Manual
Software SPECTRO3-SLA-Scope V1.0

(PC software for Microsoft® Windows® XP, VISTA, 7)

for color sensors of SPECTRO-3-...-SLA Series
with internal temperature compensation and white light balancing

This manual describes the installation of the PC software for the SPECTRO-3-...-SLA color sensor. As a support for commissioning of the color sensor this manual explains the functional elements of the Windows® user interface.

The SPECTRO-3-...-SLA provides highly flexible signal acquisition. For example, the sensor can be operated in alternating-light mode (AC mode), which makes the sensor insensitive to extraneous light. It also can be set to constant-light mode (DC mode), which makes the sensor extremely fast. An OFF function turns off the integrated light source at the sensor and changes to DC operation. The sensor then can detect so-called "self-luminous objects". With the stepless adjustment of the integrated light source as well as the selectable gain of the receiver signal and an INTEGRAL function the sensor can be set to almost any surface or any "self-luminous object".

When the integrated light source of the SPECTRO-3-...-SLA color sensor is activated, the sensor detects the radiation that is diffusely reflected from the object. As a light source the SPECTRO-3 color sensor uses a white-light LED with adjustable transmitter power. An integrated 3-fold receiver for the red, green, and blue content of the light that is reflected from the object, or the light that is emitted by a "self-luminous object", is used as a receiver.

The sensor is equipped with 3 analog outputs that either provide the red, green, blue components or the calculated color coordinates (X, Y, INT or s, i, M) from 0…+10V or 4…20mA.

Parameters and measurement values can be exchanged between a PC and the SPECTRO-3-...-SLA color sensor through the serial RS232 interface. All the parameters for color detection also can be saved to the non-volatile EEPROM of the SPECTRO-3-...-SLA color sensor through this serial RS232 interface. When parameterisation is finished, the color sensor continues to operate with the current parameters in STAND-ALONE mode without a PC.

The sensors of the SPECTRO-3-...-SLA series can be calibrated (white-light balancing). Balancing can be performed to any white surface. A ColorChecker™ table with 24 color fields is available as an alternative. White-light balancing or calibration can be performed to one of the white fields.

As a light source the SPECTRO-3-UV-SLA (or SPECTRO-3-FIO-UV-SLA) color sensor uses a UV-LED (375 nm) with adjustable transmitter power to excite the luminescent marking. These UV-sensors can be optimally adjusted to almost any luminescent colorant that can be excited in the long-wave UV range (365 nm or 375 nm). The sensors of the SPECTRO-3-…-UV-SLA series also can be calibrated. Analogous to white-light balancing with color sensors, balancing of the SPECTRO-3-UV-SLA (or SPECTRO-3-FIO-UV-SLA) could be performed to any luminescent color marking.

Sensor Instruments GmbH - Schlinding 11 - D-94169 Thurmansbang
Tel.: +49 (0)8544 / 9719-0 - Fax: +49 (0)8544 / 9719-13
E-mail: info@sensorinstruments.de - www.sensorinstruments.de

0 Contents

Shortcuts:

SEND	F9
GET	F10
GO	F11
STOP	F12

1. Installation of the SPECTRO3-SLA-Scope software

Hardware requirements for successful installation of the SPECTRO3-SLA-Scope software:

- Microsoft® Windows® XP, VISTA, 7
- IBM PC AT or compatible
- VGA graphics
- Microsoft-compatible mouse
- CD-ROM drive
- Serial RS232 interface at the PC or USB slot
- Cable **cab-las4/PC** for the RS232 interface or **cab-las4/USB** for USB slot

The SPECTRO3-SLA-Scope software can only be installed under Windows. Windows must therefore be started first, if it is not yet running.

Please install the software as described below:

1. The software can be installed directly from the installation CD-ROM. To install the software, start the SETUP program in the SOFTWARE folder of the CD-ROM.

2. The installation program displays a dialog and suggests to install the software in the C:\"FILENAME" directory on the hard disk. You may accept this suggestion with **OK** or [**ENTER**], or you may change the path as desired. Installation is then performed automatically.

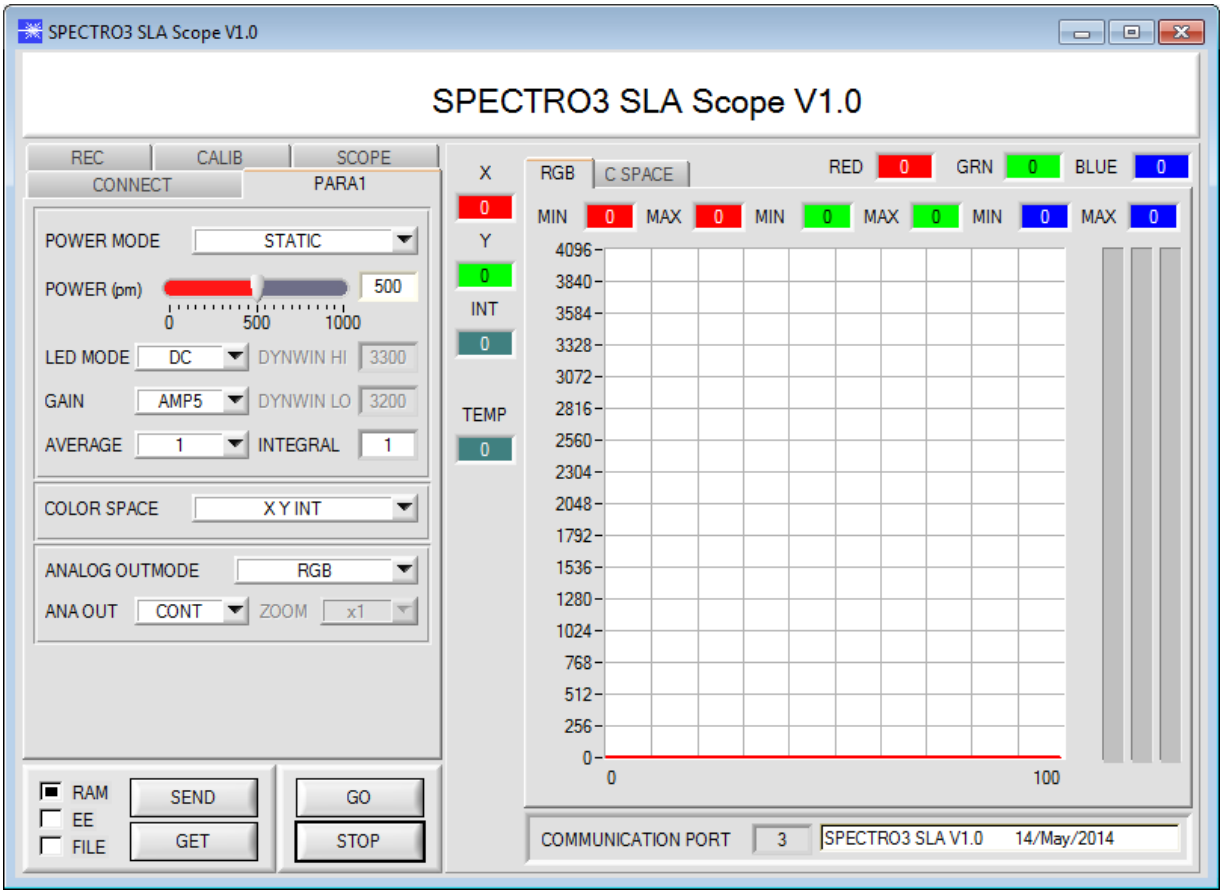
3. During the installation process a new program group for the software is created in the Windows Program Manager. In the program group an icon for starting the software is created automatically. When installation is successfully completed the installation program displays "Setup OK".

4. After successful installation the software can be started with a left mouse button double-click on the icon.

2. Operation of the SPECTRO3-SLA-Scope software

Please read this chapter first before you start to adjust and parameterise the SPECTRO-3-...-SLA color sensor.

When the SPECTRO3-SLA-Scope software is started, the following window appears on the Windows interface:



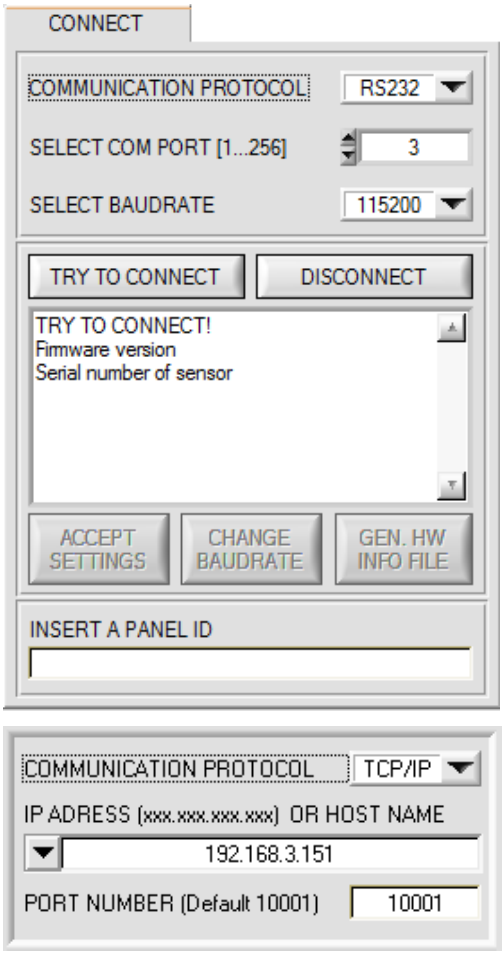
The window size and position will be the same as when the software was last closed. A double-click with the right mouse button e.g. under the minimise symbol places the window centrally in its original size.

If a connection is not established automatically, e.g. if no sensor is connected, the software can be run in OFFLINE mode. In offline mode it only is possible to exchange parameters with a file on a storage medium, which often is helpful for the purpose of analysing parameter files.

If a sensor is connected and a connection still cannot be established, either the SCOPE version (program at the PC) and the firmware version (program in the sensor) do not match, or the interface to the sensor must be correctly configured.
If different Scope and firmware versions should be the problem, please get the Scope version that matches the firmware from your supplier.
The interface configuration is described in the CONNECT tab chapter.

Pressing the right mouse button on an individual element will call up a short help text.

2.1 Tab CONNECT

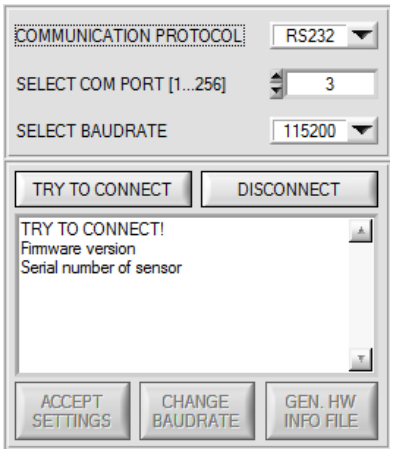


CONNECT:
Pressing the **CONNECT** tab opens a window for selecting and configuring the interface.

The **COMMUNICATION PROTOCOL** function field is used for selecting either an **RS232** or a **TCP/IP** protocol.
If **RS232** is selected, a port from 1 to 256 can be selected with **SELECT COM PORT**, depending on which port the sensor is connected to.
The sensor operates with a set baudrate that can be modified with **CHANGE BAUDRATE** (see below). The sensor and the user interface both must operate with the same baudrate. At the user interface the baudrate is set with **SELECT BAUDRATE**. If after starting the software should not automatically establish a connection, the correct baudrate can be found with **SELECT BAUDRATE**.
If an adaptor is used, the **COM PORT** number can be determined by way of the hardware manager in the system control panel.
If the sensor should communicate through a local area network, an RS232 to Ethernet adaptor will be needed. This adapter makes it possible to establish a connection to the sensor with the **TCP/IP** protocol.
The network adaptors that are available from us are based on the Lantronix XPort module. For parameterising these adapters (assigning of an IP address, setting of the Baud rate of 19200) please download the "DeviceInstaller" software that is provided free of charge by Lantronix at http://www.lantronix.com/. DeviceInstaller is based on Microsoft's ".NET" framework. Detailed operating instructions for the "DeviceInstaller" software also are available from Lantronix.

In order to establish a connection to the adaptor, its IP address or HOST name must be entered in the field **IP ADDRESS (xxx.xxx.xxx.xxx) OR HOST NAME**. The DROP DOWN menu (down arrow) shows the last 10 IP addresses that were used. An address from this list can be directly selected by clicking on the respective item. The DROP DOWN list is saved and is thus always available when the software is closed.
The **PORT NUMBER** for the XPort-based network adaptors is 10001. This port number must not be changed.
When you press the **TRY TO CONNECT** button, the software tries to establish a connection with the set parameters. The communication status is shown in the display field. If the sensor answers with its FIRMWARE ID, the set connection type can be accepted by pressing **ACCEPT SETTINGS**. You will then be returned to the **PARA** tab. If you get a **TIMEOUT** message, the software could not establish a connection to the sensor. In this case please check if the interface cable is correctly connected, if the sensor is supplied with power, and if the set parameters are correct.
If a connection has been accepted by pressing **ACCEPT SETTINGS**, the software starts automatically with these settings when called the next time.
DISCONNECT disconnects the connection between sensor and PC. The software then switches to OFFLINE mode, where it is only possible to exchange parameters with a file on a storage medium.

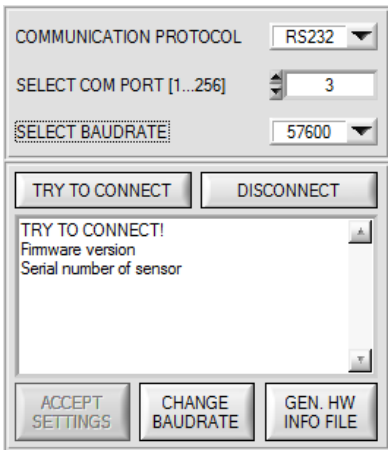
Please note:	**The stable function of the interface is a basic prerequisite for measured value transfer from the PC to the sensor.**
 **ATTENTION !**	**Due to the limited data transfer rate through the serial RS232 interface only slow changes of the raw signals at the sensor front end can be observed in the graphic output window of the PC.** **For maintaining maximum switching frequency at the sensor data communication with the PC must be stopped (press the STOP button).**



The baudrate for data transfer through the RS232 interface can be set by means of the **SELECT BAUDRATE** drop down menu and **CHANGE BAUDRATE** function field.

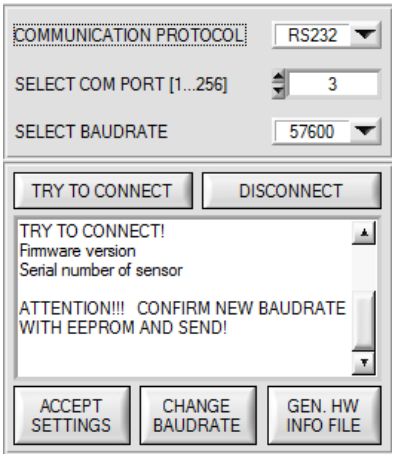
If the baudrate should be changed, a connection must first be established by clicking on **TRY TO CONNECT**.

The **CHANGE BAUDRATE** button will then be active.



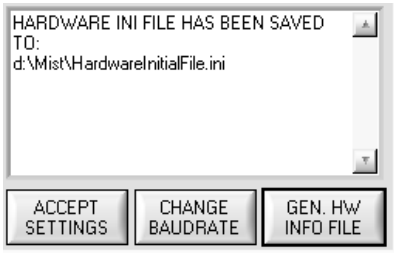
Now a new baudrate can be selected under **SELECT BAUDRATE**.

A click on **CHANGE BAUDRATE** sends the new baudrate information to the sensor.



When the new baudrate information has been successfully sent, the sensor operates with the new baudrate. A window will pop up, prompting you to select **EEPROM** and then to press **SEND**. After a hardware reset the new baudrate only will be used when **EEPROM** and **SEND** have been pressed.

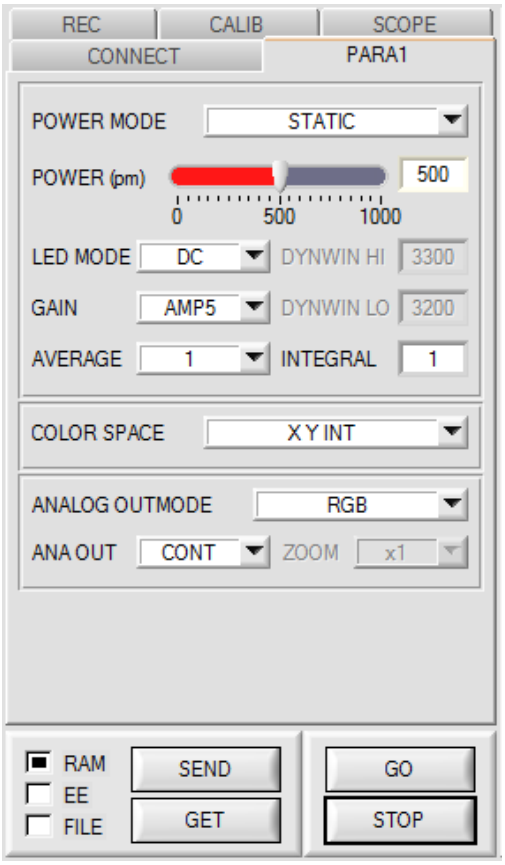
A click on **ACCEPT SETTINGS** saves the current interface settings, which will then be automatically set when the software is restarted.



A click on the **GEN. HW INFO FILE** generates a file in which all the important sensor data are stored in encrypted form.

This file can be sent to the manufacturer for diagnostic purposes.

2.2 Tab PARA1, button SEND, GET, GO, STOP



PARA1:
Pressing the **PARA1** tab opens a window for setting the sensor parameters.

ATTENTION!
A change of the parameter function groups only becomes effective at the sensor after actuation of the SEND button in the MEM function field!

SEND [F9]:
When the **SEND** button is clicked (or shortcut key button F9 is pressed), all the currently set parameters are transferred between PC and sensor. The target of the respective parameter transfer is determined by the selected button (**RAM, EEPROM,** or **FILE**).

GET [F10]:
The currently set values can be interrogated from the sensor by clicking on the **GET** button (or with shortcut key button F10). The source of data exchange is determined by the selected button (**RAM**, **EEPROM**, or **FILE**).

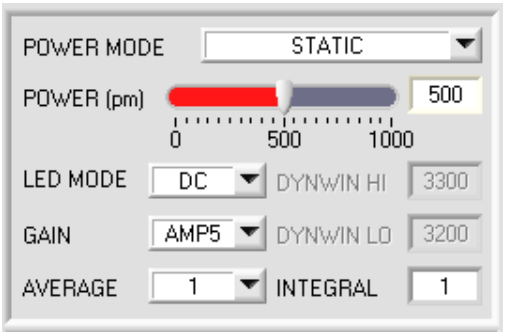
RAM:
After a click on the **SEND** button the current parameters are written into the **RAM** memory of the sensor, or they are read from the **RAM** by clicking on the **GET** button, i.e. these parameters are lost when the voltage at the sensor is switched off.

EEPROM:
After a click on the **SEND** button the current parameters are written into the non-volatile memory of the **EEPROM** in the sensor, or they are read from the **EEPROM** by clicking on the **GET** button, i.e. the parameters in the internal **EEPROM** are stored when the voltage at the sensor is switched off.

FILE:
After pressing **SEND**, the current parameters can be written to a selectable file on the hard disk. With **GET** parameters can be read from such a file. When the **SEND** or **GET** button is pressed, a dialog box opens for selecting the desired file.

GO [F11]:
A click on this button (or with shortcut key button F11) starts data transfer from the sensor to the PC through the serial RS232 interface.
SOURCE is used to select which signals should be shown in the displays and graphs.

STOP [F12]:
A click on this button (or with shortcut key button F12) stops data transfer from the sensor to the PC through the serial RS232 interface.



POWER MODE:
In this function field the operating mode of automatic power correction at the transmitter unit (transmitter LED) can be set.

STATIC:
The transmitter power is constantly kept at the value set with the **POWER [pm]** slider (recommended operation mode). The **POWER** can be set with the slider or by entering a value in the edit-box. A value of 1000 means full intensity at the transmitter unit, a value of 0 sets the lowest intensity at the transmitter.

DYNAMIC:
The LED transmitter power is dynamically controlled in accordance with the amount of radiation that is diffusely reflected from the object. By using the intensities measured at the receivers the automatic control circuit attempts to adjust the transmitter power in such a way that the dynamic range, which is determined by **DYN WIN LO** and **DYN WIN HI**, is not exceeded.

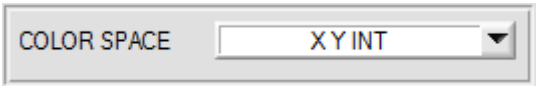
LED MODE:
This item serves for setting the control mode for the integrated light source of the sensor.
DC: In this mode the sensor operates extremely fast. Unfortunately the sensor is somewhat sensitive to extraneous light in DC mode, but if the extraneous light source does not directly shine into the sensor's receiver, the signal only is influenced to a very small extent.
AC: In this mode the sensor is insensitive to extraneous light, which is achieved by "modulating" the integrated light source, i.e. by turning the light on and off. The extraneous content in the signal is determined in off status and is simply subtracted from the on status.
OFF: The sensor's internal light source is turned off in DC mode by POWER [pm] = 0, the sensor can be used for so-called "self-luminous objects". Self-luminous objects are light sources that actively emit light (LEDs, lamps, etc.). In **OFF** mode the **POWER MODE** and **POWER** cannot be adjusted, and external teaching with **DYN1** is not possible.

GAIN:
This item is used for setting the gain of the receiver in 8 different gain stages (AMP1 to AMP8). **GAIN** should be set such that with a medium **POWER** value the sensor operates in its dynamic range (red, green, blue between 2750 and 3750).
In **AC** mode, **GAIN** directly influences the scan frequency. The current scan frequency is displayed in the **SCOPE** tab.

AVERAGE:
This function field is used for adjusting the number of scanning values (measurement values) over which the raw signal measured at the receiver is averaged. A higher **AVERAGE** default value reduces noise of the raw signals at the receiver unit and there will be a decrease of the maximal available switching frequency of the sensor

INTEGRAL:
This function field is used to set the number of scan values (measurement values) over which the raw signal measured at the receiver is summed up. This integral function allows the reliable detection even of extremely weak signals. A higher **INTEGRAL** value increases the noise of the raw signals of the receiver unit, and simultaneously decreases the maximum achievable switching frequency of the sensor.

INFO:
The **POWER** slider is only effective in the **POWER MODE = STATIC**.
DYN WIN LO and **DYN WIN HI** are only effective in **POWER MODE = DYNAMIC.**



COLOR SPACE:
This function field is used to set the color space that should be calculated.
The available settings are **X Y INT** and **s i M**

Calculation of coordinates:

X
X value of the teach color (in the color triangle the numerical value of the x-axis: RED color component)

$$X = \frac{R}{R+G+B} * 4095$$

Y
Y value of the teach color (in the color triangle the numerical value of the y-axis e: GREEN color component)

$$Y = \frac{G}{R+G+B} * 4095$$

INT
Intensity value of the respective color.

$$INT = \frac{R+G+B}{3}$$

s
s is calculated based on the L*a*b* color evaluation method.

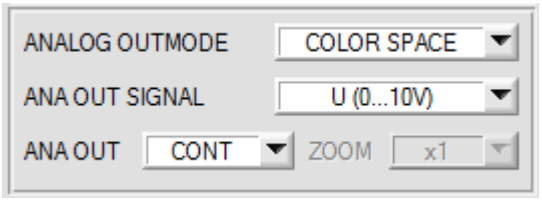
$$s = 5000 * \left(\left(\frac{R}{4096} \right)^{1/3} - \left(\frac{G}{4096} \right)^{1/3} \right) + 5000$$

i
i is calculated based on the L*a*b* color evaluation method.

$$i = 2000 * \left(\left(\frac{G}{4096} \right)^{1/3} - \left(\frac{B}{4096} \right)^{1/3} \right) + 2000$$

M
M is calculated based on the L*a*b* color evaluation method.

$$M = 1160 * \left(\frac{G}{4096} \right)^{1/3}$$



ANALOG OUTMODE:
This function field is used to determine the signals that the sensor provides at its analog outputs.

OFF: No analog signal is output.

R G B:
The **red, green, blue** signals are acquired with 12 bit resolution.
The signals therefore may have values between 0 and 4095.
These values are provided at the respective analog outputs.
The **ANA OUT SIGNAL** is used to select whether the analog signals will be output as voltage from 0…+10V or as current from 4…20mA.

RGB MM:
As long as input IN0 is HI, a maximum and minimum value is determined in the sensor for the red, green, and blue channel.
Within the respective **MIN-MAX** ranges the analog signal for **R G B** is fully output (0…+10V or 4…20mA).
In the graph window the determined **MIN-MAX** values for the **R G B** channels are displayed under the **RGB** tab.

COLOR SPACE:
X Y INT
The **X, Y, INT** color coordinates are calculated scaled to 12 bit.
The signals therefore may have values between 0 and 4095.
These values are provided at the respective analog outputs
s i M:
The **s, i, M** color coordinates are calculated according to the L*a*b* color evaluation method.
This calculation results in the following value ranges:
s: 0 to 10000 **i:** 0 to 4000 **M:** 0 to 1160
The values for **s, i, M** are scaled to 12 bit and are provided at the respective analog outputs.

CS REF:
When input IN0 changes to HI, the current color space value **X, Y, INT** or **s, i, M** are saved as 5 V reference values. This means that when the respective current color space value corresponds with the reference value, 5 V is output at the respective analog output. If the current value differs from the reference value, this difference is amplified by the **ZOOM** factor and provided at the respective analog output starting from 5 V.
Example: ZOOM = x4, **s**(reference) = 6000, **s**(current) = 6100 → difference = 100 (1 digit is approx. 2.44mV)
The output value is: 5V+(100digit * 4(zoom) * 2.44mV) = approx. 5.976V.
In the graph window the reference values for **s, i, M** are displayed in the **INTENSITY** tab.

Since the sensor only features one hardware input (IN0) this input possibility is not always available.
For example, if the setting is **ANA OUT = RGB MM,** the input is needed for **MIN-MAX** searching and can therefore not be used for a controlled output of the analog signal.
Function fields that are not available are either grayed out or deactivated.

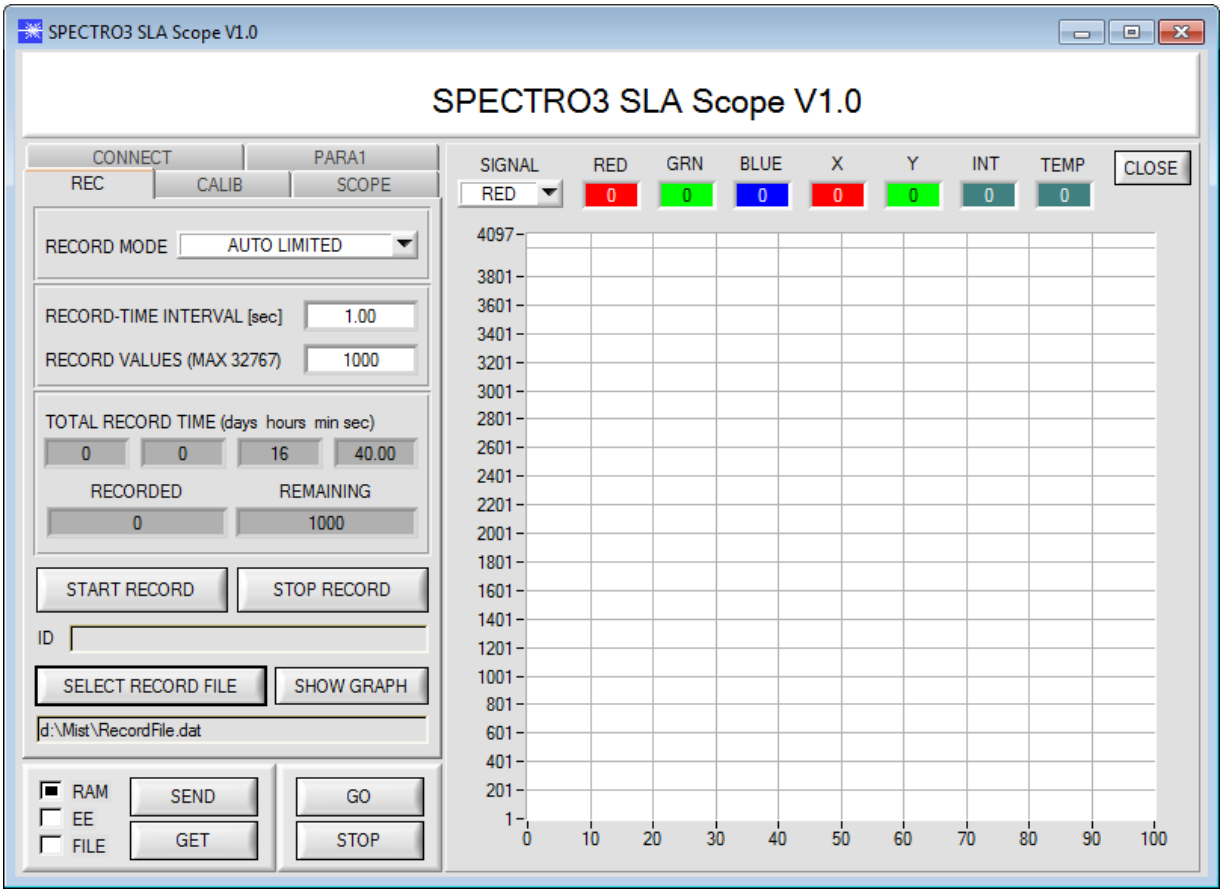
IN0:
On the software interface the status of IN0 is displayed by the **IN0** LED

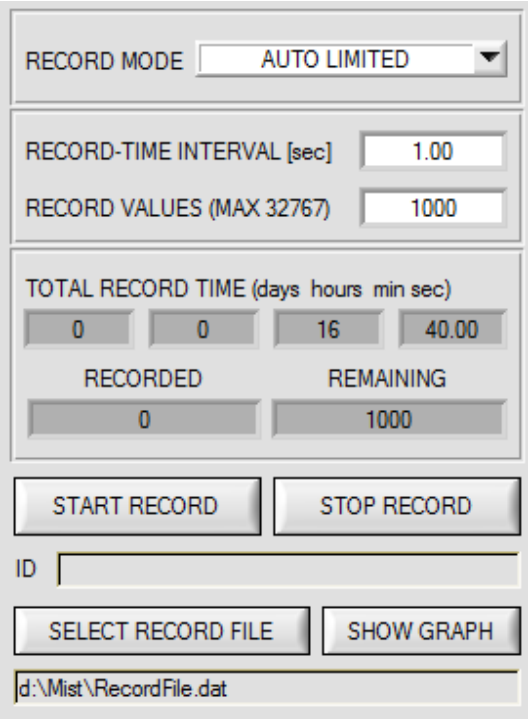
2.6 Tab REC

The SPECTRO3-SLA-Scope software features a data recorder that makes it possible to save the data that are acquired and calculated by the sensor. The recorded file is saved to the hard disk of the PC and can then be evaluated with a spreadsheet program.

The following steps describe how data frames are recorded with the recorder:

Step 1:
When the **REC** button is pressed, the following window will be displayed:
When the **SHOW GRAPH** button is pressed, a panel will be displayed that allows the user to monitor the different signals.
The individual signals can be activated from the **SIGNAL** drop-down menu.





Step 2:
If you want to automatically record several data frames, please select **AUTO LIMITED** under **RECORD MORE**.
Enter a time interval for recording under **RECORD-TIME INTERVAL [sec]**, in this example: 1, i.e. a new value is called from the sensor every second).
Then enter the maximum number of values you wish to record in the **RECORD VALUES [MAX 32767]** field.
Please note: Recording can also be stopped earlier by clicking **STOP RECORD**, the data recorded so far will not be lost.

The **TOTAL RECORD TIME** field indicates how long recording will take (in days, hours, minutes, and seconds) if all data are recorded.

Step 3:
By pressing the button **SELECT RECORD FILE** a file can be selected in which the data frame will be stored.
If you select an already existing file name, you will be asked whether you want to overwrite the existing file or not.

Step 4:
Pressing the **START RECORD** button starts automatic data recording.

The recorder starts to record data, and the button **START RECORD** is red to indicate that recording is active.
The respective data frames are shown in the display windows.

In the two display fields **RECORDED** and **REMAINING** you can check how many data frames have been recorded, and how many frames remain to be recorded.

Please note:
During recording the two input fields RECORD-TIME INTERVAL and VALUES TO BE RECORDED are inactive.

Step 5:
When as many data frames as set under **RECORD VALUES [MAX 32767]** have been recorded, or when the **STOP AUTO RECORD** button is pressed, a pop-up window will appear which confirms that the file is stored.

If you want to record an unlimited number of data, select the **AUTO UNLIMITED** function under **RECORD MORE**.
Then select the desired recording interval and press **START RECORD**.

If you want to record data "manually", select the **MANUAL RECORDING** function under **RECORD MORE**.
You can start reading data from the sensor by pressing the **GO** button. These data are visualised in the display window. Pressing the **CAPTURE DATA FRAME** button saves a data frame in the file that was selected under **SELECT RECORD FILE**. The **RECORDED** field shows the sum of the frames already recorded.

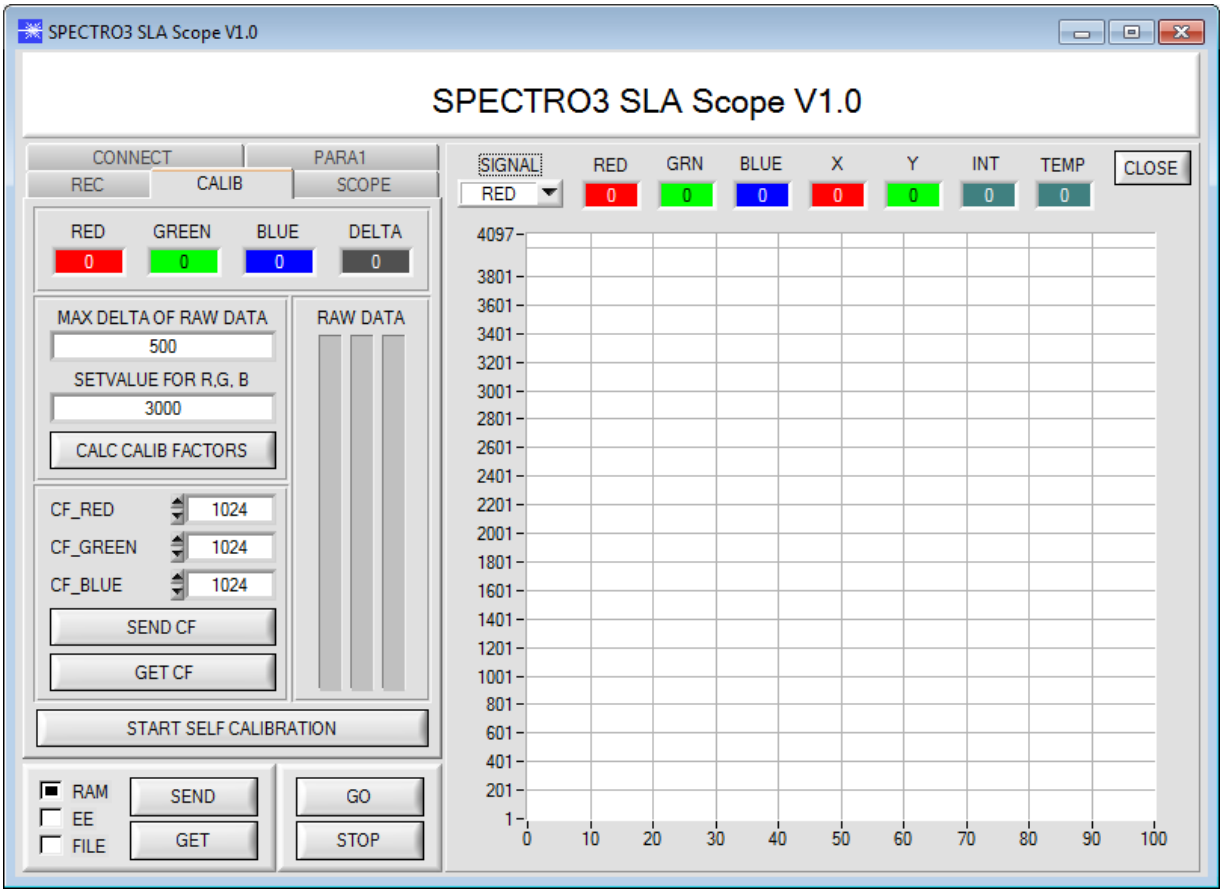
Please note:
When you press START RECORD, the file that is selected under SELECT RECORD FILE will be deleted. With RECORD FRAME MANUALLY, the file will be created if it does not already exist. If the file already exists, the data are added to the existing file.

2.4 Tab CALIB

2.4.1 White light balancing

White light balancing can be performed with the sensors of the SPECTRO-3-...-SLA series. Balancing can be performed to any white surface. A ColorChecker™ table with 24 color fields according to CIE standard is available as an alternative, and white light balancing or calibration can then be performed to one of the white fields

The following panel will be displayed after a click on **CALIB**:

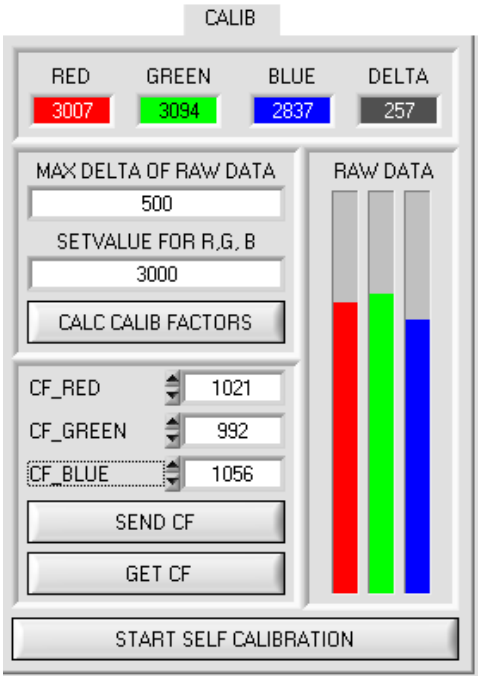


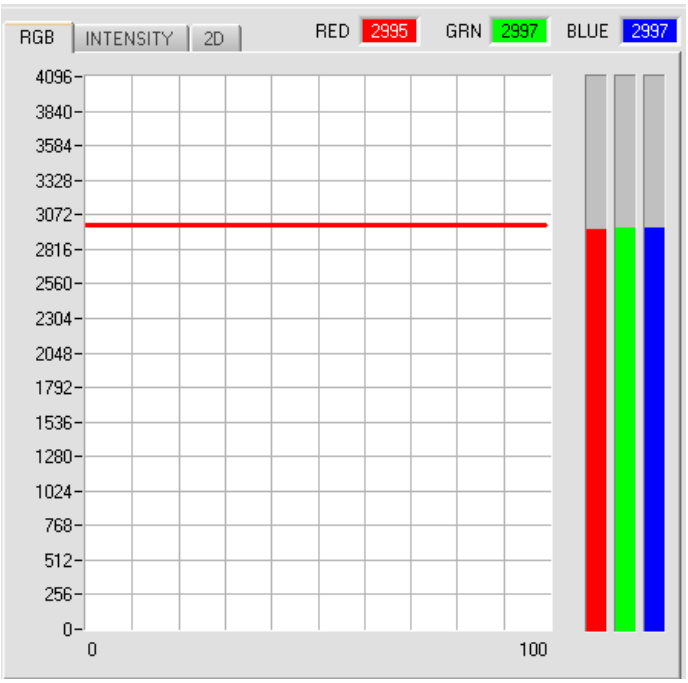
Calculation example for determining the calibration factors

In the example in the picture below, a POWER value at which the three bars of the raw signals **RAW DATA** are in the dynamic range has been set. Each of the three bars is at approx. 3000 digits. It is thus appropriate to set a setpoint value of 3000 (see **SETVALUE FOR R,G,B**) for the three bars. When calibration is now started by pressing CALCULATE CALIBRATION FACTORS, the software automatically calculates the calibration factors for channel RED, channel GREEN, and channel BLUE. The calibration factors are normalized as integers to the value 1024.

Formula:

CF_RED = (SETVALUE / RAW DATA RED) * 1024 = (3000 / 3007) * 1024 = 1021
CF_GREEN = (SETVALUE / RAW DATA GREEN) * 1024 = (3000 / 3097) * 1024 = 992
CF_BLUE = (SETVALUE / RAW DATA BLUE) * 1024 = (3000 / 2908) * 1024 = 1056





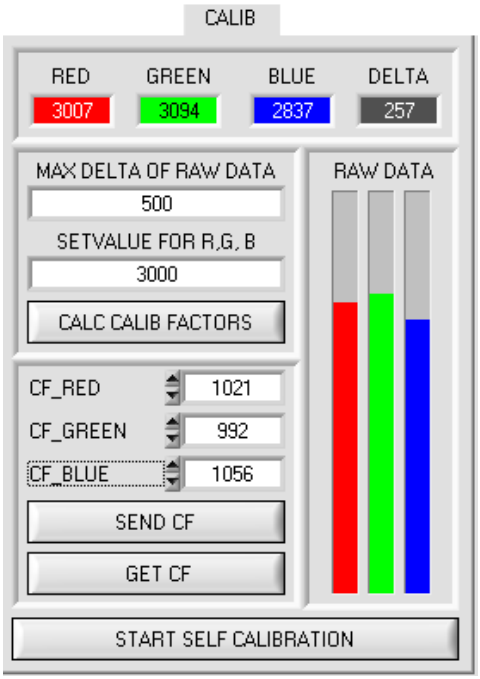
When the calibration factors have been calculated by the software on the user interface, they are automatically saved to the non-volatile **EEPROM** memory of the sensor. Calibration is then finished, work can then be continued in the main panel.

When the sensor detects a raw signal, it applies the calibration factor saved in the **EEPROM** to this raw signal, i.e. in the main panel only the calibrated data for the RED, GREEN, and BLUE channels are displayed. Evaluation by the micro-controller also is exclusively done with the calibrated data.

In the following the individual steps for calibrating the sensors are described:

INFO: The individual pop-up windows are intended as a help to guide you through the calibration process.

ATTENTION: It is a prerequisite for successful calibration that the sensor front-end is calibrated to a white surface.



Step 1:
First of all a suitable **POWER** value must be found such that the **RAW DATA** for RED, GREEN and BLUE lie in the dynamic range (upper third of the bar display).

Step 2:
When you have set a suitable POWER value, determine a **SETVALUE FOR R,G,B**. The software now calculates the calibration factors in such a way that this SETVALUE is reached for the raw data (see calculation example above).

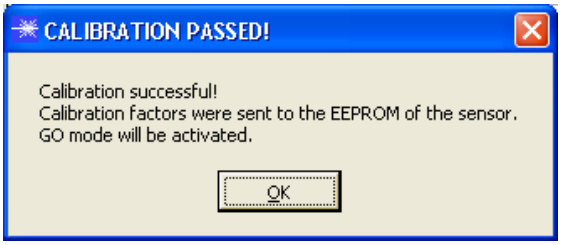
Step 3:
Determine a **MAX DELTA OF RAW DATA** (the software suggests 500).
Calibration is only permitted, if the current **DELTA** of the **RAW DATA** is smaller than the **MAX DELTA OF RAW DATA**.
DELTA is the maximum of **RED**, **GREEN**, and **BLUE**, minus the minimum of **RED**, **GREEN**, and **BLUE**. This is necessary in order to ensure that the sensor functions properly and calibration is performed on a white surface.

Step 4:
Start calibration by pressing **CALC CALIB FACTORS**.
The button starts to flash in red, and at the same time 100 raw data are recorded through the interface, of which the respective mean value of **RED**, **GREEN**, and **BLUE** is formed.
The individual calibration factors are formed from these mean values and from the **SETVALUES FOR R,G,B** and they are then entered in the corresponding edit-boxes.
The calibration software automatically saves the calculated calibration factors to the EEPROM of the sensor.
Then the software changes to the GO mode and displays the **RAW DATA** and the calibrated data in the main panel.
Please note that the values for **RED**, **GREEN**, and **BLUE** in the main panel approximately are equal to the value of SETVALUE.

You may also change the calibration factors **CF_RED**, **CF_GREEN**, **CF_BLUE** manually by entering new values in the corresponding input fields. Please note that these factors are saved to the EEPROM by pressing **SEND CF**. **GET CF** reads the calibration factors that are currently saved in the EEPROM.

If pressing **CALC CALIB FACTORS** should not be successful, please follow the information provided in the pop-up windows.

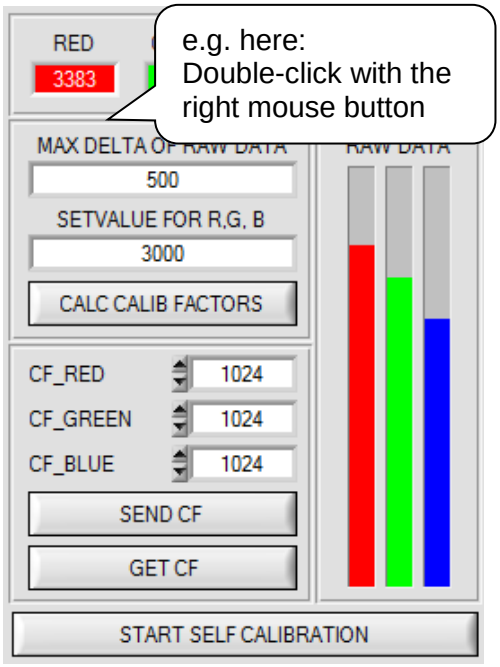
Calibration only is completed successfully, if the following pop-up window is displayed:



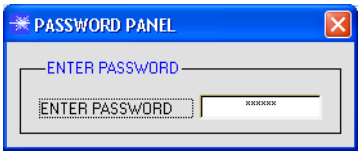
A click on **START SELF CALIBRATION** causes the sensor to calculate the calibration factors itself. It is not possible to specify a **SETVALUE** and a **MAX DELTA** here.
When the sensor has calculated the calibration factors, it displays them on the software interface. In the corresponding input fields it also displays the **SETVALUE** that it used for calculation and the **MAX DELTA** value that resulted from calculation. The **SEND CF** button must be pressed to confirm the calculated calibration factors.

2.4.2 Offset calibration

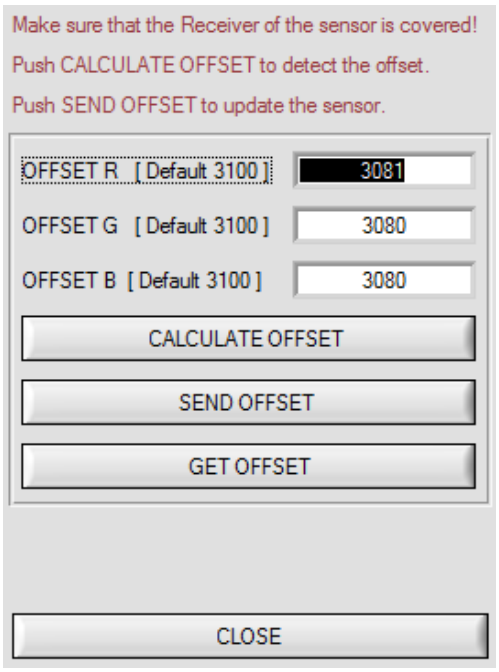
To avoid an increase of the electronic offset when using the integral function (**INTEGRAL** parameter), this offset can be eliminated by way of offset calibration or zero-point calibration. The corresponding tab is password-protected to prevent inadvertent incorrect settings.



Offset calibration can be accessed by double-clicking with the right mouse button exactly at the frame in the **CALIB** tab.



You will then be prompted to enter a password.
The password is: mellon



Now please follow the instructions provided in the tab.

ATTENTION!
It is extremely important for offset calibration that the receiver is not exposed to any extraneous light.
Please make sure that you cover the receiver of the sensor e.g. witch a black cloth that is impervious to light.

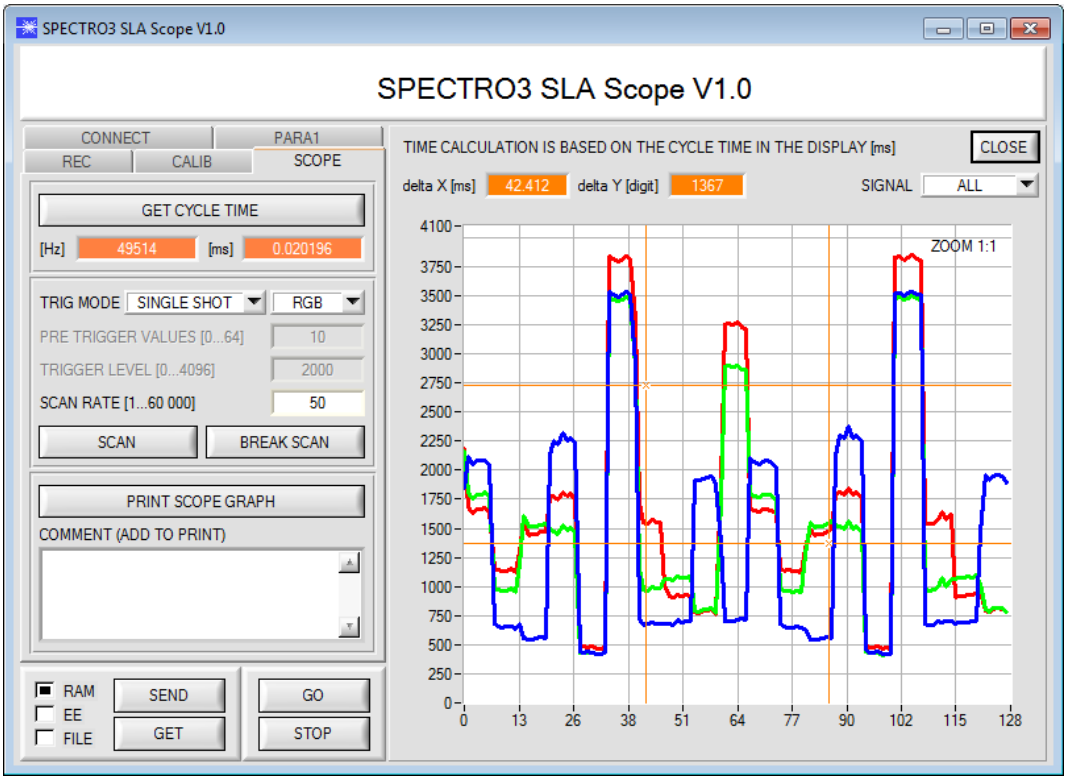
This is absolutely necessary for proper offset calculation.

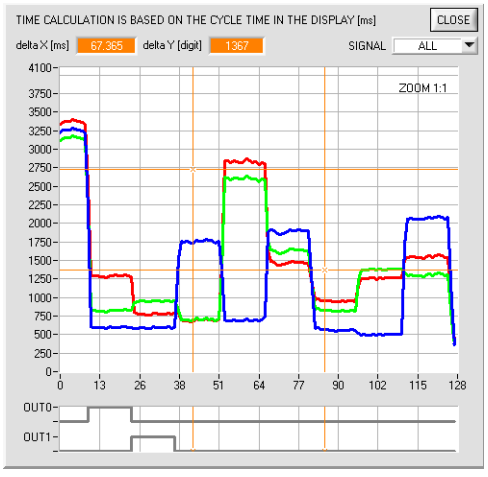
Now press **CALCULATE OFFSET**. The offset values for red, green, and blue should be approximately 3080 plus/minus 40.

The offset values can then be sent to the sensor by pressing **SEND OFFSET**.
GET OFFSET can be used to check whether the data have been sent.

2.5 Tab SCOPE

The SCOPE tab visualises an oscilloscope.
TRIG MODE now optionally displays the signals **R G B**, **X Y INT** or **s i M**, the output analog signal (depending on the setting in **ANA OUT**), or the status of the digital outputs.
A click on **GET CYCLE TIME** displays the current sensor scan frequency in **[Hz]** and **[ms]**. The current scan frequency must be determined for the correct calculation of **deltaX[ms]**. For determining the correct scan frequency please give the sensor a time of 8 seconds before you click on **GET CYCLE TIME**.
In **TRIG MODE = SINGLE SHOT** a data frame is recorded and displayed in the graph when you click on **SCAN**.
In **TRIG MODE = FALLING EDGE** and **RISING EDGE** triggered recording can be started by clicking on **SCAN**. A trigger start can be defined with **TRIGGER LEVEL**.
Triggering is performed either to **BLUE**, **INT** or **M**, depending on the signal that should be recorded. In the graph this is the blue line.
In **TRIG MODE = INTERN C-No.0** recording starts automatically when C-No. 0 is detected. **TRIG MODE= EXTERN IN0** can be used to start recording externally through input IN0.
SCAN-RATE can be used to delay or accelerate recording. This corresponds with the TIMEBASE function known in oscilloscopes. **PRE TRIGGER VALUES** can be used to define how many values should still be displayed before the actual trigger start.





The zoom function in the graph can be activated by holding the control key (CTRL) and drawing a window with the mouse.
A click on **ZOOM 1:1** cancels the zoom function again.

The two orange cursors can be moved with the mouse. The two displays **deltaX[ms]** and **deltaY[digit]** will be updated.
deltaX[ms] shows the time between the cursors in X direction.
deltaY[digit] shows the difference between the two cursors in digits or in Volt in Y direction.

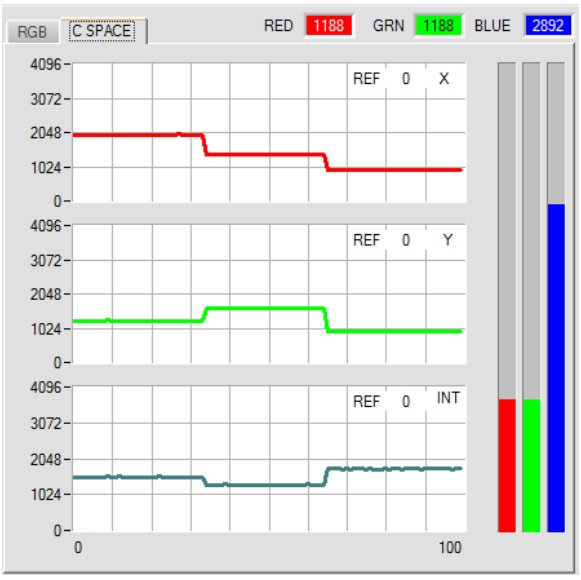
SIGNAL can be used to display individual curves.

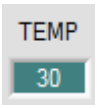
PRINT SCOPE GRAPH prints the current screen together with the text in the **COMMENT** text field.

2.6 Graphic display elements

RGB tab**:**
Display of the current raw signals of the 3-fold receiver (**red, green, blue**).

INTENSITY tab**:**
Display of the currently determined intensity of **X, Y, INT** or **s, i, M**.

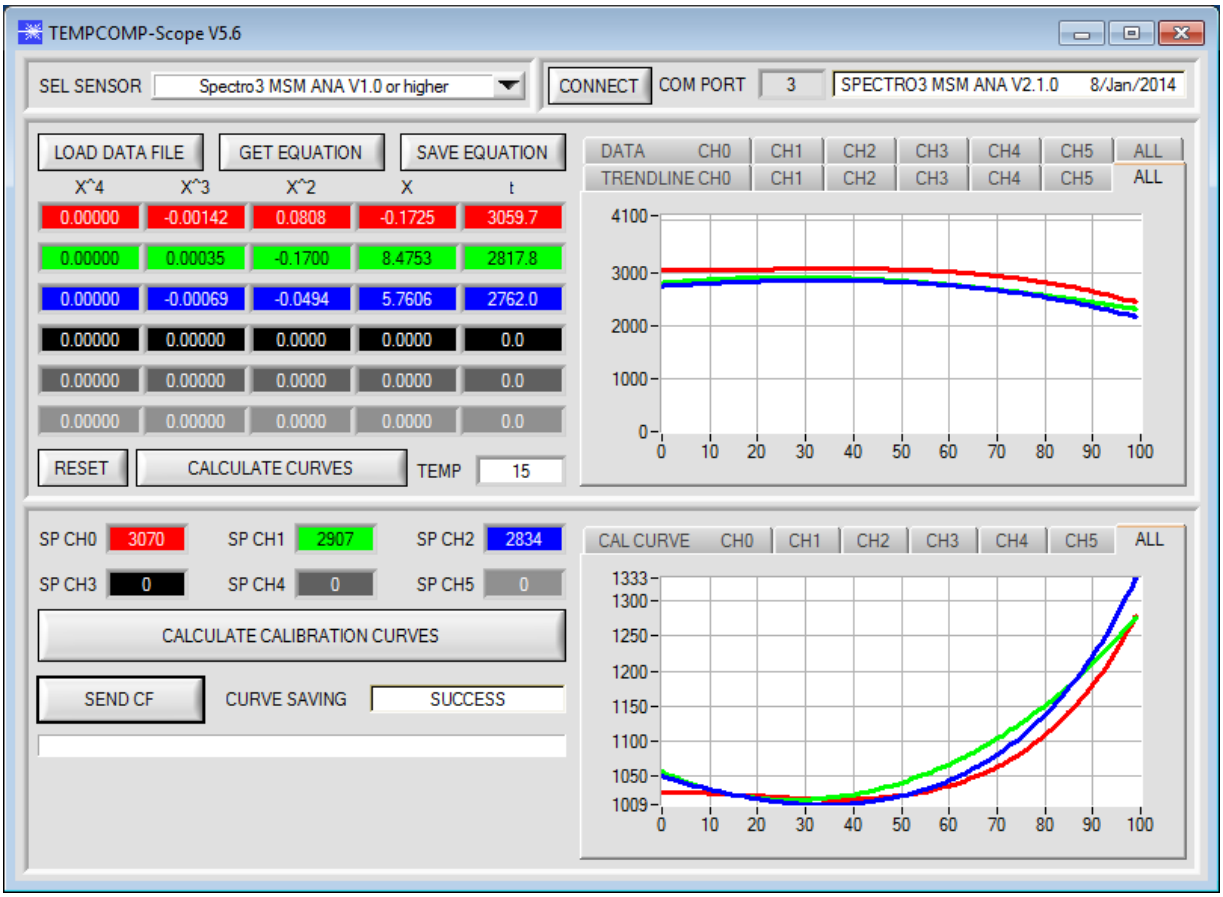




TEMP:
This display shows the temperature prevailing in the sensor housing.
The display **DOES NOT** show degrees Centigrade or Fahrenheit.

3. Operation of the TEMPCOMP-Scope software

If a firmware update should go wrong and the temperature characteristics that are stored in the EEPROM should be lost, these characteristics must be created anew. For this purpose you will need a file with the corresponding data. This file can be obtained from your supplier.
To perform temperature compensation please start the corresponding **TEMPCOMP-Scope software** that is included on the supplied CD. Please make sure that you have a functioning sensor connection. It may be necessary to select the connection with **CONNECT**. Set the correct sensor under **SELECT SENSOR**, if this is not done automatically.



Step 1: Load the temperature compensation file that you have received from your supplier with **GET EQUATION** or **LOAD DATA FILE**.

Step 2: Press **CALCULATE CURVES** to display the data in the graph.

Step 3: Select the sensor-internal operating temperature (not in °C) that the sensor has at an ambient temperature of 20°, if this has not already been done automatically. The value should be included in the file designation.

Step 4: Press **CALCULATE CALIBRATION CURVES** to calculate the mean straight line.

Step 5: Pressing the **SEND CF** button saves the mean straight lines in the **EEPROM** of the sensor.

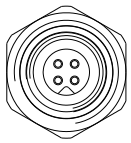
Step 6: Temperature compensation is successfully finished if the **SUCCESS** status message is then displayed.

Comment! If you do not immediately have the temperature compensation file at hand, simply start the **TempComp-Scope software**. Establish a connection, if it is not already established, and simply press **SEND-CF**. The sensor then functions as before, but it is not temperature-compensated.

4. Connector assignment of the SPECTRO-3-...-SLA color sensor

Connection of SPECTRO-3-...-SLA to PC:

4-pole fem. M5 connector (type Binder 707)
SPECTRO-3-...-SLA/PC-RS232

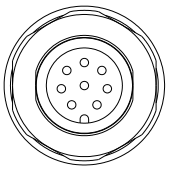


Pin No.:		Assignment:
1		+24VDC (+Ub)
2		0V (GND)
3		Rx0
4		Tx0

Available connecting cables:
cab-las4/PC-…
cab-las4/USB-…
SI-RS232/Ethernet-4-…

Connection of SPECTRO-3-...-SLA to PLC:

8-pole female connector (type Binder 712)
SPECTRO-3-...-SLA/SPS



Pin No.:	Color:	Assignment:
1	White	0V (GND)
2	Brown	+24VDC (± 10%) (+Ub)
3	Green	IN0
4	Yellow	Not connected
5	Grey	Not connected
6	Pink	ANA OUT (Analog **R** or **X** or **s**: 0...+10V or 4…20mA)
7	Blue	ANA OUT (Analog **G** or **Y** or **i**: 0...+10V or 4…20mA)
8	Red	ANA OUT (Analog **B** or **INT** or **M**: 0...+10V or 4…20mA)

Connecting cable:
cab-M12/8-…-shd (shielded)

5. RS232 communication protocol

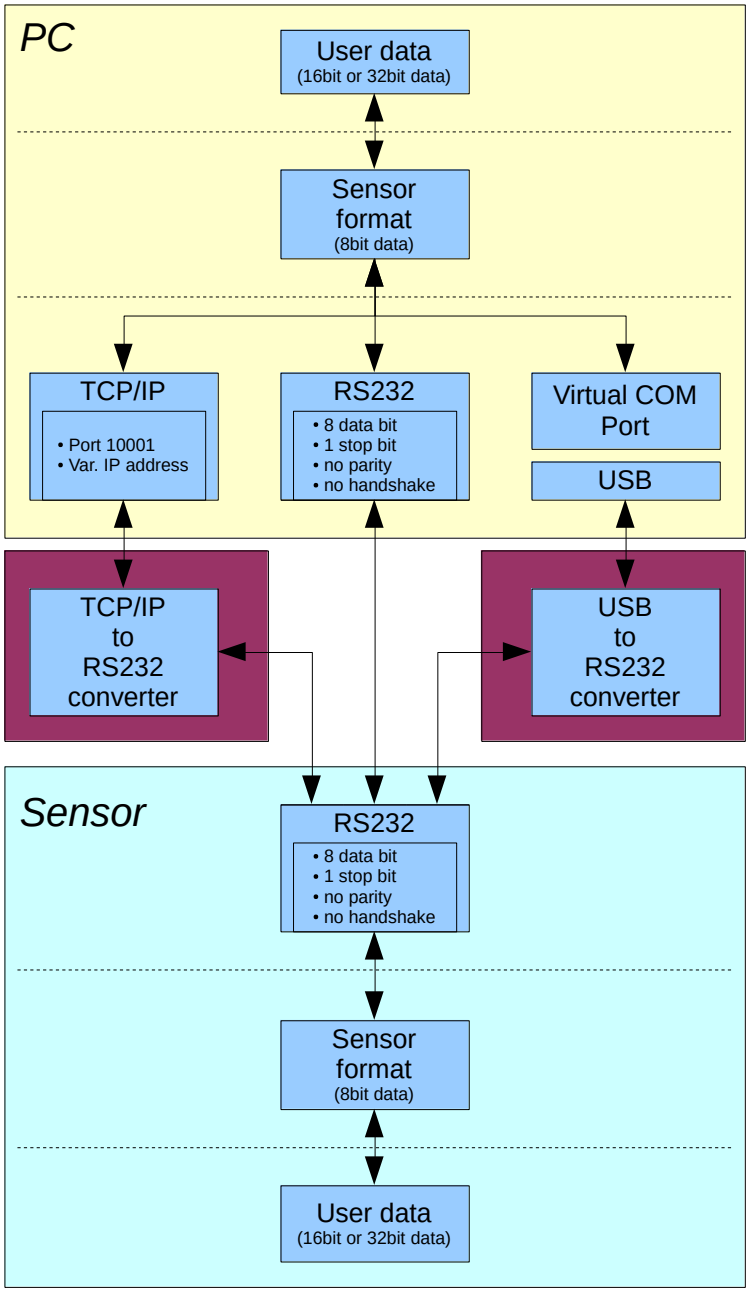
The sensors of the SPECTRO-3-...-SLA series operate with the following **parameters** that are sent to the sensor or read from the sensor in the stated sequence.
Info! 2 bytes (8bit) are one word (16bit).

Parameter		Type	Meaning
Para1:	POWER	Word	Transmitter intensity (0 ... 1000) Attention intensity in thousandth!
Para2:	POWER MODE	Word	Transmitter mode: STATIC, DYNAMIC coded to (0, 1)
Para3:	AVERAGE	Word	Signal averaging 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096, 8192, 16384 or 32768
Para4:	DYN WIN LO	Word	Low limit for dynamic window when POWER MODE=dynamic (0…4095)
Para5:	DYN WIN HI	Word	High limit for dynamic window when POWER MODE=dynamic (0…4095)
Para6:	LED MODE	Word	Control for the internal light source DC, AC, OFF coded to (0,1,2)
Para7:	GAIN	Word	Amplification of the integrated receiver AMP1, AMP2, AMP3, AMP4, AMP5, AMP6, AMP7, AMP8 coded to (1, 2, 3, 4, 5, 6, 7, 8)
Para8:	INTEGRAL	Word	Signal integration (1…250)
Para9:	COLOR SPACE	Word	Color Space X Y INT or s i M coded to (0,1)
Para10:	ANALOG OUTMODE	Word	Function of the analogue outmode: OFF, RGB, RGB MM, COLOR SPACE, CS REF coded to (0,1,2,3,4)
Para11:	ANA OUT SIGNAL	Word	Analogue output signal: U(0…10V), I(4…20mA) coded to (0,1)
Para12:	ANA OUT	Word	Function of analogue out: CONT, IN0 L--->H coded to (0,1)
Para13:	ANA ZOOM	Word	Zoom factor at ANALOG OUTMODE = CS REF: x1, x2, x4, x8, x16, x32, x64, x128 Coded to (0,1,2,3,4,5,6,7)

Upon request, the data acquired and processed by the sensor are sent by the sensor in the following sequence.

DATA VALUE		Type	Meaning
DatVal1:	RED	Word	Calibrated and temperature compensated signal from channel red
DatVal2:	GREEN	Word	Calibrated and temperature compensated signal from channel green
DatVal3:	BLUE	Word	Calibrated and temperature compensated signal from channel blue
DatVal4:	X resp. s	Word	Calculated X respectively s value
DatVal5:	Y resp. i	Word	Calculated Y respectively i value
DatVal6:	INT resp. M	Word	Calculated INT respectively M value
DatVal7:	IN0	Word	IN0 is 1 when input IN0 is HI
DatVal8:	TEMP	Word	Temperature in the sensor (**not** in °C or °F)
DatVal9:	RAW RED	Word	None Calibrated and none temperature compensated signal from channel red
DatVal10:	RAW GREEN	Word	None Calibrated and none temperature compensated signal from channel green
DatVal11:	RAW BLUE	Word	None Calibrated and none temperature compensated signal from channel blue
DatVal12:	MIN RED	Word	Calibrated and temperature compensated minimum signal from channel red
DatVal13:	MIN GREEN	Word	Calibrated and temperature compensated minimum signal from channel green
DatVal14:	MIN BLUE	Word	Calibrated and temperature compensated minimum signal from channel blue
DatVal15:	MAX RED	Word	Calibrated and temperature compensated maximum signal from channel red
DatVal16:	MAX GREEN	Word	Calibrated and temperature compensated maximum signal from channel green
DatVal17:	MAX BLUE	Word	Calibrated and temperature compensated maximum signal from channel blue
DatVal18:	REF CSX	Word	Reference Value when ANALOG OUTMODE = CS REF
DatVal19:	REF CSY	Word	Reference Value when ANALOG OUTMODE = CS REF
DatVal20:	REF CSI	Word	Reference Value when ANALOG OUTMODE = CS REF

Digital serial communication is used for the exchange of data between the software running on the PC and the sensor.
For this purpose the control unit features an EIA-232 compatible interface that operates with the (fixed) parameters "**8 data bits, 1 stop bit, no parity bit, no handshake**".
Five values are available for the baudrate: 9600baud, 19200baud, 38400baud, 57600baud and 115200baud. As an option the PC software also can communicate through TCP/IP or USB. In these cases transparent interface converters must be used that allow a connection to the RS232 interface.



A proprietary protocol format that organises and bundles the desired data is used for all physical connection variants between PC software and control unit. Depending on their type and function the actual data are 16- or 32-bit variables and represent integer or floating-point values. The protocol format consists of 8-bit wide unsigned words ("bytes"). The actual data therefore sometimes must be distributed to several bytes.

The control unit always behaves passively (except if another behaviour has been specifically activated). Data exchange therefore always is initiated by the PC software. The PC sends a data package ("frame") corresponding to the protocol format, either with or without appended data, to which the control unit responds with a frame that matches the request.

The protocol format consists of two components:
A "header" and an optional appendant ("data").
The header always has the same structure.
The first byte is a synchronisation byte and always is 85_{dez} (55_{hex}).
The second byte is the so-called order byte. This byte determines the action that should be performed (send data, save data, etc.).
A 16-bit value (argument) follows as the third and fourth byte. Depending on the order, the argument is assigned a corresponding value.
The fifth and sixth byte again form a 16-bit value. This value states the number of appended data bytes. Without appended data both these bytes are 0_{dez} or 00_{hex}, the maximum number of bytes is 512.
The seventh byte contains the CRC8 checksum of all data bytes (data byte 0 up to and incl. data byte n).
The eight byte is the CRC8 checksum for the header and is formed from bytes 0 up to and incl. 6.
The header always has a total length of 8 bytes. The complete frame may contain between 8 and 520 bytes.

Byte0 Header	Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header	Byte8 Data	Byte9 Data	…	Byte n+6 Data	Byte n+7 Data
0x55	<order>	<ARG> (lo byte)	<ARG> (hi byte)	<LEN> (lo byte)	<LEN> (hi byte)	CRC8 (Data)	CRC8 (Header)	Data1 (lo byte)	Data1 (hi byte)	…	Data n/2 (lo byte)	Data n/2 (hi byte)

The following **orders** can be sent to the sensor.

Number	ORDER (header byte no. 2)	Example
0	Sensor answers with order=0 if a communication error occurs. ARG=1: Invalid order number was sent to the sensor ARG=2: General communication error (wrong baudrate, overflow, …)	
1	Write parameter to the RAM of the sensor	order=1
2	Read parameter from the RAM of the sensor	order=2
3	Load parameter and actual Baudrate from RAM to EEPROM of the sensor	order=3
4	Load parameter from EEPROM to RAM of the sensor	order=4
5	Read CONNECTION OK and serial number from sensor	order=5
6	Free	
7	Read Firmware String and serial number from sensor	order=7
8	Read data values from sensor	order=8
103	Start white light correction and get calibration factors, setvalue and max delta of raw data.	order=103
105	Get cycle time from sensor	order=105
190	Write new baud rate to the sensor	order=190

CRC8 checksum

The so-called "Cyclic Redundancy Check" or CRC is used to verify data integrity. This algorithm makes it possible to detect individual bit errors, missing bytes, and faulty frames. For this purpose a value - the so-called checksum - is calculated over the data (bytes) to be checked and is transmitted together with the data package. Calculation is performed according to an exactly specified method based on a generator polynomial. The length of the checksum is 8 bit (= 1 byte). The generator polynomial is:

$$X^8+X^5+X^4+X^0$$

To verify the data after they have been received, CRC calculation is performed once again. If the sent and the newly calculated CRC values are identical, the data are without error.
The following pseudo code can be used for checksum calculation:

```
calcCRC8 (data[ ], table[ ])
Input:  data[ ], n data of unsigned 8bit
        table[ ], 256 table entries of unsigned 8bit
Output: crc8, unsigned 8bit

crc8  := AA_hex
for I := 1 to n do
        idx := crc8 EXOR data[ i ]
        crc8 := table[ idx ]
endfor
return  crc8
```

table[ ]

0	94	188	226	97	63	221	131	194	156	126	32	163	253	31	65
157	195	33	127	252	162	64	30	95	1	227	189	62	96	130	220
35	125	159	193	66	28	254	160	225	191	93	3	128	222	60	98
190	224	2	92	223	129	99	61	124	34	192	158	29	67	161	255
70	24	250	164	39	121	155	197	132	218	56	102	229	187	89	7
219	133	103	57	186	228	6	88	25	71	165	251	120	38	196	154
101	59	217	135	4	90	184	230	167	249	27	69	198	152	122	36
248	166	68	26	153	199	37	123	58	100	134	216	91	5	231	185
140	210	48	110	237	179	81	15	78	16	242	172	47	113	147	205
17	79	173	243	112	46	204	146	211	141	111	49	178	236	14	80
175	241	19	77	206	144	114	44	109	51	209	143	12	82	176	238
50	108	142	208	83	13	239	177	240	174	76	18	145	207	45	115
202	148	118	40	171	245	23	73	8	86	180	234	105	55	213	139
87	9	235	181	54	104	138	212	149	203	41	119	244	170	72	22
233	183	85	11	136	214	52	106	43	117	151	201	74	20	246	168
116	42	200	150	21	75	169	247	182	232	10	84	215	137	107	53

Example order=1: Write parameter to the RAM of the sensor.

DATA FRAME PC → Sensor

Byte0 Header	Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header	Byte8 Data	Byte9 Data	Byte10 Data	Byte11 Data
0x55	<order>	<ARG> (lo byte)	<ARG> (hi byte)	<LEN> (lo byte)	<LEN> (hi byte)	CRC8 (Data)	CRC8 (Header)	Para1 (lo byte)	Para1 (hi byte)	Para2 (lo byte)	Para2 (hi byte)
85 (dec)	1	0	0	26	0	95	139	244	1	0	0
		ARG=0		LEN=26				Para1=500		Para2=0	

Byte12 Data	Byte13 Data	Byte14 Data	Byte15 Data	Byte16 Data	Byte17 Data	Byte18 Data	Byte19 Data	Byte20 Data	Byte21 Data	Byte22 Data	Byte23 Data
Para3 (lo byte)	Para3 (hi byte)	Para4 (lo byte)	Para4 (hi byte)	Para5 (lo byte)	Para5 (hi byte)	Para6 (lo byte)	Para6 (hi byte)	Para7 (lo byte)	Para7 (hi byte)	Para8 (lo byte)	Para8 (hi byte)
1	0	128	12	228	12	0	0	5	0	1	0
Para3=1		Para4=3200		Para5=3300		Para6=0		Para7=2		Para8=1	

Byte24 Data	Byte25 Data	Byte26 Data	Byte27 Data	Byte28 Data	Byte29 Data	Byte30 Data	Byte31 Data	Byte30 Data	Byte31 Data
Para9 (lo byte)	Para9 (hi byte)	Para10 (lo byte)	Para10 (hi byte)	Para11 (lo byte)	Para11 (hi byte)	Para12 (lo byte)	Para12 (hi byte)	Para13 (lo byte)	Para13 (hi byte)
0	0	1	0	0	0	0	0	0	0
Para9=0		Para10=1		Para11=0		Para12=0		Para 13=0	

DATA FRAME Sensor → PC

Byte0 Header	Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header
0x55	<order>	<ARG> (lo byte)	<ARG> (hi byte)	<LEN> (lo byte)	<LEN> (hi byte)	CRC8 (Data)	CRC8 (Header)
85 (dec)	1	0	0	0	0	170	224
		ARG=0		LEN=0			

If you receive an argument greater than 0, ARG parameter where out of range and have been set to a default value.

Example order=2: Read parameter from the RAM of the sensor.

DATA FRAME PC → Sensor

Byte0 Header	Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header
0x55	<order>	<ARG> (lo byte)	<ARG> (hi byte)	<LEN> (lo byte)	<LEN> (hi byte)	CRC8 (Data)	CRC8 (Header)
85 (dec)	2	0	0	0	0	170	185
		ARG=0		LEN=0			

DATA FRAME Sensor → PC

Byte0 Header	Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header	Byte8 Data	Byte9 Data	Byte10 Data	Byte11 Data
0x55	<order>	<ARG> (lo byte)	<ARG> (hi byte)	<LEN> (lo byte)	<LEN> (hi byte)	CRC8 (Data)	CRC8 (Header)	Para1 (lo byte)	Para1 (hi byte)	Para2 (lo byte)	Para2 (hi byte)
85 (dec)	2	0	0	26	0	95	210	244	1	0	0
		ARG=0		LEN=26				Para1=500		Para2=0	

Byte12 Data	Byte13 Data	Byte14 Data	Byte15 Data	Byte16 Data	Byte17 Data	Byte18 Data	Byte19 Data	Byte20 Data	Byte21 Data	Byte22 Data	Byte23 Data
Para3 (lo byte)	Para3 (hi byte)	Para4 (lo byte)	Para4 (hi byte)	Para5 (lo byte)	Para5 (hi byte)	Para6 (lo byte)	Para6 (hi byte)	Para7 (lo byte)	Para7 (hi byte)	Para8 (lo byte)	Para8 (hi byte)
1	0	128	12	228	12	0	0	5	0	1	0
Para3=1		Para4=3200		Para5=3300		Para6=0		Para7=2		Para8=1	

Byte24 Data	Byte25 Data	Byte26 Data	Byte27 Data	Byte28 Data	Byte29 Data	Byte30 Data	Byte31 Data
Para9 (lo byte)	Para9 (hi byte)	Para10 (lo byte)	Para10 (hi byte)	Para11 (lo byte)	Para11 (hi byte)	Para12 (lo byte)	Para12 (hi byte)
0	0	1	0	0	0	0	0
Para9=0		Para10=1		Para11=0		Para12=0	

Example order=3: Load parameter and actual Baudrate from RAM to EEPROM of the sensor.

DATA FRAME PC → Sensor

Byte0 Header	Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header
0x55	<order>	<ARG> (lo byte)	<ARG> (hi byte)	<LEN> (lo byte)	<LEN> (hi byte)	CRC8 (Data)	CRC8 (Header)
85 (dec)	3	0	0	0	0	170	142
		ARG=0		LEN=0			

- DATA FRAME Sensor → PC
-

Byte0 Header	Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header
0x55	<order>	<ARG> (lo byte)	<ARG> (hi byte)	<LEN> (lo byte)	<LEN> (hi byte)	CRC8 (Data)	CRC8 (Header)
85 (dec)	3	0	0	0	0	170	142
		ARG=0		LEN=0			

Example order=4: Load parameter from EEPROM to RAM of the sensor.

DATA FRAME PC → Sensor

Byte0 Header	Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header
0x55	<order>	<ARG> (lo byte)	<ARG> (hi byte)	<LEN> (lo byte)	<LEN> (hi byte)	CRC8 (Data)	CRC8 (Header)
85 (dec)	4	0	0	0	0	170	11
		ARG=0		LEN=0			

DATA FRAME Sensor → PC

Byte0 Header	Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header
0x55	<order>	<ARG> (lo byte)	<ARG> (hi byte)	<LEN> (lo byte)	<LEN> (hi byte)	CRC8 (Data)	CRC8 (Header)
85 (dec)	4	0	0	0	0	170	11
		ARG=0		LEN=0			

Example order=5: Read CONNECTION OK from sensor.

DATA FRAME PC → Sensor
ARG determines the serial number of the sensor

Byte0 Header	Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header
0x55	<order>	<ARG> (lo byte)	<ARG> (hi byte)	<LEN> (lo byte)	<LEN> (hi byte)	CRC8 (Data)	CRC8 (Header)
85 (dec)	5	0	0	0	0	170	60
		ARG=0		LEN=0			

DATA FRAME Sensor → PC

Byte0 Header	Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header
0x55	<order>	<ARG> (lo byte)	<ARG> (hi byte)	<LEN> (lo byte)	<LEN> (hi byte)	CRC8 (Data)	CRC8 (Header)
85 (dec)	5	170	0	0	0	170	178
		ARG=170		LEN=0			

Example order=7: Read Firmware String from sensor

DATA FRAME PC → Sensor
ARG determines the serial number of the sensor

Byte0 Header	Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header
0x55	<order>	<ARG> (lo byte)	<ARG> (hi byte)	<LEN> (lo byte)	<LEN> (hi byte)	CRC8 (Data)	CRC8 (Header)
85 (dec)	7	0	0	0	0	170	82
		ARG=0		LEN=0			

DATA FRAME Sensor → PC

Byte0 Header	Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header	Byte8 Data	Byte9 Data	Byte10 Data	Byte11 Data
0x55	<order>	<ARG> (lo byte)	<ARG> (hi byte)	<LEN> (lo byte)	<LEN> (hi byte)	CRC8 (Data)	CRC8 (Header)	ASCII	ASCII	ASCII	ASCII
85 (dec)	7	0	0	72	0	183	38	F	I	R	M
		ARG=0		LEN=72							

Byte12 Data	Byte13 Data	Byte14 Data	Byte15 Data	Byte16 Data	Byte17 Data	Byte18 Data	Byte19 Data	Byte20 Data	Byte21 Data	Byte22 Data	Byte23 Data
ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII
W	A	R	E		S	T	R	I	N	G	

Byte24 Data	Byte25 Data	Byte26 Data	Byte27 Data	Byte28 Data	Byte29 Data	Byte30 Data	Byte31 Data	Byte32 Data	Byte33 Data	Byte34 Data	Byte35 Data
ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII
											R

Byte36 Data	Byte37 Data	Byte38 Data	Byte39 Data	Byte40 Data	Byte41 Data	Byte42 Data	Byte43 Data	Byte44 Data	Byte45 Data	Byte46 Data	Byte47 Data
ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII
T	:	K	W	x	x	/	x	x			

Byte48 Data	Byte49 Data	Byte50 Data	Byte51 Data	Byte52 Data	Byte53 Data	Byte54 Data	Byte55 Data	Byte56 Data	Byte57 Data	Byte58 Data	Byte59 Data
ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII

Byte60 Data	Byte61 Data	Byte62 Data	Byte63 Data	Byte64 Data	Byte65 Data	Byte66 Data	Byte67 Data	Byte68 Data	Byte69 Data	Byte70 Data	Byte71 Data
ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII

Byte72 Data	Byte73 Data	Byte74 Data	Byte75 Data	Byte76 Data	Byte77 Data	Byte78 Data	Byte79 Data	Byte80 Data	Byte81 Data
ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII

Example order=8: Read data values from sensor.

DATA FRAME PC → Sensor

Byte0 Header	Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header
0x55	<order>	<ARG> (lo byte)	<ARG> (hi byte)	<LEN> (lo byte)	<LEN> (hi byte)	CRC8 (Data)	CRC8 (Header)
85 (dec)	8	0	0	0	0	170	118
		ARG=0		LEN=0			

DATA FRAME Sensor → PC

Byte0 Header	Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header	Byte8 Data	Byte9 Data	Byte10 Data	Byte11 Data
0x55	<order>	<ARG> (lo byte)	<ARG> (hi byte)	<LEN> (lo byte)	<LEN> (hi byte)	CRC8 (Data)	CRC8 (Header)	DatVal1 (lo byte)	DatVal1 (hi byte)	DatVal2 (lo byte)	DatVal2 (hi byte)
85 (dec)	8	0	0	40	0	55	43	54	10	151	6
		ARG=0		LEN=40				DatVal1=2614		DatVal2 =1687	

Byte12 Data	Byte13 Data	Byte14 Data	Byte15 Data	Byte16 Data	Byte17 Data	Byte18 Data	Byte19 Data	Byte20 Data	Byte21 Data	Byte22 Data	Byte23 Data
DatVal3 (lo byte)	DatVal3 (hi byte)	DatVal4 (lo byte)	DatVal4 (hi byte)	DatVal5 (lo byte)	DatVal5 (hi byte)	DatVal6 (lo byte)	DatVal6 (hi byte)	DatVal7 (lo byte)	DatVal7 (hi byte)	DatVal8 (lo byte)	DatVal8 (hi byte)
153	4	162	7	237	4	34	7	0	0	32	0
DatVal3=1177		DatVal4=1954		DatVal5=1261		DatVal6=1826		DatVal7=0		DatVal8=32	

Byte24 Data	Byte25 Data	Byte26 Data	Byte27 Data	Byte28 Data	Byte29 Data	Byte30 Data	Byte31 Data	Byte32 Data	Byte33 Data	Byte34 Data	Byte35 Data
DatVal 9 (lo byte)	DatVal 9 (hi byte)	DatVal 10 (lo byte)	DatVal 10 (hi byte)	DatVal 11 (lo byte)	DatVal 11 (hi byte)	DatVal 12 (lo byte)	DatVal 12 (hi byte)	DatVal 13 (lo byte)	DatVal 13 (hi byte)	DatVal 14 (lo byte)	DatVal 14 (hi byte)
54	10	151	6	153	4	0	0	0	70	0	0
DatVal9=2614		DatVal10=1687		DatVal11=1177		DatVal12=0		DatVal13=0		DatVal14=0	

Byte36 Data	Byte37 Data	Byte38 Data	Byte39 Data	Byte40 Data	Byte41 Data	Byte42 Data	Byte43 Data	Byte44 Data	Byte45 Data	Byte46 Data	Byte47 Data
DatVal 15 (lo byte)	DatVal 15 (hi byte)	DatVal 16 (lo byte)	DatVal 16 (hi byte)	DatVal 17 (lo byte)	DatVal 17 (hi byte)	DatVal 18 (lo byte)	DatVal 18 (hi byte)	DatVal 19 (lo byte)	DatVal 19 (hi byte)	DatVal 20 (lo byte)	DatVal 20 (hi byte)
0	0	0	0	0	0	0	0	0	0	0	0
DatVal15=0		DatVal16=0		DatVal17=0		DatVal18=0		DatVal19=0		DatVal20=0	

Example order=103: Start white light correction and get calibration factors, setvalue and max delta of raw data.

DATA FRAME PC → Sensor

Byte0 Header	Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header
0x55	<order>	<ARG> (lo byte)	<ARG> (hi byte)	<LEN> (lo byte)	<LEN> (hi byte)	CRC8 (Data)	CRC8 (Header)
85 (dec)	103	0	0	0	0	170	145
		ARG=0		LEN=0			

DATA FRAME Sensor → PC

Byte0 Header	Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header	Byte8 Data	Byte9 Data	Byte10 Data	Byte11 Data
0x55	<order>	<ARG> (lo byte)	<ARG> (hi byte)	<LEN> (lo byte)	<LEN> (hi byte)	CRC8 (Data)	CRC8 (Header)	CF RED (lo byte)	CF RED (hi byte)	CF GREEN (lo byte)	CF GREEN (hi byte)
85 (dec)	103	0	0	10	0	212	28	228	3	223	3
		ARG=0		LEN=28				CF_RED = 996		CF_GREEN = 991	

Byte12 Data	Byte13 Data	Byte14 Data	Byte15 Data	Byte16 Data	Byte17 Data
CF BLUE (lo byte)	CF BLUE (hi byte)	SET VALUE (lo byte)	SET VALUE (hi byte)	MAX DELTA (lo byte)	MAX DELTA (hi byte)
65	4	134	12	43	1
CF_BLUE = 1089		SETVALUE = 3206		MAX DELTA = 299	

Example order=105: Get cycle time from sensor

DATA FRAME PC → Sensor

Byte0 Header	Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header
0x55	<order>	<ARG> (lo byte)	<ARG> (hi byte)	<LEN> (lo byte)	<LEN> (hi byte)	CRC8 (Data)	CRC8 (Header)
85 (dec)	105	0	0	0	0	170	130
		ARG=0		LEN=0			

DATA FRAME Sensor → PC

Byte0 Header	Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header	Byte8 Data	Byte9 Data	Byte10 Data	Byte11 Data
0x55	<order>	<ARG> (lo byte)	<ARG> (hi byte)	<LEN> (lo byte)	<LEN> (hi byte)	CRC8 (Data)	CRC8 (Header)	lo word lo byte	lo word hi byte	hi word lo byte	hi word hi byte
85 (dec)	105	0	0	8	0	206	163	40	28	2	0
		ARG=0		LEN=8				CYCLE COUNT = 138280			

Byte12 Data	Byte13 Data	Byte14 Data	Byte15 Data
lo word lo byte	lo word hi byte	hi word lo byte	hi word hi byte
144	1	0	0
COUNTER TIME = 400			

Cycle Time [Hz] = CYCLE COUNT / (COUNTER TIME * 0,01)

Cycle Time [ms] = (COUNTER TIME * 0,01) / CYCLE COUNT

Example order=190: Write new baud rate to the sensor.

DATA FRAME PC → Sensor

Byte0 Header	Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header
0x55	<order>	<ARG> (lo byte)	<ARG> (hi byte)	<LEN> (lo byte)	<LEN> (hi byte)	CRC8 (Data)	CRC8 (Header)
85 (dec)	190	1	0	0	0	170	14
		ARG=1		LEN=0			

New baud rate is determined by argument.
ARG=0: baud rate = 9600
ARG=1: baud rate = 19200
ARG=2: baud rate = 38400
ARG=3: baud rate = 57600
ARG=4: baud rate = 115200

DATA FRAME Sensor → PC

Byte0 Header	Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header
0x55	<order>	<ARG> (lo byte)	<ARG> (hi byte)	<LEN> (lo byte)	<LEN> (hi byte)	CRC8 (Data)	CRC8 (Header)
85 (dec)	190	0	0	0	0	170	195
		ARG=0		LEN=0			