Manual
Software SPECTRO1-SC Scope V2.0

(PC software for Microsoft® Windows® 7, 8, 10)

for sensor type SPECTRO1-CONLAS-ANA-SC

This manual describes the installation of the PC software for the sensor SPECTRO1-CONLAS-ANA-SC (electronic control unit). As a support for commissioning of the sensor this manual explains the functional elements of the Windows® user interface.

Parameters and measurement values can be exchanged between PC and sensor either through RS232 or Ethernet (using an Ethernet converter). Through the interface all the parameters can be stored in the non-volatile EEPROM of the sensor.
The PC software facilitates the parameterisation, diagnostics, and adjustment of the sensor system (oscilloscope function).

When parameterisation is finished, the sensor continues to operate with the current parameters in STAND-ALONE mode without a PC.

Sensor Instruments GmbH - Schlinding 15 - D-94169 Thurmansbang
Tel.: +49 (0)8544 / 9719-0 - Fax: +49 (0)8544 / 9719-13
E-mail: info@sensorinstruments.de - www.sensorinstruments.de

Contents

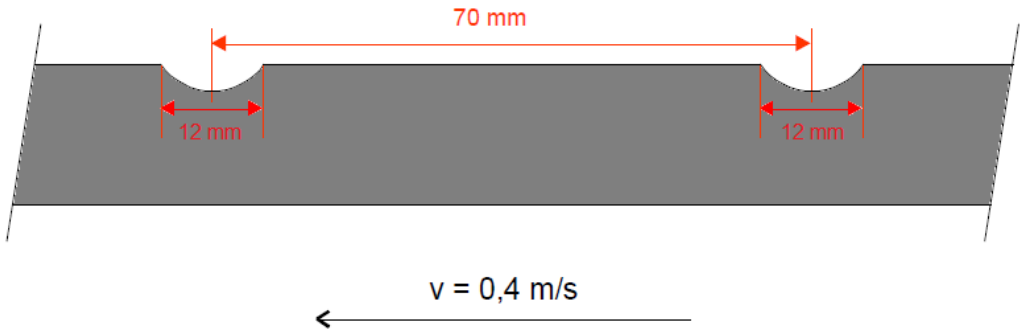
Shortcuts:

SEND	F9
GET	F10
GO	F11
STOP	F12

1 Short description of the sensor's function

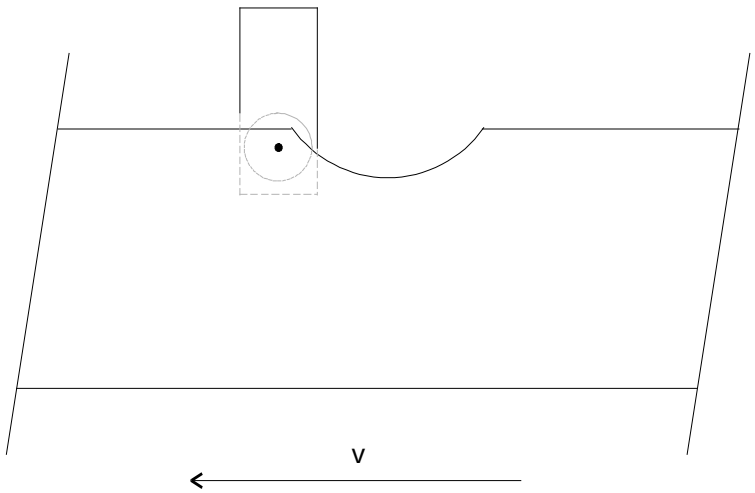
Description:
Semi-circular gaps are punched in endless cigarette paper strips.
The task is to synchronise the position of the gap with the punching stroke.
The punching stroke either may come too soon, too late, or not at all. It also may be that the light barrier sees no gap.
All possible states are output through the outputs OUT0 to OUT3. In addition, an analog output can be activated, which maps the position of the punch stroke within the recess of 0-10V or 4-20mA. The PLC can then react correspondingly and decelerate or accelerate the feed, or even switch off the feed.

Example:



Hardware used:
1. For example an A-LAS-N-F12-d0.3-20/50-C-1m is used to detect the start and end of the gap.
2. Evaluation of the signals (A-LAS-... gap) and of the punching signal (input IN0) is performed by a SPECTRO-1-CONLAS-ANA-SC control unit.
3. Corresponding cables.

Configuration:
The A-LAS-… must be positioned in such a way that the edge of the punching stroke signal that is selected in the software is present at sensor input IN0 exactly at the moment when the laser spot is exactly in the middle of the gap!!!



Evaluation:

When the laser spot of the A-LAS becomes free (gap starts), the electronic evaluation unit starts an internal counter.
The following counter readings are saved at certain moments.
CNT STROKE is the counter reading when the punching stroke occurs.
CNT GAP is the counter reading when the gap is over. It is equal to the length of the gap.
CNT PERIOD is the time from gap to gap, i.e. one complete period.

The **STROKE TOLERANCE** parameter defines a tolerance window for the punching stroke within the gap.

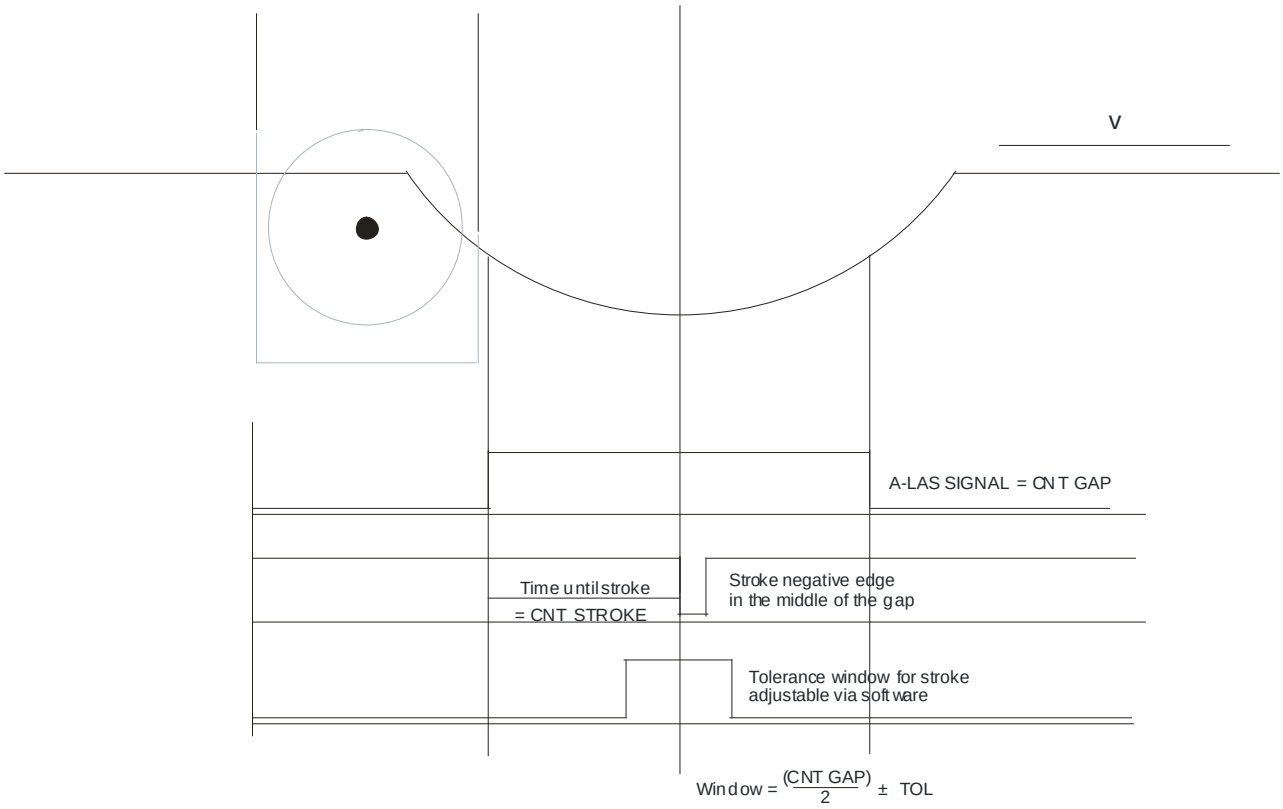
$$Tol = \frac{STROKE\ TOLERANCE * CNT\ GAP}{1000}$$

$$LOWER\ TOL\ LIMIT = \frac{CNT\ GAP}{2} - Tol \qquad UPPER\ TOL\ LIMIT = \frac{CNT\ GAP}{2} + Tol$$

Depending on the occurrence of the punching stroke within a complete period, the outputs **OUT0-OUT3** with **DIGITAL OUTMODE = DIRECT** are switched as follows.

OUT0 is LO if the punching stroke lies out of the tolerance.
OUT0 is HI if the punching stroke lies within the tolerance.
OUT1 is LO if the punching stroke lies below the tolerance.
OUT1 is HI if the punching stroke lies above the tolerance.
OUT2 is LO if the punching stroke lies in the lower half of the period.
OUT2 is HI if the punching stroke lies in the upper half of the period.
OUT3 is HI if the gap is detected, but the punching stroke not, and vice versa.
OUT3 is HI if for 60 seconds no gap and no punching stroke is detected.

The control of the analog output is described under the parameter **ANALOG OUTMODE**.



2 Installation of the SPECTRO1-SC-Scope software

The following requirements must be fulfilled for successful installation of the software:

- Microsoft® Windows® 7, 8, 10
- IBM PC AT or compatible
- VGA graphics
- Microsoft-compatible mouse
- Serial RS232 interface at the PC or USB slot or RJ45 connector
- Cable ***cab-las4/PC*** for the RS232 interface or ***cab-4/USB*** USB converter or ***cab-4/ETH*** Ethernet converter

Please install the software as described below:

1. You can download the software via a provided download link or, if applicable, install it via the provided software DVD. To install the software, start the 'SETUP' program in the 'SOFTWARE' folder.

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".

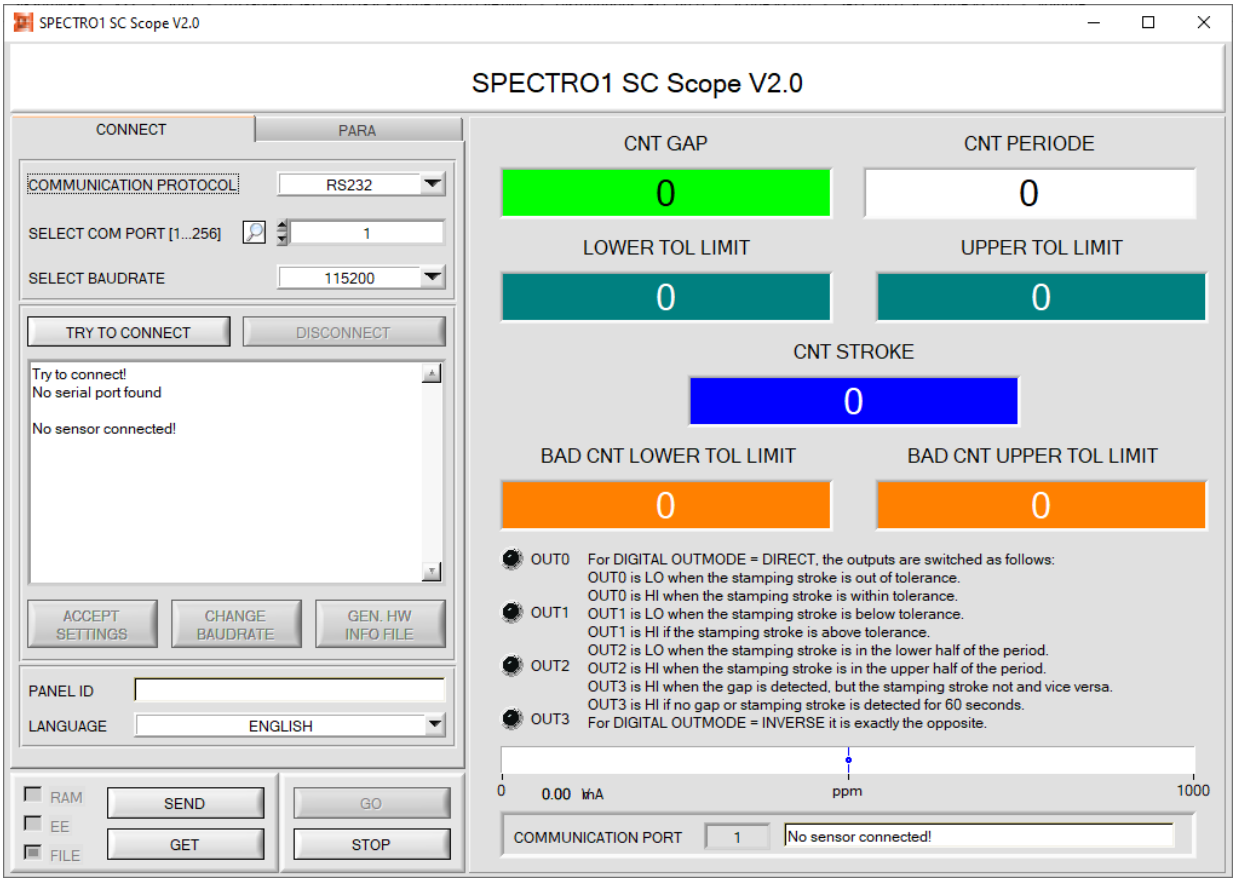
4. After successful installation the software can be started with a left mouse button double-click on the icon.

3 Operation of the SPECTRO1-SC-Scope software

Please read this chapter first before you start to adjust and parameterise the SPECTRO-1 sensor.

When the SPECTRO1-SC-Scope software is started, the following window appears on the Windows interface:

TIP! To avoid problems with the handling of the file path, it is advisable to run the software as administrator. You can either set this in the **Properties** under **Compatibility** or you start the software with a right click and choose **"Run as administrator"**.



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.

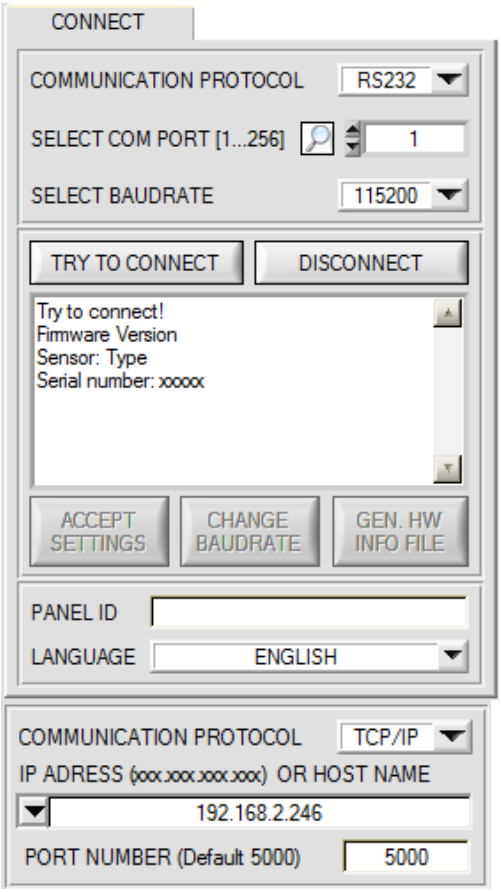
If a connection is not established automatically, e.g. if no sensor is connected, the software can be run in OFFLINE mode. In offline mode it only is possible to exchange parameters with a file on a storage medium, which often is helpful for the purpose of analysing parameter files.

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.

Due to a better overview, parameters that are not required, displays, graphs, etc., are greyed out or invisible depending on the parameterization.

3.1 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.
A click on the magnifier symbol opens a list with all the possible COM ports in the display.

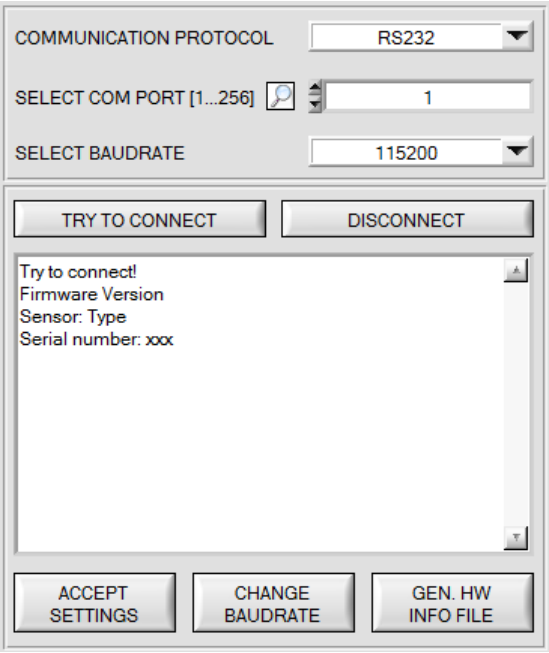
An RS232 to Ethernet adaptor **(cab-4/ETH)** is needed if the sensor should communicate through a local network. With this adaptor a connection to the sensor can be established using the **TCP/IP** protocol.
Parameterisation of the **cab-4/ETH** adaptor (assigning of IP address, baudrate setting, ...) can be done with the **SensorFinder software** that is available free of charge on the internet.

In order to establish a connection via 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 cab-4/ETH is 5000. 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.

Under **PANEL ID** a name can be entered that will be displayed at different points in the program window, and that will be recorded in different files (e.g. Record File) as well. With the input field **LANGUAGE** a language can be set with which the individual controls are displayed on the surface. This also applies to the help function that is actuated with the right mouse button.

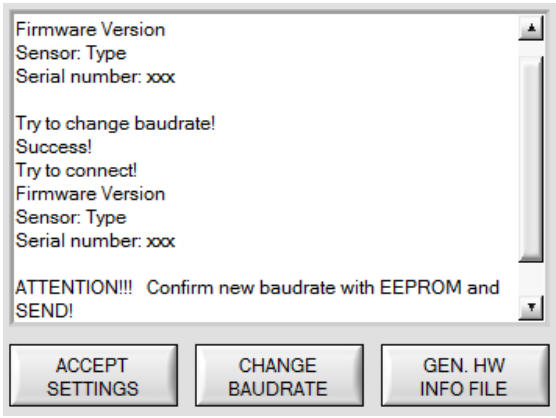
Please note:  **ATTENTION !**	**The stable function of the interface is a basic prerequisite for measured value transfer from the PC to the sensor.** **Due to the limited data transfer rate through the serial RS232 interface only slow changes of the raw signals at the sensor front end can be observed 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.

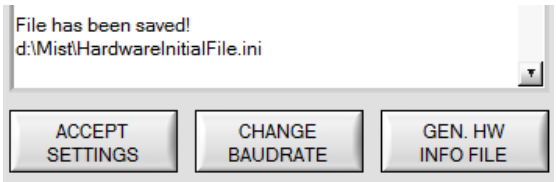
If the baudrate should be changed, a connection must first be established by clicking on **TRY TO CONNECT**.
The **CHANGE BAUDRATE** button will then be active.

Now a new baudrate can be selected under **SELECT BAUDRATE**.
A click on **CHANGE BAUDRATE** sends the new baudrate information to the sensor.



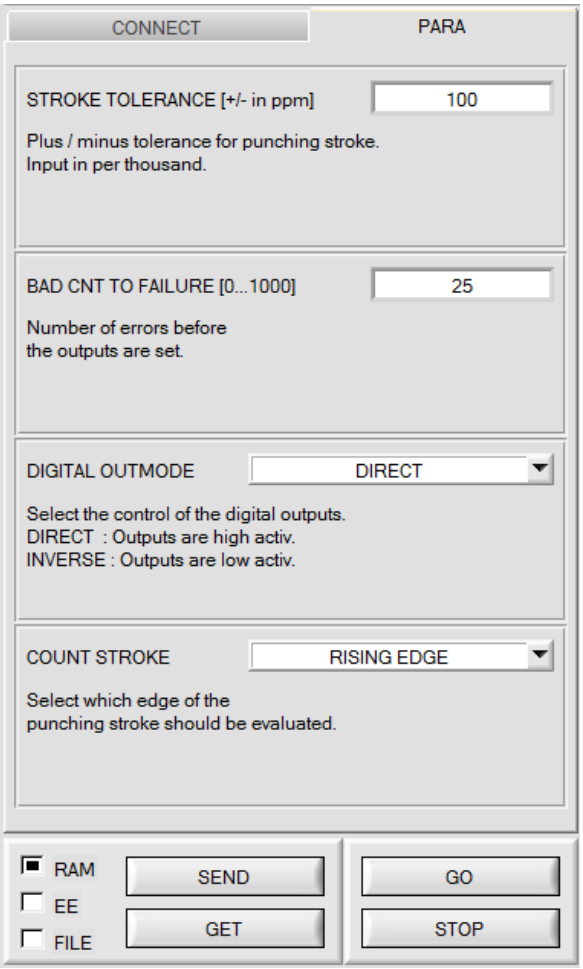
When the new baudrate information has been successfully sent, the sensor operates with the new baudrate. A window will pop up, prompting you to select **EEPROM** and then to press **SEND**. After a hardware reset the new baudrate only will be used when **EEPROM** and **SEND** have been pressed.

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.

3.2 Tab PARA, button SEND, GET, GO, STOP



PARA:
Pressing the **PARA** 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**).

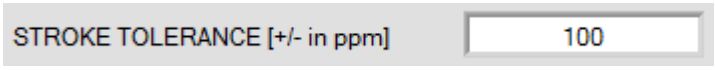
RAM:
The **RAM** is a **volatile** memory in the sensor's micro-controller, i.e. when the power at the sensor is turned off, these parameters will be lost again.
The sensor always operates with the parameters in its RAM.
If the **RAM** option is selected, a click on **SEND** writes the current parameters to the sensor's **RAM** memory, and a click on **GET** reads the parameters from the sensor's **RAM** memory.

EEPROM:
The **EEPROM** is a **non-volatile** memory in the sensor's micro-controller. When the power at the sensor is turned off the parameters in the **EEPROM** will not be lost. When power is turned on again, the parameters are loaded from the **EEPROM** to the **RAM** memory. Figuratively speaking the **EEPROM** thus is a level lower than the **RAM**. Data exchange between **PC** and **EEPROM** automatically is performed through the **RAM** memory, which means that parameters that are written to the **EEPROM** automatically are also written to the **RAM,** and data that are read from the **EEPROM** automatically are also read to the **RAM**.
If the **EEPROM** option is selected, a click on **SEND** writes the current parameters to the sensor's non-volatile **EEPROM** memory, and a click on **GET** reads the parameters from the sensor's **EEPROM**.
The **RAM** memory should always be used for parameterising the sensor. When suitable parameters have been found for the respective application, these parameters must be written to the sensor's **EEPROM** so that after restarting the sensor these parameters can be loaded from the **EEPROM** into the **RAM** memory.

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.
TIP! Once suitable parameters have been found for a certain application, these should always be saved in a file on the PC.

GO [F11]:
A click on this button starts data transfer from the sensor to the PC through the serial RS232 interface.

STOP [F12]:
A click on this button stops data transfer from the sensor to the PC through the serial RS232 interface



STROKE TOLERANCE is a +/- tolerance for the punching stroke within the gap.

The counter reading of **CNT GAP** is equal to 1000 per mill.
The tolerance limits are calculated as follows:

$$Tol = \frac{STROKE\ TOLERANCE * CNT\ GAP}{1000}$$

$$LOWER\ TOL\ LIMIT = \frac{CNT\ GAP}{2} - Tol \qquad UPPER\ TOL\ LIMIT = \frac{CNT\ GAP}{2} + Tol$$

BAD CNT TO FAILURE defines how many identical errors must occur in sequence before the corresponding outputs are activated.

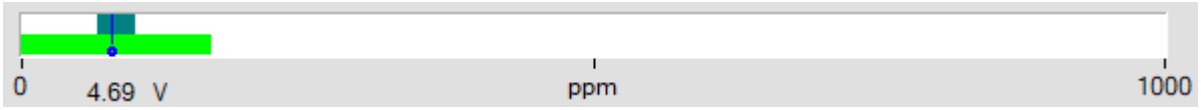
DIGITAL OUTMODE defines whether the outputs are high-active **(DIRECT)** or low-active **(INVERSE)**.

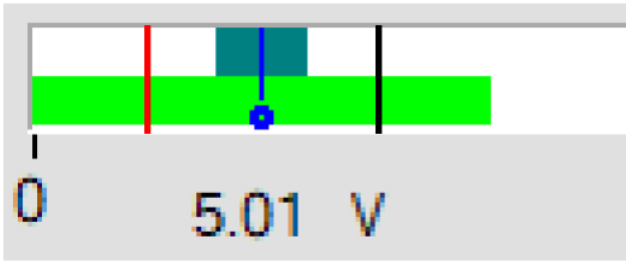
COUNT STROKE determines whether the positive or negative edge of the punching stroke should be evaluated.

ANALOG OUTMODE
In this function field, you may set how the analog signal is to be output.

OFF: No analog signal is output.
U (Voltage): The analog signal is output as a voltage from 0 to 10V.
I (Current): The analog signal is output as a current from 4 to 20mA.

Calculation of the analog signal:





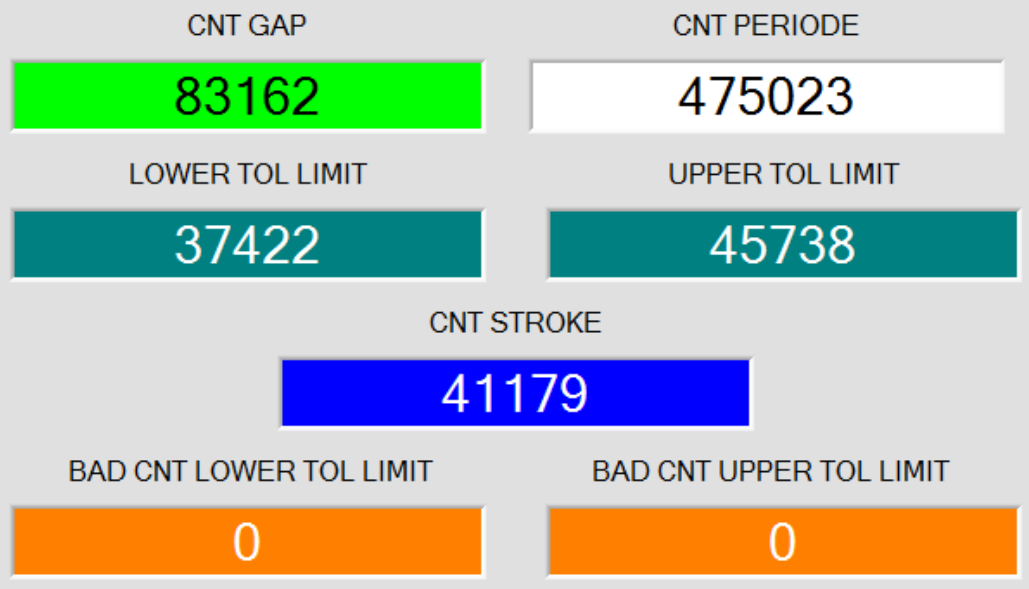
The green range **(CNT GAP)** is displayed from 0V to 10V or 4mA to 20mA.

If the cursor **(punch stroke CNT STROKE)** is on the position of the blue line, 5V or 12mA is output.
If the cursor is positioned on the red line (only symbolically drawn here), 2.5V is output.
If the cursor is positioned on the black line (only symbolically drawn here), 7.5V is output.
If the punch stroke **(CNT STROKE)** is on the far left of the green area, 0V is output.
If it comes from the far right of the green area **(CNT GAP)**, 10V is output.
From the far right in the green area **(CNT GAP)** to 500ppm in the white area **(CNT PERIODE)** 10V is output.
From 500ppm to 1000ppm in the white field **(CNT PERIODE)** 0V is output.
If no punching stroke occurs, 0V is output.
If no cutout occurs, 10V is output.
After 60 seconds without action **(no notch and no punch stroke)**, 10V is also output.

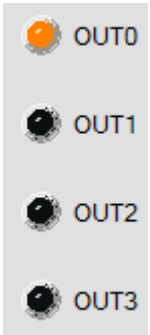
ATTENTION! When 0V or 10V is output, the digital outputs should also be interrogated for safety, because 0V or 10V can be position signals or can also mean the absence of the punching stroke or the recess or a standstill of the machine.

3.3 Graphic display elements

The software provides various display elements and a graphic window for the visualisation of all the data that are important for parameterisation. The individual display elements and the graph are explained in the chapter below.



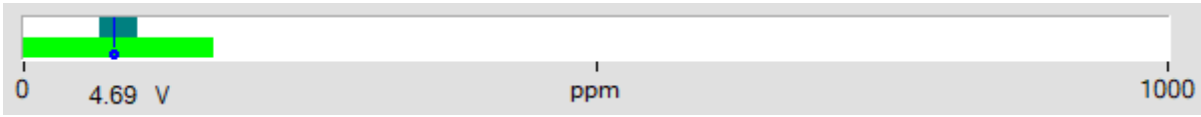
These displays show the counter readings at which an event has occurred.
The counter reading almost exactly is equal to the value in microseconds.
CNT PERIOD: Time from gap to gap.
CNT GAP: Length of the gap.
CNT STROKE: Moment of the punching stroke.
LOWER TOL LIMIT and **UPPER TOL LIMIT** show the tolerance window.
BAD CNT LOWER TOL LIMIT and **BAD CNT UPPER TOL LIMIT** show how many identical errors have occurred in sequence.



OUT0 to OUT13
These LEDs visualise the physical status of outputs OUT0 to OUT1.
When the LED is black, the output value is 0V.
When the LED is orange, the output value is +24V.

Depending on the occurrence of the punching stroke within a complete period, the outputs **OUT0-OUT3** with **DIGITAL OUTMODE = DIRECT** are switched as follows:

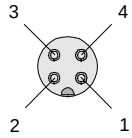
OUT0 is LO if the punching stroke lies out of the tolerance.
OUT0 is HI if the punching stroke lies within the tolerance.
OUT1 is LO if the punching stroke lies below the tolerance.
OUT1 is HI if the punching stroke lies above the tolerance.
OUT2 is LO if the punching stroke lies in the lower half of the period.
OUT2 is HI if the punching stroke lies in the upper half of the period.
OUT3 is HI if the gap is detected, but the punching stroke not, and vice versa.
OUT3 is HI if for 60 seconds no gap and no punching stroke is detected.



In the graphic display window the individual displays also are visualised scaled to per mill.
The white area is equal to **CNT PERIOD.**
The green area is equal to **CNT GAP.**
Cyan represents the tolerance window that is defined by **LOWER TOL LIMIT** and **UPPER TOL LIMIT**.
The blue cursor shows when the punching stroke occurred.
If the analog output is activated, the analog output value is displayed under the green area.

4 Connector assignment SPECTRO-1-CONLAS-ANA-SC

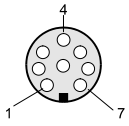
Connection of SPECTRO-1-CONLAS-ANA-SC C to PC:

4-pole M5 fem. connector (type Binder 707) ***SPECTRO-1-CONLAS-ANA-SC/PC-RS232***		

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

Connecting cables to choose from:
cab-las4/PC-…
cab-4/USB-…
cab-4/ETH-…

Connection of SPECTRO-1-CONLAS-ANA-SC to PLC:

8-pole fem. connector (type Binder 712) ***SPECTRO-1-***CONLAS-ANA***-SC/PLC***		

Pin No.:	**Color of wire:** **(cab-las8/SPS)**	**Assignment:**
1	white	0V (GND)
2	brown	+24V (± 10 %)
3	green	IN0 (Digital 0: 0 … 1V, Digital 1: +Ub – 10%)
4	yellow	ANALOG OUT (0…10V or 4…20mA)
5	grey	OUT0 (Digital 0: 0 … 1V, Digital 1: +Ub – 10%)
6	pink	OUT1 (Digital 0: 0 … 1V, Digital 1: +Ub – 10%)
7	blue	OUT2 (Digital 0: 0 … 1V, Digital 1: +Ub – 10%)
8	red	OUT3 (Digital 0: 0 … 1V, Digital 1: +Ub – 10%)

Connecting cable:
cab-las8/SPS-…

5 RS232 communication protocol

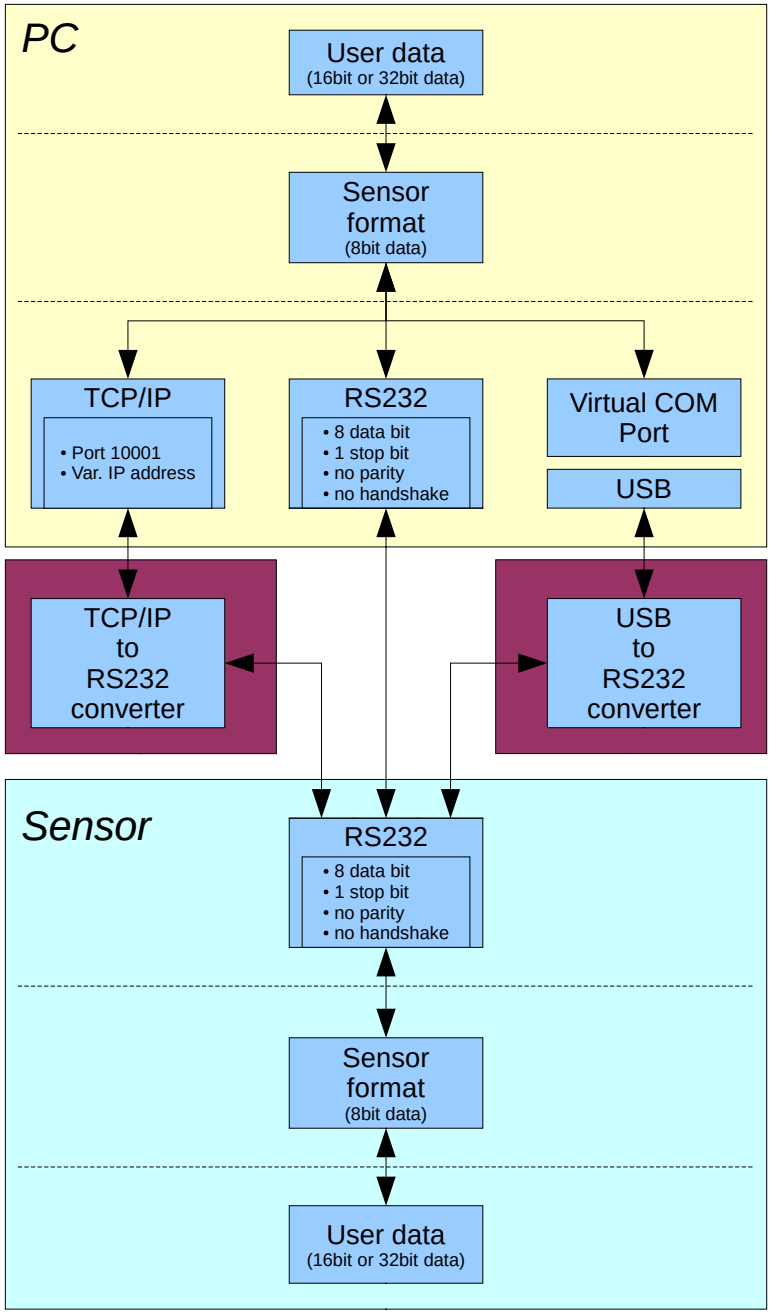
The sensors of the SPECTRO-1-...-SC series operate with the following **parameters** that are sent to the sensor or read from the sensor in the stated sequence.
Info! 1 **byte** = 8bit 1 **word** = 2 **byte** 1 **long** = 2 **word** = 4 **byte**

TABLE PARAMETER			
Parameter		**Type**	**Meaning**
Para1:	STROKE TOL	word	Stroke Tolerance (0 ... 500)
Para2:	BAD CNT TO FAILURE	word	Bad count to failure (0 … 1000)
Para3:	DIGITAL OUTMODE	word	Digital outmode: DIRECT, INVERSE coded to (0,1)
Para4:	COUNT STROKE	word	Count stroke: RISING EDGE, FALLING EDGE coded to (0,1)
Para5:	ANALOG OUTMODE	word	Function of the analog output: OFF, U, I coded to (0,1,2,3)

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

TABLE DATA VALUE			
DATA VALUE		**Type**	**Meaning**
DatVal1:	CNT PERIODE	long	Time from gap to gap
DatVal2:	CNT GAP	long	Length of gap
DatVal3:	CNT STROKE	long	Time when the stamping stroke occurs
DatVal4:	UPPER TOL LIMIT	long	Lower limit of the tolerance window
DatVal5:	LOWER TOL LIMIT	long	Upper limit of the tolerance window
DatVal6:	BAD CNT UPPER TOL LIMIT	long	Number of failures (in sequence to low)
DatVal7:	BAD CNT LOWER TOL LIMIT	word	Number of failures (in sequence to high)
DatVal8:	DIGITAL OUT	word	Status of output LED's (Bit0, Bit1, Bit2, Bit3)
DatVal9:	ANALOG OUT	word	Analogue output value (0…4095)

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.

Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header	Byte8 Header	Byte9 Data	Byte10 Data	…	Byte n+7 Data	Byte n+8 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)

Folgende **Befehle** können zum Sensor abgesetzt werden.

Number	ORDER (header byte no. 2)	Example
0	Sensor answers with order=0 if a communication error occurs. ARG=1: Invalide 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 firmware number from sensor	order=7
8	Read data values from sensor	order=8
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  := AAhex
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.

Example is with 5 parameter (Para1=500, Para2=0; Para3=3200, Para4=3300, Para5=1)
Have a look at the **TABLE PARAMETER** to check out how much parameter you have to send.

DATA FRAME PC → Sensor

Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header	Byte8 Header
0x55	<order>	<ARG> (lo byte)	<ARG> (hi byte)	<LEN> (lo byte)	<LEN> (hi byte)	CRC8 (Data)	CRC8 (Header)
85 (dec)	1	0	0	10	0	130	107
		ARG=0		LEN=10			

Byte9 Data	Byte10 Data	Byte11 Data	Byte12 Data	Byte13 Data	Byte14 Data	Byte15 Data	Byte16 Data	Byte17 Data	Byte18 Data
Para1 (lo byte)	Para1 (hi byte)	Para2 (lo byte)	Para2 (hi byte)	Para3 (lo byte)	Para3 (hi byte)	Para4 (lo byte)	Para4 (hi byte)	Para5 (lo byte)	Para5 (hi byte)
244	1	0	0	128	12	228	12	1	0
Para1=500		Para2=0		Para3=3200		Para4=3300		Para5=1	

DATA FRAME Sensor → PC

Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header	Byte8 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 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.

Example is with 5 parameter (Para1=500, Para2=0; Para3=3200, Para4=3300, Para5=1)
Have a look at the **TABLE PARAMETER** to check out how much parameter you will receive.

DATA FRAME PC → Sensor

Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header	Byte8 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

Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header	Byte8 Header
0x55	<order>	<ARG> (lo byte)	<ARG> (hi byte)	<LEN> (lo byte)	<LEN> (hi byte)	CRC8 (Data)	CRC8 (Header)
85 (dec)	2	0	0	10	0	130	50
		ARG=0		LEN=10			

Byte9 Data	Byte10 Data	Byte11 Data	Byte12 Data	Byte13 Data	Byte14 Data	Byte15 Data	Byte16 Data	Byte17 Data	Byte18 Data
Para1 (lo byte)	Para1 (hi byte)	Para2 (lo byte)	Para2 (hi byte)	Para3 (lo byte)	Para3 (hi byte)	Para4 (lo byte)	Para4 (hi byte)	Para5 (lo byte)	Para5 (hi byte)
244	1	0	0	128	12	228	12	1	0
Para1=500		Para2=0		Para3=3200		Para4=3300		Para5=1	

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

DATA FRAME PC → Sensor

Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header	Byte8 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

Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header	Byte8 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

Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header	Byte8 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

Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header	Byte8 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

Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header	Byte8 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
ARG determines the serial number of the sensor

Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header	Byte8 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

Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header	Byte8 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
ARG determines the firmware number of the sensor

Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header	Byte8 Header	Byte9 Data	Byte10 Data	Byte11 Data	Byte12 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							

Byte13 Data	Byte14 Data	Byte15 Data	Byte16 Data	Byte17 Data	Byte18 Data	Byte19 Data	Byte20 Data	Byte21 Data	Byte22 Data	Byte23 Data	Byte24 Data
ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII
W	A	R	E		S	T	R	I	N	G	

Byte25 Data	Byte26 Data	Byte27 Data	Byte28 Data	Byte29 Data	Byte30 Data	Byte31 Data	Byte32 Data	Byte33 Data	Byte34 Data	Byte35 Data	Byte36 Data
ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII
											R

Byte37 Data	Byte38 Data	Byte39 Data	Byte40 Data	Byte41 Data	Byte42 Data	Byte43 Data	Byte44 Data	Byte45 Data	Byte46 Data	Byte47 Data	Byte48 Data
ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII
T	:	K	W	x	x	/	x	x			

Byte49 Data	Byte50 Data	Byte51 Data	Byte52 Data	Byte53 Data	Byte54 Data	Byte55 Data	Byte56 Data	Byte57 Data	Byte58 Data	Byte59 Data	Byte60 Data
ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII

Byte61 Data	Byte62 Data	Byte63 Data	Byte64 Data	Byte65 Data	Byte66 Data	Byte67 Data	Byte68 Data	Byte69 Data	Byte70 Data	Byte71 Data	Byte72 Data
ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII

Byte73 Data	Byte74 Data	Byte75 Data	Byte76 Data	Byte77 Data	Byte78 Data	Byte79 Data	Byte80 Data	Byte81 Data	Byte82 Data
ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII

Example order=8: Read data values from sensor.

Example is with 5 data values (DataVal1=2000, DataVal2=4; DataVal3=3000, DataVal4=3500, DataVal5=18)
Have a look at the **TABLE DATA VALUE** to check out how much data values you will receive.

DATA FRAME PC → Sensor

Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header	Byte8 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

Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header	Byte8 Header
0x55	<order>	<ARG> (lo byte)	<ARG> (hi byte)	<LEN> (lo byte)	<LEN> (hi byte)	CRC8 (Data)	CRC8 (Header)
85 (dec)	8	0	0	10	0	28	243
		ARG=0		LEN=10			

Byte9 Data	Byte10 Data	Byte11 Data	Byte12 Data	Byte13 Data	Byte14 Data	Byte15 Data	Byte16 Data	Byte17 Data	Byte18 Data
DataVal1 (lo byte)	DataVal1 (hi byte)	DataVal2 (lo byte)	DataVal2 (hi byte)	DataVal3 (lo byte)	DataVal3 (hi byte)	DataVal4 (lo byte)	DataVal4 (hi byte)	DataVal5 (lo byte)	DataVal5 (hi byte)
208	7	4	0	184	11	172	13	18	0
DatVal1 = 2000		DatVal2 = 4		DatVal3 = 3000		DatVal4 = 3500		DatVal5 = 18	

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

DATA FRAME PC → Sensor

Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header	Byte8 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

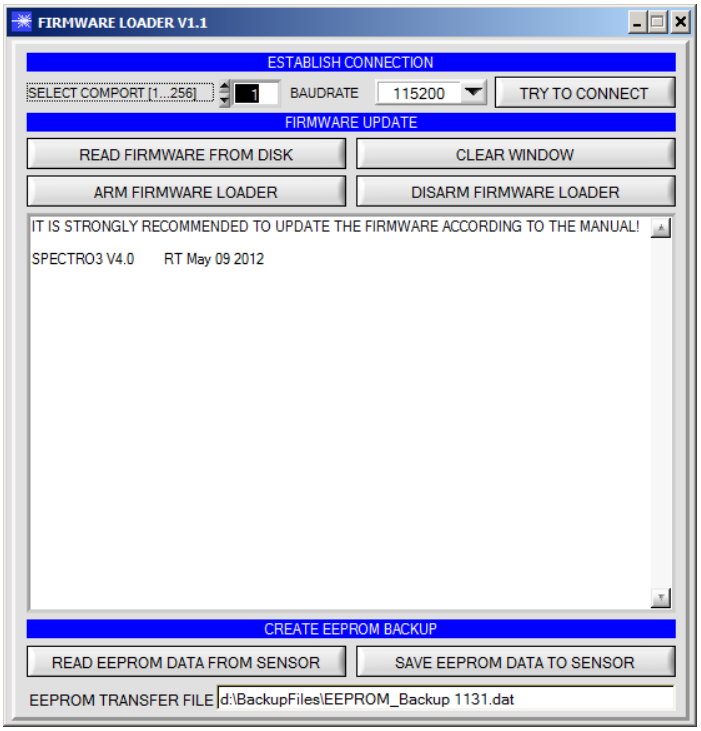
Byte1 Header	Byte2 Header	Byte3 Header	Byte4 Header	Byte5 Header	Byte6 Header	Byte7 Header	Byte8 Header
0x55	<order>	<ARG> (lo byte)	<ARG> (hi byte)	<LEN> (lo byte)	<LEN> (hi byte)	CRC8 (Data)	CRC8 (Header)
85 (dec)	190	0	0	0	0	170	195
		ARG=0		LEN=0			

Please read this chapter before you start!

In this example a software update is performed from SPECTRO3 V4.0 to SPECTRO3 V4.1.

Step 1:

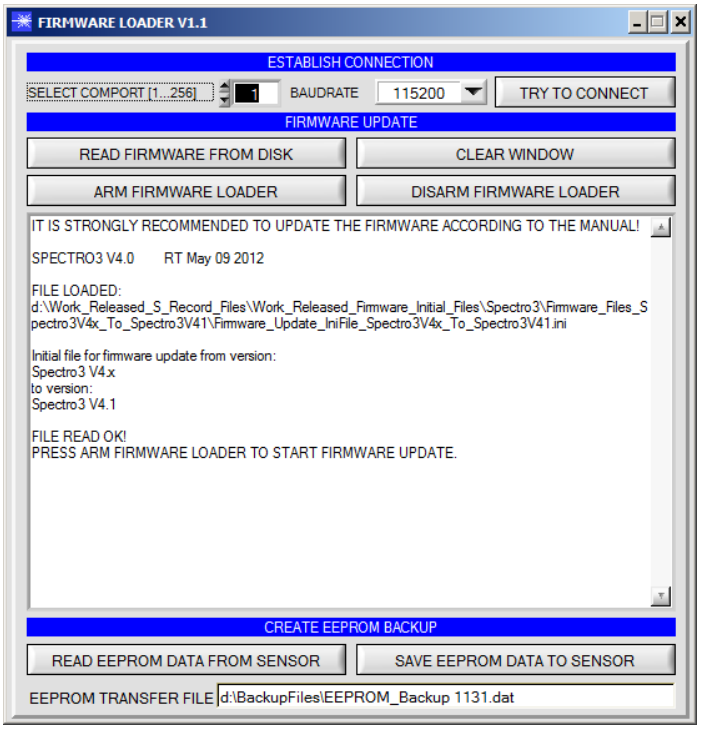
When the Firmware Loader software is started, this window opens on the Windows® user interface.



Immediately after starting, the software attempts to establish a connection to the connected sensor. If the sensor should not be connected at **COM PORT 1,** please select the corresponding **COM PORT.**
Please make sure that the correct **BAUDRATE** is selected.
Now try to establish a connection by clicking on **TRY TO CONNECT**. When the connection has been established, the sensor sends back information about the current firmware.

Step 2:

Press the **READ FIRMWARE FROM DISK** button and load the **xxx.ini** file.



The uploaded initialization file will be displayed in the status window.
As described above, a plausibility check of the initialisation file will be performed first.
If the file is OK, the following message will be displayed:
File read OK!
Press **ARM FIRMWARE LOADER** to start the firmware update.

Please read the comments that are shown in the display window. These comments allow you to make sure that you have loaded the correct initialisation file.

Step 3:

Now click on the **ARM FIRMWARE LOADER** button. The program now attempts to send a software command that interrupts the normal program run and jumps to the start address of the boot sector.
If this is successful, the sensor displays a prompt for loading the S-Record file to the sensor.

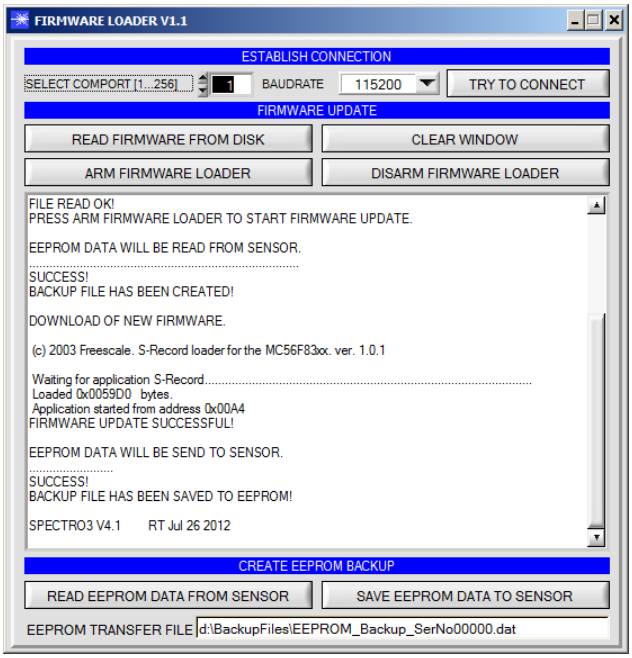
When you press the **ARM FIRMWARE LOADER** button the firmware update will be performed automatically.

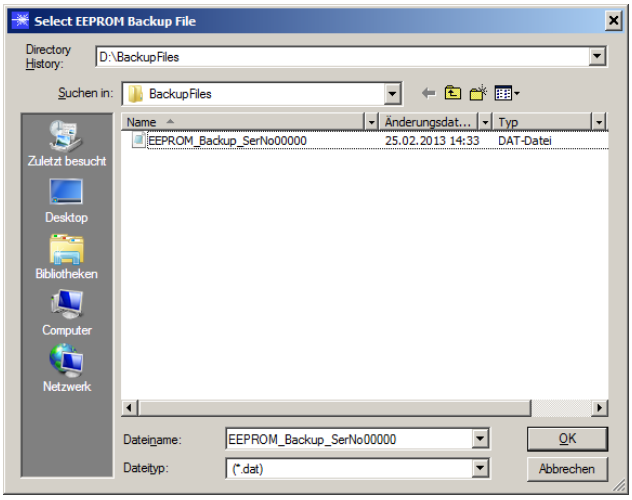
In the course of the update process you will only be prompted to enter a name for the EEPROM backup file. If the firmware update should run perfectly until the EEPRM data are read out, but should then go wrong for any reason whatsoever, the EEPROM backup file can always be written back with **SAVE EEPROM DATA TO SENSOR**.

The file name for your **EEPROM backup file** should be chosen such that the names for several sensors cannot be mixed up. Using a file name that contains the sensor serial number might be advisable. Saving this file for future updates also might be a good idea.

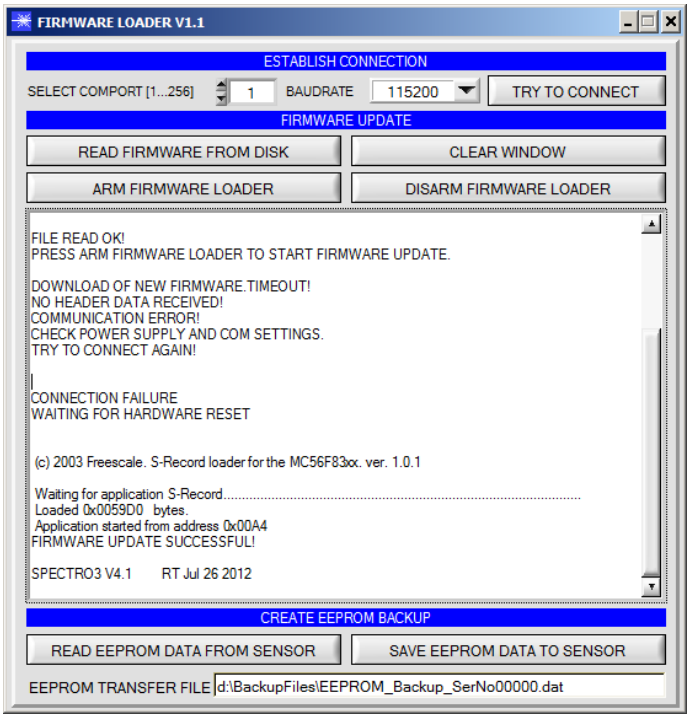
After a successful update the sensor displays the status line of the new firmware.

The complete update process may take up to 1 minute.





If, contrary to expectations, there should be any trouble with the update of the program memory, it will still be possible to perform an update, even though it may look like the sensor was "killed".
Please make sure that you have selected the correct **COM PORT** and the correct **BAUDRATE**.
You will not get any connection when you click on **TRY TO CONNECT**.
Load the corresponding **xxx.ini** file from the hard disk.
Then click on the **ARM FIRMWARE LOADER** button.
The program will try to send the software command for the update. This will not work, however, and you will get a **CONNECTION FAILURE** message.
However, the Firmware Loader software now is "armed" for 30 seconds.
If you perform a hardware reset within these 30 seconds, the firmware update will be performed.

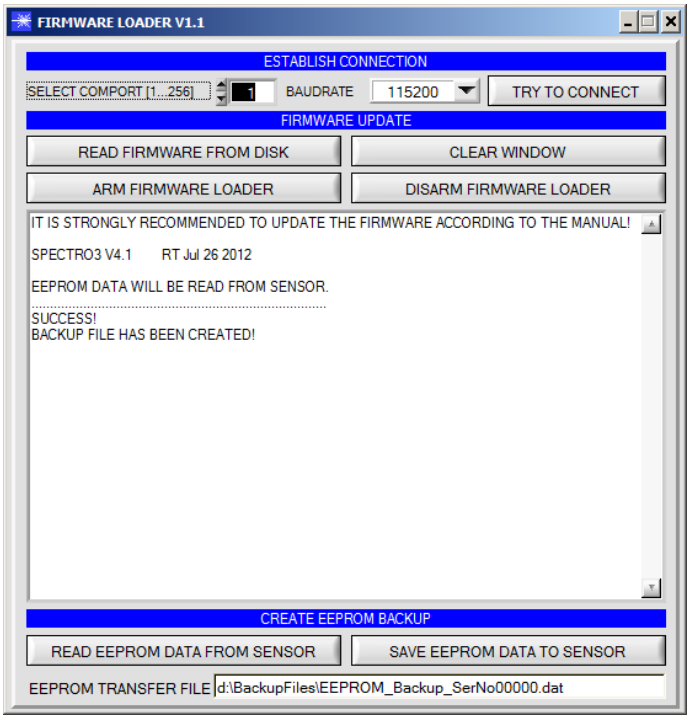


After a successful update the sensor displays the status line of the new firmware.

The complete update process may take up to 1 minute.

INFO! In case that the sensor was “killed”, the sensor will work with a BAUDRATE of 115200.

You may at any time create an EEPROM backup file for archiving it on your hard disk.
To do this, click on **READ EEPROM DATA FROM SENSOR.** You will be prompted to chose an initialization file in case that there has not yet been loaded any. Afterwards you will be asked to enter a file name. The selected name will be shown in the **EEPROM TRANSFER FILE** display.
The file name for your **EEPROM backup file** should be chosen such that the names for several sensors cannot be mixed up. Using a file name that contains the sensor serial number might be advisable.



The Firmware Loader then reads all the EEPROM data from the data memory and saves these data in the selected file.
Upon successful completion the following message will be displayed:
Success!
Backup File has been created!

If something should go wrong in a firmware update, any you have created the **backup file**, the saved **EEPROM backup file** can at any time be uploaded to the sensor again with **SAVE EEPROM DATA TO SENSOR.**

CLEAR WINDOW resets the display window.

If you should not get any response for a longer time, or if messages should be displayed in the status line, **DISARM FIRMWARE LOADER** can be used to cancel the firmware update process.
However, you should always wait for approx. 1 minute before you press this button.