameCounter: Number of the image associated with the graphics
Response: List of overlay graphics for the specified image

```
IList<ElementShapeBase> GetShapes(ushort frameCounter)
```

frameCounter: Number of the image associated with the graphics
Response: List of overlay graphics for the specified image
Returns a list of graphics for an image overlay. Receipt of these graphics is indicated by the `ShapeDataReceived(ushort frameCounter, List<IElement> shapeList)` event. The individual elements in the list are of the `ElementShapeBase` type.
This class has the following attributes:
`PointF ShapeLocation`: Coordinates of the top left corner of the graphic in the image
`Color ForeColor`: Color of the graphic
`string Type`: Type of graphic, either type="Rectangle" or type="Text".
Depending on the type, the graphic can be parsed in `ElementShapeRectangle` or `ElementShapeText` and then has the following additional attributes:
`ElementShapeRectangle`:
Size Size: The size of the rectangular graphic
`ElementShapeText`:
stringText: The text in the text graphic

```
void ResetSensor()
```

Resets all the sensor's parameters to their factory default settings. After a confirmation has been received from the sensor, a `ParameterDataReceived` event is triggered with `Modifier = LOAD_DEFAULT_DATA`.

2015-07

```
void SaveSettingsToFile(string filename)
```

Saves the current parameter set to the specified file. The current parameter set is retrieved from the sensor and saved after receipt. After a successful save, a `SaveDataOnHdd` event is triggered. If an error occurs during a save, an `InternalError` event is triggered with `ErrorType = SAVE_FILE_ERROR`.
`filename`: Valid path and file name

```
bool SetAllParameters(string filename)
```

Loads a parameter set from the specified file and sends the parameters to the sensor. After the sensor has acknowledged receipt of the parameter set, a `ParameterDataReceived` event is triggered with `Modifier = NONE`. If the data set cannot be loaded correctly, an `InternalError` event is triggered with `ErrorType = LOAD_FILE_ERROR`.
`filename`: Valid path and file name

```
void SetNetworkSettings(string ipAddress, string networkMask,
string gateway)
```

`ipAddress`: Valid IP address
`networkMask`: Valid network mask
`gateway`: Valid gateway
Converts the network parameters on the sensor. The connection to the sensor is then disconnected and a `DisconnectReceived` event is triggered. The connection must be reestablished with the new IP via `Connect`.

```
void SendImage(Bitmap image)
```

Sends an image to the sensor. This function is not supported by all device types.
`image`: Image to be sent

```
void SendVsxFile(string filepath)
```

`filepath`: Path and file name of a valid VSX file
Sends the content of a file to the sensor. This must comply with the VSX standard. If the data cannot be loaded properly or does not correspond to the VSX syntax, an `InternalError` event with `ErrorType = LOAD_FILE_ERROR` is triggered.

```
string SensorName { get; }
```

Returns the name of the sensor.

```
float SensorVsxVersion { get; }
```

Returns the VSX version installed on the sensor.

Library Events

```
event ParameterDataReceived(DataModifier modifier)
```

This is always triggered if parameter data has been received from the sensor. This is the case after the following function call:
`LoadSensorSettings (MODIFIER = LOAD_DATA)`
`GetAllParametersFromSensor (MODIFIER = NONE)`
`SetSingleParameter (MODIFIER = NONE)`
`ResetSensor (MODIFIER = LOAD_DEFAULT_DATA)`
`SetAllParameters (MODIFIER = NONE)`
Whenever this event is triggered, the internal list of parameters has been updated and the individual parameter values should be retrieved with `GetSingleParameter`.

```
event SensorResultDataReceived(ushort framecounter)
```

This is triggered when result data is received from the sensor. The parameter is an image number that allows the data to be assigned to the corresponding image. The individual results data can be retrieved with `GetResultParameter`.

```
event SingleDataReceived(string configId, string parameterId,
string value)
```

This is triggered after `SetSingleParameter` if this one value only has been changed on the sensor.

```
event AcceptReceived()
```

This is triggered after `SetSingleParameter` if the value previously set has been successfully changed on the sensor.

```
event SensorInformationDataReceived(string type, string version,
string macAddress)
```

This is triggered when any data is received from the sensor. Details of the sensor type, its firmware version, and its MAC address are transmitted.

```
event DisconnectReceived(string errorMessage)
```

This is triggered if the connection is not disconnected with `Disconnect`. A description of the reason for the lost connection is transmitted.

```
event SaveDataOnSensorReceived()
```

This is triggered after the `SaveSensorSettings` function has been called successfully.

```
event GetNetworkReceived(string ipAddress, string mask, string
gateway)
```

This is triggered after the data requested from the sensor with `GetNetworkSettings` has been received. The IP address, the network mask, and the sensor's gateway are transmitted.

```
event LogDataReceived(string logData, LogMessageTypes logType)
```

This is triggered when log data has been received from the sensor. The data and log data type are transmitted as parameters.

```
event ShapeDataReceived(ushort frameCounter)
```

This is triggered when graphics for the image overlay have been received. The image number specifies the image with which the graphics are associated. A list of all graphics can then be retrieved using the `GetShapes(ushort frameCounter)` function.

The following event is supported by laser triangulation sensors only:

```
event LineDataReceived(ushort frameCounter, LineMulti lines,
ushort status)
```

This is triggered when line data has been received from the sensor. In addition to the line data, the image number is issued so that the data can be assigned to the appropriate image. A status is issued that reflects the sensor's hardware inputs and outputs.

```
event ImageReceived(ushort frameCounter)
```

This is triggered when an image has been received from the sensor. The image number is transmitted as a parameter. The image can then be retrieved using the `GetImage(ushort frameCounter)` function.

```
event SaveDataOnHddReceived()
```

This is triggered when the current parameter set has been successfully saved to a file.

```
event ErrorReceived(string id, string name)
```

This is triggered when an error has occurred in the sensor software. The parameters contain more details about this error.

```
event InternalError(ErrorTypes errorType, string errorMessage)
```

This is triggered when an internal error has occurred. This happens in the following cases:

- When calling `GetSingleParameter (ErrorType = PARAMETER_NOT_FOUND)` if the required parameter has not been found in the internal list
- When calling `SetSingleParameter (ErrorType = PARAMETER_NOT_FOUND)` if the parameter to be set is not present
- When calling `SaveSettingsToFile (ErrorType = SAVE_FILE_ERROR)` if the parameter data could not be saved to a file
- When calling `LoadSettingsFromFile (ErrorType = LOAD_FILE_ERROR)` if the parameter file could not be loaded
- When calling `SetAllParameters (ErrorType = LOAD_FILE_ERROR)` if the parameter data could not be loaded from the file or if the file contains invalid data
- In all cases where `ParameterDataReceived` or `SensorResultDataReceived` would be triggered but the sensor data received is incorrect `(ErrorType = INVALID_DATA_RECEIVED)`.
- If log data is received from the sensor but this is incorrect `(ErrorType = INVALID_DATA_RECEIVED)`
- When calling `SendVsxFile (ErrorType = LOAD_FILE_ERROR)` if the file cannot be loaded or does not correspond to the VSX syntax

The `errorMessage` parameter contains a more detailed description of the respective error.

```
enum ErrorTypes{ INVALID_DATA_RECEIVED, SAVE_FILE_ERROR,
LOAD_FILE_ERROR, PARAMETER_NOT_FOUND }
```

See `InternalError` event.

```
enum DataModifier{ NONE, LOAD_DATA, LOAD_DEFAULT_DATA }
```

See `ParameterDataReceived` event.

```
enum LogMessageTypes { DEBUG, INFO, RESULT_OK, RESULT_NOT_OK,
WARNING, ERROR, CRITICAL, ASSERT }
```

See `LogDataReceived` event.

11.2.1 Configuration Overview

The parameters in the table below can be set using the `SetSpecificSingleParameter` and `GetSpecificSingleParameter` method in the software interface. Each parameter is identified by a **configID** and a **parameterID**.

Configuration Parameters

Config ID	Parameter ID	Value	Config Version		Description	From Firmware Version
			from	to		
System	EnableRS-232	0, 1	1		Enables RS-232	5415
	DataSize	5, 6, 7, 8	1		RS-232 data size	5415
	StopBits	1, 2	1		RS-232 stop bits	5415
	BaudRate	BAUD_115200 BAUD_19200 BAUD_38400 BAUD_57600 BAUD_9600	1		RS-232 baud rate	5415
	ParityMode	PARITY_EVEN PARITY_NONE PARITY_ODD	1		RS-232 check bit	5415
	TriggerString	string	1		RS-232 trigger string	5415
	EnableResultPort	0, 1	1		Enables TCP/IP result	5415
	UDP	0, 1	1		Enables UDP result	5415
	PortOut	1 ... 65536	1		TCP port	5415
	PortOutUdp	1 ... 65536	1		UDP port	5415
	PosLeds	0, 1	1		Switches position LEDs on or off	6106
	Enable VGA	0, 1	1		OPE only: Switches VGA output on or off	8738
	PulseLengthGOOD	1 ... 1000	1		Pulse length in ms for GOOD output signal	8474
	PulseLengthBAD	1 ... 1000			Pulse length in ms for BAD output signal	8474
	SensorName	string	1		Customizable string	8470

Config ID	Parameter ID	Value	Config Version		Description	From Firmware Version
			from	to		
	ImageFormat	752x480 752x240 752x160 752x120	1		Image capture area that is captured and processed	6022
	ImageFinePosition	-50 ... +50	1		If the image format has less than 480 lines: vertical line offset relative to the center of the image	8597
	ImageTransfer	ALL_IMG ERROR_IMG GOOD_IMG	1		Selects images to transfer to the connected PC	7818
	ImageTransferActive	0, 1	1		Enables or disables transmission of VSX image information after each cycle	6064
	ImageBuffer	ALL_IMG ERROR_IMG GOOD_IMG	1		Which images are to be saved	7818
	OverwriteBuffer	0, 1	1		Overwrites the images in the memory when the memory is full	8473
Camera	FlashTime	0 ... 255	1		Sets the flash time in ms	5415
	Gain	0 ... 255	1		Sets the image capture time in ms. The AutoExposure selection field must be disabled	5415
Command	TriggerStart	1	1		Triggers an image capture	5930
	BlockTrigger	1	1		Blocks triggers	5971
	ResetCounter	1	1		Resets the counter	8578
	GetLastImg	1	1		Requests the current image	5415
	GetErrImg	1	1		Requests the last fault pattern	5415
	SetDefaultParams	1	1		Sets the parameters to the factory settings	7900

Config ID	Parameter ID	Value	Config Version		Description	From Firmware Version
			from	to		
Trigger Modes	TriggerMode	TRIGGER_DIRECT_MODE TRIGGER_ENCODER_MODE	1		Sets the trigger operating mode	5783
	DirectMode	TRIGGER_DIRECT_EDGE TRIGGER_DIRECT_LEVEL	1		Sets the trigger signal characteristics	6022
	EdgeMode	TRIGGER_EDGE_POS TRIGGER_EDGE_LEVEL	1		Sets the trigger signal polarity Rising or falling edge in DirectMode = TRIGGER_DIRECT_EDGE High level or low level in DirectMode = TRIGGER_DIRECT_LEVEL	6022
	ContinueConnect	0, 1	1		In DirectMode operating mode = TRIGGER_DIRECT_LEVEL: trigger repeated when VSX connection active	7818
	LvlTrigCycle	8 ... 1000	1		Cycle time in DirectMode operating mode = TRIGGER_DIRECT_LEVEL	7869
	CreateDebugImage	0, 1	1		Creates and sends a fault pattern	6064
	EncoderResolution	100 ... 3000	1		OPC and OPD only: number of increment widths per meter	5988
	TriggerOffset	0 ... 32767	1		OPC and OPD only: offset of the first trigger in mm	5988
	TargetDistance	70 ... 200	1		OPC only: distance of the object from the sensor in mm	5988
	SheetLength	100 ... 300	1		OPC and OPD only: length of the sheet in mm	5988

2015-07

Config ID	Parameter ID	Value	Config Version		Description	From Firmware Version
			from	to		
	DistanceTolerance	0 ... 20	1		OPC and OPD only: tolerance for slippage	5988
	MoveDirection	MOVE_HOR_NEGATIVE MOVE_HOR_POSITIVE MOVE_VER_NEGATIVE MOVE_VER_POSITIVE	1		OPC and OPD only: direction of movement of sensor	5988
	ResetCycle	1	1		OPC and OPD only: resets an initiated cycle if TriggerMode operating mode = TRIGGER_ENCODER_MODE	5988
Window N N = 1, 2, 3, 4	WinActive	0, 1	1		Window N enabled/disabled	5415
	XStart	0 ... 751	1		X coordinate for window (ROI)	5415
	YStart	0 ... 479	1		Y coordinate for window (ROI)	5415
	RoiWidth	1 ... 752	1		Window width (ROI)	5415
	RoiHeight	1 ... 480	1		Window height (ROI)	5415
	XStart_Pattern	0 ... 751	1		X coordinate for pattern comparison window	5415
	YStart_Pattern	0 ... 479	1		Y coordinate for pattern comparison window	5415
	RoiWidth_Pattern	64, 128, 196, 256	1		Pattern comparison window width (ROI)	5415
	RoiHeight_Pattern	64, 128, 196, 256	1		Pattern comparison window height (ROI)	5415

Config ID	Parameter ID	Value	Config Version		Description	From Firmware Version
			from	to		
	InspectionTask	INSPECTION_TYPE_1_DECODE_1D2D INSPECTION_TYPE_2_MATCH_1D2D INSPECTION_TYPE_3_MATCH_LOGO INSPECTION_TYPE_4_PRESENCE_CHECK	1		Sets the inspection type	5415
	CheckDataMatrix	0, 1	1		Enables or disables Data Matrix	5415
	CheckInterleaved2of5	0, 1	1		Enables or disables Interleaved 2 of 5	5415
	CheckCode128	0, 1	1		Enables or disables Code128	5415
	CheckCode39	0, 1	1		Enables or disables Code39	5415
	CheckCode13	0, 1	1		Enables or disables Code13	5415
	CheckPharma	0, 1	1		Enables or disables Pharmacode	5415
	MinCodeLen1D	0 ... 255	1		Inputs the minimum code length	7892
	CheckSuml25	0, 1	1		Enables or disables checksum test	5716
	SuppressDuplicates	0, 1	1		Enables or disables suppression of consecutive codes with identical content	7892
	MatchString	string	1		String for reading matches	5415
	UpdateMatchString	1	1		Transfers Match String from last successful reading	5716
	CheckCheckPosition	0, 1	1		Position check enabled for barcodes	5422

2015-07

Config ID	Parameter ID	Value	Config Version		Description	From Firmware Version
			from	to		
	deltaX	0 ... 752	1		Horizontal tolerance for position check for barcodes	5415
	deltaY	0 ... 480	1		Vertical tolerance for position check for barcodes	5415
	PresenceEdgeThreshold	0 ... 255	1		Edge threshold	8102
	PresenceEdgeFilterWidth	1 ... 5	1		Edge filter width	8102
	PresenceEdgeRatioMin	0 ... 2000	1		Edge quotient	8102
	PresenceEdgesHorizontal	0, 1	1		Enables/disables horizontal edge calculation	8102
	PresenceEdgesVertical	0, 1	1		Enables/disabled vertical edge calculation	8102
	PresenceInvertOut	0, 1	1		Inverted evaluation logic	8102
	PresenceEdgeImage	0, 1	1		Shows edge image in original image	8102
	DataMatrixInvers	INVERS_autoChange INVERS_off INVERS_on	1		Decoding process for inverse Data Matrix codes	7892
	DataMatrixMirror	MIRROR_autoChange MIRROR_off MIRROR_on	1		Decoding process for mirrored Data Matrix codes	7892
	Orientation	0 ... 7	1		Permitted orientations of the Data Matrix code	8597
	Timeout1D2D	0 ... 65534	1		Timeout for 1-D/2-D decoding processes	5415
	TimeoutMatch1D2D	0 ... 65534	1		Timeout for 1-D/2-D decoding processes (match)	5415
	GoodString	string	1		Output string from good read; can contain placeholders for inspection information	8349

Config ID	Parameter ID	Value	Config Version		Description	From Firmware Version
			from	to		
	BadString	string	1		Output string from bad read; can contain placeholders for inspection information	8349

Table 11.1 Configuration parameters

11.2.2 Result overview

Accessing the result data

The sensor transmits result data in the following situations:

- After transferring each image
- Trigger
- Counter value request ("ShowCounter")

The result data is contained in the `CONFIGURATION Id="Result"` node. The node contains two different result parameters depending on the situation.

- Result data (PF.Foundation.Protocol.XML.ElementResult)
- Shape data (PF.Foundation.Protocol.XML.ElementShapeBase)

The result data is exclusively text data, which is transmitted in barcode mode and after Teach-in. The shape consists of either geometrical shapes, such as colored rectangles (ElementShapeRectangle), or a type label (ElementShapeText).

Result data

The result data is transmitted with the `SensorResultDataReceived` event and can be retrieved in the event handler using the `GetResultList` and `GetResultParameter` method.

The following event and parameter IDs are permitted:

Result data

Event	Parameter ID	Value	Description	From Firmware
Trigger	ResW**x**		Window **x**: barcode content (decoded string	8597
	ResW**x**Type	DataMatrix Code128 Code39 EAN/UPC/JAN 2/5 Interleaved Pharmacode	Window **x**: barcode type	8597
	Res**x**Time	0 … 65535 ms	Window **x**: Decoding time	8597
	Res**x**Qual	0 … 100%/ 0 … 100%	Window **x**: Location quality horizontal/vertical; 100%/100% means barcode is exactly in the center of the image	8597
ShowCounter	CtrCaptured	0 …	Number of captured images	8606
	CtrLogoMatch	0 …	Number of succesful logo comparisons	8606
	CtrFailedRead	0 …	Number of failed decodings	8606
	CtrFailedMatch	0 …	Number of mismatched decodings	8606

Table 11.2 Result data

Shape data

The shape data is transmitted with the `ShapeDataReceived` and `ImageDataReceived` events. The `GetShapes` method is used to retrieve the data.

The following parameters are transmitted:

Shape data

Type	Parameter ID	Wert	Beschreibung	Ab Firmware
ElementShape Text	Res**x**		Window x: Barcode content (decoded string)	8597
	Res**x**Time	0 65535 ms	Window x: Decoding time	8597
ElementShape Rectangle	Read**x**		Window x: Bounding box around barcode	8597

Table 11.3 Shape data

FACTORY AUTOMATION – SENSING YOUR NEEDS

Worldwide Headquarters
Pepperl+Fuchs GmbH
68307 Mannheim · Germany
Tel. +49 621 776-0
E-mail: info@de.pepperl-fuchs.com

USA Headquarters
Pepperl+Fuchs Inc.
Twinsburg, Ohio 44087 · USA
Tel. +1 330 4253555
E-mail: sales@us.pepperl-fuchs.com

Asia Pacific Headquarters
Pepperl+Fuchs Pte Ltd.
Company Registration No. 199003130E
Singapore 139942
Tel. +65 67799091
E-mail: sales@sg.pepperl-fuchs.com

www.pepperl-fuchs.com

PEPPERL+FUCHS
SENSING YOUR NEEDS

TDOCT-2828CENG
07/2015