# Bedienungsanleitung
# Software SPECTRO1-SC Scope V1.0

(PC-Software für Microsoft® Windows® 7, Vista, XP, Me, 2000)

## für Sensoren der SPECTRO1-SC Serie

Die vorliegende Bedienungsanleitung dient zur Installation und Inbetriebnahme der PC-Software für Sensoren der SPECTRO1-SC Serie. Zur Unterstützung der Inbetriebnahme der Sensoren werden in dieser Bedienungsanleitung die einzelnen Funktionselemente der graphischen Windows® Benutzeroberfläche erklärt.

Es können wahlweise über RS232 oder Ethernet (mit Hilfe eines Ethernet-Adapters) Parameter und Messwerte zwischen PC und Sensor ausgetauscht werden. Sämtliche Parameter können über die Schnittstelle im nichtflüchtigen EEPROM des Sensors abgelegt werden.
Die PC-Software erleichtert die Parametrisierung, die Diagnose und das Einjustieren des Sensorsystems (Oszilloskop-Funktion).

Nach erfolgter Parametrisierung arbeitet der Sensor im STAND-ALONE Betrieb ohne PC weiter.

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 Inhalt

Seite

1 Kurzbeschreibung zur Funktion des Sensors ..... 3

2 Installation der SPECTRO1-SC-Scope Software ..... 5

3 Bedienung der SPECTRO1-SC-Scope Software ..... 6

3.1 Registerkarte (Reiter oder Tab) CONNECT (Verbindungsaufbau) ..... 7

3.2 Registerkarte PARA, Taste SEND, GET, GO, STOP (Parametrierung und Datenaustausch) ..... 9

3.3 Graphische Anzeigeelemente ..... 11

4 Anschlussbelegung von SPECTRO1-SC Sensoren ..... 12

5 RS232 Schnittstellenprotokoll ..... 13

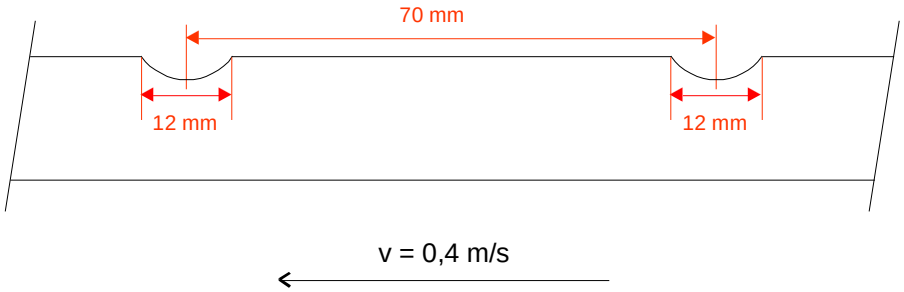
A Firmwareupdate über Software Firmware Loader ..... 22

**Shortcuts:**

| | |
|---|---|
| SEND | F9 |
| GET | F10 |
| GO | F11 |
| STOP | F12 |

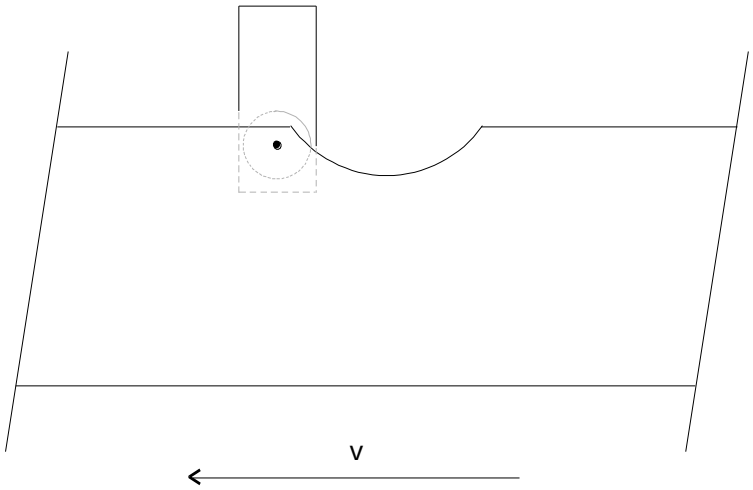
# 1 Kurzbeschreibung zur Funktion des Sensors

**Beschreibung:**
In Endlosstreifen für Zigarettenpapier (Tabakpapier) werden halbkreisförmige Aussparungen eingestanzt.
Die Aufgabe liegt darin, die Position der Aussparung mit dem Stanzhub zu synchronisieren.
Der Stanzhub kann entweder zu früh, zu spät oder gar nicht kommen. Genauso kann es sein, dass die Lichtschranke keine Aussparung sieht. Diese Fehlerzustände werden über die Ausgänge OUT0 bis OUT3 ausgegeben. Die SPS kann nun entsprechend reagieren und den Vorschub zurücknehmen oder beschleunigen oder evtl. sogar abschalten.




**Verwendete Hardware:**
1. Um den Start und das Ende der Aussparung zu detektieren wird eine A-LAS-F12-d0.3-20/50 verwendet.
2. Die Auswertung der Signale (A-LAS ... Aussparung) und des Stanzsignals (Eingang IN0) erfolgt über eine Kontrollelektronik SPECTRO1-SC.
3. Entsprechende Kabel.

**Aufbau:**
Die A-LAS muss so positioniert werden, dass die über die Software ausgewählte Flanke des Stanzhubsignals genau zu dem Zeitpunkt am Sensoreingang IN0 anliegt, wenn sich der Laserspot genau in der Mitte der Aussparung befindet!!!



**Auswertung:**

Wird der Laserspot der A-LAS frei (Aussparung beginnt), startet die Auswerteelektronik einen internen Zähler. Zu bestimmten Zeitpunkten werden folgende Zählerstände gespeichert.
**CNT STROKE** ist der Zählerstand, wann der Stanzhub auftritt.
**CNT GAP** ist der Zählerstand, wann die Aussparung vorbei ist. Er entspricht der Länge der Aussparung.
**CNT PERIODE** ist die Zeit von Aussparung zu Aussparung, also eine ganze Periode.

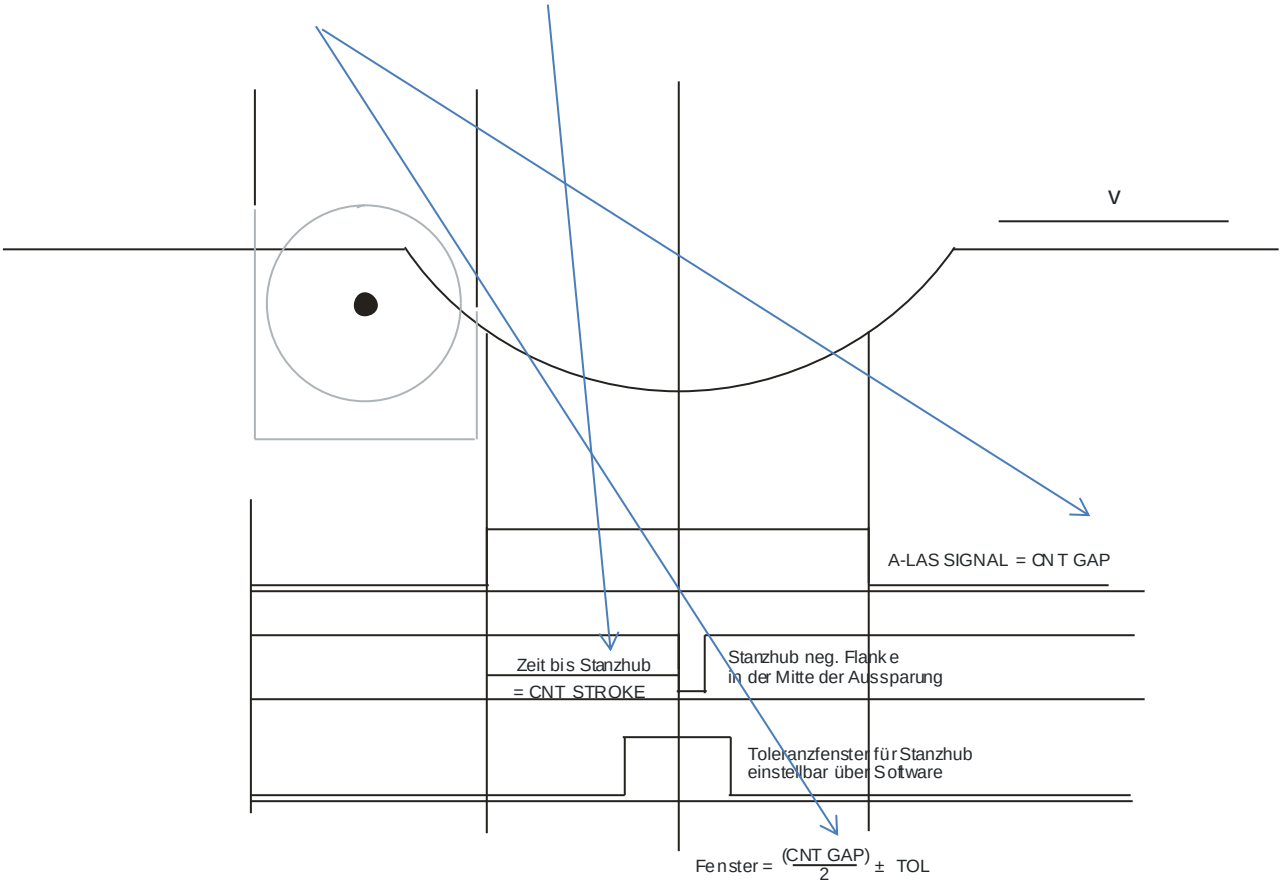
Mit dem Parameter **STROKE TOLERANCE** legt man ein Toleranzfenster für den Stanzhub innerhalb der Aussparung fest.

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

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

Abhängig vom Auftreten des Stanzhubes innerhalb einer ganzen Periode werden die Ausgänge **OU0-OUT3** bei **DIGITAL OUTMODE = DIRECT** wie folgt geschaltet.

OUT0 ist LO, wenn der Stanzhub außerhalb der Toleranz liegt.
OUT0 ist HI, wenn der Stanzhub innerhalb der Toleranz liegt.
OUT1 ist LO, wenn der Stanzhub unterhalb der Toleranz liegt.
OUT1 ist HI, wenn der Stanzhub oberhalb der Toleranz liegt.
OUT2 ist LO, wenn der Stanzhub in der unteren Periodenhälfte liegt.
OUT2 ist HI, wenn der Stanzhub in der oberen Periodenhälfte liegt.
OUT3 ist HI, wenn die Aussparung erkannt wird, der Stanzhub aber nicht und umgekehrt.
OUT3 ist HI, wenn 60 Sekunden weder eine Aussparung noch ein Stanzhub erkannt wurde.



# 2 Installation der SPECTRO1-SC-Scope Software

Für eine erfolgreiche Installation der SPECTRO1-SC-Scope Software müssen folgende Voraussetzungen erfüllt sein:

- Microsoft® Windows® XP, VISTA, 7
- IBM PC AT oder kompatibler
- VGA-Grafik
- Microsoft®-kompatible Maus
- CD-ROM-Laufwerk
- Serielle RS232-Schnittstelle am PC oder USB Slot
- Kabel **cab-las4/PC** für die RS232-Schnittstelle oder **cab-4/USB** für USB Slot

Die SPECTRO1-SC-Scope Software kann nur unter Windows installiert werden. Deshalb müssen Sie zunächst Windows starten, falls es noch nicht aktiv ist.

Installieren Sie nun die Software wie im Folgenden beschrieben:

1. Sie können die Software direkt von der Installations-CD-ROM installieren. Auf der CD-ROM befindet sich der Ordner SOFTWARE. In diesem Ordner ist eine SETUP Anwendung. Zum Installieren der Software müssen Sie diese Setup-Anwendung starten.

2. Das Installationsprogramm meldet sich mit einem Dialogfeld und schlägt vor, die Software im Verzeichnis C:\"DATEINAME" auf der Festplatte einzurichten.
   Akzeptieren Sie den Vorschlag mit **OK** oder [**ENTER**] oder ändern Sie die Pfad-Vorgaben nach Ihren Wünschen.

3. Während der Installation wird eine neue Programm-Gruppe für die Software im Windows Programm-Manager erzeugt. Außerdem wird in der erzeugten Programmgruppe ein Icon für den Start der Software automatisch generiert. Falls die Installation erfolgreich durchgeführt werden konnte, meldet sich das Installationsprogramm mit einer Dialogbox ''Setup OK''.

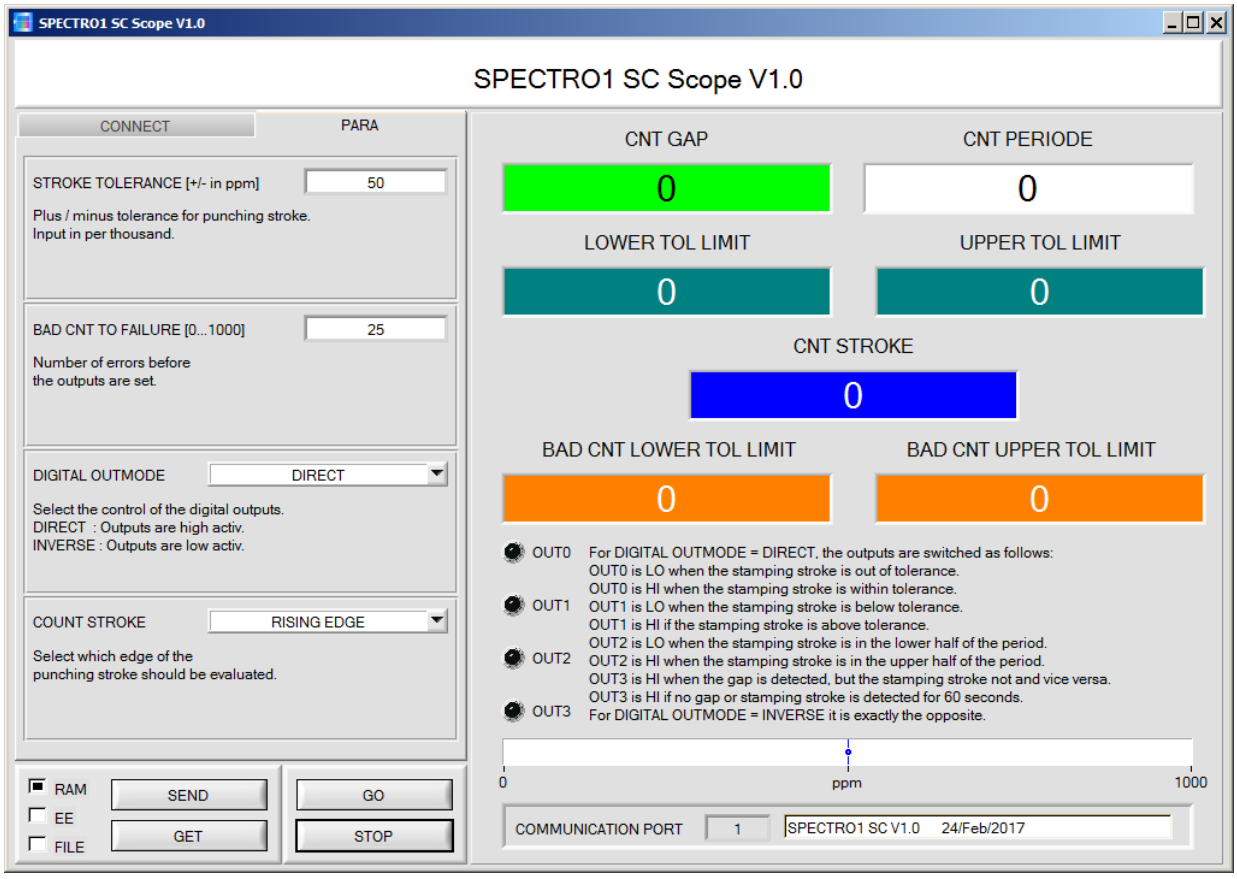
4. Nach erfolgreicher Installation kann die Software durch Doppelklick auf das Icon mit der linken Maustaste gestartet werden.

Windows™ ist ein Warenzeichen der Microsoft Corp.
VGA™ ist ein Warenzeichen der International Business Machines Corp.

# 3 Bedienung der SPECTRO1-SC-Scope Software

**Bitte lesen Sie diesen Abschnitt zuerst durch, bevor Sie die Einjustierung und Parametrisierung des Sensorsystems vornehmen.**

Nach dem Aufruf der SPECTRO1-SC-Scope Software erscheint folgendes Fenster auf der Windows Oberfläche:



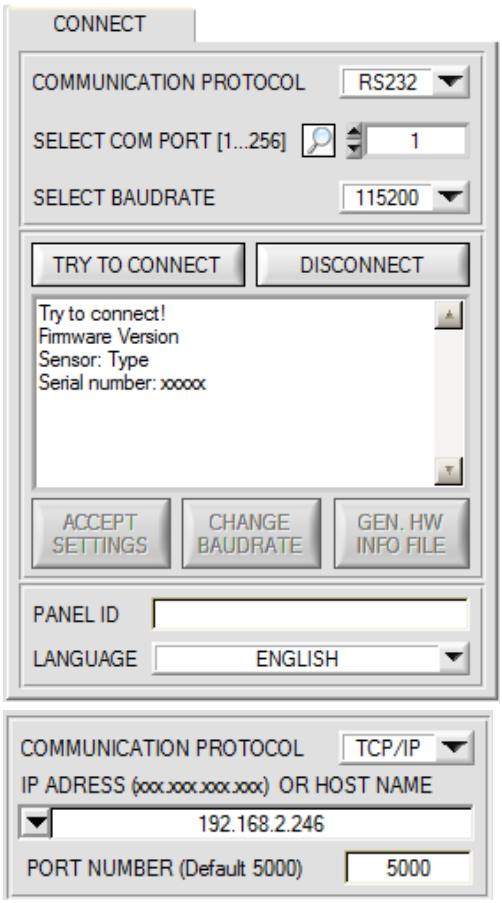

Das Fenster wird in seiner Größe und Position wieder dort platziert, wo es sich beim letzten Verlassen der Software befand. Durch einen Doppelklick mit der rechten Maustaste z.B. unterhalb des Minimierungssymbols wird das Fenster in seiner Originalgröße mittig zentriert.

Kommt es nicht automatisch zu einem Verbindungsaufbau, z.B. wenn kein Sensor angeschlossen ist, dann kann die Software im OFFLINE Modus betrieben werden. Im Offline Modus ist lediglich ein Parameteraustausch mit einer Datei auf einem Speichermedium möglich. Dies ist für Analysezwecke von Parameterfiles oft hilfreich.

Ist ein Sensor angeschlossen und es kommt trotzdem zu keinem Verbindungsaufbau, dann stimmen entweder die SCOPE Version (Programm auf PC) und die Firmware Version (Programm im Sensor) nicht überein oder man muss die Schnittstelle zum Sensor richtig konfigurieren.
Sollte das Problem eine unterschiedliche Scope und Firmware Version sein, dann muss man sich die zur Firmware passende Scope Version vom Lieferanten besorgen.
Das Konfigurieren der Schnittstelle wird in der Registerkarte CONNECT erklärt.

**Eine Kurz-Hilfe wird durch Drücken der rechten Maustaste auf ein einzelnes Element angezeigt.**

## 3.1 Registerkarte CONNECT




**CONNECT:**
Durch Drücken von **CONNECT** öffnet sich eine Ansicht, in der man die Schnittstelle wählen und konfigurieren kann.

In dem Funktionsfeld **COMMUNICATION PROTOCOL** kann entweder ein **RS232** oder ein **TCP/IP** Protokoll ausgewählt werden.
Wählt man **RS232**, kann man mit **SELECT COM PORT** einen Port von 1 bis 256 auswählen, je nachdem an welchem der Sensor angeschlossen ist. Der Sensor arbeitet mit einer eingestellten Baudrate, die über **CHANGE BAUDRATE** verändert werden kann (siehe unten). Sowohl der Sensor als auch die Benutzeroberfläche müssen mit der gleichen Baudrate arbeiten.
Über **SELECT BAUDRATE** stellt man auf der Benutzeroberfläche die Baudrate ein. Sollte die Software nach dem Starten nicht automatisch eine Verbindung aufbauen, kann mit **SELECT BAUDRATE** die richtige Baudrate gefunden werden.
Wenn man mit einem Adapter arbeitet, kann man die **COM PORT** Nummer über den Hardwaremanager in der Systemsteuerung ermitteln.
Durch Drücken auf die Lupe werden alle möglichen COM Ports im Display aufgelistet.

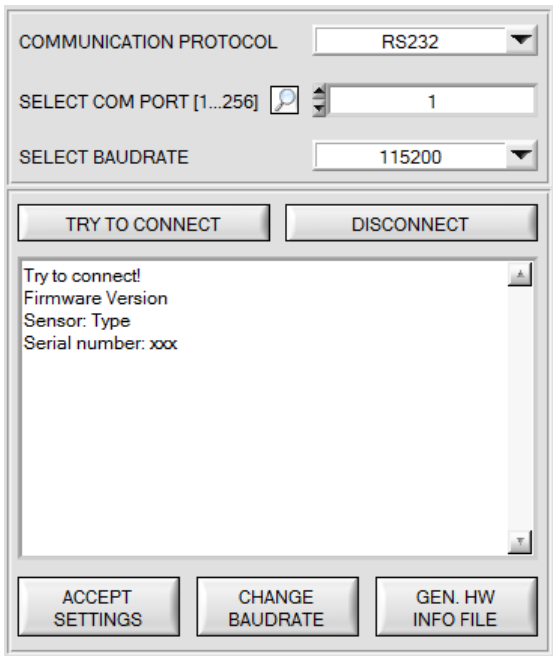
Zur Kommunikation des Sensors über ein lokales Netzwerk wird ein RS232 zu Ethernet Adapter benötigt **(cab-4/ETH)**. Dieser ermöglicht es eine Verbindung zum Sensor über das **TCP/IP** Protokoll herzustellen.
Um die **cab-4/ETH** Adapter zu parametrisieren (Vergabe von IP-Adresse, Einstellung der Baudrate, …), braucht man die im Internet kostenlos bereitgestellte **Software SensorFinder**.

Um eine Verbindung zum Adapter herzustellen, muss dessen IP-Adresse oder HOST Name in das Eingabefeld **IP ADRESS (xxx.xxx.xxx.xxx) OR HOST NAME** eingetragen werden.
Im DROP DOWN Menü (Pfeil nach unten) sind die letzten 10 verwendeten IP Adressen aufgelistet und können durch Anklicken direkt übernommen werden. Die DROP DOWN Liste bleibt auch nach Beenden der Software erhalten.
Die **PORT NUMBER** für das cab-4/ETH ist auf 5000 festgelegt und muss belassen werden.
Nach Drücken von **TRY TO CONNECT** versucht die Software eine Verbindung mit den eingestellten Parametern aufzubauen. Der Status der Kommunikation wird im Anzeigedisplay angezeigt. Meldet sich der Sensor mit seiner FIRMWARE ID, kann man mit **ACCEPT SETTINGS** die eingestellte Verbindungsart beibehalten. Die Software schaltet automatisch auf den Registerkarte **PARA** um. Erhält man ein **TIMEOUT**, konnte die Software keine Verbindung zum Sensor herstellen. In diesem Fall sollte zunächst geprüft werden, ob das Schnittstellenkabel richtig angebracht wurde, ob der Sensor an Spannung liegt und ob die eingestellten Parameter richtig gewählt wurden.
Wurde eine Verbindung mit **ACCEPT SETTINGS** bestätigt, dann startet die Software beim nächsten Aufruf automatisch mit dieser Einstellung.

Mit **DISCONNECT** trennt man die Verbindung vom Sensor zum PC. Die Software schaltet in den OFFLINE Modus in dem nur ein Parameteraustausch mit einer Datei auf einem Speichermedium möglich ist.

Unter **PANEL ID** kann man eine Bezeichnung eingeben, die an verschiedenen Stellen im Programmfenster angezeigt wird, und in verschiedene Files (z.B. Recordfile) mit abgespeichert wird.
Mit dem Eingabefeld **LANGUAGE** kann man eine Sprache einstellen, mit der die einzelnen Controls auf der Oberfläche dargestellt werden. Dies gilt auch für die Hilfe, die mit der rechten Maustaste aufgerufen wird.

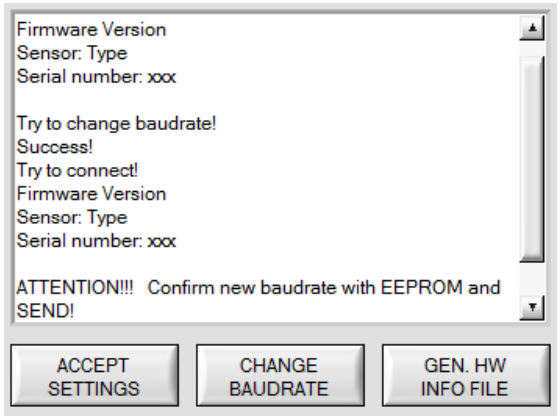
| | |
|---|---|
| **Beachte:** | **Grundvoraussetzung für die Messwertübertragung vom PC zum Sensor ist die stabile Funktion der Schnittstelle.** |
| **Achtung !** | **Aufgrund der begrenzten Datenübertragungsrate über die serielle RS232-Schnittstelle können nur langsame Veränderungen der Rohsignale am Sensor-Frontend im graphischen Ausgabefenster des PC mitverfolgt werden.**<br>**Zur Einhaltung der maximalen Schaltfrequenz am Sensor muss zudem der Datenaustausch mit dem PC beendet werden (STOP-Taste drücken).** |




Die Baudrate zur Datenübertragung über die RS232 Schnittstelle kann mit **SELECT BAUDRATE** und **CHANGE BAUDRATE** eingestellt werden.

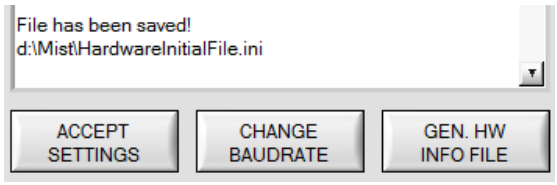
Zum Ändern muss zuerst über **TRY TO CONNECT** eine Verbindung aufgebaut werden.
Erst jetzt ist der Button **CHANGE BAUDRATE** aktiv.

Unter **SELECT BAUDRATE** kann jetzt eine neue Baudrate ausgewählt werden.
Durch Drücken von **CHANGE BAUDRATE** wird die neue Baudrate zum Sensor übertragen.



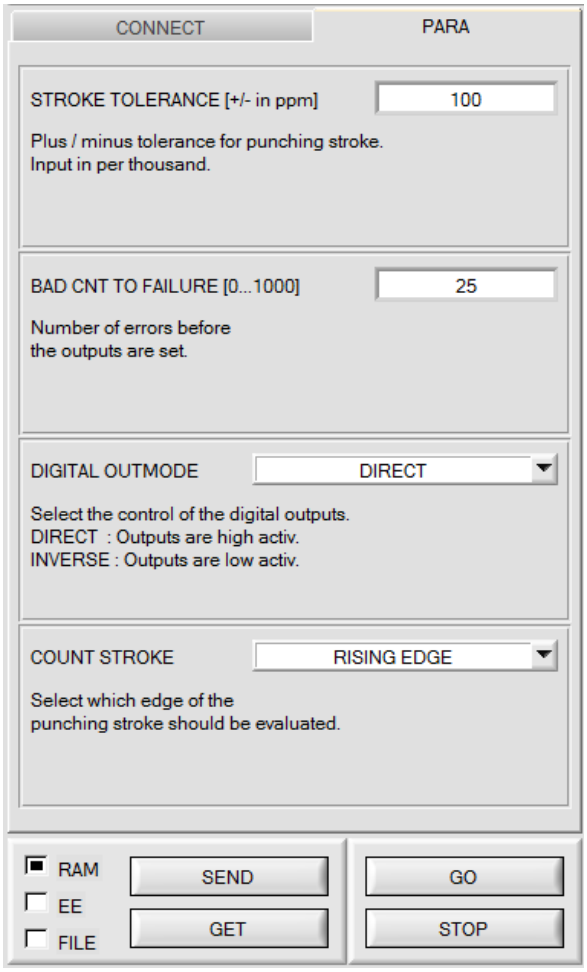

Nachdem die neue Baudrate erfolgreich übertragen worden ist arbeitet der Sensor mit der neuen Baudrate. Außerdem erscheint im Anzeigefenster eine Aufforderung **EEPROM** zu selektieren und anschließend **SEND** zu drücken. Erst nach Drücken von **EEPROM** und **SEND** wird bei einem Hardware-Reset mit der neuen Baudrate gestartet.

Durch Drücken von **ACCEPT SETTINGS** werden die aktuellen Schnittstellen-Einstellungen gespeichert und nach einem Neustart der Software automatisch eingestellt.




Über den Button **GEN. HW INFO FILE** wird ein File erzeugt, in dem alle wichtigen Sensordaten verschlüsselt hinterlegt werden.
Dieses File kann zu Diagnosezwecke an den Hersteller gesendet werden.

## 3.2 Registerkarte PARA, Taste SEND, GET, GO, STOP




**PARA:**
Durch Drücken von **PARA** öffnet sich eine Ansicht, in der man die Sensorparameter einstellen kann.

**Beachte: Eine Änderung der Funktionsgruppen Parameter wird erst nach Betätigung der SEND-Taste im MEM-Funktionsfeld am Sensor wirksam!**

**SEND [F9]:**
Durch Anklicken der Taste **SEND** (bzw. per Shortcut Keytaste F9) werden alle aktuell eingestellten Parameter zwischen PC und dem Sensor übertragen. Das Ziel der jeweiligen Parameterübertragung wird durch den selektierten Auswahlknopf (**RAM**, **EEPROM** oder **FILE**) festgelegt.

**GET [F10]:**
Durch Anklicken der Taste **GET** (bzw. per Shortcut Keytaste F10) können die aktuellen Einstellwerte vom Sensor abgefragt werden. Die Quelle des Datenaustausches wird über den selektierten Auswahlknopf (**RAM**, **EEPROM** oder **FILE**) festgelegt.

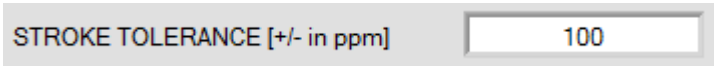
**RAM:**
Das **RAM** ist ein **flüchtiger** Speicher im Mikrokontroller des Sensors, d.h. nach Ausschalten der Spannung am Sensor gehen diese Parameter wieder verloren.
**Der Sensor arbeitet grundsätzlich mit den Parametern, die sich im RAM befinden.**
Ist der Auswahlknopf **RAM** gewählt, dann werden die aktuellen Parameter nach Drücken von **SEND** in den **RAM** Speicher des Sensors geschrieben bzw. nach Drücken von **GET** aus dessen **RAM** Speicher gelesen.

**EEPROM:**
Das **EEPROM** ist ein **nichtflüchtiger** Speicher im Mikrokontroller des Sensors. Nach Ausschalten der Spannung am Sensor gehen die Parameter im **EEPROM** nicht verloren. Nach dem Wiedereinschalten der Spannung werden die Parameter aus dem **EEPROM** ins **RAM** geladen. Bildlich gesehen liegt das **EEPROM** also eine Stufe tiefer als das **RAM**. Der Datenaustausch zwischen **PC** und **EEPROM** erfolgt automatisch über das **RAM**. D.h. Parameter die ins **EEPROM** geschrieben werden, werden automatisch auch ins **RAM** geschrieben und Daten die aus dem **EEPROM** gelesen werden, werden automatisch auch ins **RAM** gelesen.
Ist der Auswahlknopf **EEPROM** gewählt, dann werden die aktuellen Parameter nach Drücken von **SEND** in den Speicher des nichtflüchtigen **EEPROMS** im Sensor geschrieben oder durch Drücken von **GET** aus dessen **EEPROM** gelesen.
Man sollte beim Parametrieren des Sensors mit dem **RAM** arbeiten. Wenn man eine geeignete Parametrierung für die jeweilige Anwendung gefunden hat, dann muss diese im **EEPROM** des Sensors hinterlegt werden, damit die Parameter bei einem Neustart des Sensors aus dem **EEPROM** ins **RAM** geladen werden können.

**FILE:**
Die aktuellen Parameter können nach Drücken von **SEND** in ein auswählbares File auf der Festplatte geschrieben werden bzw. durch Drücken von **GET** davon gelesen werden. Nach Drücken von **SEND** oder **GET** öffnet sich eine Dialogbox, in der man das gewünschte File selektieren kann.
**TIPP!** Eine für eine bestimmte Anwendung gefundene Parametereinstellung sollte in jedem Fall in einem File auf dem PC hinterlegt werden.

**GO [F11]:**
Anklicken dieser Taste startet den Datentransfer vom Sensor zum PC über die serielle RS232 Schnittstelle.

**STOP [F12]:**
Anklicken dieser Taste beendet den Datentransfer vom Sensor zum PC über die serielle RS232 Schnittstelle.




**STROKE TOLERANCE** ist eine +/- Toleranz für den Stanzhub innerhalb der Aussparung.

Der Zählerstand von **CNT GAP** entspricht dabei 1000 Promille.
Die Toleranzgrenzen werden wie folgt berechnet:

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

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

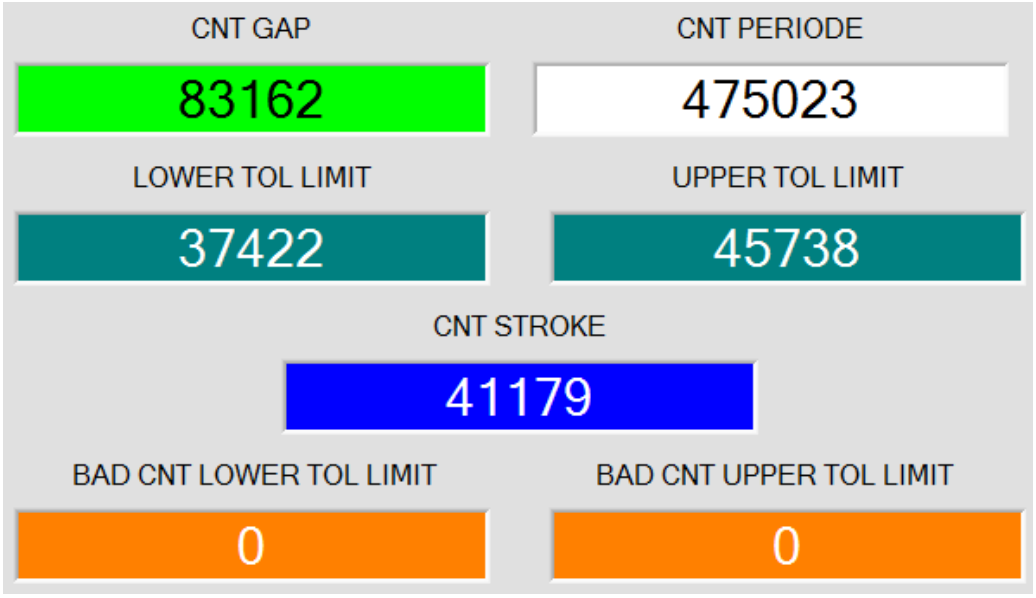
Mit **BAD CNT TO FAILURE** legt man fest, wie viele gleiche Fehler in Folge auftreten müssen, damit die entsprechenden Ausgänge geschaltet werden.

Mit **Digital OUTMODE** legt man fest, ob die Ausgänge High-Aktiv **(DIRECT)** oder Low-Aktiv **(INVERSE)** sind.

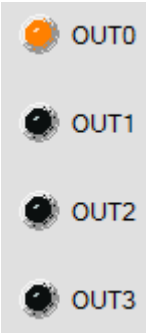
**COUNT STROKE** bestimmt ob die positive oder negative Flanke des Stanzhubes ausgewertet werden soll.

## 3.3 Graphische Anzeigeelemente

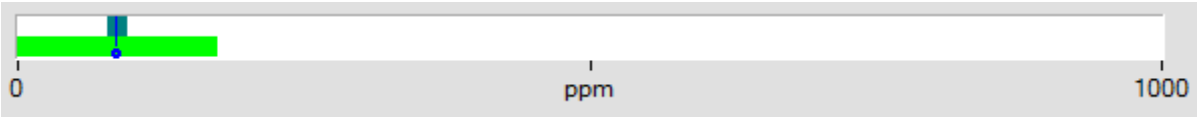
Zur Visualisierung aller zum Parametrieren wichtigen Daten stehen verschiedene Displays sowie ein graphisches Fenster zur Verfügung. Die Bedeutung der einzelnen Displays und des Graphen wird nachfolgend erklärt.




In diesen Displays wird angezeigt, bei welchem Zählerstand ein Ereignis eingetreten ist.
Der Zählerstand entspricht fast genau einer Angabe in Mikrosekunden.
**CNT PERIOD:** Zeit von Aussparung zu Aussparung.
**CNT GAP:** Länge der Aussparung.
**CNT STROKE:** Zeitpunkt des Stanzhubes.
**LOWER TOL LIMIT** und **UPPER TOL LIMIT** zeigen das Toleranzfenster an.
**BAD CNT LOWER TOL LIMIT** und **BAD CNT UPPER TOL LIMIT** zeigen an, wie viele gleiche Fehler in Folge aufgetreten sind.



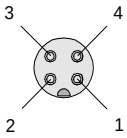

**OUT0 bis OUT13**
Die LEDs visualisieren den physikalischen Zustand der Ausgänge OUT0 bis OUT3.
Ist die LED schwarz, dann liegt der Ausgang auf 0V.
Ist die LED orange, dann liegt der Ausgang auf +24V




Im graphischen Anzeigefenster werden die einzelnen Displays nochmal normiert auf Promille visualisiert.
Die weiße Fläche entspricht **CNT PERIODE.**
Die grüne Fläche entspricht **CNT GAP.**
Cyan zeigt das Toleranzfenster welches durch **LOWER TOL LIMIT** und **UPPER TOL LIMIT** bestimmt wird.
Der blaue Cursor zeigt, wann der Stanzhub aufgetreten ist.

# 4 Anschlussbelegung von SPECTRO1-SC Sensoren

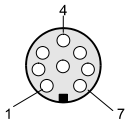
**Anschluss SPECTRO1-SC an PC:**

| **4-pol. M5 Buchse (Typ Binder 707)**<br>***SPECTRO1-SC/PC-RS232*** | |  |
|---|---|---|

| **Pin-Nr.:** | | **Belegung:** |
|---|---|---|
| 1 | | +24VDC (+Ub) |
| 2 | | 0V (GND) |
| 3 | | Rx0 |
| 4 | | Tx0 |

**Anschlusskabel zur Wahl:**
cab-las4/PC-…
cab-4/USB-…
cab-4/ETH-…

**Anschluss SPECTRO1-SC an SPS:**

| **8-pol. Buchse (Typ Binder 712)**<br>***SPECTRO1-SC/SPS*** | |  |
|---|---|---|

| **Pin-Nr.:** | **Litzenfarbe:**<br>**(cab-las8/SPS)** | **Belegung:** |
|---|---|---|
| 1 | weiß | 0V (GND) |
| 2 | braun | +24V (± 10 %) |
| 3 | grün | IN0 (Digital 0: 0 … 1V, Digital 1: +Ub – 10%) |
| 4 | gelb | IN1 (Digital 0: 0 … 1V, Digital 1: +Ub – 10%) |
| 5 | grau | OUT0 (Digital 0: 0 … 1V, Digital 1: +Ub – 10%) |
| 6 | rosa | OUT1 (Digital 0: 0 … 1V, Digital 1: +Ub – 10%) |
| 7 | blau | OUT2 (Digital 0: 0 … 1V, Digital 1: +Ub – 10%) |
| 8 | rot | OUT3 (Digital 0: 0 … 1V, Digital 1: +Ub – 10%) |

**Anschlusskabel:**
cab-las8/SPS-…

# 5 RS232 Schnittstellenprotokoll

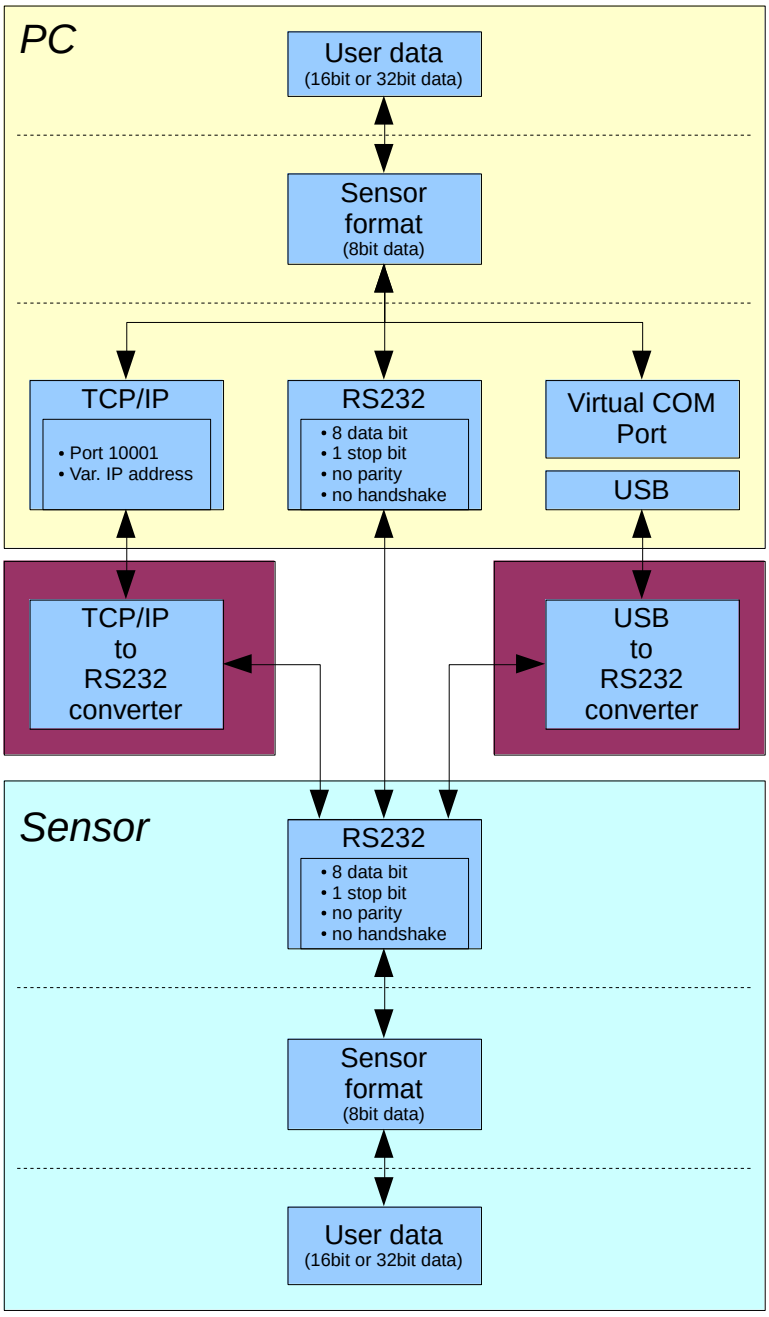
The sensors of the SPECTRO1-SC series operate with the following **parameters** that are sent to the sensor or read from the sensor in the stated sequence.
Info! 1 **byte** = 8bit 1 **word** = 2 **byte** 1 **long** = 2 **word** = 4 **byte**

| 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) |

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:** | DigOUT | word | Status of output LED's (Bit0, Bit1, Bit2, Bit3) |

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.<br>ARG=1: Invalide order number was sent to the sensor<br>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  := 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.

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 | | | |

# A. Firmwareupdate über Software Firmware Loader

## A.1 Bedienungsanleitung Software Firmware Loader V1.1

Die vorliegende Bedienungsanleitung dient zur Installation der PC-Software für den Firmware Loader. Zur Unterstützung der Inbetriebnahme des Firmware Loader werden in dieser Bedienungsanleitung die einzelnen Funktionselemente der graphischen Windows®-Benutzeroberfläche erklärt.

Die Software ermöglicht es dem Anwender, ein automatisches Firmwareupdate durchzuführen. Das Update wird dabei über die RS232 Schnittstelle durchgeführt.

Zum Firmwareupdate werden ein Initialisierungsfile (xxx.ini) sowie ein Firmwarefile (xxx.elf.S) benötigt. Diese Files sind vom Lieferanten erhältlich. In manchen Fällen wird ein zusätzliches Firmwarefile für den Programmspeicher (xxx.elf.p.S) benötigt, dieses File wird dann automatisch mit den beiden anderen Dateien zur Verfügung gestellt.

**Wichtig!** Für das Firmwareupdate ist es unbedingt erforderlich, dass alle zwei oder drei Files in dem gleichen Ordner hinterlegt sind.

Nachdem das Initialisierungsfile über den Firmware Loader geladen wurde, erfolgt ein Plausibilitätstest. Wenn das Initialisierungsfile verändert worden ist oder beschädigt wurde, ist ein Firmwareupdate nicht möglich.

Nach erfolgreichem Plausibilitätstest werden die Anweisungen, die im Initialisierungsfile hinterlegt worden sind, schrittweise durchgeführt.

Bei einem Firmwareupdate wird der komplette Mikrokontroller im Sensor gelöscht. D.h. dass sowohl das Programm im Programmspeicher als auch die Daten im Datenspeicher verloren gehen.

Der Programmspeicher wird durch die neue Firmware automatisch wieder richtig beschrieben.

Die im Datenspeicher (EEPROM) abgespeicherten Parametereinstellungen, Temperaturkurven, Linearisierungskurven etc. werden jedoch gelöscht.

**Mit dem Firmware Loader V1.1 werden die Daten im EEPROM gesichert, um sie nach einem erfolgreichen Firmware Update wieder aufzuspielen.**

**Dazu wird ein EEPROM Backup File erzeugt.**

## A.2 Installation der Software Firmware Loader V1.1

Für eine erfolgreiche Installation der Firmware Loader Software müssen folgende Hardware-Voraussetzungen erfüllt sein:

- IBM PC AT oder kompatibler
- VGA-Grafik
- Microsoft® Windows® XP, VISTA, 7
- Serielle RS232-Schnittstelle am PC
- Microsoft®-kompatible Maus
- Kabel cab-las4/PC (cab-las5/PC) für RS232 oder cab-4/USB (cab-5/USB) für USB-Slot
- CD-ROM-Laufwerk

Die Software kann nur unter Windows installiert werden. Deshalb müssen Sie zunächst Windows starten, falls es noch nicht aktiv ist.

Installieren Sie nun die Software wie im Folgenden beschrieben:

1. Sie können die Software direkt von der Installations-CD-ROM installieren. Auf der CD-ROM befindet sich der Ordner INSTALL. Im Ordner INSTALL ist eine SETUP Anwendung. Zum Installieren der Software müssen Sie diese SETUP-Anwendung starten.

2. Das Installationsprogramm meldet sich mit einem Dialogfeld und schlägt vor, die Software im Verzeichnis C:\“DATEINAME“ auf der Festplatte einzurichten.
   Akzeptieren Sie den Vorschlag mit **OK** oder [**ENTER**] oder ändern Sie die Pfad-Vorgaben nach Ihren Wünschen.

3. Während der Installation wird eine neue Programm-Gruppe für die Software im Windows Programm-Manager erzeugt. Außerdem wird in der erzeugten Programmgruppe ein Icon für den Start der Software automatisch generiert. Falls die Installation erfolgreich durchgeführt werden konnte, meldet sich das Installationsprogramm mit einer Dialogbox ‘’Setup OK’’.

4. Nach erfolgreicher Installation kann die Software durch Doppelklick auf das Icon mit der linken Maustaste gestartet werden.

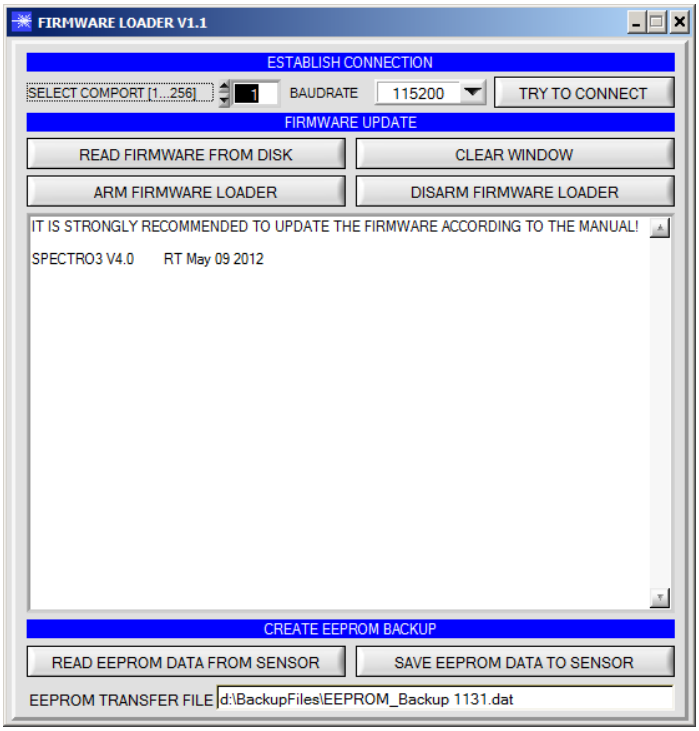
Windows® ist ein eingetragenes Warenzeichen der Microsoft Corp.
VGA™ ist ein Warenzeichen der International Business Machines Corp.

**Bitte lesen Sie diesen Abschnitt unbedingt zuerst durch, bevor Sie beginnen.**

**Im Beispiel wird ein Softwareupdate von SPECTRO3 V4.0 auf SPECTRO3 V4.1 durchgeführt.**

## Schritt 1:

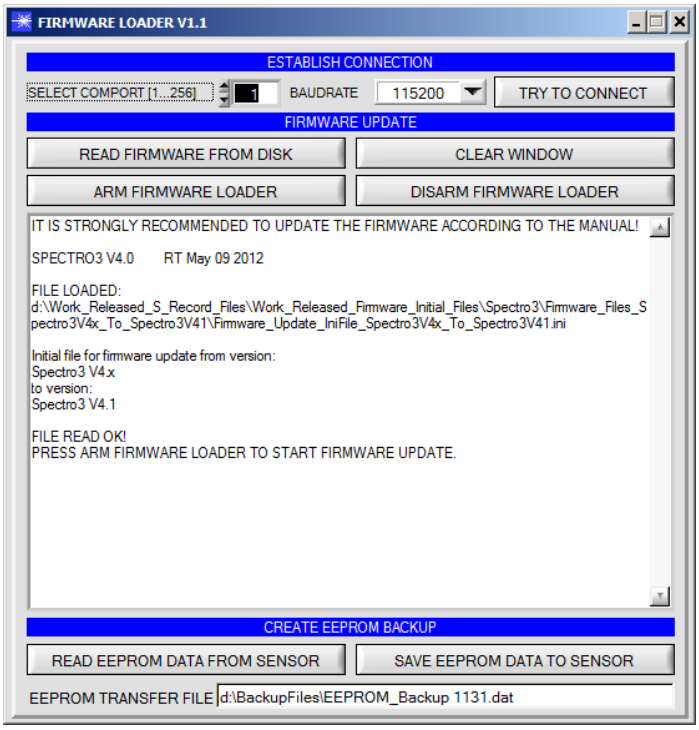
Nach dem Aufruf der Firmware Loader Software erscheint nebenstehendes Fenster auf der Windows® Oberfläche.




Die Software versucht sofort nach dem Start eine Verbindung zum angeschlossenen Sensor herzustellen. Sollte der Sensor nicht an **COM PORT 1** angeschlossen sein, wählen Sie den entsprechenden **COM PORT** aus. Beachten Sie auch, dass die richtige **BAUDRATE** eingestellt ist.
Versuchen Sie jetzt, über **TRY TO CONNECT** eine Verbindung aufzubauen. Nachdem die Verbindung steht, meldet sich der Sensor mit der momentan aufgespielten Firmware.

## Schritt 2:

Drücken Sie den Button **READ FIRMWARE FROM DISK** und laden das File **xxx.ini.**




Das geladene Initialisierungsfile wird im Statusfenster angezeigt.
Wie oben beschrieben, wird zuerst ein Plausibilitätstest des Initialisierungsfiles durchgeführt.
Wenn das File in Ordnung ist, kommt die Meldung:
File read OK!
Press **ARM FIRMWARE LOADER** to start firmware update.

Bitte beachten Sie den Kommentar, der im Anzeigefenster erscheint. Mit Hilfe des Kommentars können Sie sicherstellen, dass Sie das richtige Initialisierungsfile geladen haben.

## Schritt 3:

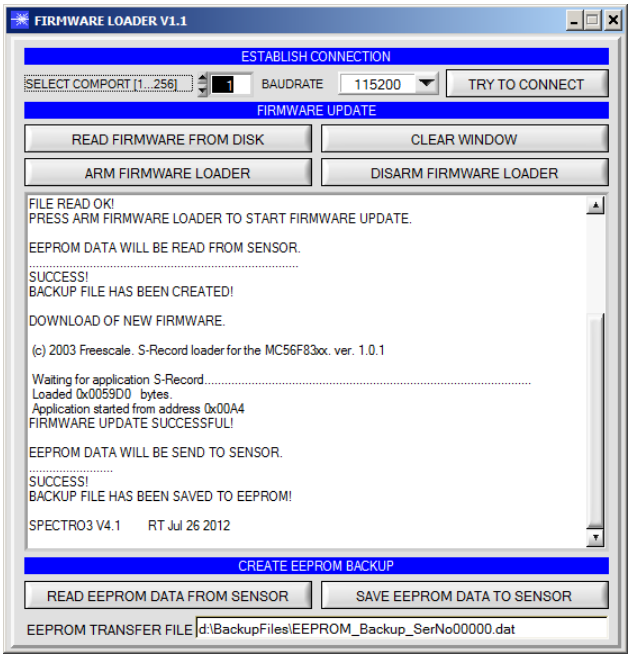
Drücken Sie jetzt den Button **ARM FIRMWARE LOADER**. Das Programm versucht nun einen Softwarebefehl abzusetzen, welcher den normalen Programmlauf unterbricht und zur Startadresse des Bootsektors springt. War dies erfolgreich, meldet sich der Sensor mit der Aufforderung, das S-Record File in den Sensor zu laden.

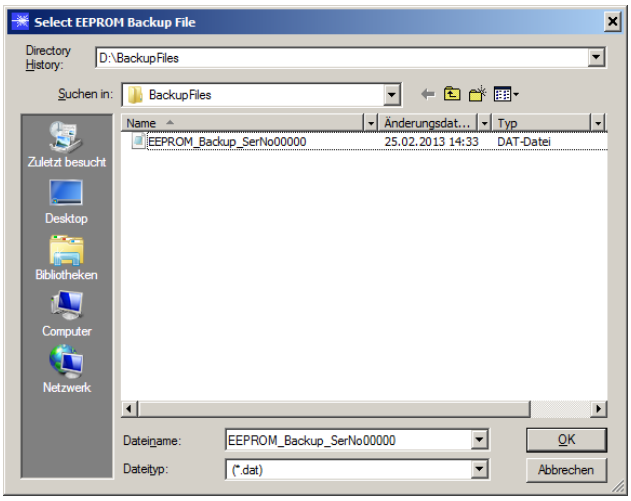
Das Firmwareupdate läuft nach Drücken von **ARM FIRMWARE LOADER** voll automatisch.

Zwischendurch werden Sie nur aufgefordert, einen Namen für das EEPROM Backup File einzugeben. Sollte das Firmwareupdate bis zum Auslesen der EEPROM Daten problemlos laufen, danach aber aus irgendwelchen Gründen schief gehen, kann das EEPROM Backup File jederzeit über **SAVE EEPROM DATA TO SENSOR** aufgespielt werden.
Den Dateinamen für Ihr **EEPROM Backup File** sollten Sie so wählen, dass es bei mehreren Sensoren nicht zu einer Verwechslung kommen kann. Ratsam wäre ein Dateiname, der die Seriennummer des Sensors enthält. Es spricht auch nichts dagegen, sich diese Datei für zukünftige Updates zu sichern.

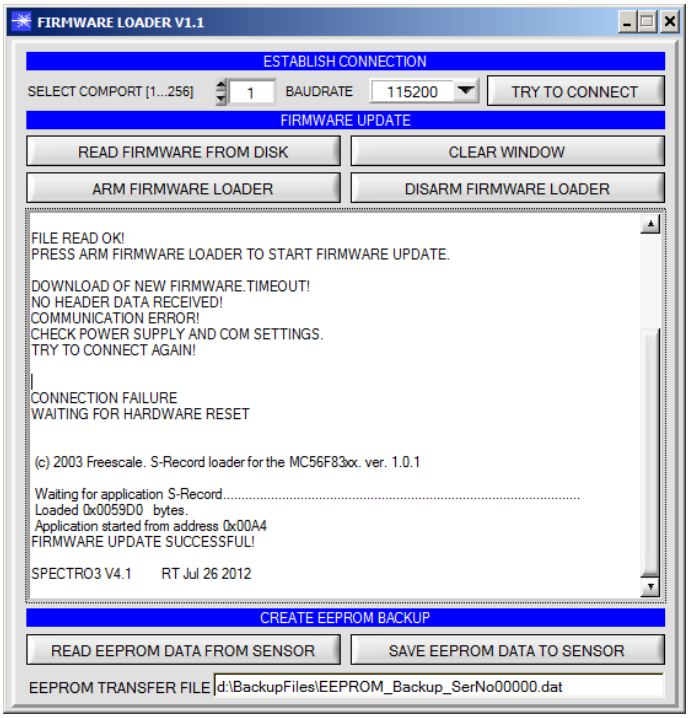
Nach erfolgreichem Update meldet sich der Sensor mit der Statuszeile der neuen Firmware.

Das gesamte Update kann bis zu 1 Minute dauern.



Sollte wider Erwarten beim Update des Programmspeichers etwas schief gegangen sein, haben Sie immer noch die Möglichkeit ein Update durchzuführen, auch wenn der Sensor „abgeschossen“ wurde.
Stellen Sie sicher, dass Sie den richtigen **COM PORT** ausgewählt haben und die richtige **BAUDRATE**.
Nach **TRY TO CONNECT** werden Sie keine Verbindung erhalten.
Laden Sie das entsprechende **xxx.ini** File von der Festplatte.
Drücken Sie **ARM FIRMWARE LOADER**.
Das Programm versucht den Softwarebefehl zum Update abzusetzen. Dies funktioniert jedoch nicht und Sie erhalten die Meldung **CONNECTION FAILURE**.
Der Firmware Loader ist aber jetzt für 30 Sekunden „scharf“.
Wenn Sie innerhalb der 30 Sekunden einen Hardware Reset durchführen, wird das Firmwareupdate automatisch durchgeführt.

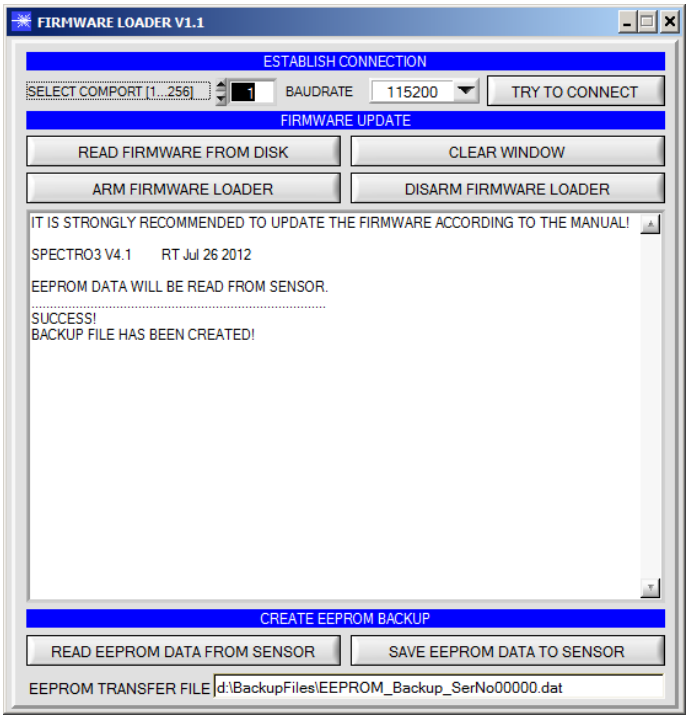



Nach erfolgreichem Update meldet sich der Sensor mit der Statuszeile der neuen Firmware.

Das gesamte Update kann bis zu 1 Minute dauern.

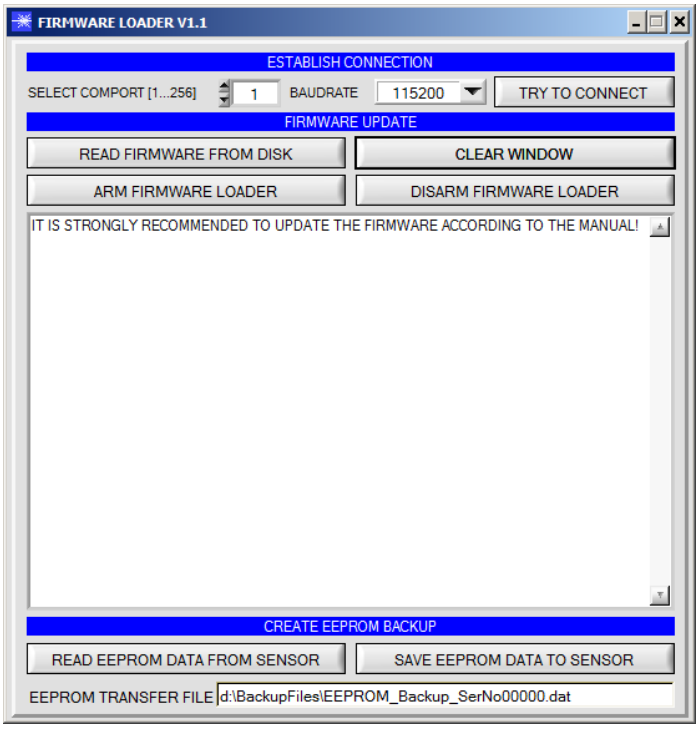
INFO! Sollte der Sensor „abgeschossen“ worden sein, dann arbeitet der Sensor mit einer Baudrate von 115200.

Sie können jederzeit ein EEPROM Backupfile erzeugen, um es auf Ihrer Festplatte zu archivieren.
Drücken Sie dazu **READ EEPROM DATA FROM SENSOR.** Sie werden aufgefordert ein Initialisierungsfile zu wählen, falls noch keines geladen wurde. Anschließend werden Sie nach einen Dateinamen gefragt. Der gewählte Name wir im Display **EEPROM TRANSFER FILE** angezeigt.
Den Dateinamen für Ihr **EEPROM Backup File** sollten Sie so wählen, dass es bei mehreren Sensoren nicht zu einer Verwechslung kommen kann. Ratsam wäre ein Dateiname der die Seriennummer des Sensors enthält.




Der Firmware Loader liest jetzt die kompletten EEPROM Daten im Datenspeicher aus und speichert diese im selektierten File.
War dies erfolgreich, erscheint die Meldung:
**Success!**
**Backup File has been created!**

Sollte bei einem Firmwareupdate etwas schief gegangen sein, das **Backup File** jedoch noch erzeugt worden sein, kann das gespeicherte EEPROM **Backup File** über **SAVE EEPROM DATA TO SENSOR** jederzeit in den Sensor geladen werden.

**CLEAR WINDOW** setzt das Anzeigedisplay zurück.

Mit **DISARM FIRMWARE LOADER** können Sie das Firmwareupdate abbrechen, wenn Sie längere Zeit keine Antwort oder Meldungen in der Statuszeile erhalten.
Warten Sie jedoch ca. 1 Minute, bevor Sie diesen Button drücken.