rexroth
A Bosch Company

Smart Flex Effector

R320103228

en

This data has been provided solely for the purpose of product description. Any references to possible uses are provided merely as a convenience and shall be understood as example applications or suggestions. Catalog data may not be construed as guaranteed properties. The information given does not release the user from the obligation of own judgment and verification. It should be noted that our products are subject to a natural process of aging and wear.

The title page contains an illustration of a sample configuration. The product as delivered can differ from the illustration.

The original instructions have been prepared in German.

Table of contents

1 About this documentation

1.1 Validity of the documentation

1.1.1 This documentation applies to the following products:

1 Smart Flex Effector (SFE) (Kernmodul)

2 Werkseitiger montierter “Standardflansch”(Aluminium) Zur einfachen Montage von
- **z. B. Greifern**
- **optional erhältlichen Adapterplatten.**

3 Flansch (Aluminium). Zur einfachen Anbindung
- **an Robotern**
- **an Linearachsen usw.**

4 Adapterplatte (Aluminium) (optional). Montage am “Standardflansch”

This product is a mechatronic component which is amended by using the related firmware.

The Smart Flex Effector (SFE) is a sensor-based compensation module with independent kinematics in six degrees of freedom. It increases precision, e.g. in handling robots, and opens up new areas of application for robots and Cartesian systems. The SFE was developed to compensate for a process-related offset in the translations X, Y, Z and the rotary motions Rx, Ry, Rz. A mechanical locking feature is integrated into the unit as standard.

1.1.2 Material numbers SFE

The following set variants of the SFE can be ordered:

Set of 2: Core module + flange cover

Set of 3: Core module + flange cover + adapter plate

Attention:

A standard pitch circle of 31.5 mm for connecting the gripper is always provided on the gripper side of the core module. (see table Sets of 2)

With the help of the adapter plates (see Sets of 3), the standard pitch circle of 31.5 mm for connecting a gripper can be changed correspondingly to other pitch circles.

The connection of the SFE to robots and grippers is designed according to the ISO 9409-1 standard.

Material numbers			
Set	**Core module**	**Flange cover**	**Adapter plate**
Sets of 2			
R124300001	R124000002	R124000012 - PC 31.5	none (PC 31.5)
R124300002	R124000002	R124000011 - PC 40	none (PC 31.5)
R124300003	R124000002	R124000010 - PC 50	none (PC 31.5)
R124300004	R124000002	R124000013 - PC 0 (blank)	none (PC 31.5)
Sets of 3			
R124300005	R124000002	R124000012 - PC 31.5	R124000021 - PC 40
R124300006	R124000002	R124000011 - PC 40	R124000021 - PC 40
R124300007	R124000002	R124000010 - PC 50	R124000021 - PC 40
R124300008	R124000002	R124000013 - PC 0 (blank)	R124000021 - PC 40
Sets of 3			
R124300009	R124000002	R124000012 - PC 31.5	R124000020 - PC 50
R124300010	R124000002	R124000011 - PC 40	R124000020 - PC 50
R124300011	R124000002	R124000010 - PC 50	R124000020 - PC 50
R124300012	R124000002	R124000013 - PC 0 (blank)	R124000020 - PC 50

Material numbers			
Set	**Core module**	**Flange cover**	**Adapter plate**
Sets of 3			
R124300013	R124000002	R124000012 - PC 31.5	R124000022 - PC 0 (blank)
R124300014	R124000002	R124000011 - PC 40	R124000022 - PC 0 (blank)
R124300015	R124000002	R124000010 - PC 50	R124000022 - PC 0 (blank)
R124300016	R124000002	R124000013 - PC 0 (blank)	R124000022 - PC 0 (blank)

1 **Material numbers SFE sets**

The following material numbers can be ordered separately:

Material number (flange cover):

- R124000010 ISO 9409-1 50-4-M6
- R124000011 ISO 9409-1 40-4-M6
- R124000012 ISO 9409-1 31.5-4-M5
- R124000013 Blank

Material numbers (adapter plate):

- R124000020 ISO 9409-1 50-4-M6
- R124000021 ISO 9409-1 40-4-M6
- R124000022 Blank

This documentation is intended for assembly personnel and service technicians.

Caution	
	**This documentation contains important information for the proper and safe installation and commissioning of the product.** ➢ Read this documentation and especially the chapter “Safety“ completely before working with the product.

1.2 Structure of safety information

Hazard classification

The hazards that can occur at the machine are divided into the following classes:

- Danger
- Warning
- Caution
- Attention

Danger

This warning notice indicates a hazard with high risk. If the safety regulations are not observed, there is a risk of death or severe injury.

⚠ Danger	
	**Type and source of hazard** Hazard consequence ➢ Remedy

Warning

This warning notice indicates a hazard with moderate risk. If the safety regulations are not observed, the hazard may result in death or serious injury.

⚠ Warning	
	**Type and source of hazard** Hazard consequence ➢ Remedy

Caution

This warning notice indicates a hazard with low risk. If the safety regulations are not observed, the hazard may result in minor injuries.

⚠ Caution	
	**Type and source of hazard** Hazard consequence ➢ Remedy

Caution

This warning notice indicates a hazard with low risk. If the safety regulations are not observed, the hazard may result in damage to property.

Caution	
	**Type and source of hazard** **Hazard consequence** ➢ Remedy

1.3 Abbreviations

The following abbreviations are used in this documentation:

Abbreviation	Meaning
GUI	**G**raphic **U**ser **I**nterface
SFE	**S**mart **F**lex **E**ffector

2 **Abbreviations**

2 Safety

2.1 Intended use

As a sensor-supported compensation element for robots and Cartesian systems, the SFE serves as a process-related offset in the translations X, Y, Z and the rotations Rx, Ry, Rz.

Intended use also includes compliance with the applicable legal provisions and safety regulations, and the operating, maintenance, and servicing requirements prescribed by the manufacturer.

Any other use is considered improper. The manufacturer shall not be liable for damage resulting thereof. Any associated risk shall be borne by the operator.

The product (SFE) is exclusively intended for incorporation into a final machine or system or for assembling with other components to build a final machine or system. The product is intended exclusively for professional use and not for private use.

Within the meaning of intended use, the SFE is not a safety component.

2.1.1 The following applications are approved for the product:

- Mounting to a handling system between the flange plate and a tool, such as a gripper.
- Installation into a machine/system or attachment to a robot. The applicable directives must be observed and adhered to.
- The SFE may be used for passive compensatory movements and transfers of position.
- The SFE may be unlocked for the duration of handling process with active contact of tools and a workpiece or device and during the necessary advancing and moving away processes. For transfer journeys or general movements at higher speeds, the SFE must be operated in a locked state. During these process stages, the status query of the locking state must be capable of ensuring that the state is not changed unexpectedly.
- The SFE may only be used in compliance with the technical data, see chapter Technical data. These must be safeguarded using further measures, if applicable.
- The SFE may only be positioned vertically in the handling process (main axis "z" in the gravitational direction).

2.2 Improper use

Using the product in any other way than as described under "Intended use" is considered to be misuse and is therefore not permitted.

Bosch Rexroth AG will not accept any liability for injury or damage caused by misuse of the product. The risks associated with any misuse of the product shall be borne by the user alone.

Misuse of the product includes in particular:

- Use of any kind to transport persons.
- Misuse of the SFE as collision protection.
- Moving the SFE to the position compensation end position. The end positions must be monitored in the LED mode STATUS (see section STATUS).
- The SFE is a dynamically excited oscillation system. As it must be prevented from reaching the position compensation end positions, it is recommended that you start the system with a travel speed lower than 100 mm/s when setting up applicative processes in an unlocked state.
- Oscillating movement patterns must be avoided during operation in an unlocked state, as this may cause damage to the device.

2.3 Residual risks

Risks of damage, failure or destruction are:

- Removing the SFE from the handling system
- Massive force impact due to incorrect use
- Collision
- Incorrect connection to power

Warning	
	**Exceeding the mechanical limit values** Exceeding the mechanical limit values may overload and destroy the mechanical system components. Components which become loose may lead to personnel injury and material damage. ➢ Only use the SFE within the permitted operating limits (see chapter Technical data). ➢ Do not carry out any transfer journeys or oscillating movements in an unlocked state.

Caution	
	**Inactive protective measures** The transmission and processing of incorrect sensor data may lead to contact and unexpected system behavior. ➢ When using the SFE, limit the machine movement by using suitable protective measures. ➢ The SFE may only be used on systems and equipment with the protective measures intended for the system. ➢ Only operate the SFE if the protective measures are enabled. ➢ Only move the system into which the SFE is installed if the safety area is active. ➢ If the protective measures are disabled, only operate the SFE in the control state: "manual with safely reduced travel speed".

Caution	
	**Bellows of TPE plastics - Restriction when used with oil and grease** If the bellows are exposed to oil and grease for a longer period of time, this may lead to damage. ➢ Avoid wetting and residues, and remove these promptly. ➢ Avoid mechanical damage which may cause tears/holes in the bellows.

Caution	
	**Bellows of TPE plastic - No absence of PWIS** Processes in which an absence of PWIS must be guaranteed may be influenced by this. ➢ Do not use the SFE in applications where absence of PWIS is required.

Caution	
	**Warm-up behavior** Temperature-related behavior can negatively affect the sensors for environment or application-specific influences during operation. ➢ By way of regular plausibility checks, ensure that the sensors are working correctly. ➢ Compensate for possible changes to the sensor setting by regularly locking and unlocking.

2.4 General safety instructions

- The SFE has no protective measures against contact with other objects and no protective equipment in the case of defects. Any protective measures must be carried out by the higher-level machine/system.
- Highly dynamic processes may lead to overloading of the SFE and require corresponding safety measures.
- Observe the applicable accident prevention and environmental regulations.
- Observe the safety rules and regulations of the country in which the product is used.
- Only use Bosch Rexroth products when they are in a technically perfect condition.
- Observe all notices on the product.
- Persons who mount/install, operate, disassemble or maintain Bosch Rexroth products must not be under the influence of alcohol, other drugs or medications which might affect their judgment or slow down their reactions.
- Only use manufacturer-approved accessories and spare parts in order to exclude any hazards to personnel and personal injuries.
- Comply with the technical data and environmental conditions stated in the product documentation.
- If unsuitable products are installed or used in safety-critical applications, this may lead to uncontrolled operating statuses in the application which can cause personal injury and/or damage to property. You should therefore only use product in safety-critical applications if this use has been expressly specified and permitted in the product documentation.
- You may only commission the product once it has been verified that the end product (for example a machine or system) into which the Bosch Rexroth products have been installed complies with the country-specific requirements, safety regulations and standards for the application.
- Except where otherwise documented, Bosch Rexroth products are intended for operation in networks which have been locally, physically and logically secured, with access restricted to authorized persons and not classified in accordance with IEC 62443-4-2.
- Only firmware or a GUI (SFE tool) provided by Bosch Rexroth may be used. This is provided by the Bosch Rexroth service team or via the Bosch Rexroth website.

2.5 Personnel qualifications

The activities described in this document require fundamental knowledge of mechanical and electrical engineering principles and familiarity with the associated technical terminology. In order to ensure safe use, these activities may therefore only be performed by appropriately trained specialists or instructed persons working under the supervision of a trained specialist.

A trained specialist is a person whose professional training, knowledge, experience and familiarity with the relevant regulations enable him/her to assess the tasks assigned to him/her, identify potential hazards and take appropriate safety precautions. A trained specialist must adhere to the relevant technical rules and standards.

2.6 Personal protective equipment

For safety reasons, safety shoes must be worn while assembling the SFE. All items of personal protective equipment must be intact.

3 Product description

The SFE is a compensation element to compensate for deviations in alignment or tolerance, for example, during a joining process with a robot or Cartesian system.

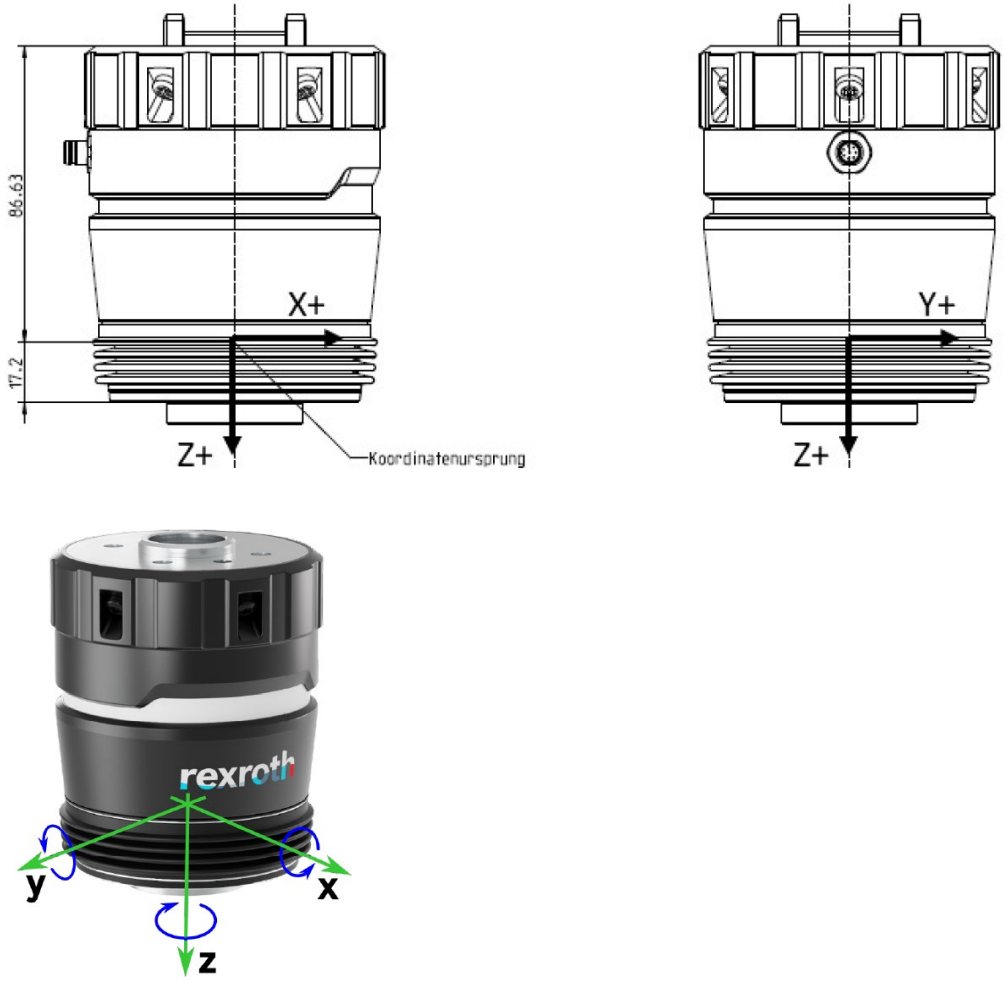
This compensation is generated passively via a freedom of movement of the compensation of elements in all 6 degrees of freedom.

The deflection is monitored by means of a sensor system and can be read via an RS-485 interface (see chapter Interface description).

In addition, locking or blocking of the freedom of movement of the element is possible. The user can choose to control the locking mechanism via the RS-485 interface or digital I/O. The element is supplied with a DC voltage of 24 V via the connection cable.

3.1 Axial alignment

The SFE axial alignment is designed based on the standard axis pattern for robots.



1 **Axes**

3.2 Technical data

Designation	Unit	Value
Housing material	-	Aluminum, anodized
Protection class	-	IP54 according to ISO 60529:2014
Maintenance	-	Lifelong lubrication
Mass of system (without adapter plate)	kg	1.3
Manipulator connection	-	Standard: ISO 9409-1 31.5-4-M5 / 40-4-M6 / 50-4-M6
Permissible ambient temperature during operation	°C	5 – 50
Permissible ambient temperature during storage and transport (thawing and freezing not permissible)	°C	-20 - 50
Compensation path XY	mm	± 3
Compensation path Z	mm	- 3
Compensation angle XY	°	± 3.4
Compensation angle Z	°	± 6.8
Max. handling weight	kg	6
Operating limits in the locked state:		
Max. sum of load moments Mx, My (locked state, relative to the coordinate origin, see also chapter Axial alignment)	Nm	13
Max. load torque Mz (locked state)	Nm	tbd
Max. load Fz (compressive force, locked state)	N	tbd
Operating limits during the locking process:		
Max. sum of load torques Mx, My during locking process	Nm	tbd
Max. load Fz during locking process (compressive force)	N	55

Designation	Unit	Value
Restoring forces in the unlocked state:		
Breakaway force spring	[N]	6
Typical spring rate path compensation in axial direction X and Y	$\frac{N}{mm}$	12
Typical spring rate path compensation in axial direction Z	$\frac{N}{mm}$	12
Max. spring rate axial angle equalization around axis X and Y	$\frac{Nm}{°}$	tbd
Max. spring rate axial angle equalization around axis Z	$\frac{Nm}{°}$	tbd
Locking time (vertically suspended load)	s	< 0.4
Properties of position feedback:		At a constant temperature of 25°C
Typical translative error (a standard deviation)	mm	0.077
Typical repeatability (a standard deviation)	mm	0.009
Typical angle alignment error (a standard deviation)	°	0.157
Typical repeatability (a standard deviation)	°	0.02
Sampling rate (determined at a baud rate of 921600 Bd)	ms	10
Power supply voltage	V	24 DC +20%/-10%
Rated current	A	1
Maximum current when locking/unlocking	A	1.5
Maximum cable length of the DC supply	m	5
Maximum cable length of the RS485 interface (point-to-point connection)	m	12

3 **Technical data**

You can find example applications and an orientation for guideline values for a “Pick and Place” application in the section: Exemplary application cases.

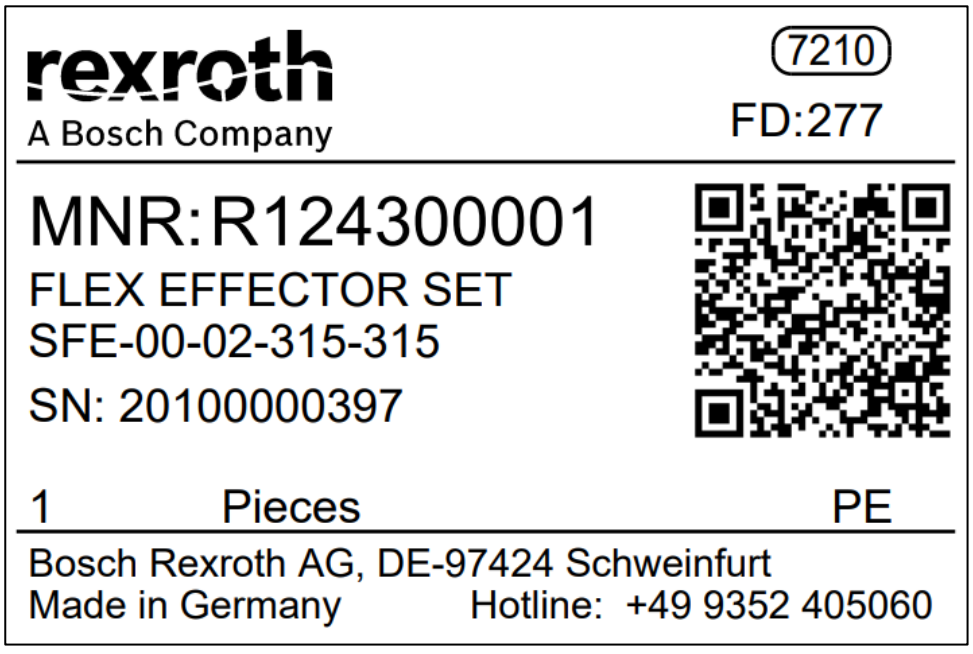
3.3 Scope of delivery

No.	Graphical representation	Description
1		• Smart Flex Effector (SFE) (1) • Screws M4x10 (6x) TORX ISO 14583 for fastening the flange (3)
2		• Shielded connection cable M8x1, 8-pole, A-coded, connector straight to female connector straight
3		• Flange (3), the required version must be selected: R124000010 / ISO 9409-50-4-M6 R124000011 / ISO 9409-40-4-M6 R124000012 / ISO 9409-31.5-4-M5 R124000013 / blank • Screws and cylinder pin included in the scope of delivery • Assembly by the customer

No.	Graphical representation	Description
4		• Adapter plate (4) optional; various versions available: R124000020 / ISO 9409-50-4-M6 R124000021 / ISO 9409-40-4-M6 R124000022 / blank • Screws and cylinder pin included in the scope of delivery • Assembly on the SFE (core module) (1) by the customer

4 **Brief instructions**

3.4 Product marking



2 **Nameplate**

The product nameplate bears the following information:

Abbreviation	Meaning
MNR	Material number
SN	Serial number
FD	Date of manufacture

5 **Nameplate**

4 Assembly

Warning
Danger due to product falling down during mounting If the SFE falls down during assembly, this may lead to personnel injury. ➢ Always carry out assembly with 2 persons present.

Caution
Damage to the SFE during assembly Mounting of the SFE in an unlocked state may lead to damage to the SFE. ➢ Mount the SFE only in the locked state. (SFE is already delivered in locked state) ➢ Lock the ready-mounted and used SFE before disassembly.

Preparation for assembly

Required tool:

- Torque screwdriver
- Bit TX20
- Hexagonal bit 5
- Hexagonal bit 4

Warning
Incorrect connection of the flange cover and the adapter plate Incorrect connection of the flange cover and adapter plate may result in parts becoming loose during assembly and production, which may then lead to personnel injury and material damage. ➢ Only connect the flange cover and the adapter plate with screws according to DIN 6912 low head) and applying the specified torque.

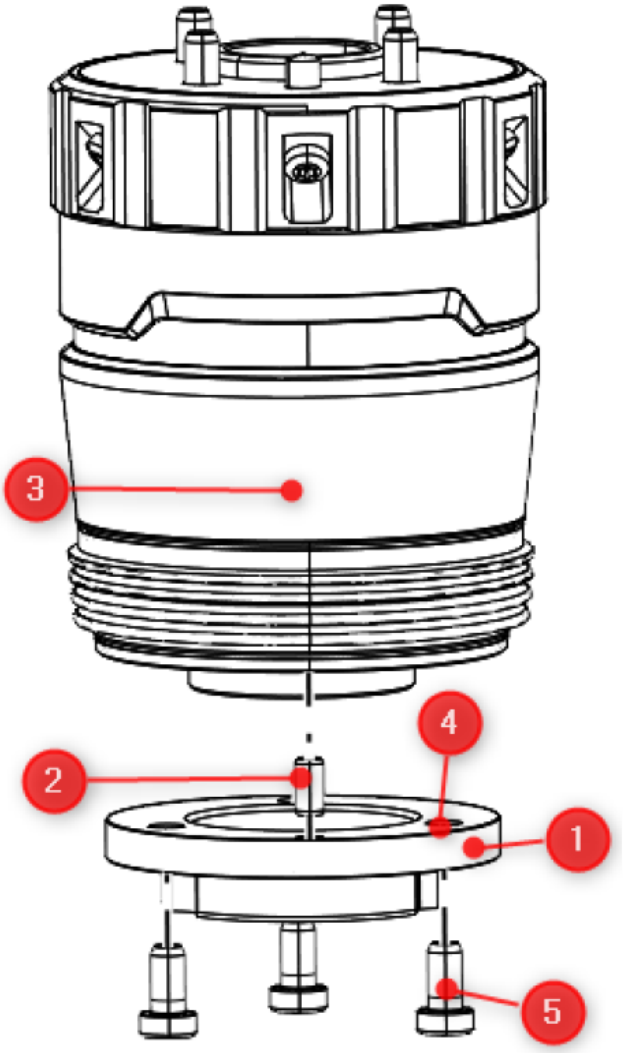
Caution
Incorrect sealing of the element Incorrect sealing of the element may mean that IP54 is not met. ➢ Seal the element on the flange side towards the robot in such a way that the protection class IP54 is maintained.

The steps described in the following chapters are necessary for mounting the SFE to the terminal selected by you. The flange cover and the optional adapter plate are supplied packaged in separate boxes.

4.1 Fitting the adapter plate

Required components:

- Adapter plate
- Cylinder pin Ø5
- 3 x screws M6 x 12



3 **Fitting the adapter plate**

➢ Position the adapter plate (1) with the straight pin Ø5 (2) at the SFE (3)

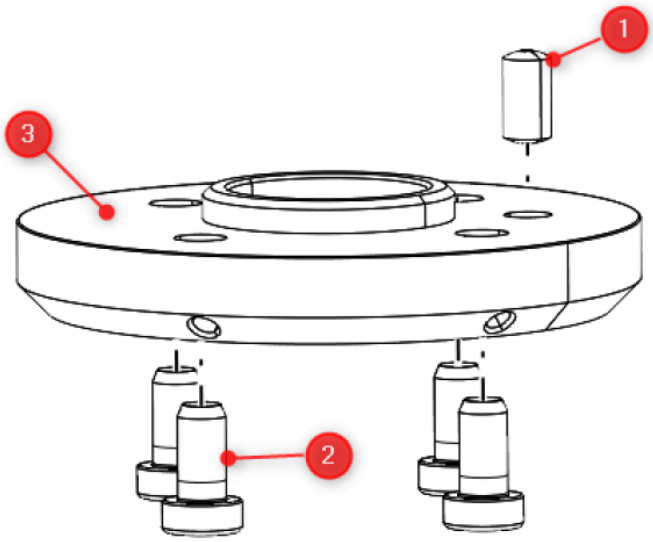
➢ Fix the adapter plate (1) in place on the SFE with the three screws (5) using the drill hole (M6 thread) (4)

- The specified screw length (12 mm) must be complied with or must not be exceeded.
- Tightening torque 10.25 Nm

4.2 Flange cover mounting

Required components:

- Flange cover
- Screws M6x12 or M5x10
- Cylinder pin Ø6 or Ø5



4 **Flange cover assembly on counter flange**

➢ Position the flange cover (3) on the counter piece using the cylinder pin (1).

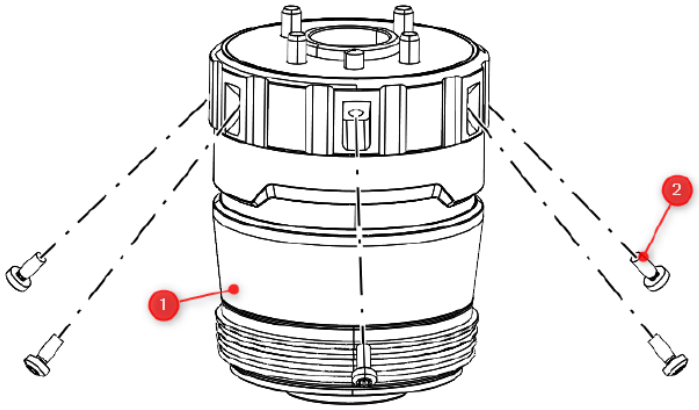
➢ Fix the flange cover (3) in place using the four supplied cylinder head screws (2) (hexagon bit 4 or hexagon bit 5).

  ☝ Tightening torque 6 Nm.

4.3 SFE assembly on flange cover

Required components:

- Flange cover
- 6 x screws M4x10



5 **SFE assembly on flange cover**

➢ Position the SFE (1) using the pre-mounted cylinder pin on the flange cover.
➢ Fasten the SFE (1) on the flange cover using the supplied screws (2).
  ☝ Tightening torque 3.13 Nm (bit TX20)

4.4 Electrical assembly

Caution	
	**Failure to comply with the technical voltage supply specifications** If the connection data is not observed, the SFE and the locking mechanism may become damaged or malfunctions may occur. ➢ Observe the defined maximum DC supply cable length. ➢ Ensure that the voltage on the device is within the specified tolerances (see chapter Technical data).

Caution	
	**Interface fault due to insufficient earthing of the cable shield** Unearthed cables may cause an interface fault, incorrect transmissions or exceedance of the EMC limit values. ➢ Earth the cable shield. ➢ Use shielded cables.

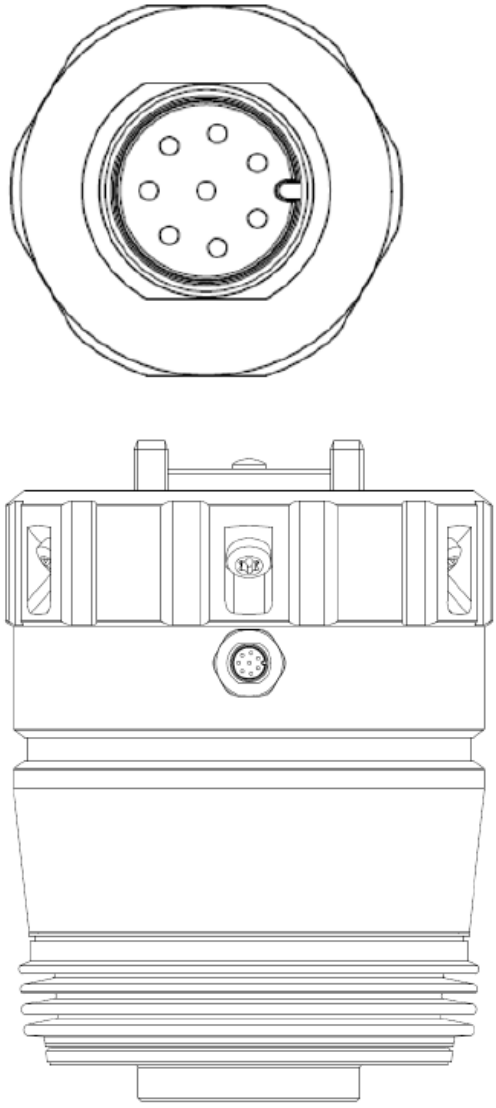
Caution	
	**Damage and short-circuit due to incorrect insertion of the connector** Incorrectly inserted connectors and bent pins may cause damage to the SFE and short-circuits. ➢ Ensure that the connector is inserted in the correct position when connecting the cable, and do not bend any pins.

Caution	
	**Damage and short-circuit due to tensile stress on the connection cable** Tension on the connection cable pins may cause damage to the SFE and short-circuits. This may cause changes to the behavior of the SFE. ➢ Avoid tensile stress on the cable.

Caution	
	**Short-circuit due to connection while the supply voltage switched on** Connecting the connector while the supply voltage switched on may cause damage to the SFE and short-circuits. ➢ Ensure that the SFE is disconnected from power before connecting or disconnecting the cable.

Electrical assembly

Once mechanical assembly of the SFE is complete, the connection cable is attached to the intended connector. During the plugging process, ensure correct alignment of the anti-twist feature of the connectors/female connectors.



6 **Connector for the SFE**

5 Desktop software: SFE tool as a file for download

The program allows you to visualize the possible SFE functions and also to update the software.

➢ Download the desktop software via the Rexroth homepage:
☞ URL: https://store.boschrexroth.com/Lineartechnik/Smart-Flex-Effector?cclcl=de_DE
➢ Then, accept the terms and conditions.

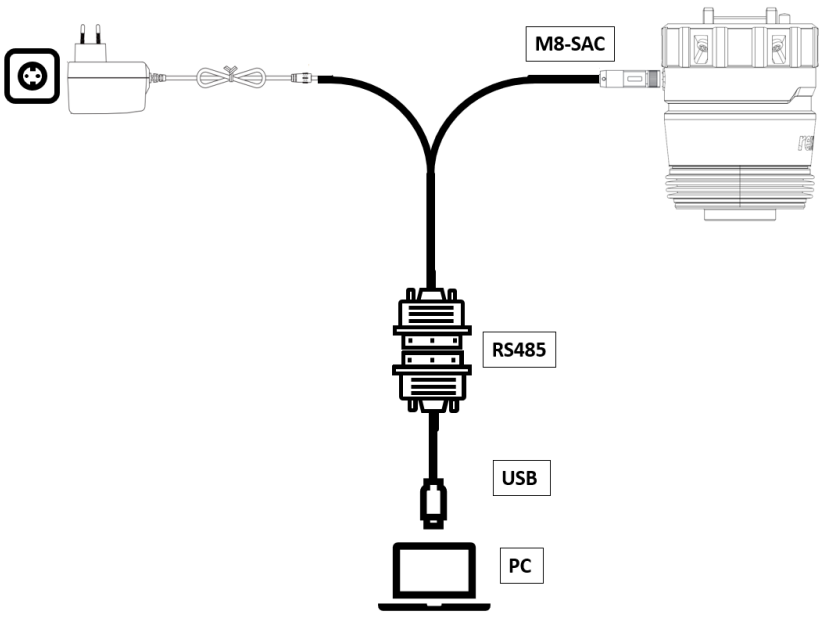
Minimum PC requirements:

- Operating system: Windows 10
- 64 bit with at least 8 GB RAM, 2 cores
- Recommended: at least 16 GB RAM
- Screen resolution: at least 1100 x 760 pixels

As accessory for the commissioning on a PC, a programming kit is available which can be ordered separately.

The set includes:

- Connection cable
- Power supply unit
- USB-RS485 adapter

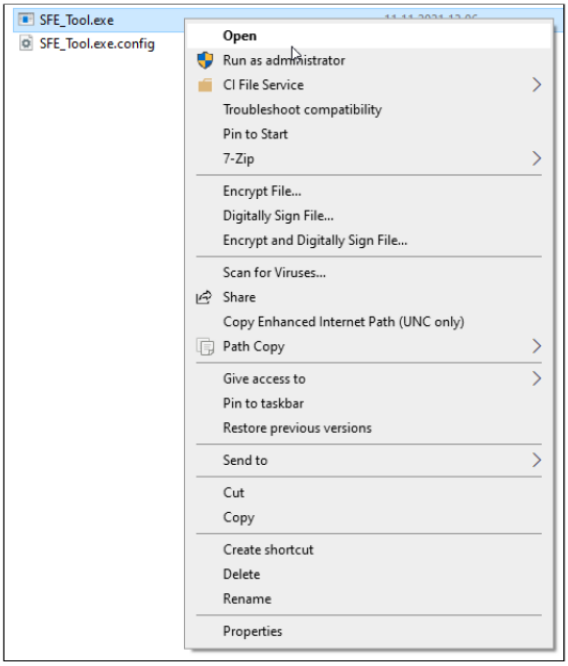


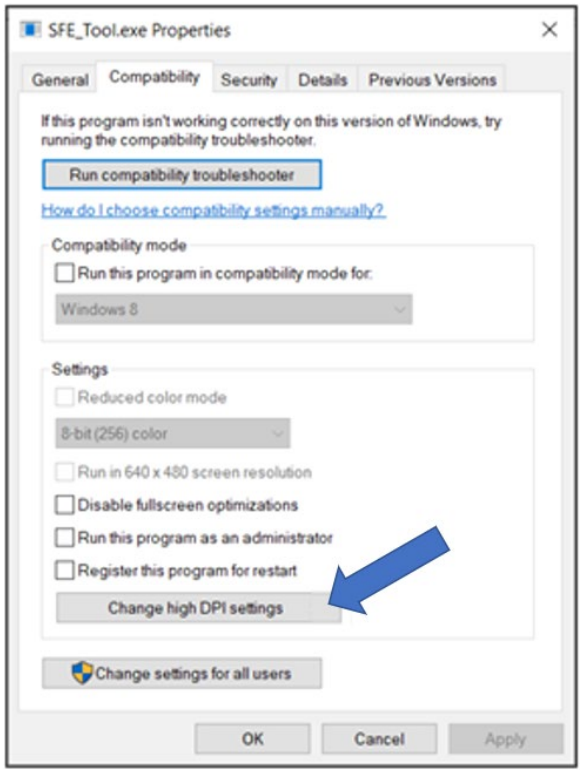
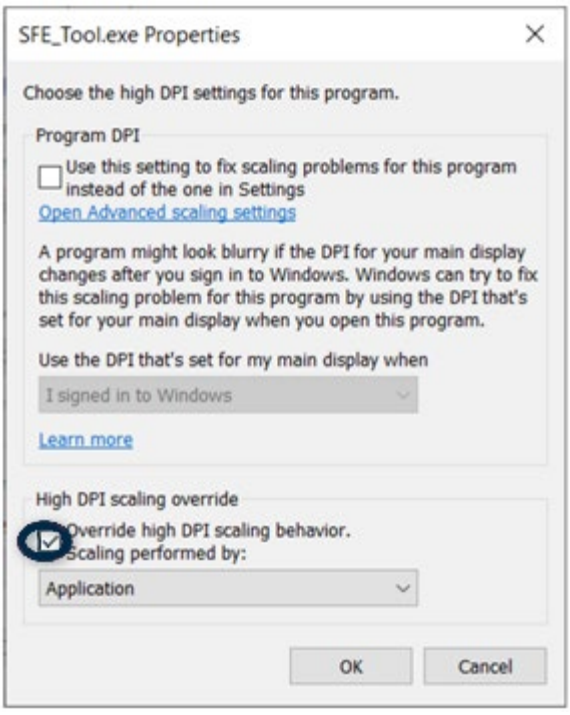
7 **Programming kit for the commissioning**

5.1 Screen settings

Note

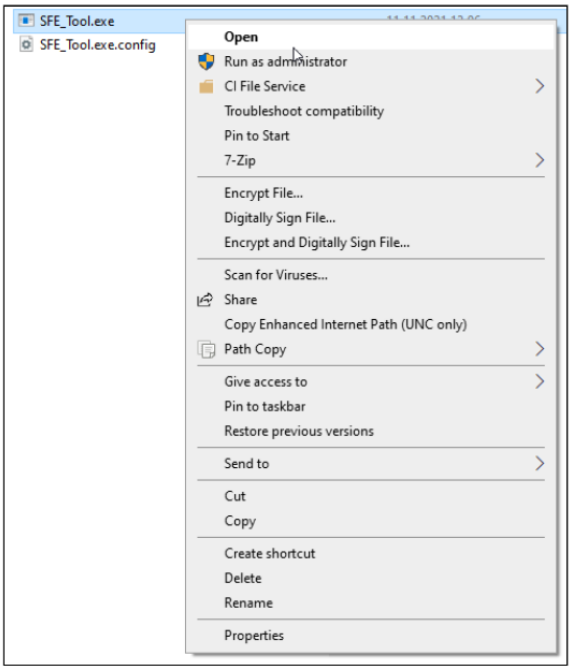
If the SFE tool cannot be shown completely on the screen of a few devices, this problem can be rectified using the settings explained in the following.

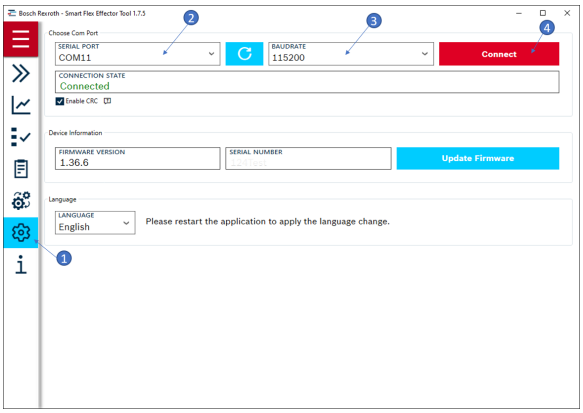
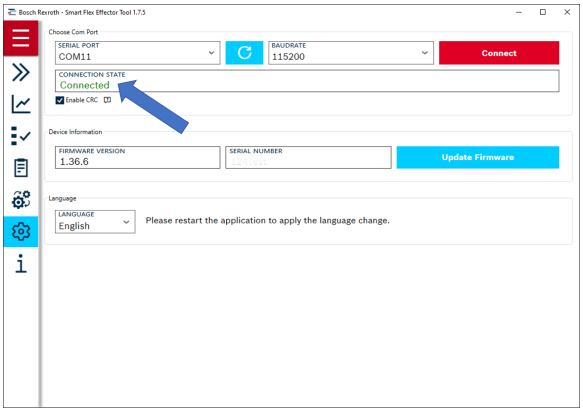
Screen	Action steps
	➢ Right-click on **SFE_Tool.exe**. → The dialog window opens. ➢ Open the **Properties** in the bottom section of the window.

Screen	Action steps
	➢ Click on **Change high DPI settings**.
	➢ Activate the checkbox in the **Override** field.

6 **Screen settings**

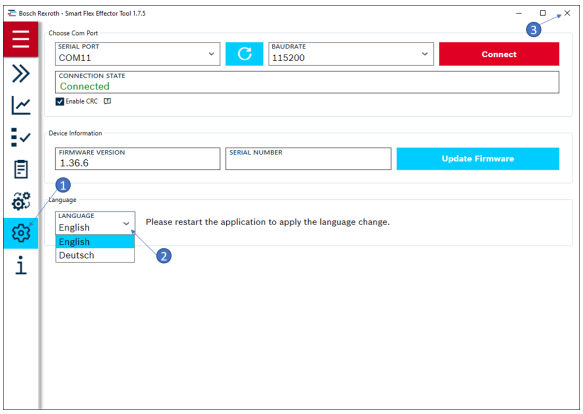
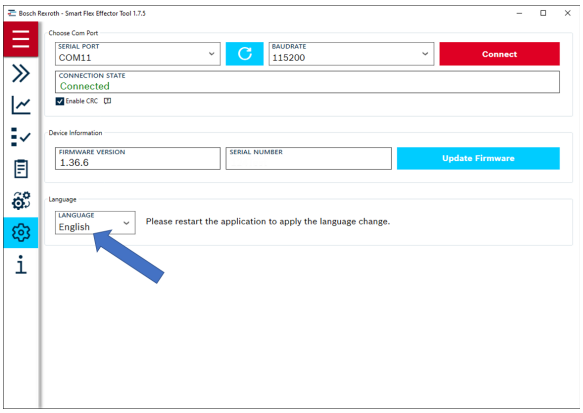
5.2 Setting up the connection

Screen	Action steps
	➢ Connect the SFE with the Windows computer via a USB-A adapter. ☝ The following RS485 adapter has been tested and may be used to connect the SFE to the PC. • Digitus DA-70157. ***Note*** *Other RS485 adapters may behave differently. Reconfiguration in the device manager may be necessary. The settings required for this vary depending on the drivers and can therefore not be stipulated with certainty.*
	➢ Open the **SFE tool**.

Screen	Action steps
	➢ Select a port. ➢ Navigate to the **Settings** (1) tab. ➢ Select **COM port** (2). When selecting, be aware that a different port applies depending on the device. ➢ Set the **Baudrate** to Auto (3). ➢ Click **Connect** (4). ***Note*** *The CRC check of the commands is activated by default. This setting can be retained.*
	→ If the connection was successful, the color of the writing under “CONNECTION STATE” “Connected – Version xyz” changes to green. → The baud rate is adjusted automatically. → New functions are available on the left hand side. ➢ Open the function bar via the **list symbol** on the top left.

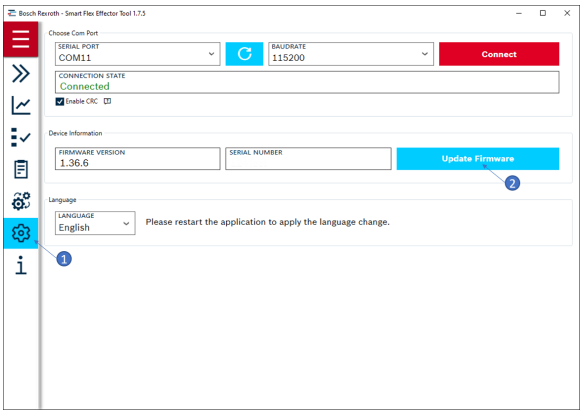
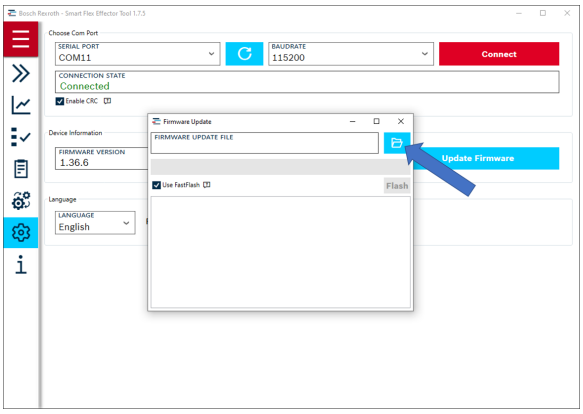
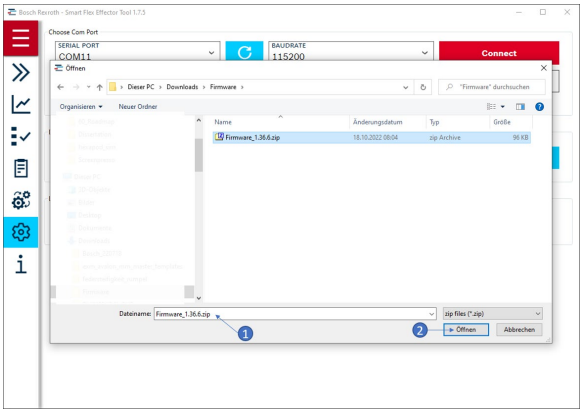
7 **Overview of SFE tool**

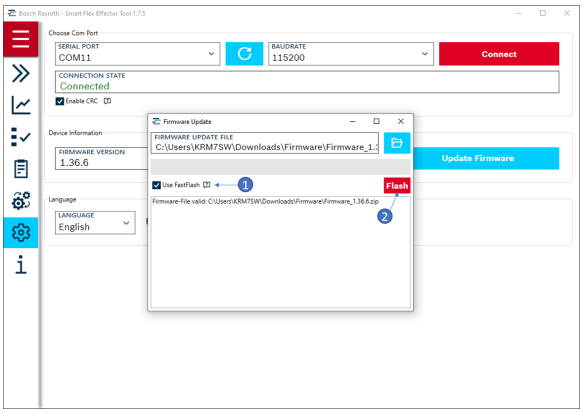
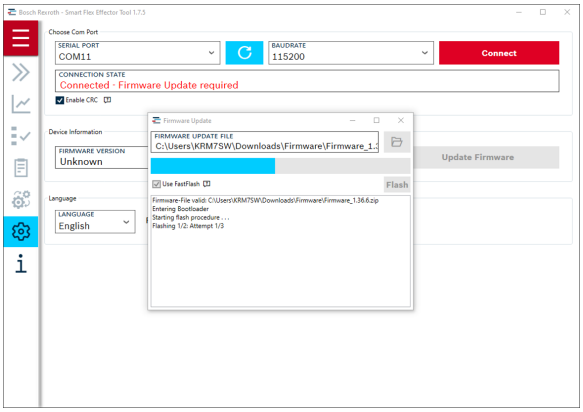
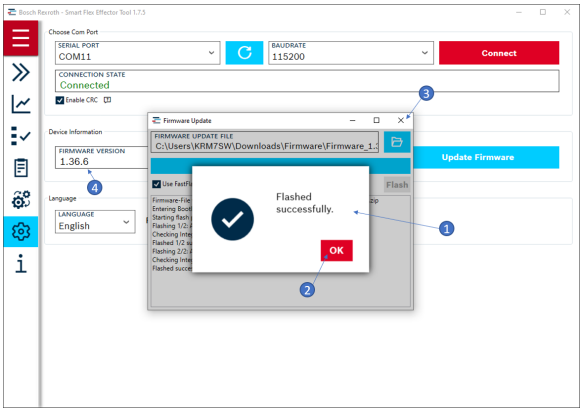
5.3 Changing the language

Screen	Action steps
	➢ Navigate to the **Settings** (1) tab. ➢ Select the desired language in the drop-down menu (2). ➢ Close the SFE tool (3). ➢ Restart the SFE tool.
	→ You can see the changed language settings when it restarts.

8 **Changing the language**

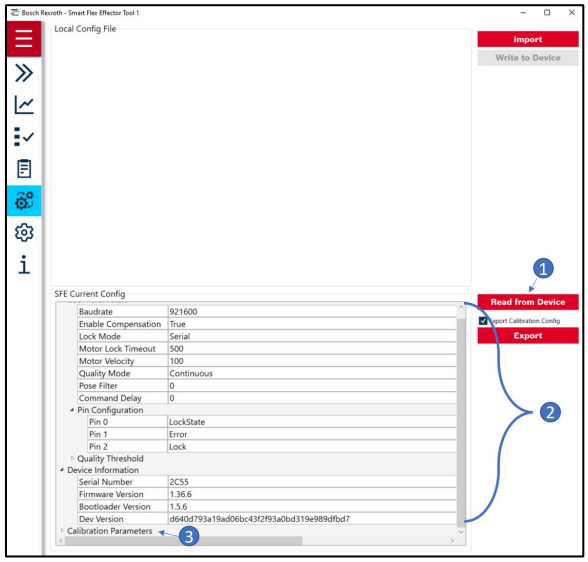
5.4 Updating the firmware

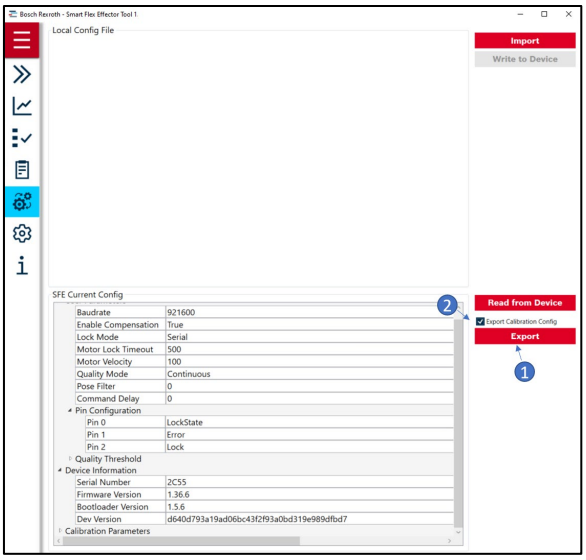
Screen	Action steps
	➢ Navigate to the **Settings** (1) tab. ➢ Click on **Update Firmware** (2).
	➢ Click on the folder symbol to select the firmware file. → An Explorer view opens.
	➢ Select the Firmware file (**zip folder**) (1). ➢ Click **Open** (2).

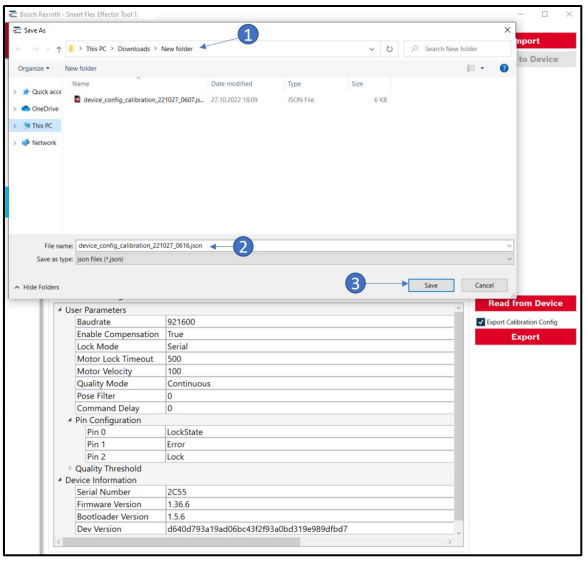
Screen	Action steps
	☝ The **FastFlash** (1) setting is activated as standard. Keep this as it is to speed up the firmware update. ***Note*** *If there are Flash problems, **FastFlash** can be deactivated. Problems may occur in particular because of high baud rates and long connection cables.* ➢ Click on **Flash** (2).
	→ The firmware update process starts. ***Note*** *During the Flash process, the power supply of the SFE unit must not be interrupted.*
	→ Once the firmware update is complete, **Flashed successfully** is shown as a message (1). ➢ Confirm with **OK** (2). ➢ Close the Firmware Update window with **X** (3). ➢ Check the **FIRMWARE VERSION** (4).

9 **Updating the firmware**

5.5 Export SFE configuration

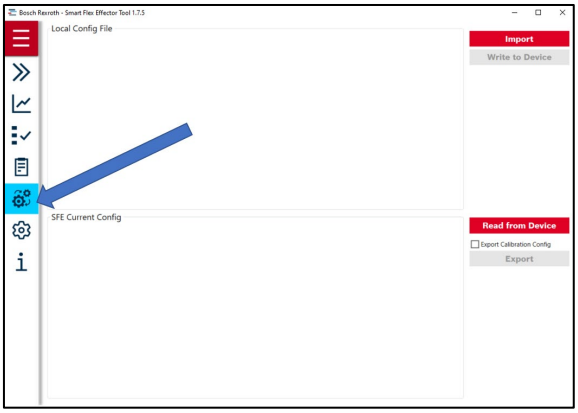
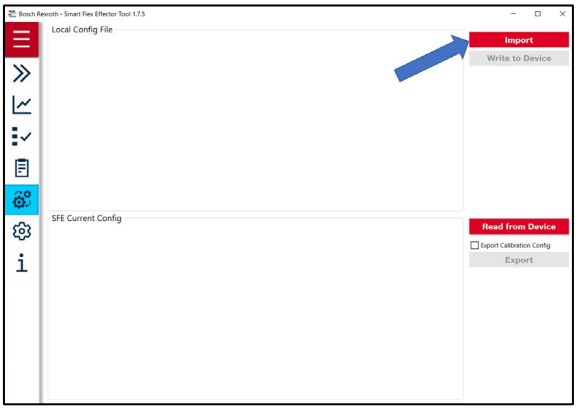
Screen	Action steps
	➢ Navigate to the **SFE config** tab.
	➢ Click **Read from Device** (1). → The configuration data is shown (2). ***Note*** *The calibration parameters for each of the six position sensors can be shown by clicking on **Calibration Parameters** (3).*

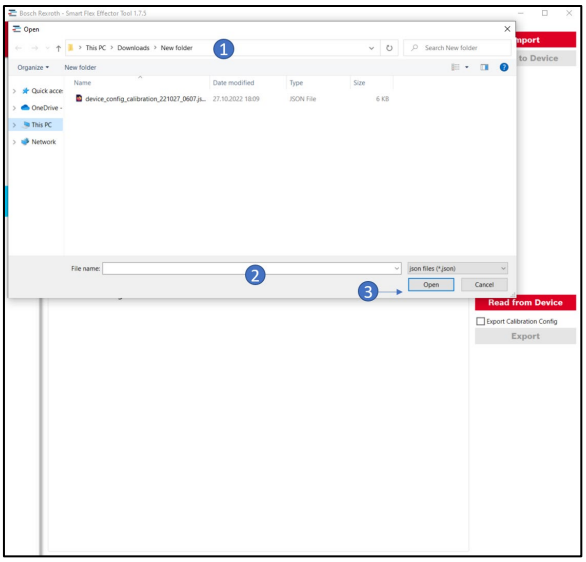
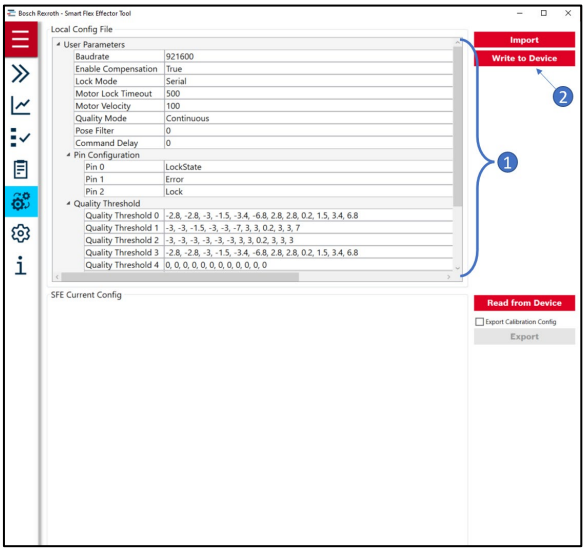
Screen	Action steps
	➢ Click **Export** (1). → The export is started. The SFE configuration can be exported without calibration data (checkbox next to **Export Calibration Config** (2) deactivated) or with calibration data (checkbox next to **Export Calibration Config** (2) activated). • Exporting without calibration data: Only fundamental user settings are exported (Baudrate, LED Quality Threshold, Motor Lock Timeout, Motor Velocity, Temperature Compensation, Lock Mode). • Exporting with calibration data: All configuration data is exported, including the motor and calibration configuration performed at the factory. ***Note*** *An export file with calibration data can only be imported in calibration mode. As calibration mode is only available for calibration at the factory and service tasks, only the user settings can be imported from an export file with calibration data.*

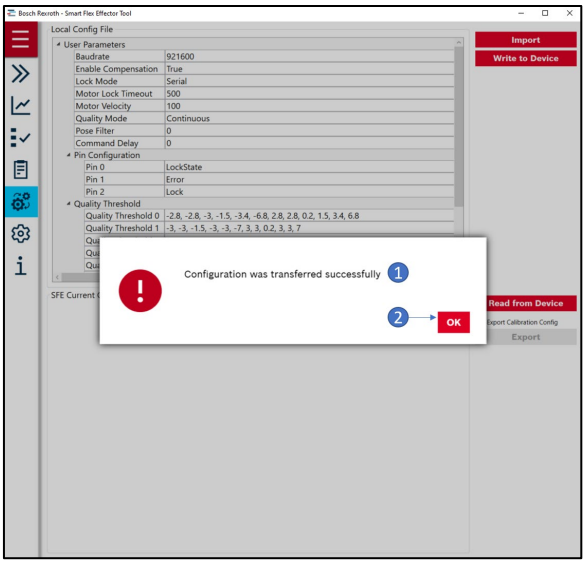
Screen	Action steps
	→ A path (1) to **save** the export file in JSON format can be selected. ***Note*** *We recommend that you do not change the name of the export file as the suggested file name contains information about the scope of data, save date and time (2).* ➢ Click **Save** (3) to save the configuration file.

10 **Export SFE configuration**

5.6 Import SFE configuration

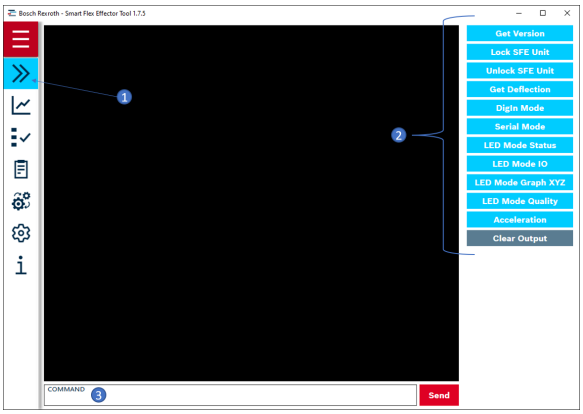
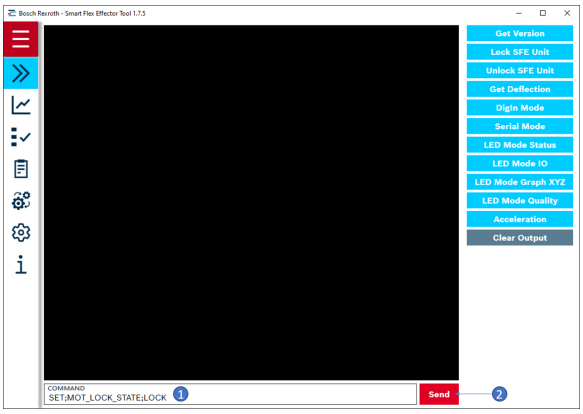
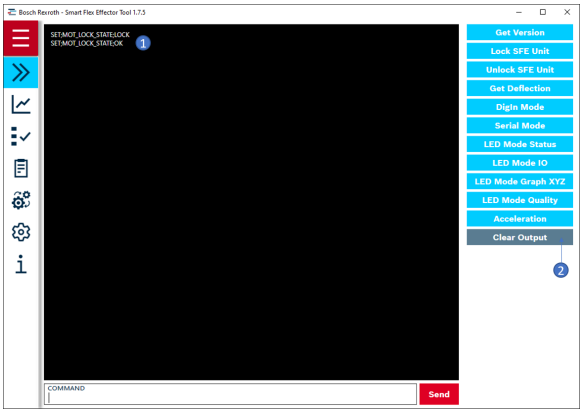
Screen	Action steps
	➢ Navigate to the **SFE Config** tab.
	➢ Click **Import**. → The import of configuration data is started.

Screen	Action steps
	➢ A **file path** (1) and an **import file in JSON format** stored in the path can be selected (2). ➢ Click **Open** (3). → The import is displayed.
	➢ Check the data to be imported (1). ➢ Click **Write to Device** (2) to import the data.

Screen	Action steps
	→ The successful import is confirmed by the message **Configuration was transferred successfully** (1). ➢ Click **OK** (2) to acknowledge the message.

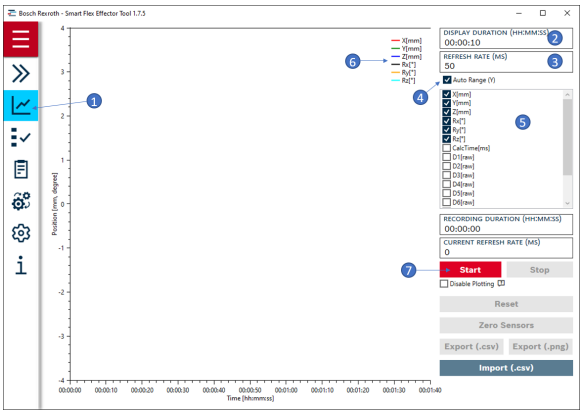
11 **Import SFE configuration**

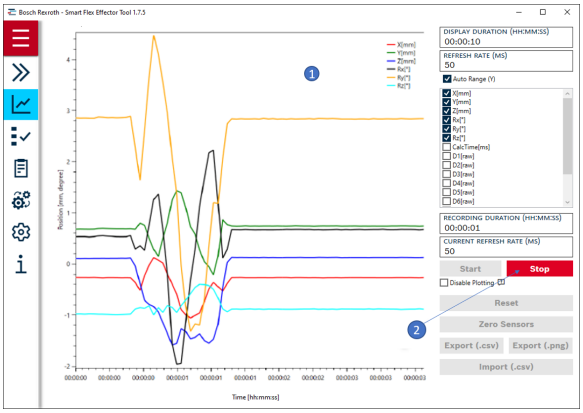
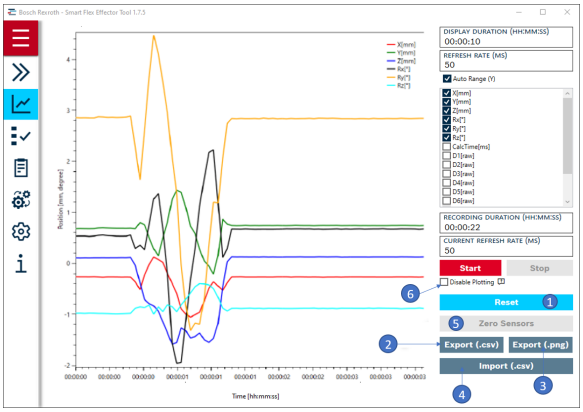
5.7 Using the console

Screen	Action steps
	➢ Navigate to the **Console** (1) tab. → Here, there are **buttons for selected commands** (2) as well as a **line for entering commands via keyboard** (3).
	➢ Commands can be entered via the **command line** (1). ➢ Clicking **Send** (2) sends the commands. ➢ Commands which have already been entered can be entered again using the Up arrow key.
	→ The sent command and the returned values are shown (1). ➢ Click **Clear Output** to empty the screen (2).

12 **Using the console**

5.8 Recording measurements

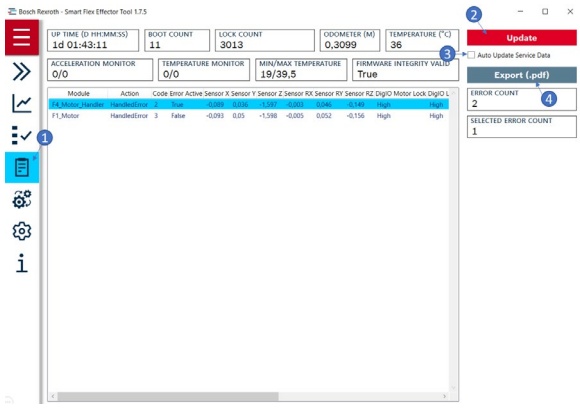
Screen	Action steps
	➢ Navigate to the **Graph** (1) tab. ➢ If necessary, adjust the **DISPLAY DURATION** (2) and **REFRESH RATE** of the display (3). ***Note*** *If the checkbox at "**Auto Range (Y)**" (4) is activated, the minimum and maximum values of the Y axis are adjusted to the current measurement.* ➢ Use the checkboxes (5) to select the signals that should be displayed. You can choose from: • X[mm]: Deflection in X direction in mm • Y[mm]: Deflection in Y direction in mm • Z[mm]: Deflection in Z direction in mm • Rx[°]: Deflection in Rx direction in ° • Ry[°]: Deflection in Ry direction in ° • Rz[°]: Deflection in Rz direction in ° • CalcTime[ms]: Required time for calculating the measured values in mm/° from the raw values • D1[raw]: Raw value sensor 1 in counts • D2[raw]: Raw value sensor 2 in counts • D3[raw]: Raw value sensor 3 in counts • D4[raw]: Raw value sensor 4 in counts • D5[raw]: Raw value sensor 5 in counts • D6[raw]: Raw value sensor 6 in counts • Temperature[°C]: Actual temperature in °C • Acceleration X [g]: Actual acceleration value in X direction in g (=9.81 m/s2) • Acceleration Y [g]: Actual acceleration value in Y direction in g (=9.81 m/s2)

Screen	Action steps
	• Acceleration Z [g]: Actual acceleration value in Z direction in g (=9.81 m/s2) → The selected measured values appear in the recording area including the color of the graph in the form of a key (6). ➢ Click **Start** (7) to start the measurement.
	➢ If necessary, click **Stop** (2) to terminate an ongoing measurement (1).
	➢ If necessary, click **Reset** (1) to clear the shown measurement. ➢ If necessary, click **Export (.csv)** (2) to save the measurement in csv format. ➢ If necessary, click **Export (.png)** (3) to save an image of the measurement in png format. ➢ If necessary, click **Import (.csv)** (4) to show past measurements. ➢ If necessary, click on **Zero Sensors** (5) to remove the offset of the measured values. ***Note*** *This option is not available in the default settings. This is because, as part of the temperature compensation in the locked state, the measured values are compensated depending on the ambient temperature at which the offset is removed. Even if the temperature compensation is switched off, it is not possible to only zero the sensors in the locked state and for a measurement which is not ongoing.* ➢ If necessary, activate the checkbox **Disable plotting** (6) for long-term measurements to deactivate the measurement display. The measurement continues in the background.

Screen	Action steps
	Note *This option should be used for long-term measurements, as otherwise this can lead to too high a load when visualizing the measurement data.*

13 **Recording measurements**

5.9 Viewing and exporting service data

Screen	Action steps
	➢ Navigate to the **Service** (1) tab. ➢ Click **Update** to open the error log (2). ☝ Selecting the **Auto Update Service Data** checkbox continuously updates the service data (first line in the screen - runtime, etc.) (3). ➢ If necessary, click **Export (.pdf)** (4) to export the service data. **Note** *If the baud rates are low, updating the error log can take longer.*

14 **Viewing and exporting service data**

5.10 Using the testcases

The structure of Testfiles in .json format is explained using the following examples.

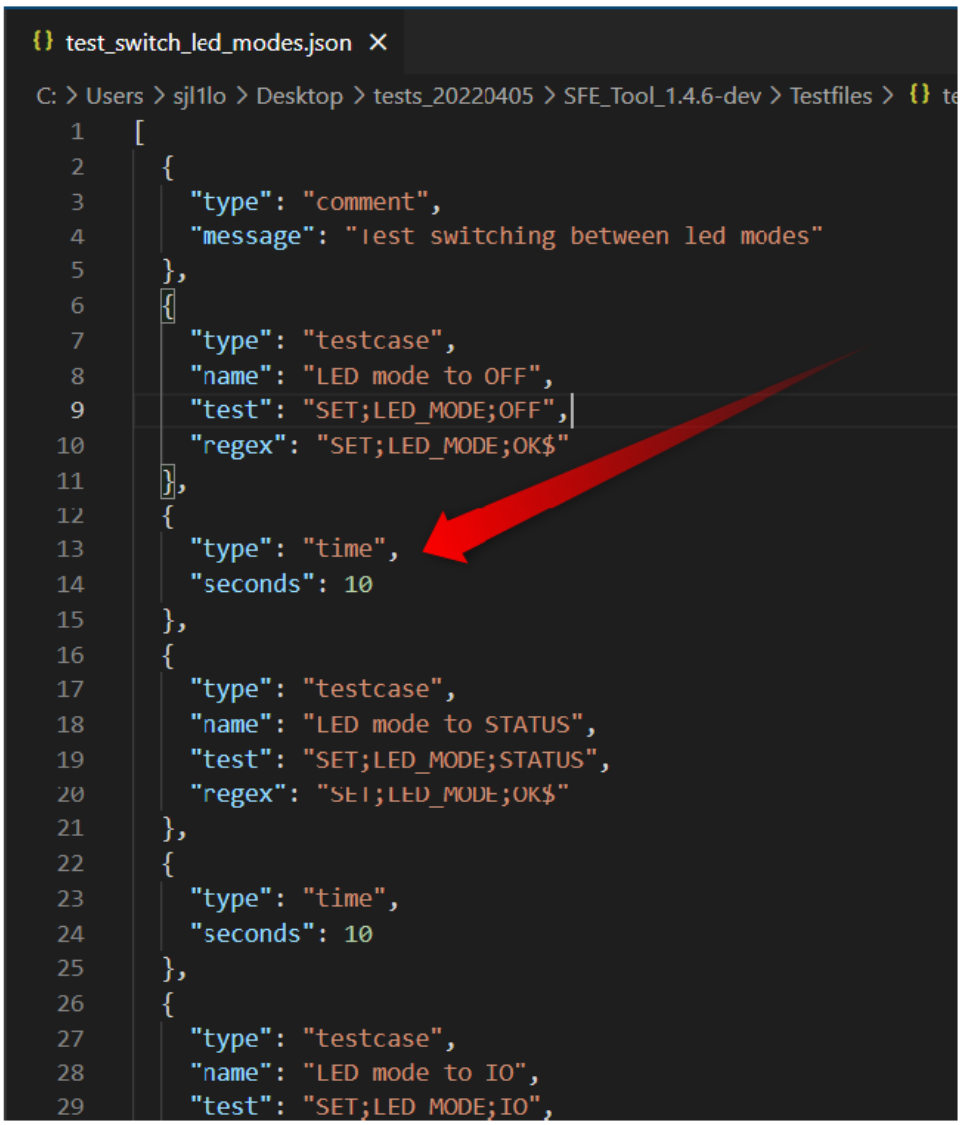
test_get_commands.json

C: > Users > sjl1lo > Desktop > tests_20220405 > SFE_Tool_1.4.6-dev > Testfiles > test_get_commands.json > 3

```
[
  {
    "type": "comment",
    "message": "Test get commands"
  },
  {
    "type": "testcase",
    "name": "Get the firmware version",
    "test": "GET;VERSION",
    "regex": "GET;VERSION;\\d+[.]\\d+[.]\\d+"
  },
  {
    "type": "testcase",
    "name": "Get the serial number",
    "test": "GET;SNO",
    "regex": "GET;SNO;\\w+$"
  },
  {
    "type": "testcase",
    "name": "Get the current baudrate",
    "test": "GET;BAUD",
    "regex": "GET;BAUD;(9600|38400|115200|921600)$"
  },
  {
    "type": "testcase",
    "name": "Get the lock state",
    "test": "GET;MOT_LOCK_STATE",
    "regex": "GET;MOT_LOCK_STATE;(LOCKED|UNLOCKED|TIMEOUT|RUNNING|ERROR)$"
  },
  {
    "type": "testcase",
    "name": "Get the lock timeout",
    "test": "GET;MOT_LOCK_TIMEOUT",
    "regex": "GET;MOT_LOCK_TIMEOUT;\\d+$"
  },
  {
```

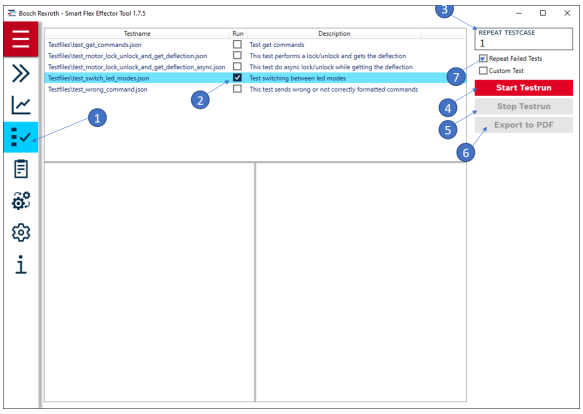
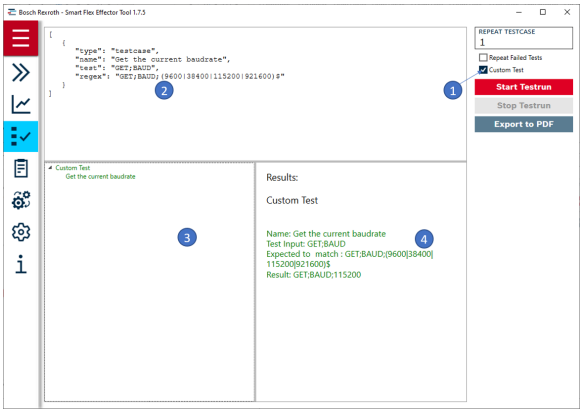
8 **Using the testcases**

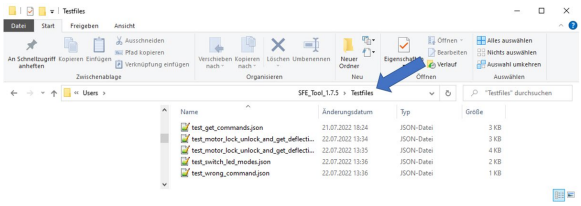
The entire test process must be placed in square brackets. Individual test stages are written in curly brackets and separated from one another using commas. After "type": the user sets whether this is a comment or a test stage. Once the designation of the test stage has been stated after "name":, the command to be tested can be determined after the label "test":. There are two ways to check the return of the command. With the variant with the field “expected:”, the return is checked for exact compliance with the content after “expected:”. With the variant with “regex:” as in the example above, the content after “regex:” is interpreted as a regular expression and thus the return is evaluated. If these match, the test was successful and appears in green in the SFE tool. If the return does not match the expected result, the test has failed and is marked red.



9 **Using the testcases – "type" type**

A defined delay of the subsequent test stages can be implemented using the "type" type.

Screen	Action steps
	➢ Navigate to the **Test** (1) tab. ➢ Select the desired testcase using the **checkboxes** (2). ➢ Set the desired **number of repetitions** (3). ➢ Click **Start Testrun** (4) to start the tests. ➢ If necessary, click **Stop Testrun** (5) to interrupt the test during execution. ➢ If necessary, click **Export (.pdf)** (6) to export the test results in pdf format. ☝ By activating the checkbox **Repeat Failed Tests** (7), only the failed tests are repeated; successful tests are only performed once. ***Note*** ☝ *Additional testcases can be implemented and saved in .json format in the folder structure of the software tool. The user-specific tests then also appear in the lists of tests which can be selected.*
	☝ When you activate the checkbox **Custom Test** (1), individual testcases can be written within the GUI (2). ***Note*** ☝ *Important information or note regarding handling* → The tests carried out (3) and their results (4) can be viewed in the bottom section.

Screen	Action steps
	The selectable test cases are stored in the **Testfiles** folder. In this folder, you can also create your own Testfiles. These can be selected and opened via the tool after restarting the SFE tool.

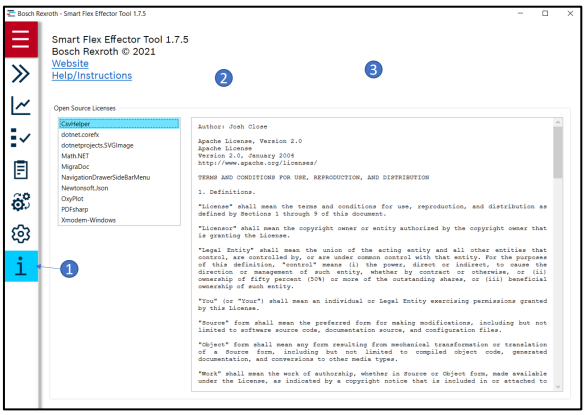
15 **Using the testcases**

5.11 Calling the SFE website

Screen	Action steps
	➢ Navigate to the **About** (1) tab. ➢ Click the **Website** link (2) to go to the SFE's main website. or ➢ Click **Help/Instructions** (3) to go directly to the help page with SFE documents. → The corresponding website opens in the browser.

16 **Calling the SFE website**

5.12 Viewing open source license texts

Screen	Action steps
	➢ Navigate to the **About** (1) tab. ➢ Select a package (2) from the **list of all open source software packages used in the SFE tool GUI**. → The related **license text** is shown (3).

17 **Viewing open source license texts**

6 Interface description

6.1 Pin assignment

The following part describes the pin assignment of the supplied cable:

10 **Cable**

Pin	Wire color	Signal	Input/output	Description
1	White	RS485+	Input / output	RS485 communication: non-inverted signal
2	Brown	RS485-	Input / output	RS485 communication: inverted Signal
3	Green	Ground (I/O)	Power supply	IO interface mass
4	Yellow	IO pin 0	Input / output	Pin 0 of the configurable digital IO pins (see chapter Use and configuration of the digital IO pins). Default assignment: Lock status (DIGIO_LOCK_STATE) • low = unlocked • high = locked
5	Grey	IO pin 1	Input / output	Pin 1 of the configurable digital IO pins (see chapter Use and configuration of the digital IO pins). Default assignment: Error output (DIGIO_ERROR)

Pin	Wire color	Signal	Input/output	Description
6	Pink	IO pin 2	Input	Pin 2 of the configurable digital IO pins (see chapter Use and configuration of the digital IO pins). Default assignment: Triggering of locking and unlocking (DIGIO_LOCK) • low = unlock • high = lock
7	Blue	0 V	Power supply	Power supply voltage mass
8	Red	24 V	Power supply	Power supply voltage

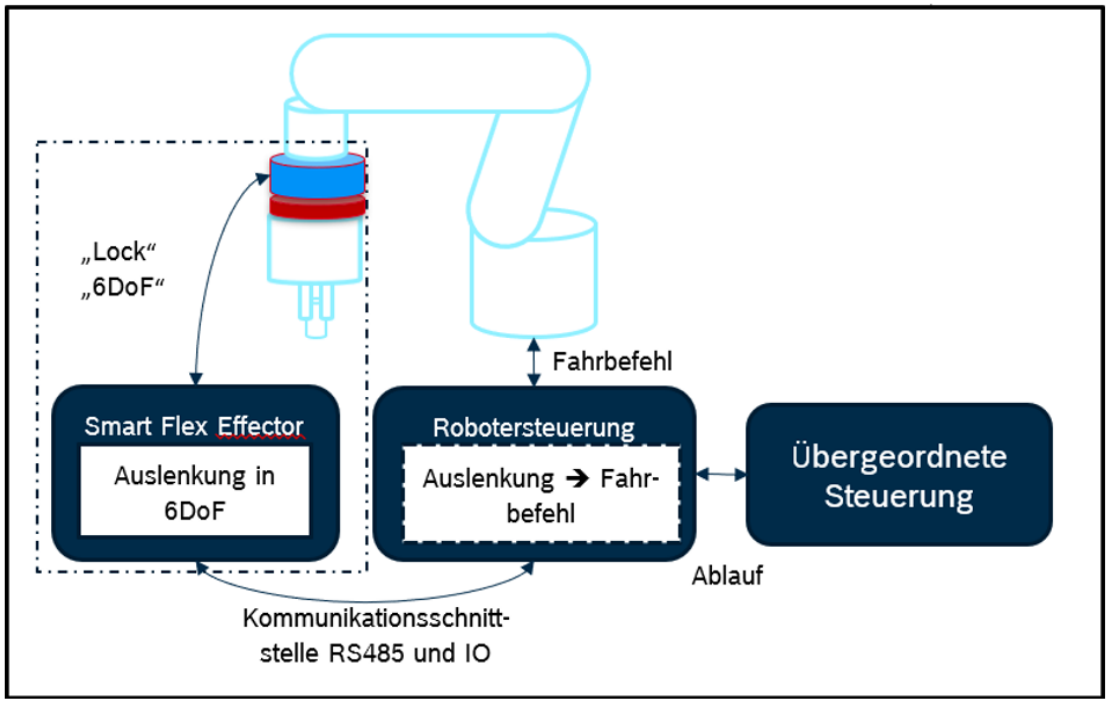
18 **Cable pin assignment**

Key data for digital inputs	Specification
Vil (Volt in low)	< 3.3 V
Vih (Volt in high)	> 9.7 V

19 **Key data for digital inputs**

6.2 Protocol

6.2.1 System connection and application concept



11 **System connection figure**

The above figure shows a schematic connection of the SFE to a robot controller. For this, the SFE is mounted on the robot flange. Then, various grippers can be mounted onto the SFE with ISO 9409-1 31.5-4-M5, or with the adapter plates supplied. The SFE communicates via a serial interface (RS-485) and digital inputs/outputs with a controller.

The SFE measures a displacement for the 6 degrees of freedom (X, Y, Z, RX, RY, RZ) which is caused by the contact between the gripper and a component. The SFE deflection is read by the robot controller via the communication interface and can be processed there, in order to align the robot correctly over the component being gripped, for instance.

The SFE can be locked via the RS-485 interface or via a digital input, i.e. it loses its flexibility. Locking resets the deflection to zero, fixing in a deflected position is not possible. Locking is required, for example, for dynamic movement between two positions.

An LED ring provides status information on the SFE, the various LED modes are explained in more detail in the chapter Range of functions.

6.2.2 Serial interface RS-485

Communication takes place via a serial interface (RS-485). To simplify commissioning and incorporation into the customer's application, the SFE uses a text-based communication protocol. This interface is used to communicate using ASCII strings. Each string transmission ends with a line feed (/n oder LF). The various commands are explained in more detail in the Range of functions section. To familiarize yourself with the command set, we recommend establishing a connection between SFE and the “SFE Tool” and sending a few commands manually.

When the SFE is started up for the first time, a baud rate of 38400 is set as standard. If the baud rate is changed during operation, this is saved. When you restart, the newly set baud rate is active, it is not reset to 38400.

Settings for the first connection setup:

You must select the following parameters for the connection setup

- Baud rate -> 38400 (may vary after the initial startup)
- Bits -> 8
- Parity -> none
- stopBits -> 1
- timeout -> 0

6.3 Structure of the commands

6.3.1 Fundamental structure of the commands

The command set is divided into three groups: GET, SET and CTR. These groups are subdivided once more into the areas: locking, sensors, diagnostics, LED modes, IO pins and system.

GET commands allow you to query parameters and states, e.g. whether the SFE is locked. The SET commands allow you to set parameters and statuses and execute commands to change the state of the SFE unit, e.g. to trigger locking of the unit.

CTR commands trigger actions (e.g. restarting the SFE unit), however parameter values are not changed by CTR commands.

In principle, the commands are structured as follows:

- GET commands: GET;COMMAND;PARAMETER
- SET commands: SET;COMMAND;VALUE (when setting several values, these are separated by |)
- CTR commands: CTR;ACTION;PARAMETER

For GET and CTR commands, the PARAMETER part is often omitted.

After each sent command, the SFE unit sends a response which depends on the sent command type.

For GET commands, the response contains the queried values next to the executed command. If a SET or CTR type command was executed, the response contains an ENUM which states whether the command execution was successful.

6.3.2 CRC checksum

The **cyclic redundancy check** (**CRC**) is a method for determining a checksum for data in order to identify errors during transmission or saving. As an option, the SFE also allows you to use this procedure to ensure the correctness of the data which is sent by the user to the SFE unit, or from the SFE unit to the user.

Modbus CRC16 is used as the checksum.

Using the checksum when communicating via serial interface

The data (responses) from the SFE unit always contain a checksum which depends on the actual response value. The response value is separated by a "!". Example: GET;BAUD;921600!37**EF**

With this checksum, the data received by the user can be checked for transmission errors. This check is optional. The checksum can be calculated by the user. Example function in C (Copyright (c) 1999-2016 Lammert Bies - https://github.com/lammertb/libcrc, license text: MIT License in the chapter Third-party license information):

Checksum Modbus CRC16

```
1    /*
2    * uint16_t crc_modbus( const unsigned char *input_str, size_t num_bytes );
3    *
4    * The function crc_modbus() calculates the 16 bits Modbus CRC in one pass for
5    * a byte string of which the beginning has been passed to the function. The
6    * number of bytes to check is also a parameter.
7    */
8
9    uint16_t crc_modbus( const unsigned char *input_str, size_t num_bytes ) {
10
11           uint16_t crc;
12           const unsigned char *ptr;
13           size_t a;
14
15           if ( ! crc_tab16_init ) init_crc16_tab();
16
1717         crc = CRC_START_MODBUS;
18           ptr = input_str;
19
20           if ( ptr != NULL ) for (a=0; a<num_bytes; a++) {
21
22           crc = (crc >> 8) ^ crc_tab16[ (crc ^ (uint16_t) *ptr++) & 0x00FF ];
23           }
24
25           return crc;
26
27   }               /* crc_modbus */
```

The checksum calculated by the user can now be compared with the checksum sent by the SFE unit. Examples:

Command	Response from SFE unit	Checksum calculated by the user	Result of the check
GET;BAUD	GET;BAUD;921600 ! 37EF	crc_modbus(GET;BAUD;9 2 1600) = 0x37EF	The checksum calculated by the user is identical to the checksum sent by the SFE unit. This means that the data transfer was without error.
GET;BAUD	GET;BAUD;921601 ! 37EF	crc_modbus(GET;BAUD;9 2 1601) = 0xF72E	The checksum calculated by the user does not match the checksum sent by the SFE unit. The data transfer contained an error.

20 **Examples of calculated checksums**

If the user sends commands to the SFE unit via the serial interface, they can add the checksum Modbus CRC16 if they wish. The SFE unit then checks the transmitted data from the user for errors. If there is a transmission error, the SFE unit responds with an error message. Examples:

Command from the user including CRC checksum	Response from SFE unit	Result of the check
GET;LOCK_MODE!FD83	GET;LOCK_ MODE;SERI AL!811C	The checksum calculated by the user is identical to the checksum calculated by the SFE unit. This means that the data transfer was without error.
GET;LOCK_MODE!FD83	ERROR;CRC! D1C4	The checksum calculated by the user does not match the checksum calculated by the SFE unit. The data transfer contained an error.

21 **Examples of calculated checksums for serial interfaces**

6.3.3 Handling the checksum when working with the SFE tool

As standard, the SFE tool attaches the checksum Modbus CRC16 to every command and checks the response of the SFE unit for correctness using the checksum.

Important note!

The checksum which is attached to the command by the SFE tool is not displayed in the SFE tool console. In addition, the checksum of the response attached by the SFE unit is also not shown in the console.

6.4 GET functions

6.4.1 Locking

Functional description	Querying the lock state. An example application case is a query whether the SFE is locked before the robot will move.		
Transmission example	1	GET;MOT_LOCK_STATE	Calling the lock state
Reception example	1	GET;MOT_LOCK_STATE;LOCKED	SFE unit is locked
Description of response parameters	Type		Description
	ENUM(LOCKED,UNLOCKED,TIMEOUT, RUNNING, ERROR)		LOCKED=SFE is locked, UNLOCKED=SFE is unlocked, TIMEOUT=A timeout occurred during the last locking/unlocking, RUNNING=Locking/unlocking is in progress, ERROR=An error occurred during the last locking/unlocking
Available from FW version	1.27.5		

Functional description	Querying the lock mode		
Transmission example	1	GET;MOT_LOCK_MODE	Calling the lock mode
Reception example	1	GET;LOCK_MODE;SERIAL	Lock mode is serial
Description of response parameters	Type		Description
	ENUM(SERIAL, DIGIN)		SERIAL=Locking can only be changed via the serial interface, DIGIN=Locking can only be changed via IO pins
Available from FW version	1.27.5		

Functional description	Querying the maximum duration for a locking/unlocking procedure. If the locking procedure is not completed within the time, an entry is made in the error history and the error state of the SFE is activated		
Transmission example	1	GET;MOT_LOCK_TIMEOUT	Calling the maximum duration for a locking/unlocking procedure
Reception example	1	GET;MOT_LOCK_TIMEOUT;500	Receipt of a maximum duration for a 500ms locking/unlocking procedure
Description of response parameters	Type		Description
	UINT32		Maximum duration for a locking/unlocking procedure in ms
Available from FW version	1.27.5		

Functional description	Querying the persistently set velocity used for the SFE locking/unlocking		
Transmission example	1	GET;MOT_VEL	Calling the locking/unlocking speed
Reception example	1	GET;MOT_VEL;50	Receipt of a locking/unlocking speed of 50
Description of response parameters	Type		Description
	UINT32		Locking/unlocking speed (without unit)
Available from FW version	1.36.6		

6.4.2 Sensors

Functional description	Querying the deflection of the adapter plate and status		
Transmission example	1	GET;POSE	Querying the deflection of the adapter plate
Reception example	1	GET;POSE;0.171|0.157|-0.140|-0.072|0.461|-0.197;OK	Receipt of the deflection with status OK

Description of response parameters

Type	Description
FLOAT	X position(mm)
FLOAT	Y position(mm)
FLOAT	Z position(mm)
FLOAT	Rx position(°)
FLOAT	Ry position(°)
FLOAT	Rz position(°)
ENUM(OK, ERROR_FTOL, ERROR_GTOL, ERROR_XTOL, ERROR_MAX_ITERATIONS, ERROR_TAPPED, ERROR)	OK = Calculation of deflection with high accuracy successful, ERROR_FTOL = Calculation error(cancel tolerance ftol exceeded), ERROR_GTOL = Calculation error(cancel tolerance gtol exceeded), ERROR_XTOL = Calculation error(cancel tolerance xtol exceeded), ERROR_MAX_ITERATIONS = No valid result could be calculated within the maximum iterations, ERROR_TAPPED = calculation error(Calculation does not approach solution), ERROR = Calculation error(error during the calculation)

Available from FW version	1.27.5

Functional description	Querying the length of the moving average filter for actual position values		
Transmission example	1	GET;POSE_FILTER	Calling the length of the moving average filter
Reception example	1	GET;POSE_FILTER;10	Receipt of the length of the moving average filter of 10

Description of response parameters

Type	Description
UINT32	Length of moving average filter

Available from FW version	1.27.5

Functional description	Querying the activation of the temperature compensation.		
Transmission example	1	GET;ENABLE_COMPENSATION	Querying whether the temperature compensation is enabled
Reception example	1	GET;ENABLE_COMPENSATION; TRUE	Receipt that temperature compensation is enabled

Description of response parameters

Type	Description
ENUM(TRUE, FALSE)	TRUE=Compensation is enabled, FALSE=Compensation is disabled

Available from FW version	1.27.5

Functional description	Querying the actual acceleration values.
Transmission example	1 | GET;ACC | Calling the actual acceleration values
Reception example	1 | GET;ACC;0.035|-0.027|-1.053 | Receipt of the actual acceleration values
Description of response parameters	Type: FLOAT — Description: Acceleration value in X direction (unit g=9.81 m/s2) Type: FLOAT — Description: Acceleration value in Y-direction (unit g=9.81 m/s2) Type: FLOAT — Description: Acceleration value in Z-direction (unit g=9.81 m/s2)
Available from FW version	1.27.5

Functional description	Querying the absolute acceleration, corrected by the gravitational acceleration
Transmission example	1 | GET;ACC_ABS | Calling the absolute acceleration
Reception example	1 | GET;ACC_ABS;0.532 | Receipt of the absolute acceleration
Description of response parameters	Type: FLOAT — Description: Absolute acceleration (unit g=9.81 m/s2)
Available from FW version	1.36.6

Functional description	Query whether the data logger is active
Transmission example	1 | GET;LOGGER_RUNNING | Calling the logger status
Reception example	1 | GET;LOGGER_RUNNING;TRUE | Receipt that logger is active
Description of response parameters	Type: ENUM(TRUE,FALSE) — Description: TRUE=Logger is active, FALSE=Logger is inactive
Available from FW version	1.36.6

Functional description

Reading the data recorded by the logger

Transmission example

1	GET;LOGGER_DATA;2	Calling the two latest logger data records
2	GET;LOGGER_DATA	Calling all existing logger data records

Description of transmission parameters

Type	Value range	Optional	Description
UINT32	1-2000	Yes	Number of data series that are to be read out

Reception example

1	GET;LOGGER_DATA;39073919;0|-0.041|-0.455|0.782|-0.617|-2.521|1.454|0.037|-0.016|-1.088;5|-0.040|-0.453|0.782|-0.612|-2.528|1.452|0.039|-0.016|-1.078	Receipt of the two latest logger data records
2	GET;LOGGER_DATA;39073919;0|-0.041|-0.455|0.782|-0.617|-2.521|1.454|0.037|-0.016|-1.088;5|-0.040|-0.453|0.782|-0.612|-2.528|1.452|0.039|-0.016|-1.078;…;…	Receipt of all existing logger data records
3	GET;LOGGER_DATA;BUSY	Data recording in progress
4	GET;LOGGER_DATA;NONE	No data records recorded

Description of response parameters

Return variant 1

Type	Description
UINT32	Absolute timestamp (ms)
LIST(1-2000)	

Type	Description
UINT32	Time axis (ms)
FLOAT	X position(mm)
FLOAT	Y position(mm)
FLOAT	Z position(mm)
FLOAT	Rx position(°)
FLOAT	Ry position(°)
FLOAT	Rz position(°)
FLOAT	X acceleration(unit g=9.81 m/s2)
FLOAT	Y acceleration(unit g=9.81 m/s2)
FLOAT	Z acceleration(unit g=9.81 m/s2)

Return variant 2

Type	Description
ENUM(NONE, BUSY, ERROR)	NONE=No data records recorded, BUSY=Data recording in progress, ERROR=Command processing error

Available from FW version

1.36.6

6.4.3 Diagnostics

<table>
<tr><td>Functional description</td><td>Querying the last error from the error memory</td></tr>
<tr><td>Transmission example</td><td>
<table>
<tr><td>1</td><td>GET;ERROR_LAST</td><td>Querying the last error</td></tr>
</table>
</td></tr>
<tr><td>Reception example</td><td>
<table>
<tr><td>1</td><td>GET;ERROR_LAST;1|13|2|1|-0.009|0.113|-1.088|0.162|0.175|0.138|1|2|0|0|1|0|0|0|119596078</td><td>Reception of last error</td></tr>
<tr><td>2</td><td>GET;ERROR_LAST;NONE</td><td>No error in the error memory</td></tr>
</table>
</td></tr>
<tr><td>Description of response parameters</td><td>
Return variant 1:
<table>
<tr><td>Type</td><td>Description</td></tr>
<tr><td>UINT8</td><td>Error type (1= HandledError(Error is handled by the software), 2 = UnhandledError(Error could not be handled internally), 3 = CriticalError(Crtical error, the SFE can no longer perform any more functions)</td></tr>
<tr><td>UINT8</td><td>Error module (error module number)</td></tr>
<tr><td>UINT8</td><td>Error number (The number of the error in the relevant error module)</td></tr>
<tr><td>UINT8</td><td>Error active status (Indicates whether the error state is still active (0 = not active, 1 = active)</td></tr>
<tr><td>FLOAT</td><td>X position(mm)</td></tr>
<tr><td>FLOAT</td><td>Y position(mm)</td></tr>
<tr><td>FLOAT</td><td>Z position(mm)</td></tr>
<tr><td>FLOAT</td><td>Rx position(°)</td></tr>
<tr><td>FLOAT</td><td>Ry position(°)</td></tr>
<tr><td>FLOAT</td><td>Rz position(°)</td></tr>
<tr><td>UINT16</td><td>Motor speed</td></tr>
<tr><td>UINT8</td><td>Current motor state (SFE is locked = 0, SFE is unlocked = 1, A timeout occurred during locking/unlocking = 2, Locking/unlocking is in progress = 3, An error occurred during locking/unlocking = 4, The initial motor state could not be determined) = 254</td></tr>
<tr><td>UINT8</td><td>Motor target state (SFE is locked = 0, SFE is unlocked = 1, A timeout occurred during locking/unlocking = 2, Locking/unlocking is in progress = 3, An error occurred during locking/unlocking = 4, The initial motor state could not be determined = 254)</td></tr>
<tr><td>UINT16</td><td>RESERVED</td></tr>
<tr><td>UINT8</td><td>State of the digital IO pin 2 (LOW=0/HIGH=1).</td></tr>
<tr><td>UINT8</td><td>State of the digital IO pin 0 (LOW=0/HIGH=1).</td></tr>
<tr><td>UINT8</td><td>State of the digital IO pin 1 (LOW=0/HIGH=1).</td></tr>
<tr><td>UINT8</td><td>Bootloader Error Flag(Indicates whether the SFE was in the bootloader at the time of the error. (0=Firmware error, 1=Bootloader error)</td></tr>
<tr><td>UINT64</td><td>Error time(The time the error occurred in ms)</td></tr>
</table>
Return variant 2:
<table>
<tr><td>Type</td><td>Description</td></tr>
<tr><td>ENUM(NONE)</td><td>NONE=No error in the error memory</td></tr>
</table>
</td></tr>
<tr><td>Available from FW version</td><td>1.27.5</td></tr>
</table>

<table>
<tr><td>Functional description</td><td>Calling the error history. The last 100 errors are saved</td></tr>
<tr><td>Transmission example</td><td>
<table>
<tr><td>1</td><td>GET;ERROR_HISTORY</td><td>Calling the error history</td></tr>
</table>
</td></tr>
<tr><td>Reception example</td><td>
<table>
<tr><td>1</td><td>GET;ERROR_HISTORY;1|6|2|0|0.000|0.000|0.000|0.000|
0.000|0.000|100|3|0|0|1|1|0|0|1381367;1|7|21|0|0.000|
0.000|0.000|0.000| 0.000|0.000|1|4|4|0|0|0|0|0|119580058</td><td>Receipt of an error history with 2 errors</td></tr>
<tr><td>2</td><td>GET;ERROR_HISTORY;NONE</td><td>No errors in the error memory</td></tr>
</table>
</td></tr>
<tr><td>Description of response parameters</td><td>
Return variant 1
<table>
<tr><td>Type</td><td>Description</td></tr>
<tr><td>LIST
(1-100)</td><td>
<table>
<tr><td>Type</td><td>Description</td></tr>
<tr><td>UINT8</td><td>Error type (1= HandledError(Error is handled by the software), 2 = UnhandledError(Error could not be handled internally), 3 = CriticalError(Crtical error, the SFE can no longer perform any more functions)</td></tr>
<tr><td>UINT8</td><td>Error module (error module number)</td></tr>
<tr><td>UINT8</td><td>Error number (The number of the error in the relevant error module)</td></tr>
<tr><td>UINT8</td><td>Error active status (Indicates whether the error state is still active. 0 = not active, 1 = active)</td></tr>
<tr><td>FLOAT</td><td>X position(mm)</td></tr>
<tr><td>FLOAT</td><td>Y position(mm)</td></tr>
<tr><td>FLOAT</td><td>Z position(mm)</td></tr>
<tr><td>FLOAT</td><td>Rx position(°)</td></tr>
<tr><td>FLOAT</td><td>Ry position(°)</td></tr>
<tr><td>FLOAT</td><td>Rz position(°)</td></tr>
<tr><td>UINT16</td><td>Motor speed</td></tr>
<tr><td>UINT8</td><td>Current motor state (SFE is locked = 0, SFE is unlocked = 1, A timeout occurred during the last locking/unlocking = 2, Locking/unlocking is in progress = 3, An error occurred during locking/unlocking = 4, The initial motor state could not be determined = 254)</td></tr>
<tr><td>UINT8</td><td>Motor target state (SFE is locked = 0, SFE is unlocked = 1, A timeout occurred during the last locking/unlocking = 2, Locking/unlocking is in progress = 3, An error occurred during locking/unlocking = 4, The initial motor state could not be determined = 254)</td></tr>
<tr><td>UINT16</td><td>RESERVED</td></tr>
<tr><td>UINT8</td><td>State of the digital IO pin 2 (LOW=0/HIGH=1)</td></tr>
<tr><td>UINT8</td><td>State of the digital IO pin 0 (LOW=0/HIGH=1)</td></tr>
<tr><td>UINT8</td><td>State of the digital IO pin 1 (LOW=0/HIGH=1)</td></tr>
<tr><td>UINT8</td><td>Bootloader Error Flag(Indicates whether the SFE was in the bootloader at the time of the error. (0=Firmware error, 1=Bootloader error)</td></tr>
<tr><td>UINT64</td><td>Error time(The time the error occurred in ms)</td></tr>
</table>
</td></tr>
</table>
Return variant 2
<table>
<tr><td>Type</td><td>Description</td></tr>
<tr><td>ENUM(NONE)</td><td>NONE=No error in the error memory</td></tr>
</table>
</td></tr>
<tr><td>Available from FW version</td><td>1.27.5</td></tr>
</table>

6.4.4 LED modes

Functional description	Querying the active LED modes (For more details, see chapter Displaying various operating states (LED modes))		
Transmission example	1	GET;LED_MODE	Calling the LED modes
Reception example	1	GET;LED_MODE;STATUS	Return LED modes status
Description of response parameters	Type		Description
	ENUM(OFF, STATUS, IO, GRAPH_XYZ, QUALITY, SENSOR, CUSTOM)		OFF = The LEDs are switched off, STATUS = Graphically displays the lock mode, sensor status and lock status, IO = Graphically displays the levels of the digital IOs in addition to the status mode, GRAPH_XYZ = Graphically displays the deflection in X/Y/Z-direction, QUALITY = Graphically displays the quality mode, SENSOR = Graphically displays the sensor diagnosis, CUSTOM = Shows that a custom LED display is active)
Available from FW version	1.27.5		

Functional description	Reading out the active quality mode.		
Transmission example	1	GET;QUALITY_MODE	Calling the active quality mode
Reception example	1	GET;QUALITY_MODE;QUALITY_TRIGGER	Return quality mode quality trigger
Description of response parameters	Type		Description
	ENUM(QUALITY_TRIGGER, QUALITY_CONTINUOUS, QUALITY_PEAK, QUALITY_OFF)		Set quality mode(QUALITY_TRIGGER = Quality check when triggering through digital inputs or serial interface, QUALITY_CONTINUOUS = Continuous quality check, QUALITY_PEAK = Continuous quality check which does not reset to OK in case of NOK, QUALITY_OFF = quality check switched off)
Available from FW version	1.36.6		

Functional description	Check whether quality check according to set limits was successful (For a more detailed description, see chapter Range of functions)		
Transmission example	1	GET;QUALITY_IN_RANGE	Calling the quality check
Reception example	1	GET;QUALITY_IN_RANGE;TRUE	Quality check OK
Description of response parameters	Type		Description
	ENUM(TRUE,FALSE)		Quality check(TRUE = quality check OK, FALSE = quality check NOK)
Available from FW version	1.36.6		

<table>
<tr><td>Functional description</td><td>Reading a quality mode threshold data record (For more details see chapter Quality mode)</td></tr>
<tr><td>Transmission example</td><td>
<table>
<tr><td>1</td><td>GET;QUALITY_THRES</td><td>Calling the active data record</td></tr>
<tr><td>2</td><td>GET;QUALITY_THRES;3</td><td>Calling the data record at index 3</td></tr>
</table>
</td></tr>
<tr><td>Description of transmission parameters</td><td>
<table>
<tr><td>Type</td><td>Value range</td><td>optional</td><td>Description</td></tr>
<tr><td>UINT32</td><td>0-9</td><td>Yes</td><td>Index of the data record to be called</td></tr>
</table>
</td></tr>
<tr><td>Reception example</td><td>
<table>
<tr><td>1</td><td>GET;QUALITY_THRES;-2.800|-2.800|-3.000|-1.500|-3.400|-6.800|2.800|2.800|0.200|1.500|3.400|6.800</td><td>Receipt of a threshold data record</td></tr>
<tr><td>2</td><td>GET;QUALITY_THRES;ERROR</td><td>Error due to invalid index</td></tr>
</table>
</td></tr>
<tr><td>Description of response parameters</td><td>
Return variant 1
<table>
<tr><td>Type</td><td>Description</td></tr>
<tr><td>FLOAT</td><td>Lower limit translative displacement in x direction in mm</td></tr>
<tr><td>FLOAT</td><td>Lower limit translative displacement in y-direction in mm</td></tr>
<tr><td>FLOAT</td><td>Lower limit translative displacement in z-direction in mm</td></tr>
<tr><td>FLOAT</td><td>Lower limit rotary motion around x-axis in °</td></tr>
<tr><td>FLOAT</td><td>Lower limit rotary motion around y-axis in °</td></tr>
<tr><td>FLOAT</td><td>Lower limit rotary motion around z-axis in °</td></tr>
<tr><td>FLOAT</td><td>Upper limit translative displacement in x direction in mm</td></tr>
<tr><td>FLOAT</td><td>Upper limit translative displacement in y-direction in mm</td></tr>
<tr><td>FLOAT</td><td>Upper limit translative displacement in z-direction in mm</td></tr>
<tr><td>FLOAT</td><td>Upper limit rotary motion around x-axis in °</td></tr>
<tr><td>FLOAT</td><td>Upper limit rotary motion around y-axis in °</td></tr>
<tr><td>FLOAT</td><td>Upper limit rotary motion around z-axis in °</td></tr>
</table>
Return variant 2
<table>
<tr><td>Type</td><td>Description</td></tr>
<tr><td>ENUM(ERROR)</td><td>ERROR=Error due to invalid index or general error</td></tr>
</table>
</td></tr>
<tr><td>Available from FW version</td><td>1.27.5, function extension in 1.36.6</td></tr>
</table>

Functional description	Reading the index of the active set quality threshold data record.		
Transmission example	1	GET;QUALITY_ACTIVE_THRESH	Calling the index of the active threshold data record
Reception example	1	GET;QUALITY_ACTIVE_THRESH;1	Threshold data record at index 1 is active
Description of response parameters	Type		Description
	UNIT32		Index of the active threshold data record
Available from FW version	1.36.6		

6.4.5 IO pins

Functional description	Calling the pin configuration of a selected pin		
Note	IO pin 2 can only be configured as input. Input functions can only be assigned once.		
Transmission example	1	GET;IO_CONFIG;0	Calling of configuration with pin ID 0
Description of transmission parameters	Type	Value range	Description
	UINT32	0-2	IO pin ID
Reception example	1	GET;IO_CONFIG;IO_LOCK_STATE	Motor state
Description of response parameters	Type		Description
	ENUM(IO_LOCK_STATE, IO_LOCK, IO_ERROR, IO_OFF, IO_QUALITY_TRIGGER, IO_QUALITY_IN_RANGE, IO_INPUT, IO_OUTPUT)		IO_LOCK_STATE = Motor state, IO_LOCK = Lock/Unlock, IO_ERROR = Error state, IO_OFF = Off, IO_QUALITY_TRIGGER = Quality trigger, IO_QUALITY_IN_RANGE = Quality state, IO_INPUT = Configure pin as input, IO_OUTPUT = Configure pin as output
Available from FW version	1.36.6		

Functional description	Calling the pin state of a selected pin		
Transmission example	1	GET;IO_STATE;0	Calling of pin state with pin ID 0
Description of transmission parameters	Type	Value range	Description
	UINT32	0-2	IO pin ID
Reception example	1	GET;IO_CONFIG;HIGH	Return of pin state
Description of response parameters	Type		Description
	ENUM(HIGH, LOW)		HIGH = Set, LOW = Not set
Available from FW version	1.36.6		

6.4.6 System

Functional description	Calling the firmware version		
Transmission example	1	GET;VERSION	Calling of firmware version
Reception example	1	GET;VERSION;1.36.6	Return of version 1.36.6
Description of response parameters	Type		Description
	STRING		Version
Available from FW version	1.27.5		

Functional description	Calling the baud rate		
Transmission example	1	GET;BAUD	Calling the baud rate
Reception example	1	GET;BAUD;38400	Return of baud rate with value 38400
Description of response parameters	Type		Description
	UINT32		Baud rate in Baud
Available from FW version	1.27.5		

Functional description	Calling the serial number		
Transmission example	1	GET;SNO	Calling the serial number
Reception example	1	GET;SNO;20100000000	Return of the serial number
Description of response parameters	Type		Description
	STRING		Serial number
Available from FW version	1.27.5		

Functional description	Calling the system time		
Transmission example	1	GET;SYS_UP_TIME	Calling the system time
Reception example	1	GET;SYS_UP_TIME;11052230	Return of the system time of 11052230ms
Description of response parameters	Type		Description
	UINT64		Operating time in ms
Available from FW version	1.27.5		

Functional description	Calling the system temperature		
Transmission example	1	GET;SYS_TEMP	Calling the system temperature
Reception example	1	GET;SYS_TEMP;21.5	Return of the system temperature return of 21.5°C
Description of response parameters	Type		Description
	FLOAT		System temperature in °C
Available from FW version	1.27.5		

Functional description	Calling the number of locking/unlocking procedures		
Transmission example	1	GET;SYS_LOCK_COUNT	Calling the number of locking/unlocking procedures
Reception example	1	GET;SYS_LOCK_COUNT;1000	Return of the locking/unlocking procedures with value 1000
Description of response parameters	Type		Description1
	UINT32		Number of lock/unlock procedures
Available from FW version	1.27.5		

Functional description	Calling the response time delay		
Transmission example	1	GET;CMD_DELAY	Calling the response time delay
Reception example	1	GET;CMD_DELAY;1	Return of the response time delay with value 1
Description of response parameters	Type		Description
	UINT32		Set response delay in word length
Available from FW version	1.36.6		

Functional description	Calling the number of boot operations		
Transmission example	1	GET;SYS_BOOT_COUNT	Calling the number of boot operations
Reception example	1	GET;SYS_BOOT_COUNT;1000	Return of the number of boot operations
Description of response parameters	Type		Description
	UINT32		Number of boot operations
Available from FW version	1.27.5		

6.5 SET functions

6.5.1 Locking

Functional description	Persistent setting of the lock mode		
Transmission example	1	SET;LOCK_MODE;SERIAL	Setting the lock mode to serial control
Description of transmission parameters	Type		Description
	ENUM(SERIAL,DIGIN)		SERIAL=Locking/unlocking via serial interface, DIGIN=Locking/unlocking via digital inputs
Reception example	1	SET;LOCK_MODE;OK	Successful changing of the lock mode
Description of response parameters	ENUM(OK, ERROR)		OK = No error, ERROR = Error occurred
Available from FW version	1.27.5		

Functional description	Setting of motor state in asynchronous mode. The setting is confirmed directly and the use of the serial interface is possible again. The motor state changes in the background and must be checked manually if necessary.		
Note	The command can be overloaded with the locking speed, or with it and additionally the timeout. This makes it possible to set the motor state with frequently changing requirements without the parameters having to be set individually. Setting the motor state by overloading uses the parameters once without overwriting the stored values		
Transmission example	1	SET;MOT_LOCK_STATE_A;LOCK	Performs Lock with permanent velocity and permanent timeout
	2	SET;MOT_LOCK_STATE_A;UNLOCK|STORED	Performs Unlock with permanent velocity and timeout calculated therefrom
	3	SET;MOT_LOCK_STATE_A; LOCK|50	Performs Lock with transmitted velocity and timeout calculated therefrom
	4	SET;MOT_LOCK_STATE_A;UNLOCK|STORED|STORED	Performs Unlock with permanent velocity and permanent timeout
	5	SET;MOT_LOCK_STATE_A;LOCK|STORED|1800	Performs Lock with permanent velocity and transmitted timeout
	6	SET;MOT_LOCK_STATE_A;UNLOCK|90|STORED	Performs Unlock with transmitted velocity and permanent timeout
	7	SET;MOT_LOCK_STATE_A;LOCK|80|1800	Performs Lock with transmitted velocity and transmitted timeout

Description of transmission parameters	Type	Value range	optional	Description
	ENUM(LOCK,UNLOCK)		No	Motor state
	ENUM(STORED)/ UINT32	1-100	Yes	Locking speed in percent
	ENUM(STORED)/ UINT32	200-2000	Yes	Timeout in ms
Reception example	SET;MOT_LOCK_STATE_A;OK			State successfully reached
Description of response parameters	Type			Description
	ENUM(OK, RUNNING, ERROR)			OK = State successfully reached, RUNNING = Locking/unlocking is in progress, ERROR = State not reached
Available from FW version	1.27.5, Overloading with velocity and timeout from 1.36.6			

Functional description	Setting the motor state in synchronous mode. The setting is confirmed after the operation has been completed. During this time, the serial interface is blocked. This process does not take longer than the period defined for the timeout.
Note	The command can be overloaded with the locking speed, or with it and additionally the timeout. This makes it possible to set the motor state with frequently changing requirements without the parameters having to be set individually. Setting the motor state by overloading uses the parameters once without overwriting the stored values.

Transmission example

1	SET;MOT_LOCK_STATE;LOCK	Performs Lock with permanent velocity and permanent timeout
2	SET;MOT_LOCK_STATE;UNLOCK|STORED	Performs Unlock with permanent velocity and timeout calculated therefrom
3	SET;MOT_LOCK_STATE;LOCK|50	Performs Lock with transmitted velocity and timeout calculated therefrom
4	SET;MOT_LOCK_STATE;UNLOCK|STORED|STORED	Performs Lock with permanent velocity and permanent timeout
5	SET;MOT_LOCK_STATE;LOCK|STORED|1800	Performs Lock with permanent velocity and transmitted timeout
6	SET;MOT_LOCK_STATE;UNLOCK|90|STORED	Performs Unlock with transmitted velocity and permanent timeout
7	SET;MOT_LOCK_STATE;LOCK|80|1800	Performs Lock with transmitted velocity and transmitted timeout

Description of transmission parameters

Type	Value range	optional	Description
ENUM(LOCK, UNLOCK)		No	Motor state
ENUM(STORED)/ UINT32	1-100	Yes	Locking speed in percent
ENUM(STORED)/ UINT32	200-2000	Yes	Timeout in ms

Reception example

SET;MOT_LOCK_STATE;OK	State successfully reached

Description of response parameters

Type	Description
ENUM(OK, TIMEOUT, RUNNING, ERROR)	OK = State successfully reached, TIMEOUT = State not reached within the given time, RUNNING = Locking/unlocking is in progress, ERROR = State not reached

Available from FW version	1.27.5, Overloading with velocity and timeout from 1.36.6

Functional description	Persistent setting of motor timeout
Note	Setting the motor timeout overrides the motor timeout calculated by setting the locking speed. This may result in parameter combinations which lead to a TIMEOUT

Transmission example

1	SET;MOT_LOCK_TIMEOUT;500	Setting the motor timeout

Description of transmission parameters

Type	Value range	Description
UINT32	200-2000	Timeout in ms

Reception example

1	SET;MOT_LOCK_TIMEOUT;OK	

Description of response parameters

ENUM(OK, ERROR)	OK=No error, ERROR=Error occurred

Available from FW version	1.27.5

Functional description	Persistent setting of the locking speed
Note	Setting the locking speed overrides the motor timeout with a calculated set-point value to ensure that it is correctly dimensioned
Transmission example	1 | SET;MOT_VEL;1 | Setting the locking speed
Description of transmission parameters	Type | Value range | Description UINT32 | 1-100 | Velocity (without unit)
Reception example	1 | SET;MOT_VEL;OK |
Description of response parameters	ENUM(OK, ERROR) | OK = No error, ERROR = Error occurred
Available from FW version	1.36.6

6.5.2 Sensors

Functional description	Persistent activation or deactivation of the compensation. When compensation is enabled, the position measuring system is compensated in the locked state, which minimizes errors due to temperature fluctuations, for example
Transmission example	1 | SET;ENABLE_COMPENSATION;TRUE | Enabling the compensation
Description of transmission parameters	Type | Description ENUM(TRUE,FALSE) | Target state of the compensation (TRUE = enabled compensation, FALSE = disabled compensation)
Reception example	1 | SET;ENABLE_COMPENSATION;OK | Compensation successfully enabled
Description of response parameters	Type | Description ENUM(OK, ERROR) | Feedback about success (OK = Compensation successfully enabled/disabled, ERROR = Error enabling/disabling the compensation)
Available from FW version	1.27.5

Functional description	Persistent setting of the length of the moving average filter for actual position values
Transmission example	1 | SET;POSE_FILTER;10 | Setting the length of the average filter to 10 position values
Description of transmission parameters	Type | Value range | Description UINT32 | 0-128 | Length of the average filter in position values
Reception example	1 | SET;POSE_FILTER;OK | Length of the average filter set successfully
Description of response parameters	Type | Description ENUM(OK, ERROR) | Feedback about success (OK = Length of the average filter set successfully, ERROR = Error setting length of the average filter)
Available from FW version	1.27.5

6.5.3 LED modes

Functional description	Setting the LED mode. For a description of the different modes, see chapter Displaying various operating states (LED modes)		
Transmission example	1	SET;LED_MODE;STATUS	Setting the LED mode to “STATUS” mode
Description of transmission parameters	Type		Description
	ENUM (OFF, STATUS, QUALITY, SENSOR, GRAPH_XYZ)		OFF = The LEDs are switched off, STATUS = Graphically displays the lock mode, sensor status and lock status, IO = Graphically displays the levels of the digital IOs in addition to the status mode, GRAPH_XYZ = Graphically displays the deflection in X/Y/Z-direction, QUALITY = Graphically displays the quality mode, SENSOR = Graphically displays the sensor diagnosis, CUSTOM = Shows that a custom LED display is active)
Reception example	1	SET;LED_MODUS;OK	LED mode set successfully
Description of response parameters	Type		Description
	ENUM(OK, ERROR)		Feedback about success (OK = LED mode successfully set, ERROR = Error setting LED mode)
Available from FW version	1.27.5		

Functional description	Persistent activation of a quality mode, see chapter Displaying various operating states (LED modes)		
Transmission example	1	SET;QUALITY_MODE; QUALITY_TRIGGER	Enabling the “QUALITY_TRIGGER” quality mode
Description of transmission parameters	Type		Description
	ENUM(QUALITY_TRIGGER, QUALITY_CONITNUOUS, QUALITY_PEAK, QUALITY_OFF)		Set quality mode(QUALITY_TRIGGER = Quality check when triggering through digital inputs or serial interface, QUALITY_CONTINUOUS = Continuous quality check, QUALITY_PEAK = Continuous quality check which does not reset to OK in case of NOK, QUALITY_OFF = quality check switched off)
Reception example	1	SET;QUALITY_MODE;OK	Quality mode successfully set
Description of response parameters	Type		Description
	ENUM(OK, ERROR)		Feedback about success (OK = Quality mode successfully set, ERROR = Error setting quality mode)
Available from FW version	1.36.6		

<table>
<tr><td>Functional description</td><td>Persistent setting of minimum and maximum quality mode thresholds (see chapter Quality mode); they may vary depending on the application. Up to 10 parameter sets can be stored (indexes 0-9). Switching via SET;QUALITY_ACTIVE_THRESH command.</td></tr>
<tr><td>Transmission example</td><td>
<table>
<tr><td>1</td><td>SET;QUALITY_THRES;-1.000|-1.000|-2.000|-1.000|-1.000|-3.000|1.000|1.000|0.100|1.000|1.000|3.000</td><td>Setting the currently active threshold data record for the quality check in the quality mode to the specified limits.</td></tr>
<tr><td>2</td><td>SET;QUALITY_THRES;1|-1.000|-1.000|-2.000|-1.000|-1.000|-3.000|1.000|1.000|0.100|1.000|1.000|3.000</td><td>Setting the threshold data record with index 1 for the quality check in the quality mode to the specified limits.</td></tr>
</table>
</td></tr>
<tr><td>Description of transmission parameters</td><td>
<table>
<tr><td>Type</td><td>Value range</td><td>optional</td><td>Description</td></tr>
<tr><td>UINT32</td><td>0 - 9</td><td>Yes</td><td>Index of the threshold data record to be set</td></tr>
<tr><td>Float</td><td>-3.5 - 3.5</td><td>No</td><td>Lower limit translative displacement in x direction in mm</td></tr>
<tr><td>Float</td><td>-3.5 - 3.5</td><td>No</td><td>Lower limit translative displacement in y-direction in mm</td></tr>
<tr><td>Float</td><td>-3.5 - 3.5</td><td>No</td><td>Lower limit translative displacement in z-direction in mm</td></tr>
<tr><td>Float</td><td>-3.9 - 3.9</td><td>No</td><td>Lower limit rotary motion around x-axis in °</td></tr>
<tr><td>Float</td><td>-3.9 - 3.9</td><td>No</td><td>Lower limit rotary motion around y-axis in °</td></tr>
<tr><td>Float</td><td>-7.3 - 7.3</td><td>No</td><td>Lower limit rotary motion around z-axis in °</td></tr>
<tr><td>Float</td><td>-3.5 - 3.5</td><td>No</td><td>Upper limit translative displacement in x direction in mm</td></tr>
<tr><td>Float</td><td>-3.5 - 3.5</td><td>No</td><td>Upper limit translative displacement in y-direction in mm</td></tr>
<tr><td>Float</td><td>-3.5 - 3.5</td><td>No</td><td>Upper limit translative displacement in z-direction in mm</td></tr>
<tr><td>Float</td><td>-3.9 - 3.9</td><td>No</td><td>Upper limit rotary motion around x-axis in °</td></tr>
<tr><td>Float</td><td>-3.9 - 3.9</td><td>No</td><td>Upper limit rotary motion around y-axis in °</td></tr>
<tr><td>Float</td><td>-7.3 - 7.3</td><td>No</td><td>Upper limit rotary motion around z-axis in °</td></tr>
</table>
</td></tr>
<tr><td>Reception example</td><td>
<table>
<tr><td>1</td><td>SET;QUALITY_THRES;OK</td><td>Threshold data record set successfully</td></tr>
</table>
</td></tr>
<tr><td>Description of response parameters</td><td>
<table>
<tr><td>Type</td><td>Description</td></tr>
<tr><td>ENUM(OK, ERROR)</td><td>Feedback about success
(OK = Threshold data record set successfully,
WARNING = Exceedance of the permissible deflection according to the technical data sheet. Values < 0.5 mm/° above or below the specification in the data sheet are accepted, but this warning is issued.
ERROR = Error setting the threshold data record)</td></tr>
</table>
</td></tr>
<tr><td>Available from FW version</td><td>1.27.5, Switching function with indexes (optional parameter) and “Warning” return type from 1.36.6</td></tr>
</table>

Functional description	Setting the currently active QUALITY_THRESHOLD_INDEX to determine the applicable threshold data record for the quality mode
Transmission example	1 | SET;QUALITY_ACTIVE_THRES;0 | Enabling the threshold data record with index 0
Description of transmission parameters	Type | Value range | Description UINT32 | 0-9 | ID of the threshold data record to be activated
Reception example	1 | SET;QUALITY_ACTIVE_THRES;OK | Active threshold data record set successfully
Description of response parameters	Type | Description ENUM(OK, ERROR) | Feedback about success feedback(OK = Active threshold data record set successfully, ERROR = Error setting the active threshold data record)
Available from FW version	1.36.6

Functional description	Setting the LED color of all LEDs to the same color value
Transmission example	1 | SET;LED_SINGLE;255|255|0 | Setting of all LEDs to the color value (RGB) 255|255|0 (yellow).
Description of transmission parameters	Type | Value range | Description UINT8 | 0-255 | Red share of the RGB color value UINT8 | 0-255 | Green share of the RGB color value UINT8 | 0-255 | Blue share of the RGB color value
Reception example	1 | SET;LED_SINGLE;OK | LED color successfully set
Description of response parameters	Type | Description ENUM(OK, ERROR) | Feedback about success(OK = LED color set successfully ERROR = Error setting the LED color)
Available from FW version	1.27.5

Functional description	Setting every single one of the 36 LEDs to a defined RGB value
Transmission example	1 | SET;LED_MULTI; 138,43,226|138,43,226|138,43,226|138,43,226|138,43,226|138,43,226| 138,43,226|138,43,226|138,43,226|138,43,226|138,43,226|138,43,226| 138,43,226|138,43,226|138,43,226|138,43,226|138,43,226|138,43,226| 138,43,226|138,43,226|138,43,226|138,43,226|138,43,226|138,43,226| 138,43,226|138,43,226|138,43,226|138,43,226|138,43,226|138,43,226| 138,43,226|138,43,226|138,43,226|138,43,226|138,43,226 | Setting the individual LEDs to the specified color values (RGB)
Description of transmission parameters	Type: LIST(36) Description: Type: UINT8 | Value range: 0-255 | Description: Red share of the RGB color value Type: UINT8 | Value range: 0-255 | Description: Green share of the RGB color value Type: UINT8 | Value range: 0-255 | Description: Blue share of the RGB color value
Reception example	1 | SET;LED_MULTI;OK | LED colors successfully set
Description of response parameters	Type: ENUM(OK, ERROR) | Description: Feedback about success(OK = LED colors successfully set ERROR = Error setting the LED colors)
Available from FW version	1.27.5

6.5.4 IO pins

Functional description	Persistent setting of the configuration of a digital IO pin
Transmission example	1 | SET;IO_CONFIG;1|IO_LOCK | Setting the configuration with pin ID 0 to function “IO_LOCK”, i.e. to trigger a locking
Description of transmission parameters	Type: UINT32 | Value range: 0-2 | Description: IO pin ID Type: ENUM(IO_LOCK_STATE, IO_LOCK, IO_ERROR, IO_OFF, IO_QUALITY_TRIGGER, IO_QUALITY_IN_RANGE, IO_INPUT, IO_OUTPUT) | Value range: - | Description: IO_LOCK_STATE = Display status of the locking, IO_LOCK = Trigger locking, IO_ERROR = Show error state, IO_OFF = Disabled, IO_QUALITY_TRIGGER= Triggering a trigger in quality mode,, IO_QUALITY_IN_RANGE = Indication whether deflection in the configured range is in quality mode, IO_INPUT = General input pin, IO_OUTPUT = General output pin
Reception example	1 | SET;IO_CONFIG;OK | IO pin configuration set successfully
Description of response parameters	Type: ENUM(OK, ERROR) | Description: Feedback about success(OK = IO pin configuration set successfully, ERROR = Error setting IO pin configuration)
Available from FW version	1.36.6

Functional description	Setting the level of a pin configured as output (IO_OUTPUT)
Transmission example	1 | SET;IO_STATE;0|LOW | Setting the level of the pin with ID 0 to “LOW”
Description of transmission parameters	Type | Value range | Description UINT32 | 0-2 | IO pin ID ENUM(LOW, HIGH) | - | LOW = Level low, HIGH = Level high
Reception example	1 | SET;IO_STATE;OK | IO pin levels set successfully
Description of response parameters	Type | Description ENUM(OK, ERROR) | Feedback about success (OK = IO pin levels set successfully, ERROR = Error setting the IO pin level)
Available from FW version	1.36.6

6.5.5 System

Functional description	Persistent setting of the baud rate.
Transmission example	1 | SET;BAUD;115200 | Setting the baud rate to 115200 Baud
Description of transmission parameters	Type | Value range | Description UINT32 | discrete: 9600, 38400, 115200, 921600 | Baud rate
Reception example	1 | SET;BAUD;OK | Baud rate set successfully
Description of response parameters	Type | Description ENUM(OK, ERROR) | Feedback about success (OK = Baud rate set successfully, ERROR = Error setting the baud rate)
Available from FW version	1.27.5

Functional description	Setting the response delay (see chapter Setting the command delay)
Transmission example	1 | SET;CMD_DELAY;1 | Setting the response delay to one word length
Description of transmission parameters	Type | Value range | Description UINT32 | 0-10 | Response delay in word lengths (baud rate-dependent)
Reception example	1 | SET;CMD_DELAY;OK | Response delay set successfully
Description of response parameters	Type | Description ENUM(OK, ERROR) | Feedback about success(OK = Response delay set successfully, ERROR = Error setting the response delay)
Available from FW version	1.36.6

6.6 CTR functions

Functional description	SFE unit restart
Transmission example	1 | CTR;REBOOT | SFE unit restart
Reception example	1 | CTR;REBOOT;OK | Restart triggered successfully
Description of response parameters	Type: ENUM(OK, ERROR) — Description: Feedback about success(OK = Restart triggered successfully, ERROR = Error triggering the restart)
Available from FW version	1.27.5

Functional description	Starting the data logger which allows measurement data to be recorded over a longer period of time and retrieved subsequently (For more details see chapter Data logger)
Transmission example	1 | CTR;LOGGER_START | Start the data logger with recording every 5ms for an indefinite number of data records (free running) 2 | CTR;LOGGER_START;10 | Start the data logger with recording every 10ms for an indefinite number of data records (free running) 3 | CTR;LOGGER_START;10|500 | Start the data logger with recording every 10ms for 500 data records
Description of transmission parameters	Type: UINT32 — Value range: 5-100 (multiples of 5 only) — Optional: Yes — Description: Recording interval in ms Type: UINT32 — Value range: 1-2000 — Optional: Yes — Description: Number of data records that are to be recorded
Reception example	1 | CTR;LOGGER_START;OK | Data logger started successfully
Description of response parameters	Type: ENUM(OK, ERROR) — Description: Feedback about success(OK = Data logger started successfully, ERROR = ERROR starting the data logger)
Available from FW version	1.36.6

Functional description	Stop recording with data logger (For details see chapter Data logger)
Transmission example	1 | CTR;LOGGER_STOP | Recording with data logger is stopped
Reception example	1 | CTR;LOGGER_STOP;OK | Data logger stopped successfully
Description of response parameters	Type: ENUM(OK, ERROR) — Description: Feedback about success(OK = Data logger stopped successfully, ERROR = Error stopping data logger)
Available from FW version	1.36.6

Functional description	Trigger quality check when QUALITY_TRIGGER quality mode is active (For details see chapter Quality mode)		
Transmission example	1	CTR;QUALITY_TRIGGER	Quality check is triggered
Reception example	1	CTR;QUALITY_TRIGGER;OK	Quality check successfully triggered
Description of response parameters	Type		Description
	ENUM(OK, ERROR)		Feedback about success(OK = Quality check successfully triggered, ERROR = Error triggering quality check)
Available from FW version	1.36.6		

Functional description	Resetting the quality check result (For details see chapter Quality mode)		
Transmission example	1	CTR;QUALITY_RESET	Quality check result is reset
Reception example	1	CTR;QUALITY_RESET;OK	Quality check result successfully reset
Description of response parameters	Type		Description
	ENUM(OK, ERROR)		Feedback about success (OK = Quality check result successfully reset, ERROR = Error resetting the quality check result)
Available from FW version	1.36.6		

6.7 Example communication

Received position:

Sent	Received
GET;POSE	GET;POSE;-0.004|0.003|0.003|-0.003|0.003|0.005;OK

If this command is executed in a loop, a constant deflection or position monitoring of the SFE tool flange may take place.

This command can be used to develop many strategies, for example for joining.

Checking the version

The following communication may serve as an example for checking the current version of the embedded software:

Sent	Received
GET;VERSION	GET;VERSION;1.36.6

The version can only be updated via the desktop software.

You can find a more detailed description of the desktop software in the Initialization section.

Changing and checking the lock state

Sent	Received
SET;MOT_LOCK_STATE;LOCK	SET;MOT_LOCK_STATE;OK
GET;MOT_LOCK_STATE	GET;MOT_LOCK_STATE;LOCKED

Here, the first command is for locking, whereupon the SFE response states that the command was received and carried out.

The second command is for querying the current status, whether the device is locked or unlocked. It is locked in the example.

7 Range of functions

7.1 Transformation chain

For the supplied SFE position values to be used correctly in the controller, the tool information cannot be described as a transformation. The transformation must be described from a chain of several transformations. This is necessary, as the SFE flange is mobile and can therefore also be displaced by the TCP. If the SFE offset were not restored, the tool transformation would not be updated and you would receive incorrect values about the current position of the TCP.

It is therefore recommended to build up two transformation chains to compare deflected and fixed/locked SFE.

- The first chain contains the transformations with the locked SFE.
- The second is fed with the SFE values which you obtain using GET;POSE.

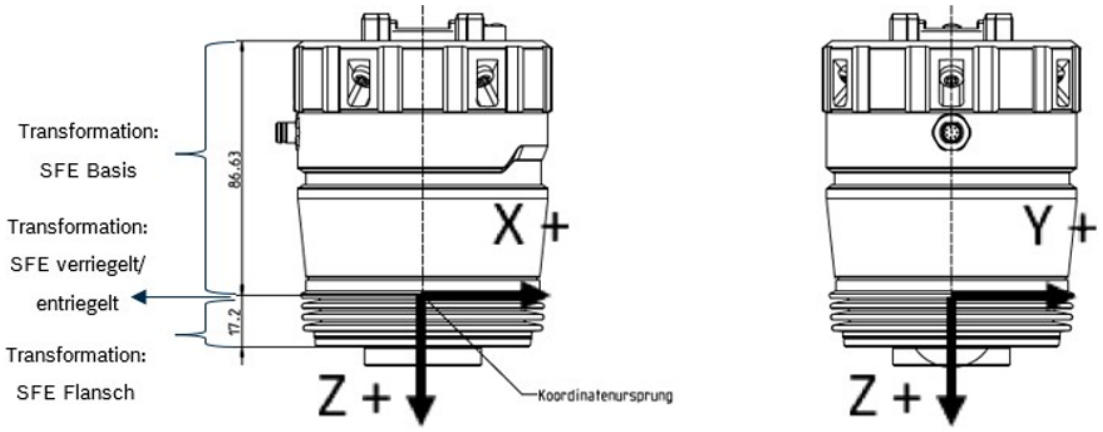
→ The values are written in the SFE unlocked transformation.

☞ See figure 14 Transformation chain for unlocked SFE.

The SFE is made up of three parts in the transformation chain:

- Base (the top part with a length of 86.63 mm)
- Flange (17.2 mm in length)
- Transformation (created by the displacement between the base and flange)

The displacement is in the origin of coordinates in the following figure:



12 **Displacement**

→ This displacement can be obtained using the command GET;POSE.

➢ Open the command GET;POSE cyclically.

➢ The writing of the updated values in the transformation must by cyclical so that this is kept up to date.

In the **Displacement** figure, you can see the SFE without affixed flange plates. The transformation chain must then contain the respective mounted flange.

The following transformations must be entered for the supplied flanges:

Adapter plate	Transformation Z
R124500039	13.50 mm
R124500041	16.00 mm
R124500055	16.00 mm

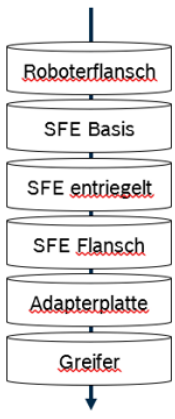
22 **Transformation**

Depending on how the robot flange coordinate system is, the transformation to the SFE base must be expanded more, for example, by a rotation on Z. This depends on the robot.

In the following example, flange R124500039 is installed.

	X	Y	Z	RX	RY	RZ
1.	-	-	-	-	-	-
2.	0	0	86.63	0	0	0
3.	0	0	0	0	0	0
4.	0	0	17.2	0	0	0
5.	0	0	13.5	0	0	0
6.	0	0	40	0	0	0

13 **Transformation chain for locked SFE**



	X	Y	Z	RX	RY	RZ
1.	-	-	-	-	-	-
2.	0	0	86.63	0	0	0
3.	0.010	0.011	0.009	0.001	0.002	0.001
4.	0	0	17.2	0	0	0
5.	0	0	13.5	0	0	0
6.	0	0	40	0	0	0

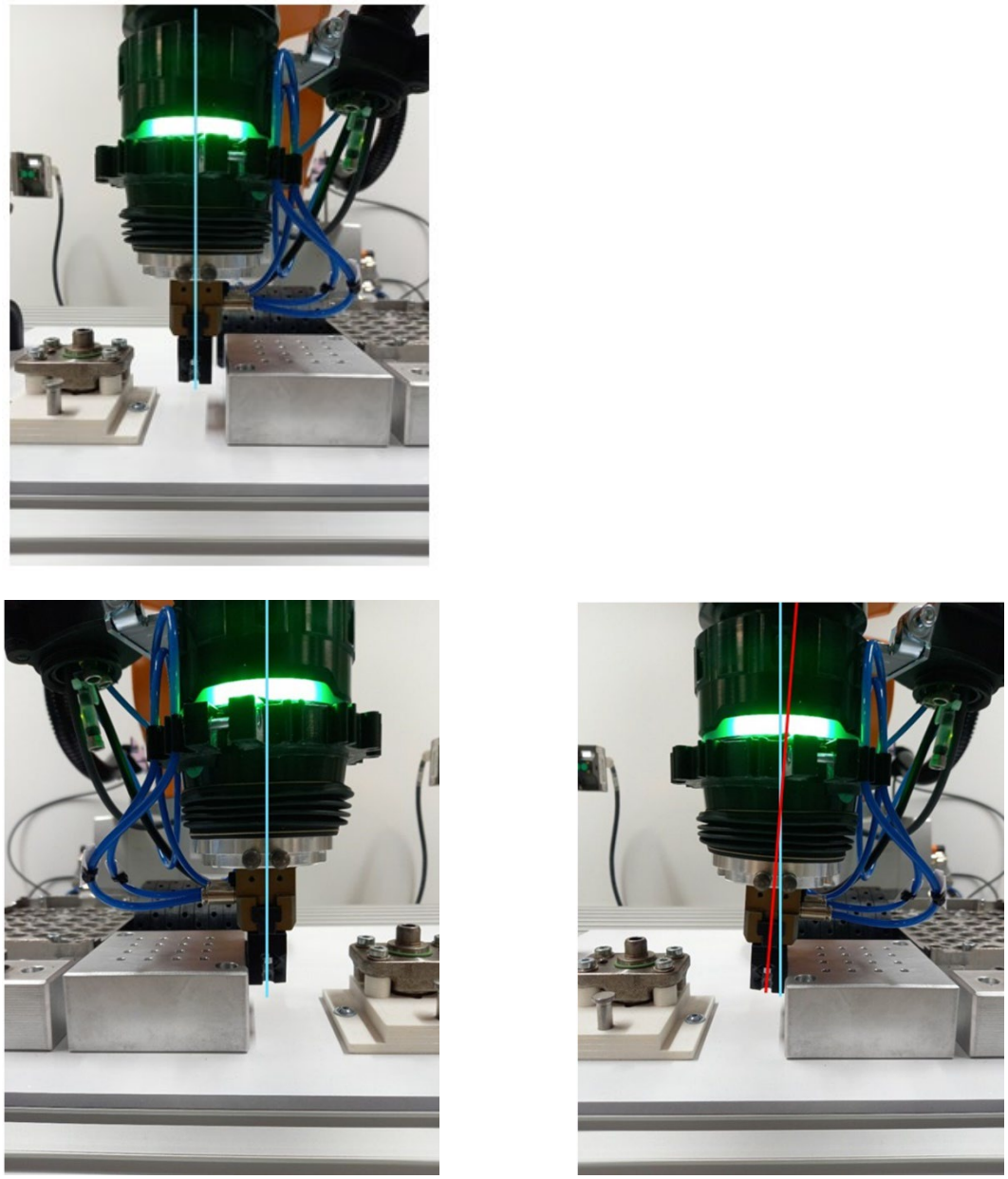
14 **Transformation chain for unlocked SFE**

The values in 3 are obtained from the response values of GET;POSE.

Any installed intermediate plates, gripper jaws or components must then also be included in the transformation chain, depending on the application.

In the following figures, you can see how an SFE deflection occurs. Here, you can see a rotation on the X axis.

The SFE must be unlocked for this.



15 **SFE deflection**

The robot moves against the block. Once the gripper touches the block, the SFE yields. You can obtain the current deflection again via GET;POSE.

In the last figure, you can clearly see why you must update the transformation chain cyclically with GET;POSE, as the TCP experiences a displacement when contact is made with a component.

The blue line represents the SFE in the non-deflected state, the red line shows the displacement.

7.2 Handling

7.2.1 Initial temperature compensation when starting up and after each restart

Before using the SFE for the first time and after each restart, a locking procedure should be carried out using SET;MOT_LOCK_STATE;LOCK before working with the device. During this kind of locking procedure, the temperature compensation, activated by default, is carried out, which allows the greater temperature-related position deviations which occur due to the temperature dependency of the sensors to be avoided.

7.2.2 Communication protocol error responses

The communication protocol distinguishes between protocol errors and command errors. Protocol errors are shown in Table 23. For example, SET;BAUD;4294967296 is a protocol error because there is an overflow of UINT32. Protocol errors are always detected before command errors.

Error class	Description
ERROR;UNKNOWN_CMD	Command class or command does not exist
ERROR;WRONG_MODE	• No permission for the command • Digital mode
ERROR;WRONG;CRC	The checksum sent in the request is incorrect
ERROR;STORAGE_PROTECTION	Protection of the EEprom was triggered due to too frequent writing
ERROR;SYNTAX	• The number of transmitted parameters does not match the command definition • Error in the command syntax
ERROR;INVALID_PARAMETER	• Parameters out of range (e.g. UINT8=256) • Example: SET;BAUD;4294967296

23 **Overview of protocol errors**

Command errors are individual for each command. They only occur if, for example, a parameter is invalid in a specific case (for example, SET;BAUD;100 → SET;BAUD;ERROR). In the example stated here, the command form is correct "SET;BAUD;UINT32", but a baud rate of 100 baud cannot be set. Such parameter constraints are documented in the respective command description under the description of the transfer parameters in the chapter Interface description.

7.2.3 Errors during ongoing operation

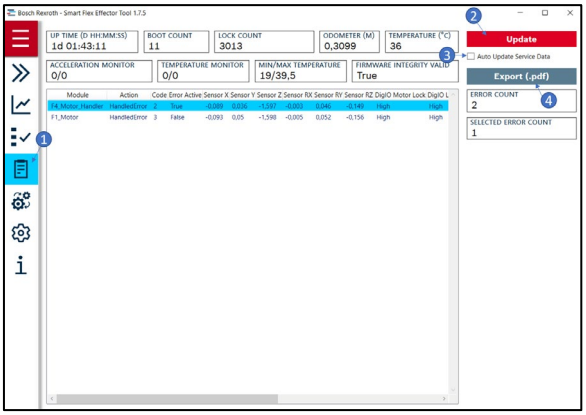
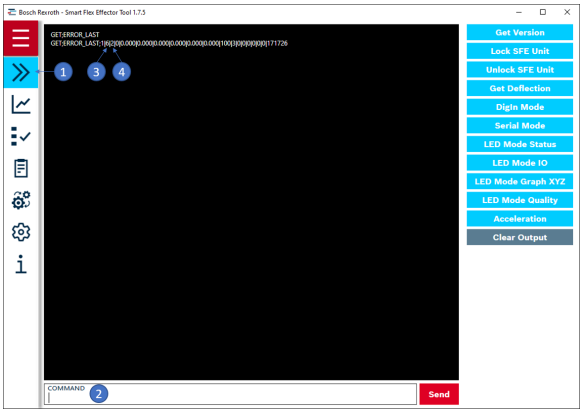
Unlike the commands of the communication protocol, these errors can occur even without a command having been sent beforehand. For example, an internal electronics defect may be detected and reported during operation. This leads to an entry in the error history (error log). In addition, if it is configured, an IO pin is set to level HIGH (see chapter Use and configuration of the digital IO pins).

This pin must be connected to the control so that this can be detected by the control. In the event of an error, the pin is set to the HIGH level for 10 seconds and then back again to the LOW level. The pin only remains on HIGH permanently if errors are pending for a long time.

The error must be rectified using the response values and the enclosed guide on error rectification The movement can be continued afterwards.

Identification of error sources and possible causes and measures to rectify them

In order to determine possible sources of error and measures to rectify them, the error module and error number (error code) must be read. This can be done via the error log in the SFE tool or alternatively via the command GET;ERROR_LAST (last error) or GET;ERROR_HISTORY (reading of the entire error memory):

Screen	Action steps
	Read error module and error code with the SFE tool: ➢ Navigate to the Service (1) tab. ➢ Click the "Update" button to display the current error table (2). ➢ Read the error module (here: F4_Motor_Handler) (3) and error code (here: 2) (4).
	Alternatively: Read error module and error code with the command GET;ERROR_LAST: ➢ Navigate to the Console (1) tab. ➢ Send the command GET;ERROR_LAST (2). ➢ The error module is the second digit in the response (here: 13) (3). ➢ The error code is the third digit in the response (here: 2) (4).

24 **Reading the error module and error code**

Using the error module and error code, you can now determine the error description and possible sources of error using the following table:

Error module	Module number	Error code	Possible error sources	Comments
F4_Bootloader	0	0-8	Internal electronics	

Error module	Module number	Error code	Possible error sources	Comments
F4_Bootloader	0	9	RS485	
F4_Bootloader	0	10-19	Internal electronics	
F4_Bootloader	0	20	RS485	
F4_Bootloader	0	21	RS485, internal electronics	
F4_Bootloader	0	22	RS485	
F4_Bootloader	0	23	RS485, internal electronics	
F4_Bootloader	0	24-25	Internal electronics	
F4_Bootloader	0	26-27	RS485	
F4_Bootloader	0	28	Internal electronics	
F4_Bootloader	0	29-30	RS485	
F4_Bootloader	0	31-34	Internal electronics	
F1_Bootloader	1	0-1	Internal electronics	
F1_Bootloader	1	2	RS485	
F1_Bootloader	1	5-8	Internal electronics	
F1_Bootloader	1	9	RS485	
F1_Bootloader	1	10	RS485, internal electronics	
F1_Bootloader	1	11	RS485	
F1_Bootloader	1	12	RS485, internal electronics	
F1_Bootloader	1	13-18	Internal electronics	
F4_Intercom	2	0-27	Internal electronics	
F4_Bootloader_Jumping	4	0-6	Internal electronics	
F1_Bootloader_Jumping	5	0-6	Internal electronics	

Error module	Module number	Error code	Possible error sources	Comments
F1_Motor	6	1	Motor, internal electronics	
F1_Motor	6	2	Motor	
F1_Motor	6	3	Motor	Error when performing a locking/unlocking procedure (target state was not reached) First, increase timeout with SET;MOT_LOCK_TIMEOUT (lock process can take longer if the load is greater) If this does not work, proceed with rectification measures under error source "Motor".
F1_Motor	6	4-5	Internal electronics	
F4_Settings	7	0-1	Internal electronics	
F4_Settings	7	22	Internal electronics	
F4_Parameter _Storage	8	0-62	Internal electronics	
F4_RS485	9	0-6	RS485	
F4_Position sensors	10	0-22	Internal electronics	
F4_Temperature sensor	11	0-1	Internal electronics	
F4_System	12	0	Internal electronics, RS485, motor	Internal initialization processes not successful
F4_System	12	1-2	Internal electronics	
F4_System	12	3	See error sources of the other errors	A large number of errors with one or more error sources has occurred within a short time

Error module	Module number	Error code	Possible error sources	Comments
F4_Motor_ Handler	13	0-1	Motor, internal electronics	
F4_Motor_ Handler	13	2	Motor	Error when performing a locking/unlocking procedure (target state was not reached) First, increase timeout with SET;MOT_LOCK_TIMEOUT (lock process can take longer if the load is greater) If this does not work, proceed with rectification measures under error source "Motor".
F4_Led_ Handler	14	0	Internal electronics	
F4_DIGIO_ Handler	15	0	Motor, internal electronics	Dig IO status could not be updated correctly
F4_Hexapod	16	0	Internal electronics	
F4_Protocol_ Handler	17	0-3	RS485	

25 **Error description and error sources**

You can now take measures to rectify the error, depending on the possible error sources:

Error source	Error pattern	Possible cause	Rectification/measures
Internal electronics	Error in communication with the EEPROM, with position sensors, with the temperature sensors or with the interprocessor communication	EEPROM defective, position sensor defective, temperature sensor defective, F100 MCR defective, cold solder joint, loose contact, electromagnetic interference	1. Restart the SFE and observe whether the error continues to occur 2. Check the dimensioning of the power supply unit (see Technical Data) 3. Contact support
RS485	Communication with the RS485 interface not possible / defective	Unidirectional communication protocol (request/response) is not adhered to, connection line too long, defective connection line, electromagnetic interference, RS485 transceiver or interface defective, cold solder joint, loose contact	1. Check whether communication is taking place according to the request-response principle 2. Check the length of the connection line (for maximum length see Technical Data) 3. Check for defects on the connection line 4. Identify possible sources of electromagnetic interference 5. Restart the SFE and observe whether the error continues to occur 6. Contact support
Motor	Error when installing / moving with the motor	Motor overload, overcurrent shutdown, low supply voltage, cable too long, cable defective, overload (weight), voltage breakdown, short-circuit, hardware defective, position sensor defective, magnet for position determination too weak, incorrect size of power supply unit	1. Check the load 2. Ensure that the SFE is freely mobile when locking/unlocking 3. Ensure that it is not locked/unlocked while moving 4. Check the cable length (for maximum length see) 5. Check cable for defects 6. Check the dimensioning of the power supply unit (see Technical data) 7. Restart the SFE and observe whether the error continues to occur 8. Contact support

26 **Rectification measures**

When contacting support, please generate the following error analysis information with the SFE tool in advance and have it ready:

- Configuration export with calibration configuration (see chapter Export SFE configuration)
- Error log (see Viewing and exporting service data for more information about this)

7.2.4 Moving at high travel speed

The SFE must be locked when moving at high velocity. Otherwise, the forces generated by the movement of accelerated masses may lead to damage.

Moving the SFE to the position compensation end position is not permitted. These end positions are specified in the chapter Technical data. In the unlocked state, this must be observed when selecting the travel speed and acceleration. When setting up in the unlocked state, we recommend starting with a travel speed lower than 100 mm/s.

An example process may look as follows:

1. The robot is at the home position, the command for locking the SFE, SET;MOT_LOCK_STATE;LOCK, is sent. The response SET;MOT_LOCK_STATE;OK is returned. The robot may move to the preliminary position at high travel speed.
2. When the preliminary position is reached, the SFE is unlocked via SET;MOT_LOCK_STATE;UNLOCK. If the command SET;MOT_LOCK_STATE;OK is sent, the robot can move to the pickup position at reduced travel speed.
3. Once the product has been picked up, the robot moves back to the preliminary position and the SFE is locked again. Once locking is complete, the robot can move to the next position with increased travel speed again.

7.2.5 Using the locked and unlocked transformation chains

- ➢ Teach the positions with the locked transformation chain and move to these as well.
- ➢ Compare the deviations using the unlocked transformation chain.
- ➢ For example, determine the offset of the TCP from the transformation chain with the unlocked SFE in the pickup position and carry out a compensatory movement with the robot to eliminate the offset.

7.3 Exemplary application cases

In an example like the one shown below, the following is to be observed:

For a movement in the unlocked state, the system must never be brought to the limit stop in order to avoid damage. The maximum travel speed for a newly setup application should therefore be determined through repetition and while observing the internal sensors. The starting point for a secured, collision-free setup is the stated travel speed of 100 mm/s. The parameters set for the safe operation of an example application is provided for orientation purposes, but cannot be interpreted explicitly as a guideline value for any and all applications.

Max. operating data for example application:		
Acceleration **	[m/s^2]	2.5
Travel speed **	[mm/s]	250
Rotation velocity **	[°/s]	550
Rotation acceleration **	[°/s^2]	5000

27 **Operating data for example applications**

** was determined for a trial "pick and place" allocation with a nominal load: Mass 6 kg, lever arm load center 100 mm, aluminum cuboid 3.1645: length 200 mm, width/depth 103 mm, suspension centrally over standard flange: ISO 9409-1 31,5-4-M5, no oscillating movement takes place.

7.3.1 Exact alignment of the robot above a component

If the robot moves over a component in a pickup, drop or joining position and is not set correctly above the position due to various tolerances, this is expressed by a deflection of the SFE.

If the offset which can be read by the robot from the SFE is compensate by a position correction by the robot, the robot is correctly above the joining position and the process can continue without disruption.

7.3.2 Connecting connector contacts

Connector connections can be fitted using the SFE; the process for clicking them into place can be detected by analyzing the position data. This allows an automatic quality check without any visual check afterwards.

7.3.3 Determining the orientation of pallets

Pallets which are no longer in the same position as the previous ones after a change may be touched by the SFE, allowing it to remeasure the position in by communicating the displacement to the robot.

7.3.4 Joining µ fits

Application-specific searching and joining strategies can be used to join fits with tolerances in the µ range.

7.3.5 Measuring a tool changer

The drop positions for individual tools can be measures automatically using the SFE. This saves a laborious process of teaching the positions.

7.4 Displaying various operating states (LED modes)

7.4.1 OFF

With the command SET;LED_MODE;OFF, all LEDs are switched off permanently.

7.4.2 STATUS

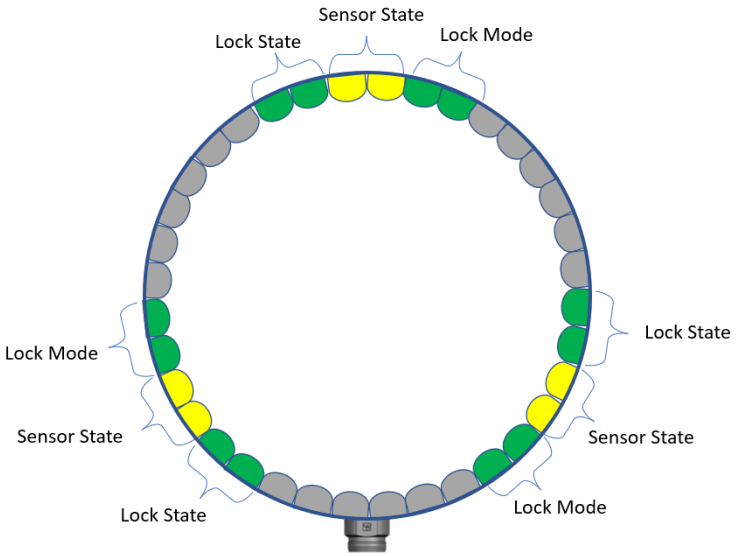
With the command SET;LED_MODE;STATUS, the status mode is changed in the status. The status mode is also the LED mode which is displayed during a boot process after the startup animation.

To make this easier to read, the three states are shown three times in the light bar, each time shifted by 120°. A status light is formed of two LEDs.

The following three states can be identified on the SFE:

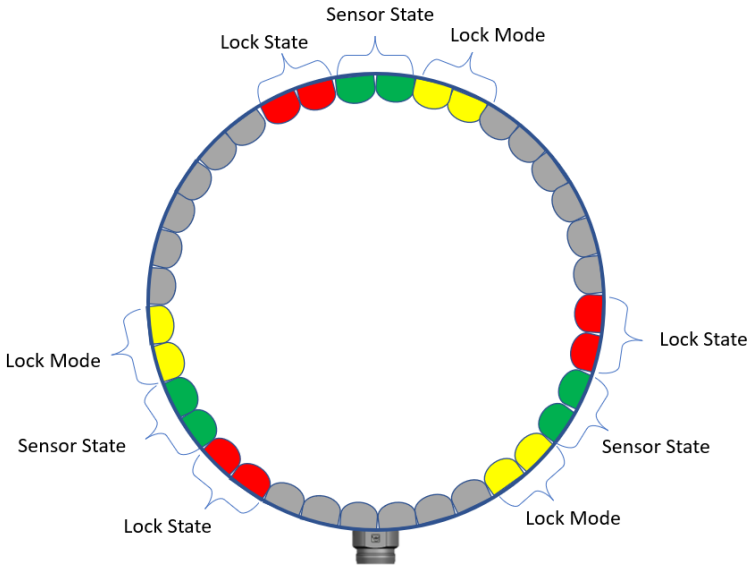
- selected lock mode
    - SERIAL: LEDs illuminated in green
    - DIGIN: LEDs illuminated in yellow
- Sensor status
    - Sensor calibration not set: LEDs illuminated in blue
    - Sensors in the range 0-66% of the maximum deflection: LEDs illuminated in green
    - Sensors in the range 67-83% of the maximum deflection: LEDs illuminated in yellow
    - Sensors in the range 84-100% of the maximum deflection: LEDs illuminated in red
- Lock status
    - Motor calibration not set: LEDs illuminated in blue
    - Unlocked: LEDs illuminated in green
    - In motion: LEDs illuminated in blue
    - Locked: LEDs illuminated in yellow
    - Timeout/error: LEDs illuminated in red

The following LED figure visualizes the information that SERIAL was selected as the lock mode, the degree of modulation of the sensors is between 67-83% and that the SFE unit is currently unlocked.



16 **Example 1 for a possible LED pattern, top view**

The following LED figure visualizes the information that DIGIN was selected as the lock mode, the degree of modulation of the sensors is between 0-66% and that an error has occurred during the locking/unlocking procedure.



17 **Example 2 for a possible LED pattern, top view**

7.4.3 IO

With the command SET;LED_MODE;IO, the display changes into the IO mode. Both, the status of the SFE and the status of the digital IO pins 0-2 are shown.

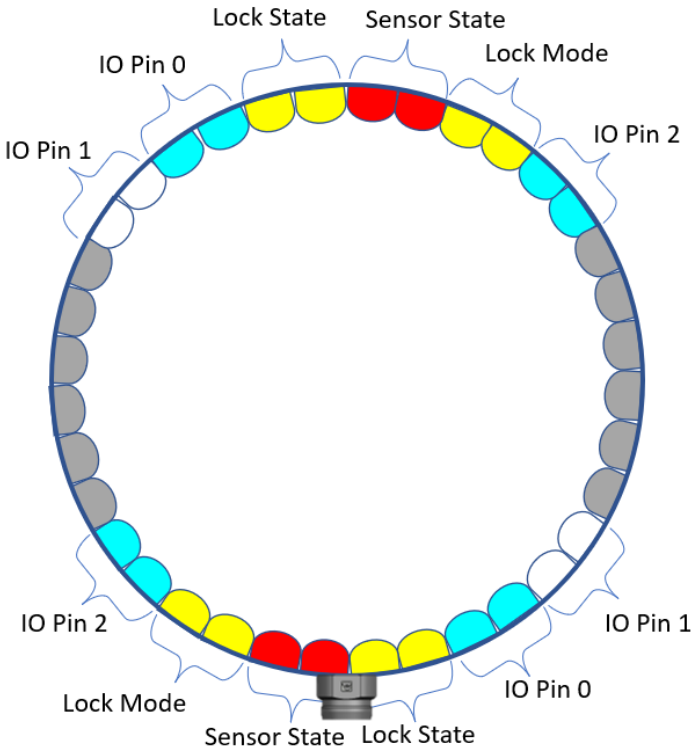
The “Lock status”, “Sensor status” and “Lock mode” LED pairs behave as in STATUS mode, see chapter STATUS.

The level of a pin configured as “IO_ERROR” is shown in red when it is HIGH (error), and white when it is LOW (no error).

The level of pins configured otherwise is shown in blue when it is HIGH and white when it is LOW.

IO_ERROR (IO pin 1)	There is no error.
IO_LOCK_STATE (IO pin 0)	SFE is locked.
Lock status	The SFE is locked.
Sensor status	The control of a proximity sensor is in the red range.
Lock mode	Digital communication is selected.
IO_LOCK (IO pin 2)	The digital input for locking is set.

28 **States**



18 **Example 3 for a possible LED pattern in IO mode, top view**

7.4.4 GRAPH_XYZ

With the command SET;LED_MODE;GRAPH_XYZ, the mode is switched to graph mode. In this mode, the current deflection of the X, Y and Z axis is displayed in graphic form. The greater the deflection in x/y direction, the more LEDs are lit up. The deflection in z direction is shown by a color change from white to dark blue.

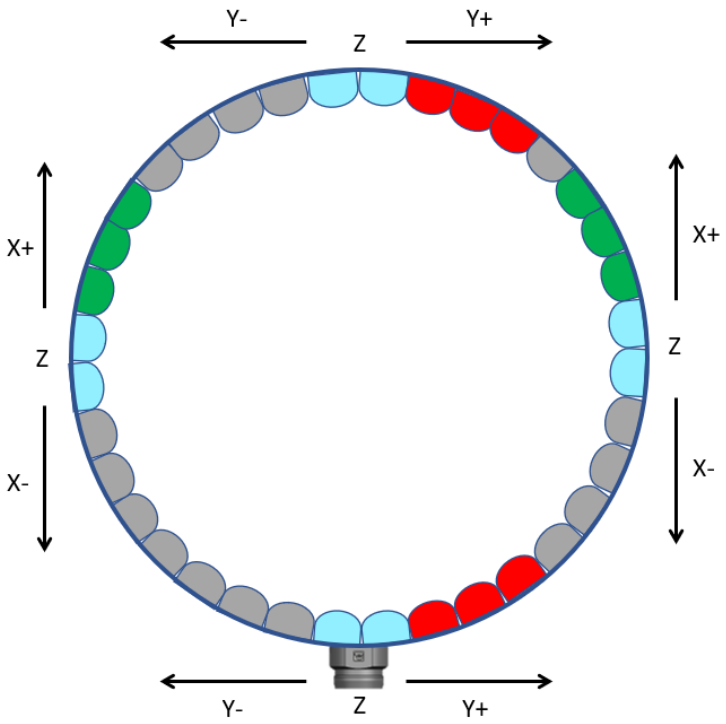
Examples for a possible representation can be found after the breakdown of the individual ranges. The mode can be used for teaching new positions during pickup, for example. You move to the part, unlock the SFE and grip the part. If the robot is not correctly positioned at the part, the SFE deflects. The robot then uses the LED response to move to the correct position for the part.

- x axis representation as a graph (green)
  For values ≤ 5% of the maximum deflection in x direction: 0 green LEDs
  For values > 5% and ≤ 66% of the maximum deflection in x direction: 1 green LED
  For values > 66% and ≤ 83% of the maximum deflection in x direction: 2 green LEDs
  For values > 83% of maximum deflection in x direction: 3 green LEDs
- y axis representation as a graph (red)
  For values ≤ 5% of the maximum deflection in y-direction: 0 red LEDs
  For values > 5% and ≤ 66% of the maximum deflection in y-direction: 1 red LED
  For values > 66% and ≤ 83% of the maximum deflection in y-direction: 2 red LEDs
  For values > 83% of the maximum deflection in y-direction: 3 green LEDs
- z axis representation as a graph (various blue shades)
  For values ≤ 5% of the maximum deflection in z-direction: white
  For values > 5% and ≤ 66% of the maximum deflection in z-direction: light blue
  For values > 66% and ≤ 83% of the maximum deflection in z-direction: blue
  For values > 83% of the maximum deflection in z-direction: dark blue

In the following, two examples for deflections and their visualization are shown in graph mode:

Axis	Current deflection in the range between
X	+2.49 to 3.00 mm
Y	+2.49 to 3.00 mm
Z	0 to -2.00mm

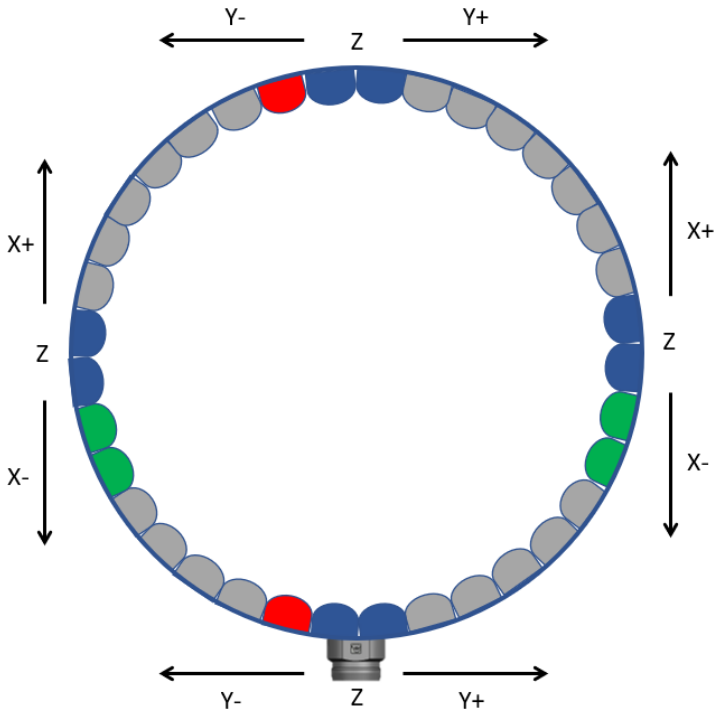
29 **Deflection of flange plate example 4**



19 **Example 4 for a possible LED pattern, top view**

axis	Current deflection in the range between
X	-+2.00 to -2.49 mm
Y	-0.15 to -2.00 mm
Z	-2.49 to -3.00 mm

30 **Deflection of flange plate example 5**



20 **Example 5 for a possible LED pattern, top view**

7.4.5 QUALITY

With the command SET;LED_MODE;QUALITY, the quality mode is shown with the LEDs.

Quality mode shows whether a deflection of the SFE is within a defined range or has left it. The exact type of the check is determined by the quality mode sub-modes, which are described in the chapter Quality mode.

Basically, there are three different states of the LED display:

- Deflection is within the defined range → green
- Deflection is outside the defined range → red
- Range not defined (no threshold values set) or quality mode off → blue

7.4.6 SENSOR

With the command SET;LED_MODE;SENSOR, sensor mode is shown.

It includes a representation of the sensor deflection with a color shift from minimum sensor value (green) to maximum sensor value (red). This LED mode is only used by the Bosch Rexroth service team for position sensor analyses. This is why this description does not go into more detail.

7.5 Quality mode

In quality mode, it is checked whether a deflection of the SFE is within a defined range or has left it. The range is determined by the command “SET;QUALITY_THRES”, see command description.

The quality mode has four different submodes which are described below. You can switch between the submodes using the “SET;QUALITY_MODE” command, see command description. The submodes are:

- QUALITY_TRIGGER
  The check whether the deflection of the SFE unit is within the range takes place when the command CTR;QUALITY_TRIGGER or a triggering by an IO pin is executed (see chapter 7.6). The result of the check remains in the respective state until there is a new trigger.

- QUALITY_CONTINUOUS
  The check whether the deflection of the SFE unit is within the range takes place continuously with every measurement.

- QUALITY_PEAK
  The check whether the deflection of the SFE unit is within the range takes place continuously with every measurement. In contrast to the “QUALITY_CONTINUOUS” mode, the evaluation will, however, remain in the “out of range” state when the boundaries of the range have been exceeded once. Reset is possible by executing the CTR;QUALITY_RESET command

- QUALITY_OFF
  Quality mode is switched off, no check of the deflection range takes place.

The quality mode is expressed in the LED display when the “QUALITY” LED mode (see chapter QUALITY) is activated and in the level of the IO pin that is activated as “IO_QUALITY_IN_RANGE” (see chapter Use and configuration of the digital IO pins).

7.6 Use and configuration of the digital IO pins

The SFE has three digital IO pins, the configuration of which is freely configurable by the user.

The pins have IDs and are numbered from 0 to 2. Pin 0 and pin 1 can be used as input and output. Pin 2 can only be used as input. The same input functionality cannot be used at two different pins at the same time.

The functionality is determined by the “SET;IO_CONFIG” command (see chapter Interface description). The options and their description are listed in the following table:

IO configuration	Input / output	Description
IO_LOCK	Input	Pin triggers an SFE locking or unlocking. (low = unlock, high = lock)
IO_OFF	Input	Pin is not used.
IO_QUALITY_TRIGGER	Input	Pin triggers a check in the Quality Trigger mode, i.e. the deflection of the SFE is compared to the active quality threshold data record. A check takes place after every level change at the pin, no matter in which direction.
IO_INPUT	Input	Pin is a general input.
IO_LOCK_STATE	Output	Pin indicates whether the SFE is locked or unlocked. (low = unlocked, high = locked)
IO_ERROR	Output	Pin indicates whether an error condition exists. (high = error pending, low = no error pending)
IO_QUALITY_IN_RANGE	Output	Pin indicates whether the current deflection of the SFE lies outside the active quality threshold data record. The exact type of the check depends on the quality mode sub-mode, see chapter Quality mode.
IO_OUTPUT	Output	Pin is a general output.

31 **Options and description of the digital IO pins**

If the generic input/output functions are used, the levels can be set using “SET;IO_STATE” and queried using “GET;IO_STATE”.

The default configuration is IO pin 0 as “IO_LOCK_STATE”, IO pin 1 as “IO_ERROR” and IO pin 2 as “IO_LOCK”.

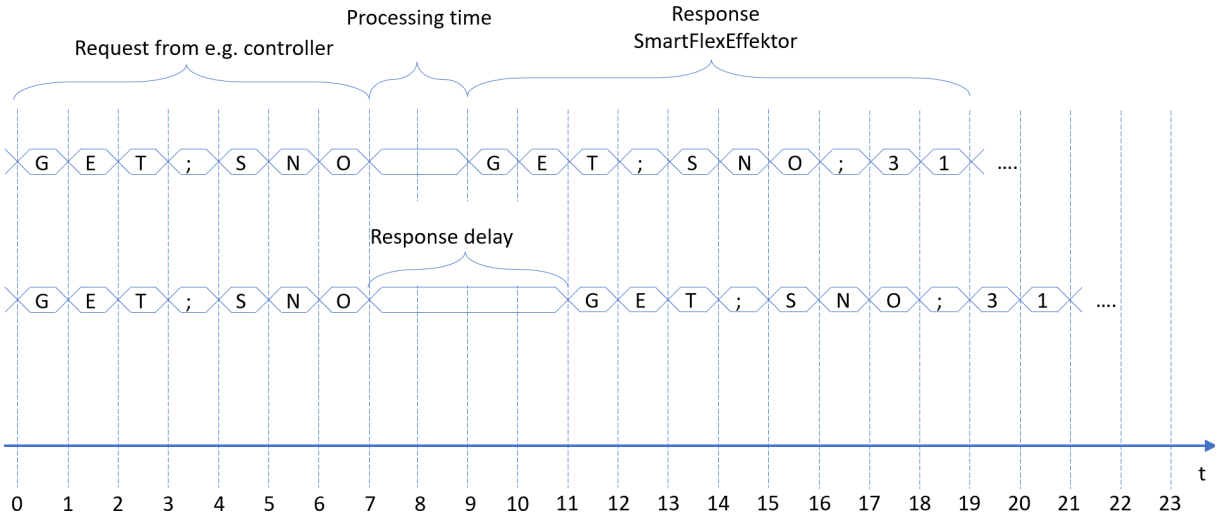
7.7 Setting the command delay

The command structure of the SFE is constructed in such a way that a command which is sent to the unit is always followed by a response. This occurs either immediately or when a related operation has been completed.
The “SET;CMD_DELAY;x” command can be used to set a delay of x word lengths, which the SFE unit will wait for after receiving the command before sending a response; you can set a value between 0 (default) and 10 word lengths. Using the command “GET;CMD_DELAY”, you can query the set delay.

In this connection, a word length is the length of a character which is transmitted via the serial RS485 interface, i.e. 8 bits and in each case one start and stop bit, i.e. a total of 10 bits. The target delay in seconds thus depends on the set baud rate and can be calculated using the following formula:

$$t = \frac{10}{\text{Baud rate}}\ s$$

The following graphic illustrates the procedure for a delay of 4 word lengths in the lower procedure as compared to the upper procedure without delay.



21 **Comparison procedure with delay and procedure without delay**

As shown, the processing time of the SFE is already included in the delay.

7.8 Data logger

Instead of reading out process data manually, these data for deflection and acceleration (e.g. as part of a process monitoring) can be collected in a defined grid on the SFE unit and retrieved afterwards.

The command to start the data logger is:

CTR;LOGGER_START;(interval)|(number of measurements)

The interval determines the frequency of the recording, the minimum and default value is the setting 5, the maximum is 100. Only multiples of 5 are allowed.
A maximum of 2000 data records can be saved. The number of measurements can be defined between 1 and 2000. If no number is specified, the measurement runs indefinitely; of 2000 values, the oldest data record is overwritten by the latest. When the logger starts, all existing data records are discarded.

The command to cancel a running recording is:

CTR;LOGGER_STOP.

The command to read out the data is:

GET;LOGGER_DATA.

The command to read whether the data logger is active (TRUE or FALSE) is:

GET;LOGGER_RUNNING.

For a description of the input and output parameters of the commands, see chapter Interface description.

8 Third-party license information

8.1 BSD 2-clause FreeBSD License

Component name	Version	Copyright
lmfit	8.2.2	Copyright © 1980-1999 University of Chicago 2004-2018 Joachim Wuttke, Forschungszentrum Juelich GmbH

32 **BSD 2-clause FreeBSD License**

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE FREEBSD PROJECT ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE FREEBSD PROJECT OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The views and conclusions contained in the software and documentation are those of the authors and should not be interpreted as representing official policies, either expressed or implied, of the FreeBSD Project.

8.2 BSD (Three Clause License)

Component name	Version	Copyright
STM32Cube HAL	6.2.0	Copyright © 2016 STMicroelectronics Copyright © 2009-2015 ARM Limited

33 **BSD (Three Clause License)**

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

8.3 RSA MD4 or MD5 Message-Digest Algorithm License

Component name	Copyright
md5	Copyright © 1991-2 RSA Data Security, Inc. Created 1991. All rights reserved.

34 **RSA MD4 or MD5 Message-Digest Algorithm License**

License to copy and use this software is granted provided that it is identified as the "RSA Data Security, Inc. MD4 or MD5 Message-Digest Algorithm" in all material mentioning or referencing this software or this function.

License is also granted to make and use derivative works provided that such works are identified as "derived from the RSA Data Security, Inc. MD4 or MD5 Message-Digest Algorithm" in all material mentioning or referencing the derived work.

RSA Data Security, Inc. makes no representations concerning either the merchantability of this software or the suitability of this software for any particular purpose. It is provided "as is" without express or implied warranty of any kind.

These notices must be retained in any copies of any part of this documentation and/or software.

8.4 MIT License

Component name	Copyright
stm32-bootloader	Copyright © 2018 Ferenc Németh - https://github.com/ ferenc-nemeth/
Example code in the customer documentation for calculating the checksum Modbus CRC16	Copyright © 1999-2016 Lammert Bies - https://github.com/lammertb/libcrc

35 **MIT License**

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

9 Disposal

The product and its components must be recycled correctly and in compliance with all applicable national and international guidelines and regulations. Collect any leaking lubricant and dispose of it properly.

9.1 Return

Products manufactured by Bosch Rexroth can be returned free-of-charge for disposal.

A prerequisite for this is that there are no objectionable films such as oil, grease or other contamination on the device. Moreover, the device must be free of inordinate foreign materials and/or components.

Please send the products to be returned carriage paid to the following address:

Bosch Rexroth AG

Linear Motion Technologies

Ernst-Sachs-Straße 100

97424 Schweinfurt

Germany

9.2 Packaging

Packaging materials basically consist of cardboard. They can be easily recycled. For ecological reasons, please refrain from returning the packaging.

9.3 Recycling

Due to the high metal content, the products can mostly be recycled. To achieve an optimum metal recovery, dismantling into individual assemblies is required.

10 Service and support

We have a dense global service network for fast and optimal support. Our experts will be happy to assist you in any way they can. You can reach us 24/7 – even on weekends and holidays.

Service Germany

You can reach our service hotline and our service helpdesk under:

Phone: +49 9352 40 5060

Fax: +49 9352 18 4941

Email: service.svc@boschrexroth.de

Internet: http://www.boschrexroth.com

Supplementary notes on service, repair work
(e.g. delivery addresses) and training can be found on our website.

International service

If you are located outside of Germany, please first contact your local service representative. For hotline numbers, please refer to the sales addresses online.

Preparation of information

We will be able to help you quickly and efficiently if you have the following information ready:

- A detailed description of the malfunction and conditions
- Information on the name plate of the affected product, particularly the type code and serial numbers
- Your contact information (phone and fax number and email address)

11 Appendix

11.1 Table of figures

11.2 Table of figures