

# Software manual

## SMP

### SCHUNK motion protocol V3.03

## Imprint

### Copyright:

This manual is protected by copyright. The author is SCHUNK GmbH & Co. KG. All rights reserved. Any reproduction, processing, distribution (making available to third parties), translation or other usage - even excerpts - of the manual is especially prohibited and requires our written approval.

### Technical changes:

We reserve the right to make alterations for the purpose of technical improvement.

**Document number:** 1006506

**Version:** 01.00 | 09/11/2017 | en

© SCHUNK GmbH & Co. KG

All rights reserved.

Dear Customer,

thank you for trusting our products and our family-owned company, the leading technology supplier of robots and production machines.

Our team is always available to answer any questions on this product and other solutions. Ask us questions and challenge us. We will find a solution!

Best regards,

Your SCHUNK team

SCHUNK GmbH & Co. KG  
Spann- und Greiftechnik

Bahnhofstr. 106 – 134  
D-74348 Lauffen/Neckar

Tel. +49-7133-103-0  
Fax +49-7133-103-2399

info@de.schunk.com  
schunk.com

## Table of contents

<b>1</b>	<b>General.....</b>	<b>7</b>
1.1	Presentation of Warning Labels .....	7
1.2	Applicable documents .....	7
1.3	User administration.....	7
1.3.1	USER .....	8
1.3.2	PROFI .....	8
1.3.3	Advanced .....	8
1.3.4	ROOT.....	8
1.3.5	SCHUNK .....	8
1.3.6	DISABLED .....	8
1.4	Unit system.....	9
1.5	Booting .....	9
<b>2</b>	<b>Software function .....</b>	<b>10</b>
2.1	Pseudo-absolute encoder.....	10
2.1.1	Precondition .....	10
2.1.2	Function.....	10
2.2	Standstill commutation .....	11
2.2.1	Precondition .....	11
2.2.2	Function .....	11
<b>3</b>	<b>Communication .....</b>	<b>12</b>
3.1	Data format .....	12
3.2	Data frame.....	12
3.3	Communication via USB .....	13
3.4	Communication via CAN bus .....	13
3.5	Communication via PROFIBUS.....	14
3.6	Fragmentation .....	15
<b>4</b>	<b>Commands.....</b>	<b>17</b>
4.1	Movement.....	17
4.1.1	CMD REFERENCE.....	17
4.1.2	MOVE POS .....	18
4.1.3	MOVE POS REL.....	19
4.1.4	MOVE POS TIME .....	20
4.1.5	Move pos time rel.....	21
4.1.6	MOVE VEL .....	22
4.1.7	SET TARGET VEL.....	23
4.1.8	SET TARGET ACC .....	23
4.1.9	SET TARGET JERK .....	23
4.1.10	SET TARGET CUR .....	24
4.1.11	SET TARGET POS .....	24
4.1.12	SET TARGET POS REL.....	25

4.1.13	CMD STOP .....	25
4.1.14	CMD FAST STOP .....	26
4.2	Spontaneous messages .....	26
4.2.1	CMD INFO .....	26
4.2.2	CMD MOVE BLOCKED .....	27
4.2.3	CMD POS REACHED .....	27
4.2.4	CMD ERROR .....	28
4.2.5	GET STATE .....	28
4.2.6	CMD TOGGLE IMPULSE MESSAGE .....	30
4.3	Settings .....	30
4.3.1	SET CONFIG EXT .....	30
4.3.2	GET CONFIG EXT .....	31
4.3.3	CALIB CURRENT .....	32
4.4	Other .....	32
4.4.1	CMD REBOOT .....	32
4.4.2	CHANGE USER .....	33
4.4.3	CMD DIO .....	33
4.5	Fragmentation .....	34
4.5.1	FRAG ACK .....	34
4.5.2	FRAG START .....	34
4.5.3	FRAG MIDDLE .....	34
4.5.4	FRAG END .....	35
4.6	Troubleshooting .....	35
4.6.1	CMD ERROR .....	35
4.6.2	CMD WARNING .....	36
4.6.3	CMD INFO .....	36
4.6.4	CMD ACK .....	36
<b>5</b>	<b>Configuration parameters .....</b>	<b>37</b>
5.1	Value range .....	37
5.2	Parameter code display .....	37
5.3	Parameter .....	38
5.3.1	Device .....	38
5.3.2	Motor .....	43
5.3.3	Controller .....	51
5.3.4	Referencing .....	55
5.3.5	Positioning .....	62
5.3.6	Gear .....	69
5.3.7	Brake .....	70
5.3.8	Voltage .....	72
5.3.9	Communication .....	73
5.3.10	General .....	75
5.3.11	Info .....	77
5.3.12	Asynchronous .....	82

<b>6</b>	<b>Info and error messages.....</b>	<b>93</b>
6.1	Detailed error information .....	93
6.2	Info codes .....	93
6.2.1	INFO BOOT.....	93
6.2.2	INFO NO RIGHTS .....	93
6.2.3	INFO UNKNOWN COMMAND .....	94
6.2.4	INFO FAILED.....	94
6.2.5	NOT REFERENCED .....	94
6.2.6	INFO SEARCH SINE VECTOR .....	94
6.2.7	INFO NO ERROR.....	94
6.2.8	INFO COMMUNICATION ERROR .....	94
6.2.9	INFO TIMEOUT.....	95
6.2.10	INFO WRONG DATA TYPE .....	95
6.2.11	INFO RESTART.....	95
6.2.12	INFO CHECKSUM.....	95
6.2.13	INFO VALUE LIMIT MAX.....	95
6.2.14	INFO VALUE LIMIT MIN.....	95
6.2.15	INFO MESSAGE LENGTH .....	95
6.2.16	INFO WRONG PARAMETER.....	95
6.2.17	INFO UNKNOWN PARAMETER.....	96
6.3	Error codes .....	96
6.3.1	ERROR FILE NOT FOUND.....	96
6.3.2	ERROR FILE IS CORRUPT .....	96
6.3.3	ERROR FILE TYPE WRONG.....	96
6.3.4	ERROR FILE SYSTEM WRONG.....	96
6.3.5	ERROR FILE READ .....	96
6.3.6	ERROR FILE IS NOT CREATED .....	96
6.3.7	ERROR FILE WRITE .....	96
6.3.8	ERROR REBOOT.....	97
6.3.9	ERROR MOTOR PHASE.....	97
6.3.10	ERROR WRONG RAMP TYPE .....	97
6.3.11	ERROR WRONG DIRECTION .....	97
6.3.12	ERROR CONFIG MEMORY .....	97
6.3.13	ERROR SOFT LOW .....	97
6.3.14	ERROR SOFT HIGH .....	97
6.3.15	ERROR SERVICE.....	98
6.3.16	ERROR FAST STOP .....	98
6.3.17	ERROR TOW .....	98
6.3.18	ERROR VPC3.....	98
6.3.19	ERROR FRAGMENTATION .....	98
6.3.20	ERROR COMMUTATION.....	99
6.3.21	ERROR I2T .....	99
6.3.22	ERROR CURRENT.....	99
6.3.23	ERROR TOO FAST .....	99

6.3.24	ERROR POS SYSTEM.....	99
6.3.25	ERROR RESOLVER CHECK FAILED .....	99
6.3.26	ERROR MATH.....	100
6.3.27	ERROR CALIB CURRENT .....	100
6.3.28	ERROR INITIALIZE.....	100
6.3.29	ERROR INTERNAL.....	100
6.3.30	ERROR CONNECTION TEMP LOW .....	100
6.3.31	ERROR CONNECTION TEMP HIGH .....	101
6.3.32	ERROR MOTOR TEMP LOW .....	101
6.3.33	ERROR MOTOR TEMP HIGH.....	101
6.3.34	ERROR TEMP LOW OPTION .....	101
6.3.35	ERROR TEMP HIGH OPTION.....	101
6.3.36	ERROR TEMP LOW .....	102
6.3.37	ERROR TEMP HIGH .....	102
6.3.38	ERROR LOGIC LOW .....	102
6.3.39	ERROR LOGIC HIGH.....	102
6.3.40	ERROR MOTOR VOLTAGE LOW .....	102
6.3.41	ERROR MOTOR VOLTAGE HIGH.....	103
6.3.42	ERROR CABLE BREAK .....	103
6.3.43	ERROR LIFE SIGN.....	103
6.3.44	ERROR CUSTOM DEFINED.....	103
6.3.45	ERROR OVERSHOOT.....	103
6.3.46	ERROR HARDWARE VERSION .....	104
6.3.47	ERROR SOFTWARE VERSION.....	104

## 1 General

### 1.1 Presentation of Warning Labels

To make risks clear, the following signal words and symbols are used for safety notes.



#### **⚠ DANGER**

##### **Danger for persons!**

Non-observance will inevitably cause irreversible injury or death.



#### **⚠ WARNING**

##### **Dangers for persons!**

Non-observance can lead to irreversible injury and even death.



#### **⚠ CAUTION**

##### **Dangers for persons!**

Non-observance can cause minor injuries.

#### **NOTICE**

##### **Material damage!**

Information about avoiding material damage.

### 1.2 Applicable documents

- General terms of business\*
- "SCHUNK Motion Tool (MTS)" software manual \*
- Documentation for the products used \*

The documents marked with an asterisk (\*) can be downloaded on our homepage **[schunk.com](https://www.schunk.com)**

### 1.3 User administration

The module is equipped with a user administration to specifically protect certain actions. The user rights can be changed via "CHANGE USER" [CHANGE USER](#) [► 33] or by the "User" parameter (0x7DDA) [User](#) [► 92].

#### **NOTE**

After restarting the module, the user remains active as "USER".

### **1.3.1 USER**

The standard user, active after the module has been switched on. The module can only be parameterized to a limited extent, but has full operational features.

### **1.3.2 PROFI**

Many important parameters can be changed.  
The standard password for the professional rights is "Schunk".

---

#### **NOTE**

Incorrect parameterization can result in an unanticipated module behavior. However, the module cannot be destroyed.

---

### **1.3.3 Advanced**

The most important parameters can be changed.

---

#### **NOTE**

Incorrect parameterization can result in destruction of the module.

---

### **1.3.4 ROOT**

Full access to the module, all parameters can be changed.

---

#### **NOTE**

Incorrect parameterization can result in destruction of the module.

---

### **1.3.5 SCHUNK**

Parameters can only be changed by SCHUNK.

### **1.3.6 DISABLED**

Parameters with cannot be changed.



## 1.4 Unit system

For linear and gripper products, the following unit system applies:

- Position [mm]
- Speed [mm/s]
- Acceleration [mm/s<sup>2</sup>]
- Jerk [mm/s<sup>3</sup>]
- Current values [A]
- Times [s]

For rotary units, the following unit system applies:

- Position [degree]
- Speed [degree/s]
- Acceleration [degree/s<sup>2</sup>]
- Jerk [degree/s<sup>3</sup>]
- Current values [A]
- Times [s]

## 1.5 Booting

Default values for motions are predefined for the module as standard values. This allows the module to be commissioned directly without having to set the parameters beforehand. The following default values apply after the restart:

- "Target speed"  
as [%] of the maximum value -> 10%, [Max. speed](#) [► 46]
- "Target acceleration"  
as [%] of the maximum value -> 10%, [Max. acceleration](#) [► 46]
- "Target jerk"  
as [%] of the maximum value -> 50%, [Max. jerk](#) [► 47]
- "Target current"  
Rated current, [Nom. Current](#) [► 46]
- Spontaneous messages activated.
- User is set to "User" [User administration](#) [► 7]

## 2 Software function

### 2.1 Pseudo-absolute encoder

#### 2.1.1 Precondition

If the following preconditions are met, then the modules support the "pseudo absolute encoder" function:

- Brake available
- Position measuring system resolver [Positioning type](#) [► 62]
- OR: position measuring system encoder with index track and
  - DC motor type, [Motor Type](#) [► 44]
  - OR: BLDC motor type
  - OR: PMSM motor type

#### 2.1.2 Function

When the brake is applied, the current position is saved in a non-volatile memory. If the logic voltage is switched off, an attempt is made to save the current position with the remaining residual energy.

##### Resolver

When the module is switched on again, the position saved beforehand will then be compared with a control value. Should this check be successful, then the position saved will be compared with the current position of the resolver. If these positions are also equal, then the module does not need to be re-referenced.

If the resolver is turned by precisely one revolution when deenergized, the displayed position will be faulty when reactivation takes place.

##### Encoder with index track

When the module is switched on again, the position saved beforehand will be compared with a control value. Should this check be successful, then the saved position will become the current position. The interval between the next index pulse is calculated at the same time.

In the first motion command that follows, the calculated interval is compared with the measured interval when the index pulse is reached. The module is deemed as referenced if the two values agree with each other. In addition, the index pulse must be reached within a certain period of time after the first motion command has been sent.

If an error occurs during the movement to the index pulse, then the referencing will be deleted.

After a successful reference movement, the index pulse has to be run over at least once in order to activate the function.

If the encoder is moved in a deenergized state, it is possible that the module will carry out a motion of no more than one motor revolution to the next index pulse after it has been switched on again with the incorrect position.

If the encoder is turned by precisely one revolution when deenergized, the displayed position could be faulty when reactivation takes place.

## 2.2 Standstill commutation

### 2.2.1 Precondition

If the following preconditions are met, then the modules support the "standstill commutation" function:

- DC motor type, [Motor Type](#) [► 44]
- OR: BLDC motor type
- OR: PMSM motor type
  - Position measurement system "Encoder with index track" and available hall sensors [Positioning type](#) [► 62]
  - OR: position measuring system resolver

#### NOTE

The direction of motion for block commutation and sine commutation must agree. If the directions of rotation are different, then the phases must be changed and the commutation table [Commutation table](#) [► 49] adjusted.

### 2.2.2 Function

If all prerequisites are met, then the module will attempt to carry out the standstill commutation. In modules with absolute-value measuring systems, the sine commutation can be activated directly after switch-on, since the position of the "sine pointer" is known.

In modules with an encoder measuring system, the position of the "sine pointer" is not known until the index pulse has been reached. Therefore, the module is moved with block commutation up to the first index pulse and then converted to sine commutation.

The position of the "sine pointer" to the index pulse is set or readjusted using the parameter *positioning offset* Positioning. If this value is set to "0", then a "sine pointer" - which is *saved* Positioning- is searched for in the next motion command by energizing the motor phases. The *referencing* [Pseudo-absolute encoder](#) [► 10] is deleted in the process.

#### NOTE

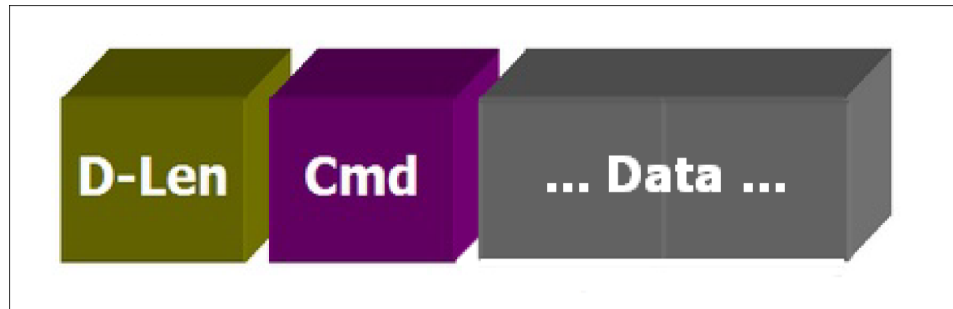
The module should be able to move freely in all directions for the pointer search. The module is moved in a jerking fashion up to two motor revolutions. Communication with the module is not possible during this time.

### 3 Communication

#### 3.1 Data format

Data is sent from the modules in the Intel format (little-endian) and interpreted in this same format when being received.

#### 3.2 Data frame



*Data frame*

The data frame of the SCHUNK motion protocol always includes the following elements:

- D-Len (length byte) - 1 byte
  - Specifies the number of subsequent items of user data including the command byte.
- Command code (1 byte)
  - Specifies the command to run. The command code can also be extended further using "Subcommand codes".
- Command parameters - optional, variable lengths
  - Specifies any other required parameters.

The data frame is used for the following messages from the SCHUNK motion protocol:

- Commands to the module
- Responses of the module
- Spontaneous message of the module

A motion protocol message can transfer a maximum of 254 data bytes, as the D-Len is 1 byte wide.

All commands are confirmed by the module with a response. If the command has been successfully processed, D-Len always has a value that is not equal to "0x02". If the command failed, D-Len has the value "0x02".

An "spontaneous message" is triggered in the following events:

- When a motion has been concluded correctly.
- When a serious error has occurred.
- When there are regular status messages, if activated, [Spontaneous messages](#) [► 26].

### 3.3 Communication via USB

USB communication is only designed for parameterization and commissioning. Use the "Motion Tool SCHUNK" software via USB to access the "Motion Tool SCHUNK (MTS)" software manual.

### 3.4 Communication via CAN bus

In addition to the data frame, CAN bus requires a unique identifier allocated to the message. The modules support the standard 11-bit identifier. The lower 8 bits are used here for the unique module ID. A maximum of 255 modules can be addressed.

The remaining 3 bits are coded as follows:

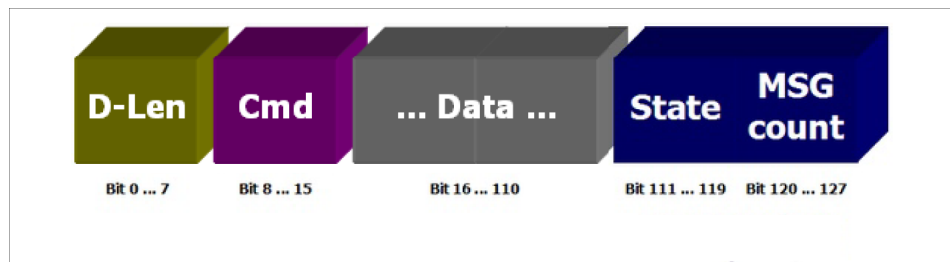
- 0x03 - Module error message
- 0x05 - Command from the control unit to the module
- 0x07 - Response from module to control unit

The following identifiers are used (XX module address in hex format):

- Module error message - identifier 0x3XX
- Command from the control unit to the module - Identifier 0x5XX
- Response from module to control unit - Identifier 0x7XX

A maximum of 8 bytes can be sent in a CAN message. If it is necessary to package a longer data frame (D-Len > 7) into several CAN messages, this happens via fragmentation, [Fragmentation](#) [► 15]. Fragmentation is normally not required, since all commands needed to operate the modules can be packed into one CAN message.

### 3.5 Communication via PROFIBUS



*Data frame for Profibus*

The maximum length of the data to be transmitted all at once from the control unit to the module is limited to 8 bytes. This is sufficient to fully operate a module, as 7 bytes are required for a message.

The maximum length of the data transmitted from the control unit to the module is limited to 16 bytes. If larger amounts of data are to be sent/received, this happens via fragmentation, [Fragmentation](#) [► 15].

The longest message in normal operation occurs in 16 bytes from the module to the controller, leaving two occupied bytes.

The last 2 bytes are occupied as follows:

1. the current status of the module (byte 14), [GET STATE](#) [► 28]
2. Command counter "MsgCount" (byte 15)

When all data (position, speed, and current) have been requested, bytes 10-13 are used only in response to a status message, [GET STATE](#) [► 28].

Bytes 10-15 are used for data in fragmented messages.

If bytes 10-13 are not required, the current position in the respective unit system can be read out here, [Unit system](#) [► 9].

#### NOTE

Only the upper 8 bits of the status word are written. The error code is not used. You can use the extended diagnosis under PROFIBUS for this on the one hand, and on the other, the error code is displayed in the output data in the event of an error, [Troubleshooting](#) [► 35].

If a message is sent from the master to the module, then with PROFIBUS the MsgCount is increased by 1 in addition to the response. This ensures that every request is confirmed despite possible spontaneous messages.

#### NOTE

A spontaneous message [Spontaneous messages](#) [► 26] does not increase the MsgCount.

If, for example, you want to move to a position where the module is currently located, the module will signal "Command understood" and "Position reached" immediately in the next PROFIBUS cycle. Since in some circumstances a control unit connected to the PROFIBUS does not query the data in every PROFIBUS cycle, the acknowledgment of (response to) the motion command could go missing. The MsgCount ensures that a confirmation of the request was received.

---

**NOTE**

The last bit for MsgCount can be evaluated as a toggle bit. During data transmission from the control unit to the module, the unused byte can be used as toggle byte, or bit 63 as toggle bit.

---

Groups are fully supported by the SYNC / FREEZE mechanism implemented in PROFIBUS.

If consistent data transmission is not possible, then the following options can be used to operate the module:

- Using the SYNC, UNSYNC mechanism.
- Set D-Len to "0". Populate all data and set the D-Len as soon as all data are present.

### 3.6 Fragmentation

---

**NOTE**

The fragmentation of messages is not required for normal operation.

---

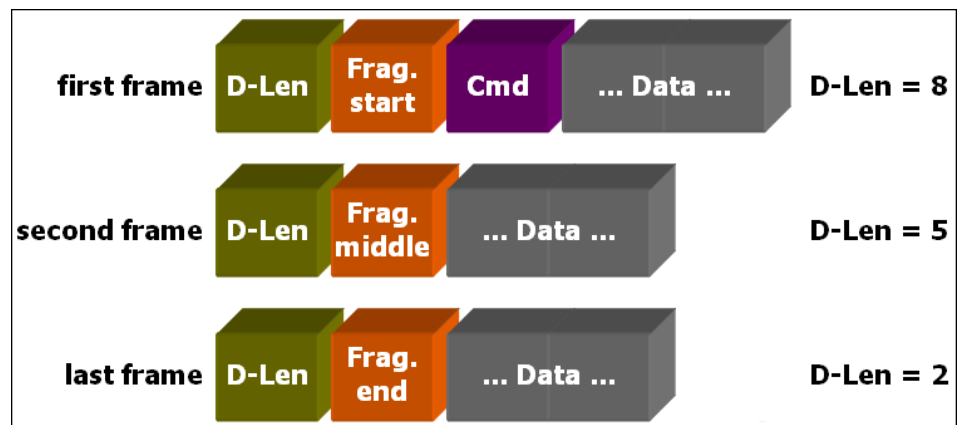
If a fragmentation of messages is necessary, the length of the bytes that follow and afterward a fragment identifier are sent at the beginning of each message. This fragment identifier is not recorded in the D-Len (length byte).

Structure of fragmentation:

- FragStart -> *First fragment*
- FragMiddle -> *A middle fragment*
- FragEnd -> *Last fragment*

The sending process divides the data frame into individual fragments. These individual fragments are reassembled into a complete data frame that can then be interpreted.

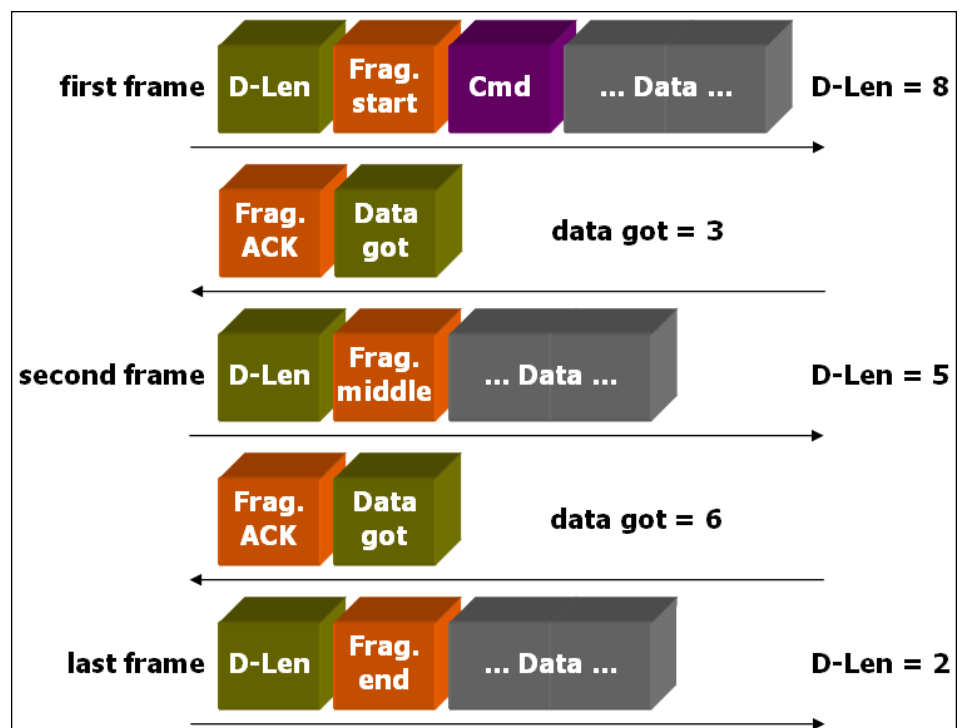
### Fragmentation CAN bus



*Fragmentation*

A confirmation of the fragments is not necessary.

### PROFIBUS fragmentation



*PROFIBUS fragmentation*

Each fragment must be confirmed with the command "FRAG ACK" as well as the D-Len byte of the fragment obtained. The last fragment does not have to be confirmed.



## 4 Commands

### 4.1 Movement

#### 4.1.1 CMD REFERENCE

A reference run is executed.

Code: **0x92**

Access rights: USER

- Command from the control unit to the module
  - No command parameter
- Reply from module to control unit
  - "OK" (0x4F 0x4B) if successful. The module executes the command.
- Spontaneous message of the module
  - A spontaneous message from the module is possible if, depending on the type of referencing, an interruption of the movement command or a position movement occurs, [CMD MOVE BLOCKED](#) [► 27] and [CMD POS REACHED](#) [► 27]. If the "MOVE ZERO AFTER REFERENCING" flag is set, a positioning movement is triggered after referencing, [Approach 0 after referencing](#) [► 60].

---

#### NOTE

Prior to a reference movement, all workpieces must be removed for a gripper.

The type of referencing is defined by the "Referencing type" parameter. Observe the special features of the respective position measuring system, [Positioning type](#) [► 62].

During referencing, the set parameter values are accepted for the movements. After referencing, the values set for the movement to 0 in the process data are used.

Under certain conditions, a successfully performed referencing process is retained even after the module has been switched off [Pseudo-absolute encoder](#) [► 10].

#### 4.1.2 MOVE POS

Moves the module to a fixed position.

Code:	<b>0xB0</b>
Access rights	USER

- Command from the control unit to the module
  - Position (optional), in the configured unit system.
  - Speed (optional), which is used for the positioning movement. Not relevant for the "No ramp" motion profile.
  - Acceleration (optional), which is used for the positioning movement. Not relevant for the "No ramp" motion profile.
  - Current (optional), which may not be exceeded. If the controller structure is "Current speed", then this value must be transferred, since jerk is required. The value must be at least "0", otherwise an info message will be issued, [INFO WRONG PARAMETER](#) [► 95].
  - Jerk (optional), which is used for the positioning movement. Should the motion profile not be equal to "jerk-limited", then this value cannot also be transferred. An info message is issued, [Info and error messages](#) [► 93].
- Reply from module to control unit
  - If possible, the time that the module is expected to need for the movement is returned. If a calculation of the time is not possible, then the request will be confirmed with "OK" (0x4F 0x4B) in the event of success, and the module will execute the movement.
- Spontaneous message of the module
  - When the "CMD POS REACHED" position is reached or when the "MOVE BLOCKED" positioning movement is blocked.

The change of position is set in the configured unit system, [Unit system](#) [► 9]. The positioning movement is based on the configured motion profile, [Position ramp](#) [► 67].

If the motion profile is configured to "jump", the jerk-limited motion profile is used, otherwise the drive will experience violent movements.

All parameters must be transferred in the specified order. If only the current is to be specified, then it is vital that the position change, speed, and acceleration are also specified. The following parameters do not have to be transferred as well. All parameters are retained up to the restart or until these parameters are changed.

### 4.1.3 MOVE POS REL

The module moves a specified distance.

Code:	<b>0xB8</b>
Access rights	USER

- Command from the control unit to the module
  - Position change (optional), in the configured unit system
  - Speed (optional), which is used for the positioning movement. Not relevant for the *"No ramp"* motion profile.
  - Acceleration (optional), which is used for the positioning movement. Not relevant for the *"No ramp"* motion profile.
  - Current (optional), which may not be exceeded. If the controller structure is "Current speed", then this value must be transferred, since jerk is required. The value must be at least "0", otherwise an info message will be issued, [INFO WRONG PARAMETER](#) [► 95].
  - Jerk (optional) which is used for the positioning movement. Should the motion profile not be equal to "jerk-limited", then this value cannot also be transferred. An info message is issued, [Info and error messages](#) [► 93].
- Reply from module to control unit
  - If possible, the time that the module is expected to need for the movement is returned. If a calculation of the time is not possible, then the request will be confirmed with "OK" (0x4F 0x4B) in the event of success, and the module will execute the movement.
- Spontaneous message of the module
  - When the "CMD POS REACHED" position is reached or when the "MOVE BLOCKED" positioning movement is blocked.

The change of position is set in the configured unit system, [Unit system](#) [► 9]. The positioning movement is based on the configured motion profile, [Position ramp](#) [► 67].

If the motion profile is configured to "jump", the jerk-limited motion profile is used, otherwise the drive will experience violent movements.

All parameters must be transferred in the specified order. If only the current is to be specified, then it is vital that the position change, speed, and acceleration are also specified. The following parameters do not have to be transferred as well. All parameters are retained up to the restart or until these parameters are changed.

#### 4.1.4 MOVE POS TIME

Moves the module to a fixed position (optional: within a specified time).

Code:	<b>0xB1</b>
Access rights	USER

- Command from the control unit to the module
  - Position (optional), in the configured unit system. The position must be offset by the position variation when compared with the start position.
  - Speed (optional), which may not be exceeded.
  - Acceleration (optional), which may not be exceeded.
  - Current (optional), which may not be exceeded. If the controller structure is "Current speed", then this value cannot be transferred as well.
  - Time (optional), in which the positioning movement is to be concluded in accordance with the speed and acceleration.
- Reply from module to control unit
  - The time that the module needs for the movement is returned.
- Spontaneous message of the module
  - When the "CMD POS REACHED" position is reached or when the "MOVE BLOCKED" positioning movement is blocked.

The change of position is set in the configured unit system, [Unit system](#) [► 9]. During the movement, new positions can be preset, which are then immediately moved to. The calculation takes into account the set speed and set acceleration as well as the current actual speed and actual acceleration. If the time parameter is also given, then the speed and acceleration are adjusted such that the position is reached in the specified time while taking the specified speed and acceleration limits into account.

All parameters must be transferred in the specified order. If only the current is to be specified, then it is vital that the position change, speed, and acceleration are also specified. The following parameters do not have to be transferred as well. All parameters are retained up to the restart or until these parameters are changed.

---

#### NOTE

It is possible to move along curved paths. Due to calculating time problems for curved paths, the motion profile is temporarily changed to "trapezoid".

---

**NOTE**

If the motion profile "jump" is used, the internal ramp generator switches off. This enables you to specify your own position ramps externally. An adjustment to the controller parameters could be necessary depending on the interpolation interval of the external specification.

**4.1.5 Move pos time rel**

The module moves along a predetermined distance (optional: within a predetermined time).

Code:	<b>0xB9</b>
Access rights	USER

- Command from the control unit to the module
  - Position change (optional), in the configured unit system The change of position may not be smaller than the position variation.
  - Speed (optional), which may not be exceeded.
  - Acceleration (optional), which may not be exceeded.
  - Current (optional), which may not be exceeded. If the controller structure is "Current speed", then this value cannot be transferred as well.
  - Time (optional), in which the positioning movement is to be concluded in accordance with the parameterized speed and acceleration limits.
- Reply from module to control unit
  - The time that the module needs for the movement is returned.
- Spontaneous message of the module
  - When reaching the position "CMD POS REACHED".

The change of position is set in the configured unit system, [Unit system](#) [► 9]. During the movement, new positions can be preset, which are then immediately moved to. The calculation takes into account the set speed and set acceleration as well as the current actual speed and actual acceleration. If the time parameter is also given, then the speed and acceleration are adjusted such that the position is reached in the specified time while taking the specified speed and acceleration limits into account.

All parameters must be transferred in the specified order. If only the current is to be specified, then it is vital that the position change, speed, and acceleration are also specified. The following parameters do not have to be transferred as well. All parameters are retained up to the restart or until these parameters are changed.

### NOTE

It is possible to move along curved paths. Due to calculating time problems for curved paths, the motion profile is temporarily changed to "trapezoid".

---

### NOTE

If the motion profile "jump" is used, the internal ramp generator switches off. This enables you to specify your own position ramps externally. An adjustment to the controller parameters could be necessary depending on the interpolation interval of the external specification.

---

#### 4.1.6 MOVE VEL

A speed movement is executed.

Code:	<b>0xB5</b>
Access rights	USER

- Command from the control unit to the module
  - Speed in the configured unit system, [Unit system](#) [► 9].
  - Current (optional), which may not be exceeded in the speed movement, [INFO WRONG PARAMETER](#) [► 95].
- Reply from module to control unit
  - "OK" (0x4F 0x4B) if successful.
- Spontaneous message of the module
  - Spontaneous message "CMD MOVE BLOCKED" is possible if the module was blocked during the movement, [CMD MOVE BLOCKED](#) [► 27].

#### 4.1.7 SET TARGET VEL

The speed is set.

Code:	<b>0xA0</b>
Access rights	USER

- Command from the control unit to the module
  - Speed in the specified unit system.
- Reply from module to control unit
  - "OK" (0x4F 0x4B) if successful.

This value is retained after it has been written successfully once until the module is restarted or this value is changed again. The value is not adopted until the next motion command and is not immediately active.

#### 4.1.8 SET TARGET ACC

The acceleration is set.

Code:	<b>0xA1</b>
Access rights	USER

- Command from the control unit to the module
  - Acceleration in the specified unit system.
- Reply from module to control unit
  - "OK" (0x4F 0x4B) if successful.

This value is retained after it has been written successfully once until the module is restarted or this value is changed again. The value is not adopted until the next motion command and is not immediately active.

#### 4.1.9 SET TARGET JERK

The jerk is set.

Code:	<b>0xA2</b>
Access rights	USER

- Command from the control unit to the module
  - Jerk in the specified unit system.
- Reply from module to control unit
  - "OK" (0x4F 0x4B) if successful.
- Spontaneous message of the module
  - None

This value is retained after it has been written successfully once until the module is restarted or this value is changed again. The value is not adopted until the next motion command and is not immediately active.

#### 4.1.10 SET TARGET CUR

The current is set.

Code:	<b>0xA3</b>
Access rights	USER

- Command from the control unit to the module
  - Current in the specified unit system.
- Reply from module to control unit
  - "OK" (0x4F 0x4B) if successful.

The target specifications for the current can be changed during the movement. With grippers, the gripping force can be increased or reduced in the process. This is only considered for movements if permitted by the controller structure, [Structure](#) [► 54].

This value is retained after it has been written successfully once until the module is restarted or this value is changed again. The value is not adopted until the next motion command and is not immediately active.

#### 4.1.11 SET TARGET POS

The position for the next command "MOVE POS" and "MOVE POS TIME" is set.

Code:	<b>0xA6</b>
Access rights	USER

- Command from the control unit to the module
  - Relative position in the specified unit system
- Reply from module to control unit
  - "OK" (0x4F 0x4B) if successful.

This value is retained after it has been written successfully once until the module is restarted or this value is changed again. The value is not adopted until the next motion command and is not immediately active.



#### 4.1.12 SET TARGET POS REL

The relative position for the next command "MOVE POS REL" and "MOVE POS TIME REL" is set.

Code: **0xA7**  
Access rights: USER

- Command from the control unit to the module
  - Relative position in the specified unit system
- Reply from module to control unit
  - "OK" (0x4F 0x4B) if successful.

This value is retained after it has been written successfully once until the module is restarted or this value is changed again. The value is not adopted until the next motion command and is not immediately active.

#### 4.1.13 CMD STOP

The module is braked and stopped in the current position.

Code: **0x91**  
Access rights: USER

- Command from the control unit to the module
  - No command parameter
- Reply from module to control unit
  - "OK" (0x4F 0x4B) if successful.

The brake will be immediately activated in modules with an appropriately configured holding brake, otherwise, the module will be actively controlled.

In modules without a brake or appropriately configured holding brake, the maximum permissible current for the control is limited to the nominal current to prevent overheating of the motor, [Nom. Current](#) [► 46]. This is why the module may "stall".

#### 4.1.14 CMD FAST STOP

A fast stop will be triggered.

Code:	<b>0x90</b>
Access rights	USER

- Command from the control unit to the module
  - No command parameter
- Reply from module to control unit
  - Error message "ERROR FAST STOP" is triggered, [Error codes](#) [► 96].

The motor is switched to a de-energized state during a fast stop. The holding brake is activated immediately for modules with an appropriately configured holding brake, and the motor phases are short circuited. A fast stop during a movement can lead to significant mechanical wear of the brake.

The fast stop can only be reset using the command "CMD ACK", [Troubleshooting](#) [► 35].

---

#### NOTE

In modules without a brake, the module may "stall" since the motor is de-energized during a fast stop.

---

### 4.2 Spontaneous messages

The module independently reports spontaneous messages for certain events. These messages are sent via the standard data frame. Spontaneous messages can be deactivated, [CMD TOGGLE IMPULSE MESSAGE](#) [► 30].

In PROFIBUS, the MsgCount is not increased for these types of messages, since no data has been requested by the control unit.

#### 4.2.1 CMD INFO

The module sends an info message, [INFO BOOT](#) [► 93], [INFO NO ERROR](#) [► 94] and [INFO UNKNOWN PARAMETER](#) [► 96].

Code:	<b>0x8A</b>
-------	-------------

- Spontaneous message of the module
  - Info code

#### 4.2.2 CMD MOVE BLOCKED

The current motion command was interrupted.

Code: **0x93**

- Spontaneous message of the module
  - Current position in the set unit system

The motor was briefly blocked or is still blocked. The motor is deemed as blocked if all of the following prerequisites are met:

- The motor turns with a speed below the movement threshold, [Motion threshold](#) [► 66].
- The target current is reached (+/-15%).
- The time parameterized in the parameter "Wait time motion blocked" has expired, [Motion waiting time blocked](#) [► 68].

When a block is detected, the set current is reduced by root (2) during sine commutation - form factor effective value to peak value for sinusoidal currents.

---

#### NOTE

Reliable detection as to whether an object has been gripped is not possible using this.

---

#### 4.2.3 CMD POS REACHED

A positioning movement has reached the target position.

Code: **0x94**

- Spontaneous message of the module
  - Current position in the set unit system.

The module has stopped. If the brake is present, it will be engaged depending on the configuration, [Brake usage](#) [► 71].

#### 4.2.4 CMD ERROR

A serious error has occurred that makes user intervention necessary.

Code: **0x88**

- Spontaneous message of the module
  - Error code

This error must be acknowledged with "CMD ACK", [CMD ACK](#) [► 36]. The module is not ready for operation. The power to the motor is switched off and the brake is applied. The error messages are sent regularly every 15 seconds from the module to the control unit until the error is acknowledged.

With CAN, the first three bits result in the identifier 0x3. With PROFIBUS, an extended diagnosis is generated

#### 4.2.5 GET STATE

Displays the status and other information of the module.

Code: **0x95**

Access rights **USER**

- Command from the control unit to the module
  - No command parameter  
Module provides the data once. This can be used to switch off the previously set cyclical sending of data.
  - Time (4 bytes) Options (1 byte)  
The module independently transmits its status at the time interval entered (in the respective *unit system* 18).  
The code "Options" is used to parameterize which data should be supplied in addition to the status:  
Bit 1 (0x01): Position  
Bit 2 (0x02): Speed  
Bit 3 (0x04): Current
- Reply from module to control unit
  - Optional data (n byte) and the status (2 bytes)
- Spontaneous message of the module
  - Optional data (n byte) and the status (2 bytes)

*Module status*

Bit 1	Referenced	0x01	The module is referenced.
Bit 2	Movement	0x02	The module moves.
Bit 3	Program sequence	0x04	The module is in program mode. An internal sequential program is active.
Bit 4	Warning	0x08	There is a warning.
Bit 5	Error	0x10	There is an error.
Bit 6	Brake	0x20	The brake is activated.
Bit 7	Motion blocked	0x40	A movement was interrupted.
Bit 8	Position reached	0x80	The target position was reached.

In the case of an error, the bits 9-16 also contain the error code.

**CAN bus**

If the position, speed, and current need to be received in one message, then the fragmentation protocol must be used.

**PROFIBUS**

All information is placed in a PROFIBUS message. An "Options" code is set once and retained, meaning that it does not need to be reset each time. When the module is switched on, the code "Options" is set to "0x07" and all status information is transmitted completely.

If all parameters (position, speed, current) have been transferred, only the lower 8 bits of the status are displayed in Profibus. These now come to byte 14, where the status is always provided up-to-date Profibus specifically 15. The MsgCount that overwrites the upper 8 bits of the status word follows in byte 15.

The current position is transferred in bytes 10-13 as standard. If GET STATE is used to request all parameters, then the up-to-date position in bytes 10-13 is overwritten with the up-to-date value of the current. SCHUNK recommends setting the option bytes to a maximum of "0x06" (speed and current will be transferred).

Under PROFIBUS, it may be more favorable to call the data cyclically than to call it up automatically. This is particularly true when the FREEZE mechanism is used.

### 4.2.6 CMD TOGGLE IMPULSE MESSAGE

This can be used to activate or deactivate spontaneous messages.

Code: **0xE7**  
Access rights: USER

- Command from the control unit to the module
  - None
- Reply from module to control unit
  - If the command is confirmed with "ON" (0x4F 0x4E), then spontaneous messages are active.
  - If the command is confirmed with "OFF" (0x4F 0x46 0x46), then spontaneous messages are deactivated.

Spontaneous messages are always activated in the restart of the module.

## 4.3 Settings

### 4.3.1 SET CONFIG EXT

Individual configuration parameters in the module are set.

Code: **0x83**  
Access rights: USER

- Command from the control unit to the module
  - Configuration code (2 bytes)
  - Data type (1 byte)
  - Length of the parameter value (1 byte, only with data types CHAR\_ARRAY and BINARY)
  - Parameter value (n bytes, the length depending on the data type)
- Reply from module to control unit
  - "OK" (0x4F 0x4B)
  - Configuration code (2 bytes)

Name	Value	Description
UINT8	1	Unsigned integer with 8 bits (1 byte)
INT8	2	Signed integer with 8 bits (1 byte)
UINT16	3	Unsigned integer with 16 bits (2 bytes)
INT16	4	Signed integer with 16 bits (2 byte)
UINT32	5	Unsigned integer with 32 bits (4 bytes)

Name	Value	Description
INT32	6	Signed integer with 32 bits (4 bytes)
UINT64	7	Unsigned integer with 64 bits (8 bytes)
INT64	8	Signed integer with 64 bits (8 bytes)
FLOAT	9	Floating point number with single precision (4 bytes)
DOUBLE	10	Floating point number with double precision (8 bytes)
CHAR_ARRAY	11	String
BOOL	12	Boolean value (1 byte)
BINARY	13	Byte sequence
ENUM	14	Enumeration (2 bytes)

The "Configuration code" identifies the configuration parameter to be set. The returned configuration code matches the requested values, allowing an assignment of the response to the request. The significance of a configuration code depends on the module connected.

Some configuration parameters can only be changed if the control is deactivated and the module is stopped, if necessary a fast stop can be triggered, [CMD FAST STOP](#) [► 26].

If access is made to configuration parameters that cannot be changed, an info message appears, [INFO NO RIGHTS](#) [► 93].

#### 4.3.2 GET CONFIG EXT

Configuration parameters are read out from the module.

Code: **0x82**  
Access rights: USER

- Command from the control unit to the module
  - Configuration code (2 bytes)
- Reply from module to control unit
  - Configuration code (2 bytes)
  - Data type (1 byte)
  - Length of the parameter value (1 byte, only with data types CHAR\_ARRAY and BINARY)
  - Parameter value (n bytes, the length depending on the data type)

The "Configuration code" identifies the configuration parameter to be set. The returned configuration code matches the requested values, allowing an assignment of the response to the request.

Only one single configuration parameter can be set specifically per command. The configuration parameters are identified via a 16-bit wide configuration code . The value of a configuration parameter to be set can be transferred in various data formats, depending on the module.

### 4.3.3 CALIB CURRENT

The zero point adjustment is carried out for the current sensors.

Code:	<b>0x8F</b>
Access rights	ADVANCED

- Command from the control unit to the module
  - No command parameter
- Reply from module to control unit
  - Offset values for phases A, B and C (each UINT 16)

If the third current sensor is deactivated, the value for phase C is always 2048, [Max. measurement discrepancies](#) [► 49]. If the measured values of the current sensors are beyond the tolerance limits, an error message will appear, [ERROR CALIB CURRENT](#) [► 100].

## 4.4 Other

### 4.4.1 CMD REBOOT

The module will restart.

Code:	<b>0xE0</b>
Access rights	USER

- Command from the control unit to the module
  - No command parameter
- Reply from module to control unit
  - The command is confirmed with "OK" (0x4F 0x4B).



#### 4.4.2 CHANGE USER

The user of the module is changes.

Code: **0xE3**  
Access rights: USER

- Command from the control unit to the module
  - Password (n bytes)
- Reply from module to control unit
  - Password

The command is always confirmed with "OK (0x4F4B)". The current user is added (1 Byte), [User](#) [► 92].

If an incorrect password is entered, then "User" is always set. "User" is always active when restarting a module.

#### 4.4.3 CMD DIO

Digital inputs/outputs can be set and read.

Code: **0xE1**  
Access rights: USER

- Command from the control unit to the module
  - No command parameter
  - The current status of the digital inputs/outputs is read.
  - 1 Byte
  - The upper 4 bits can be used to set the 4 digital outputs.
- Reply from module to control unit
  - If successful, "OK" (0x4F 0x4B); with attached byte for the current status of the digital inputs in the lower 4 bits and the status of the digital outputs in the upper 4 bits.

Input 1	Bit 1	0x01
Input 2	Bit 2	0x02
Input 3	Bit 3	0x04
Input 4	Bit 4	0x08
Output 1	Bit 5	0x10
Output 2	Bit 6	0x20
Output 3	Bit 7	0x40
Output 4	Bit 8	0x80

Observe the setting for digital inputs and outputs. When the module is switched on, the outputs briefly have undefined states. Under certain circumstances, this may lead to the destruction of the connected up hardware.

## 4.5 Fragmentation

---

### NOTE

Fragmentation is not generally required for operating the modules, [Fragmentation](#) [► 15].

---

#### 4.5.1 FRAG ACK

A properly processed fragment is acknowledged.

Code: **0x87**

Access rights USER

The command is required if PROFIBUS is required to fragment messages.

- Command from the control unit to the module
  - D-Len code for received fragment if the control unit confirms the fragment to the module.
- Reply from module to control unit
  - D-Len code for received fragment if the module confirms the fragment to the control unit.

#### 4.5.2 FRAG START

Indicates in a fragmented message that it is the first fragment.

Code: **0x84**

Access rights USER

- Command from the control unit to the module
  - None
- Reply from module to control unit
  - None

It is written directly after the D-Len byte. However, it is not included in D-Len since it only serves as a marker.

#### 4.5.3 FRAG MIDDLE

Indicates in a fragmented message that it is a middle fragment.

Code: **0x85**

Access rights USER

- Command from the control unit to the module
  - None
- Reply from module to control unit
  - None

It is written directly after the D-Len byte. However, it is not included in D-Len since it only serves as a marker.

#### 4.5.4 FRAG END

Indicates in a fragmented message that it is the last fragment.

Code: **0x86**  
Access rights: USER

- Command from the control unit to the module
  - None
- Reply from module to control unit
  - None

It is written directly after the D-Len byte. However, it is not included in D-Len since it only serves as a marker.

### 4.6 Troubleshooting

#### 4.6.1 CMD ERROR

A serious error has occurred which makes user intervention necessary.

Code: **0x88**

---

#### NOTE

In modules without a brake, the module may "stall", since the motor is de-energized in the event of a serious error.

These error messages are sent regularly every 15 seconds from the module to the control unit until the error is acknowledged.

The motor is switched to a deenergized states and the brake is engaged. The module is not ready for operation. The error message must be acknowledged with the "CMD ACK" command, [CMD ACK](#) [► 36].

#### 4.6.2 CMD WARNING

A software limit stop was exceeded or the temperature limit of the main board was exceeded.

Code: **0x89**

The following error messages are regularly sent from the module to the control unit every 30 s.

- Software limit range exceeded
  - A fast stop is triggered and must be acknowledged. The module is only ready for operation to a limited degree. Only motion commands out of the software limit range are permitted. If the module is moved out of the software limit range, then the warning will be deleted.
- Maximum temperature limit exceeded
  - The module will shut down in 1 min if the temperature is not lowered.

#### 4.6.3 CMD INFO

The module sends an info message.

Code: **0x8A**

Info messages are also sent when errors are rectified and the module is restarted, [INFO NO ERROR](#) [► 94] and [INFO BOOT](#) [► 93].

#### 4.6.4 CMD ACK

A pending error message is acknowledged.

Code: **0x8B**

Access rights **USER**

- Command from the control unit to the module
  - No command parameter
- Reply from module to control unit
  - The command is confirmed with "OK" (0x4F 0x4B). The CMD INFO message with the subcode INFO NO ERROR (0x08 0x00) follows.

## 5 Configuration parameters

### 5.1 Value range

The following value ranges are used:

- MAX\_BOOL = 1
- MAX\_INT8 = 127
- MAX\_INT16 = 32767
- MAX\_INT32 = 2147483647
- MAX\_UINT8 = 255
- MAX\_UINT16 = 65535
- MAX\_UINT32 = 4294967295
- MAX\_CHAR = 255
- MAX\_ENUM = 65535
- MAX\_FLOAT = 3.402823E+38
- MIN\_BOOL = 0
- MIN\_INT8 = -128
- MIN\_INT16 = -32768
- MIN\_INT32 = -2147483648
- MIN\_UINT8 = 0
- MIN\_UINT16 = 0
- MIN\_UINT32 = 0
- MIN\_CHAR = 0
- MIN\_ENUM = 0
- MIN\_FLOAT = -3.402823E+38

### 5.2 Parameter code display

The parameter code is displayed as a hexadecimal number and is structured as follows:

**Example:** System of units, code 0x7D75

0x	Hexadecimal display
7D	Parameter prefix
75	Parameter index

## 5.3 Parameter

### NOTE

If a parameter value is exceeded or not reached during the writing process, an info message appears, [INFO VALUE LIMIT MIN](#) [► 95] or [INFO VALUE LIMIT MAX](#) [► 95].

### 5.3.1 Device

#### 5.3.1.1 Device serial number

The parameter shows the serial number of the module.

Code: **0x7D73**  
 Access rights, read - write: USER - SCHUNK  
 Data type: UINT32  
 Parameter value, min. - max.: 0 - MAX\_UINT32

#### 5.3.1.2 Actual gripper

The parameter shows whether the module is a gripper.

Code: **0x7D74**  
 Access rights, read - write: USER - PROFI  
 Data type: BOOL  
 Parameter value, min. - max.: false - true

#### 5.3.1.3 Unit system

The parameter shows the unit system of the module.

A restart of the module is required after the writing.

Code: **0x7D75**  
 Access rights, read - write: USER - USER  
 Data type: ENUM  
 Parameter value, min. - max.: see the following table

Parameter value	Designation
0	mm
1	m
2	Inch
3	Rad
4	Degrees
5	INTERNAL

#### 5.3.1.4 Inverted direction of rotation of motor

The parameter shows the direction of rotation of the motor.  
A restart of the module is required after the writing.

Code:	<b>0x7D76</b>
Access rights, read - write:	USER - PROFI
Data type:	BOOL
Parameter value, min. - max.:	false - true

#### NOTE

An incorrect setting may lead to unexpected effects, e.g. the module turns at an unexpectedly high speed.

The parameter has direct interaction with the following parameter:

- Parameter [Inverted position measurement](#) [► 39]
  - If the direction of rotation of the motor and the position measuring system are inverted at the same time, then a right-turning module can be configured from a left-turning one, or a positive closing gripper can be configured from a positive opening gripper.

#### 5.3.1.5 Inverted position measurement

The parameter shows the measuring direction of the position measuring system.

A restart of the module is required after the writing.

Code:	<b>0x7D77</b>
Access rights, read - write:	USER - PROFI
Data type:	BOOL
Parameter value, min. - max.:	false - true

#### NOTE

An incorrect setting may lead to unexpected effects, e.g. the module turns at an unexpectedly high speed.

If the A and B encoder tracks are interchanged, this can be exchanged using the software.

The parameter has direct interaction with the following parameter:

- Parameter [Inverted direction of rotation of motor](#) [► 39]
  - If the direction of rotation of the motor and the position measuring system are inverted at the same time, then a right-turning module can be configured from a left-turning one, or a positive closing gripper can be configured from a positive opening gripper.

#### 5.3.1.6 Endless

The parameter shows whether the axis rotates endlessly or if software end stops are taken into account.

Code: **0x7D78**  
 Access rights, read - write: USER - PROFI  
 Data type: BOOL  
 Parameter value, min. - max.: false - true

With rotary modules, you can set whether or not the module rotates endlessly.

The parameter has direct interaction with the following parameters:

- Parameter [Min. position](#) [► 41]
  - Not taken into account when "Endless" is set.
- Parameter [Max. position](#) [► 41]
  - Not taken into account when "Endless" is set.

#### 5.3.1.7 Digital outputs

The parameter shows how digital outputs are used.

Code: **0x7D7A**  
 Access rights, read - write: USER - PROFI  
 Data type: ENUM  
 Parameter value, min. - max.: see the following table

Parameter value	Designation
0	Normal
1	Status + movement OUT2
2	Status + position reached OUT2
3	Status + brake OUT2
4	Status + warning OUT2
5	Status + program sequence OUT2



### 5.3.1.8 Min. position

The parameter shows the minimum position of the software stop.

Code:	<b>0x7D7B</b>
Access rights, read - write:	USER - PROFI
Data type:	FLOAT/INT32
Parameter value, min. - max.:	MIN_FLOAT - <a href="#">Max. Position</a> [▶ 41]

If the position setting exceeds this value, an info message appears, [INFO VALUE LIMIT MIN](#) [▶ 95] and the position setting is automatically corrected according to this value.

The parameter has direct interaction with the following parameters:

- Parameter [Endless](#) [▶ 40]
  - Not taken into account when "Endless" is set.
- Parameter [Referencing type](#) [▶ 55]
  - Is used for referencing with stroke control.

### 5.3.1.9 Max. position

The parameter shows the maximum position of the software stop.

Code:	<b>0x7D7C</b>
Access rights, read - write:	USER - PROFI
Data type:	FLOAT/INT32
Parameter value, min. - max.:	<a href="#">Min. Position</a> [▶ 41] - MAX_FLOAT

If the position setting exceeds this value, an info message appears, [INFO VALUE LIMIT MAX](#) [▶ 95] and the position setting is automatically corrected according to this value.

The parameter has direct interaction with the following parameters:

- Parameter [Endless](#) [▶ 40]
  - Not taken into account when "Endless" is set.
- Parameter [Referencing type](#) [▶ 55]
  - Is used for referencing with stroke control.

#### 5.3.1.10 Min. motor temperature

The parameter shows the minimum permissible operating temperature for the motor when the motor temperature sensor is connected.

Code:	<b>0x7D7F</b>
Access rights, read - write:	USER - ADVANCED
Data type:	FLOAT
Parameter value, min. - max.:	MIN_FLOAT - <a href="#">Max. motor temperature</a> [► 42]

If the parameter value is not reached, an error message occurs, [ERROR MOTOR TEMP LOW](#) [► 101].

#### 5.3.1.11 Max. motor temperature

The parameter shows the minimum permissible operating temperature for the motor when the motor temperature sensor is connected.

Code:	<b>0x7D80</b>
Access rights, read - write:	USER - ADVANCED
Data type:	FLOAT
Parameter value, min. - max.:	<a href="#">Min. motor temperature</a> [► 42] - MAX_FLOAT

If the parameter value is exceeded, an error message occurs, [ERROR MOTOR TEMP HIGH](#) [► 101].

#### 5.3.1.12 Min. main board temperature

The parameter shows the minimum permissible working temperature for the main board.

Code:	<b>0x7D7D</b>
Access rights, read - write:	USER - ADVANCED
Data type:	FLOAT
Parameter value, min. - max.:	MIN_FLOAT - <a href="#">Max. main board temperature</a> [► 42]

If the parameter value is not reached, an error message occurs, [ERROR TEMP LOW](#) [► 102].

#### 5.3.1.13 Max. main board temperature

The parameter shows the maximum permissible working temperature for the main board.

Code:	<b>0x7D7E</b>
Access rights, read - write:	USER - ADVANCED
Data type:	FLOAT

Parameter value, min. - max.: [Min. main board temperature](#)  
[► 42] - MAX\_FLOAT

If the temperature value exceeds the working temperature, there will be an warning. There will be an error message if the temperature does not drop within 1 min, [ERROR TEMP HIGH](#) [► 102].

#### 5.3.1.14 Min. temperature of the communication board

The parameter shows the minimum permissible working temperature for the communication board.

Code: **0x7D81**  
Access rights, read - write: USER - ADVANCED  
Data type: FLOAT  
Parameter value, min. - max.: MIN\_FLOAT - [Max. temperature of the communication board](#)  
[► 43]

If the parameter value is not reached, an error message occurs, [ERROR CONNECTION TEMP LOW](#) [► 100].

#### 5.3.1.15 Max. temperature of the communication board

The parameter shows the maximum permissible working temperature for the communication board.

Code: **0x7D82**  
Access rights, read - write: USER - ADVANCED  
Data type: FLOAT  
Parameter value, min. - max.: [Min. temperature of the communication board](#) [► 43] -  
MAX\_FLOAT

If the parameter value is exceeded, an error message occurs, [ERROR CONNECTION TEMP HIGH](#) [► 101].

### 5.3.2 Motor

#### 5.3.2.1 Motor serial number

The parameter shows the serial number of the motor.

Code: **0x7D1E**  
Access rights, read - write: USER - SCHUNK  
Data type: UINT32  
Parameter value, min. - max.: 0 - MAX\_UINT32

The parameter value can only be written if the motor is de-energized.

### 5.3.2.2 Motor voltage

The parameter shows the nominal voltage of the motor [V].

Code: **0x7D1F**  
 Access rights, read - write: USER - ROOT  
 Data type: UINT8  
 Parameter value, min. - max.: 24 - 48

The parameter value can only be written if the motor is de-energized.

#### NOTICE

An incorrect input can destroy the electronics.

### 5.3.2.3 Motor Type

The parameter shows the selected motor.

A restart of the module is required after the writing.

Code: **0x7D20**  
 Access rights, read - write: USER - ADVANCED  
 Data type: ENUM  
 Parameter value, min. - max.: see the following table

Parameter value	Designation
0	DC Brush type DC motor
1	BLDC Electronically commutated brushless DC motor with block commutation.
2	PMSM Electronically commutated brushless DC motor with sine commutation.

The parameter value can only be written if the motor is de-energized.

#### NOTICE

An incorrect input can destroy the electronics.

**5.3.2.4 I<sup>2</sup>T**

The parameter shows the strength of the I<sup>2</sup>T monitoring [%].

Code:	<b>0x7D21</b>
Access rights, read - write:	USER - ADVANCED
Data type:	UINT8
Parameter value, min. - max.:	10 - 100

The parameter value can only be written if the motor is de-energized.

An I<sup>2</sup>T *error* [ERROR I2T](#) [► 99] will be triggered if the load is too high. With I<sup>2</sup>T monitoring, it is assumed that the maximum current may be present for three seconds (corresponds to 100%). If a value of <100% is entered, the time is reduced and the I<sup>2</sup>T monitoring triggers earlier.

**5.3.2.5 Max. current**

The parameter shows the maximum permissible current.

Code:	<b>0x7D22</b>
Access rights, read - write:	USER - ADVANCED
Data type:	FLOAT/INT32
Parameter value, min. - max.:	0 - Metering current range

The parameter value can only be written if the motor is de-energized.

**NOTICE**

An incorrect input can destroy the electronics.

If the value is exceeded for a period of several milliseconds, a fast stop is triggered and an error message appears, [ERROR CURRENT](#) [► 99].

#### 5.3.2.6 Nom. Current

The parameter shows the maximum current that is allowed to flow permanently.

Code:	<b>0x7D23</b>
Access rights, read - write:	USER - ADVANCED
Data type:	FLOAT/INT32
Parameter value, min. - max.:	0 - <a href="#">Max. current</a> [► 45]

The parameter value can only be written if the motor is de-energized.

#### NOTICE

An incorrect input can destroy the electronics.

---

If the value is exceeded for a period of several seconds, an error message appears, [ERROR I2T](#) [► 99].

#### 5.3.2.7 Max. speed

The parameter shows the maximum permissible speed (output side).

Code:	<b>0x7D24</b>
Access rights, read - write:	USER - ADVANCED
Data type:	FLOAT/INT32
Parameter value, min. - max.:	0 - MAX_FLOAT/MAX_INT32

The parameter value can only be written if the motor is de-energized.

#### 5.3.2.8 Max. acceleration

The parameter shows the maximum permissible acceleration (output side).

Code:	<b>0x7D25</b>
Access rights, read - write:	USER - ADVANCED
Data type:	FLOAT/INT32
Parameter value, min. - max.:	0 - MAX_FLOAT/MAX_INT32

The parameter value can only be written if the motor is de-energized.

**5.3.2.9 Max. jerk**

The parameter shows the maximum permissible jerk (output side).  
The parameter value can only be written if the motor is de-energized.

Code:	<b>0x7D26</b>
Access rights, read - write:	USER - ADVANCED
Data type:	FLOAT/INT32
Parameter value, min. - max.:	0 - MAX_FLOAT/MAX_INT32

The jerk is the temporal alteration of acceleration.

The parameter is only analyzed if a position movement with jerk limitation is executed.

The parameter has direct interaction with the following parameter:

- Parameter [Position ramp](#) [► 67]

**5.3.2.10 Pole pairings**

The parameter shows the electrical pole pairing of the motor.

Code:	<b>0x7D27</b>
Access rights, read - write:	USER - ROOT
Data type:	UINT8
Parameter value, min. - max.:	0 - MAX_UINT8

The parameter value can only be written if the motor is de-energized.

The parameter is only required for brushless DC motors and affects the calculation of the speed, position and commutation pattern.

The parameter has direct interaction with the following parameter:

- Parameter [Motor Type](#) [► 44]

#### 5.3.2.11 Terminal resistance

The parameter shows the value of the terminal resistance [Ohm].

Code:	<b>0x7D28</b>
Access rights, read - write:	USER - ROOT
Data type:	FLOAT
Parameter value, min. - max.:	0 - MAX_FLOAT

The parameter value can only be written if the motor is de-energized.

#### **NOTICE**

An incorrect input can destroy the electronics.

---

The terminal resistance is used for test functions to limit maximum currents as well as for automatic controller configuration.

#### 5.3.2.12 Inductance

The parameter shows the value of the inductance [H].

Code:	<b>0x7D29</b>
Access rights, read - write:	USER - ROOT
Data type:	FLOAT
Parameter value, min. - max.:	0 - MAX_FLOAT

The parameter value can only be written if the motor is de-energized.

#### **NOTICE**

An incorrect input can destroy the electronics.

---

#### 5.3.2.13 Motor constant

The parameter shows the value of the motor constant.

Code:	<b>0x7D2A</b>
Access rights, read - write:	USER - ROOT
Data type:	FLOAT/INT32
Parameter value, min. - max.:	0 - MAX_FLOAT/MAX_INT32

The parameter value can only be written if the motor is de-energized.



**5.3.2.14 Commutation table**

The parameter shows the number of the valid hall sensor table for block commutation.

Code:	<b>0x7D2B</b>
Access rights, read - write:	USER - ROOT
Data type:	UINT8
Parameter value, min. - max.:	0 - 5

The parameter value can only be written if the motor is de-energized.

If the entry is incorrect, the motor will not move at all or only generates very little torque.

**5.3.2.15 Metering current range**

The parameter shows the maximum metering current range for the current sensor used internally.

A restart of the module is required after the writing.

Code:	<b>0x7D2C</b>
Access rights, read - write:	USER - SCHUNK
Data type:	FLOAT/INT32
Parameter value, min. - max.:	0 - MAX_FLOAT/MAX_INT32

The parameter value can only be written if the motor is de-energized.

**5.3.2.16 Max. measurement discrepancies**

The parameter shows the maximum measurement difference [A].

Code:	<b>0x7D2D</b>
Access rights, read - write:	USER - ROOT
Data type:	FLOAT
Parameter value, min. - max.:	1 MAX_FLOAT

The parameter value can only be written if the motor is de-energized.

A third current sensor is necessary to calculate the maximum measurement difference. The maximum permitted deviation is  $A + B + C = 0$ . If the deviation is greater than the set value, an error message appears, [ERROR MOTOR PHASE](#) [► 97].

The value "-1" deactivates the third current sensor. This is not supported by all hardware variants. A motor phase break or motor phase short circuit is not detected by the deactivation of the sensor.

When the third current sensor is activated, the current sensors must be calibrated.

#### 5.3.2.17 Offset phase A

The parameter shows the zero point alignment for the first current sensor.

Code:	<b>0x7D2E</b>
Access rights, read - write:	USER - SCHUNK
Data type:	UINT16
Parameter value, min. - max.:	1500 - 2600

The parameter value can only be written if the motor is de-energized.

---

#### NOTE

An incorrect value may lead to an unpredictable behavior in the drive (e.g., only travels in one direction, jerks badly).

---

#### 5.3.2.18 Offset phase B

The parameter shows the zero point alignment for the second current sensor.

Code:	<b>0x7D2F</b>
Access rights, read - write:	USER - SCHUNK
Data type:	UINT16
Parameter value, min. - max.:	1500 - 2600

The parameter value can only be written if the motor is de-energized.

---

#### NOTE

An incorrect value may lead to an unpredictable behavior in the drive (e.g., only travels in one direction, jerks badly).

---

**5.3.2.19 Offset phase C**

The parameter shows the zero point alignment for the third current sensor.

Code:	<b>0x7D30</b>
Access rights, read - write:	USER - SCHUNK
Data type:	UINT16
Parameter value, min. - max.:	1500 - 2600

The parameter value can only be written if the motor is de-energized.

**NOTE**

An incorrect value may lead to an unpredictable behavior in the drive (e.g., only travels in one direction, jerks badly).

If the hardware does not support third current sensors, this value is always 2048.

**5.3.3 Controller****5.3.3.1 KR current**

The parameter shows the proportional part of current controller.

Code:	<b>0x7D4B</b>
Access rights, read - write:	USER - PROFI
Data type:	FLOAT
Parameter value, min. - max.:	0 - MAX_FLOAT

**5.3.3.2 TN current**

The parameter shows the integral part of the current controller.

Code:	<b>0x7D4C</b>
Access rights, read - write:	USER - PROFI
Data type:	FLOAT
Parameter value, min. - max.:	0 - MAX_FLOAT

This is not required for current limiting control, MCSL\_Struktur.

**5.3.3.3 TD current**

The parameter shows the differential part of the current controller.

Code:	<b>0x7D4D</b>
Access rights, read - write:	USER - ROOT
Data type:	FLOAT
Parameter value, min. - max.:	0 - MAX_FLOAT

#### 5.3.3.4 KC current

The parameter shows the correction factor of the current controller for the integral part (anti-windup).

Code:	<b>0x7D4E</b>
Access rights, read - write:	USER - ROOT
Data type:	FLOAT
Parameter value, min. - max.:	0 - MAX_FLOAT

#### 5.3.3.5 KR speed

The parameter shows the proportional part of the speed controller.

Code:	<b>0x7D4F</b>
Access rights, read - write:	USER - PROFI
Data type:	FLOAT
Parameter value, min. - max.:	0 - MAX_FLOAT

#### 5.3.3.6 TN speed

The parameter shows the integral part of the speed controller.

Code:	<b>0x7D50</b>
Access rights, read - write:	USER - PROFI
Data type:	FLOAT
Parameter value, min. - max.:	0 - MAX_FLOAT

#### 5.3.3.7 TD speed

The parameter shows the differential part of the speed controller.

Code:	<b>0x7D51</b>
Access rights, read - write:	USER - ROOT
Data type:	FLOAT
Parameter value, min. - max.:	0 - MAX_FLOAT

#### 5.3.3.8 KC speed

The parameter shows the correction factor of the current controller for the integral part (anti-windup).

Code:	<b>0x7D52</b>
Access rights, read - write:	USER - ROOT
Data type:	FLOAT
Parameter value, min. - max.:	0 - MAX_FLOAT

#### 5.3.3.9 KR position

The parameter shows the proportional part of position controller.

Code:	<b>0x7D53</b>
Access rights, read - write:	USER - PROFI
Data type:	FLOAT

Parameter value, min. - max.: 0 - MAX\_FLOAT

#### 5.3.3.10 TN position

The parameter shows the integral part of position controller.

Code: **0x7D54**  
 Access rights, read - write: USER - PROFI  
 Data type: FLOAT  
 Parameter value, min. - max.: 0 - MAX\_FLOAT

#### 5.3.3.11 TD position

The parameter shows the differential part of the position controller.

Code: **0x7D55**  
 Access rights, read - write: USER - ROOT  
 Data type: FLOAT  
 Parameter value, min. - max.: 0 - MAX\_FLOAT

#### 5.3.3.12 KC position

The parameter shows the correction factor of the current controller for the integral part (anti-windup).

Code: **0x7D56**  
 Access rights, read - write: USER - ROOT  
 Data type: FLOAT  
 Parameter value, min. - max.: 0 - MAX\_FLOAT

#### 5.3.3.13 Position deviation

The parameter shows the position window in which the position control is terminated.

Code: **0x7D59**  
 Access rights, read - write: USER - PROFI  
 Data type: FLOAT/INT32  
 Parameter value, min. - max.: 0 - MAX\_FLOAT/MAX\_INT32

Whether the brake will be applied or the "Position reached" will be triggered is controlled depending on the brake configuration [Brake usage](#) [► 71].

#### 5.3.3.14 Max. overshoot

The parameter shows how much overshoot the module is allowed during a movement.

Code: **0x7D5A**  
 Access rights, read - write: USER - ADVANCED  
 Data type: FLOAT/INT32  
 Parameter value, min. - max.: 0 - MAX\_FLOAT/MAX\_INT32

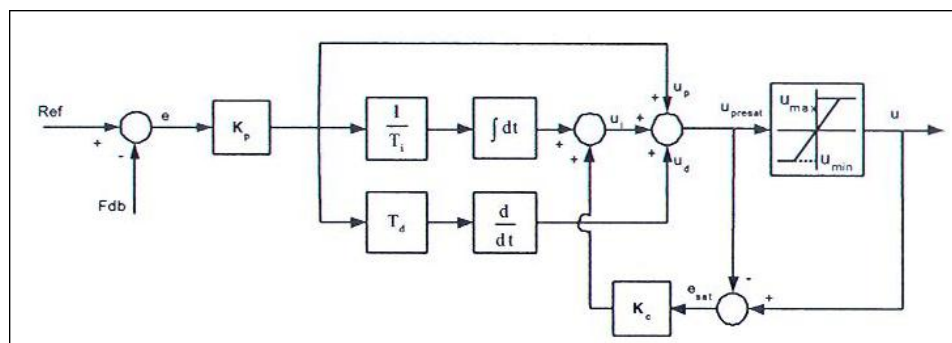
If the module moves past this position window during a positioning movement, then the error message "ERROR OVERSHOOT" is generated, [ERROR OVERSHOOT](#) [► 103]. This value must be set greater than the maximum permissible position variation. [Position deviation](#) [► 53].

### 5.3.3.15 Structure

The parameter shows the structural composition of the control circuit.

Code: **0x7D5B**  
 Access rights, read - write: USER - PROFI  
 Data type: ENUM  
 Parameter value, min. - max.: see the following table

Parameter value	Designation
0	Current speed
1	Cascade
2	Speed with current limiting
3	Speed with PWM limiting
4	Position cascade



Controllers, structure

All controllers are realized as PID controllers with anti-windup functionality. The complete parameter set is only to be accessed via the root rights in SCHUNK mode.

The following parameters can be set:

- KR: proportional part of the respective controller
- TN: integral part of the respective controller
- TD: differential part of the respective controller
- KC: correction factor for the integral part
- Current - Speed  
Current control and speed control operate independently of each other.
- Cascade  
Position, speed, and current controllers are cascade connected  
=> Current controlled (set current is not to be exceeded) posi-

tioning or speed movements are possible (e.g. no pre-positioning required for a gripping process necessary). In this mode, the specified current is not exceeded in all motion types.

- **Speed with current limiting**  
Current control is not active. The specified current is limited for speed or position movements. In contrast to the cascade, the current is not controlled, but limited (current limiting control).
- **Speed with PWM limiting**  
Current control is not active. The duty cycle of the PWM is limited for speed or position movements. The ratio of current to duty cycle is calculated using the terminal resistance of the motor, [Terminal resistance](#) [► 48].

Since the PWM's duty cycle is directly limited (voltage limitation), it is possible that the motor no longer reaches its full speed. Position movements may take considerably longer than anticipated.

If the controller structure is changed, the controller parameters may have to be adapted.

### 5.3.4 Referencing

#### 5.3.4.1 Referencing type

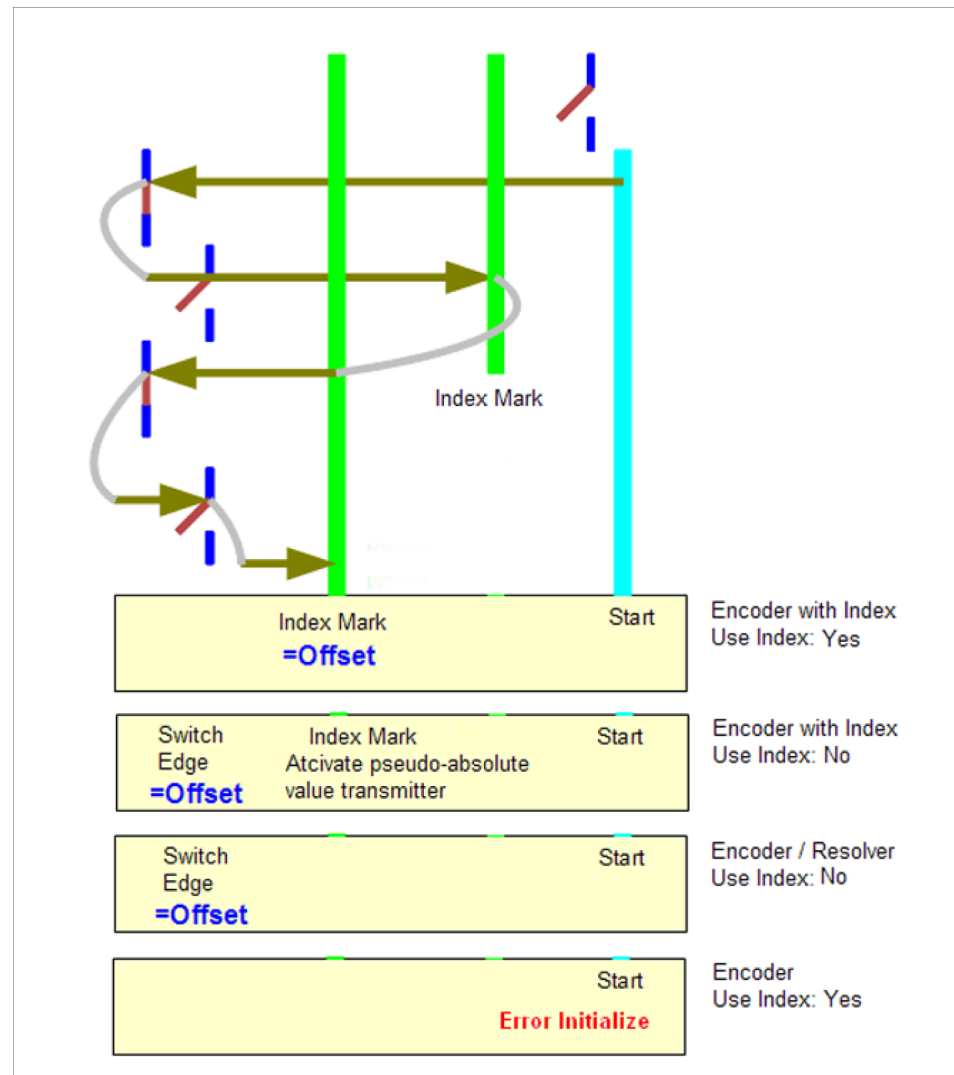
The parameter shows the type of referencing.

A restart of the module is required after the writing.

Code: **0x7D41**  
 Access rights, read - write: USER - PROFI  
 Data type: ENUM  
 Parameter value, min. - max.: see the following table

Parameter value	Designation
0	Left side internal switch
1	Right side internal switch
2	Left side external IN0 switch
3	Right side external IN0 switch
4	Speed, left
5	Speed, right
6	Speed, left (stroke monitoring)
7	Speed, right (stroke monitoring)
8	Current, left
9	Current, right
10	Current, left (stroke monitoring)
11	Current, right (stroke monitoring)
12	None

When using an encoder with index track, observe the "Positioning type" parameter, [Positioning type](#) [► 62].



#### Referencing with switch

- Internal switch left/right
  - The internal reference switch is used for referencing. The direction of motion when the reference switch is active is determined by the direction "left" or "right".
- External IN1 switch left/right
  - An external reference switch (IN1) is used for referencing. The direction of motion when the reference switch is active is determined by the direction "left" or "right".

When referencing with a switch, it must be ensured that the switch flack of the proximity switch is applied for at least 200 ms. If necessary, also adjust the referencing speed and the switching cams.



**NOTE**

SCHUNK recommends a basic referencing process after installing the machine/system on the module. If the position or load is changed during referencing, SCHUNK also recommends a basic referencing process. Without a basic referencing process, the error message "NOT REFERENCED" may occur, [NOT REFERENCED](#) [► 94].

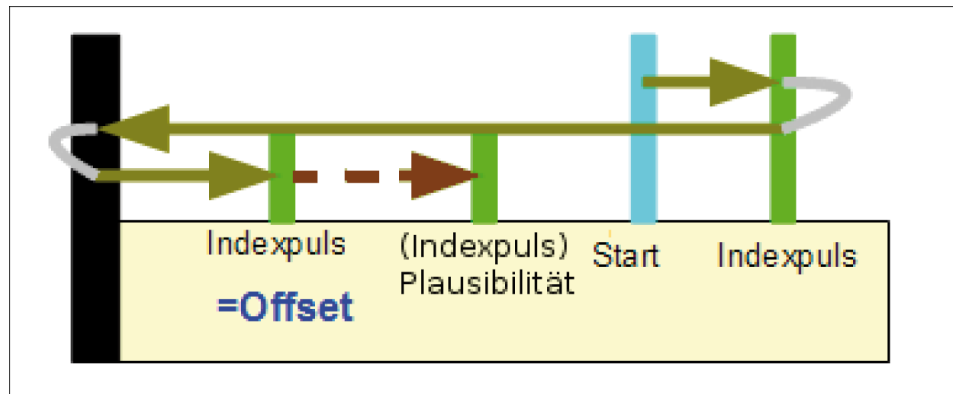
If a fixed end stop is available, SCHUNK recommends the following referencing types:

- Speed, left/right
  - A speed movement is performed for referencing. If the module moves to a fixed stop, then this will be recognized as a reference point. The direction of rotation is defined via "left" or "right".
- Speed with stroke monitoring left/right
  - In addition to the procedure described above, a movement is made to the opposite fixed stop after the first fixed stop has been approached. The traveled distance must be greater than the difference in the software limit ranges => Referencing successful, [Min. position](#) [► 41].
- Current, left/right

**NOTE**

Jamming, sluggishness in the mechanical system, or a "forgotten" workpiece can also lead to the rated current being exceeded. This would then also be interpreted as a fixed end stop, although none are present.

- A current move is executed. The current is increased until the module moves. If the current exceeds the max. reference current, then it is assumed that a fixed stop has been reached which is recognized as a reference point, [Referencing max. current](#) [► 60].
- Current with stroke monitoring left/right
  - In addition to the procedure described above, a movement is made to the opposite fixed end stop after the first fixed stop has been approached. The traveled distance must be greater than the difference in the software limit ranges => Referencing successful, [Min. position](#) [► 41].
- None



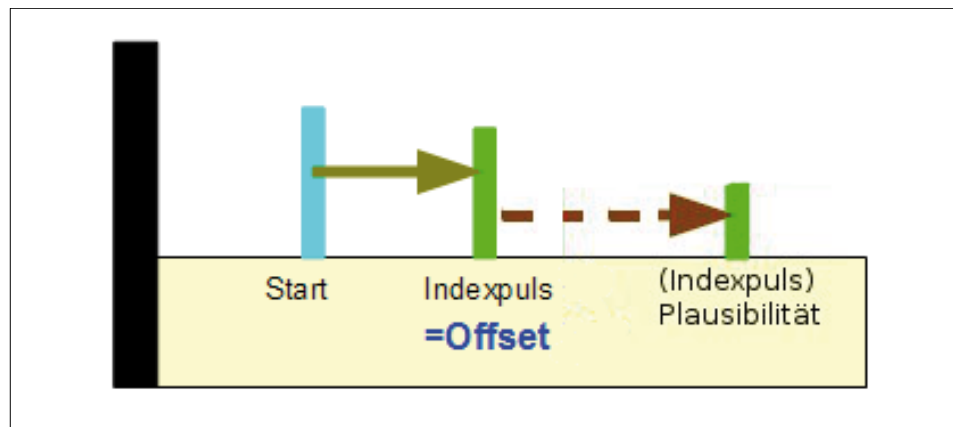
*Referencing up to stop with activated index track*

- The current position is seen as the reference position.

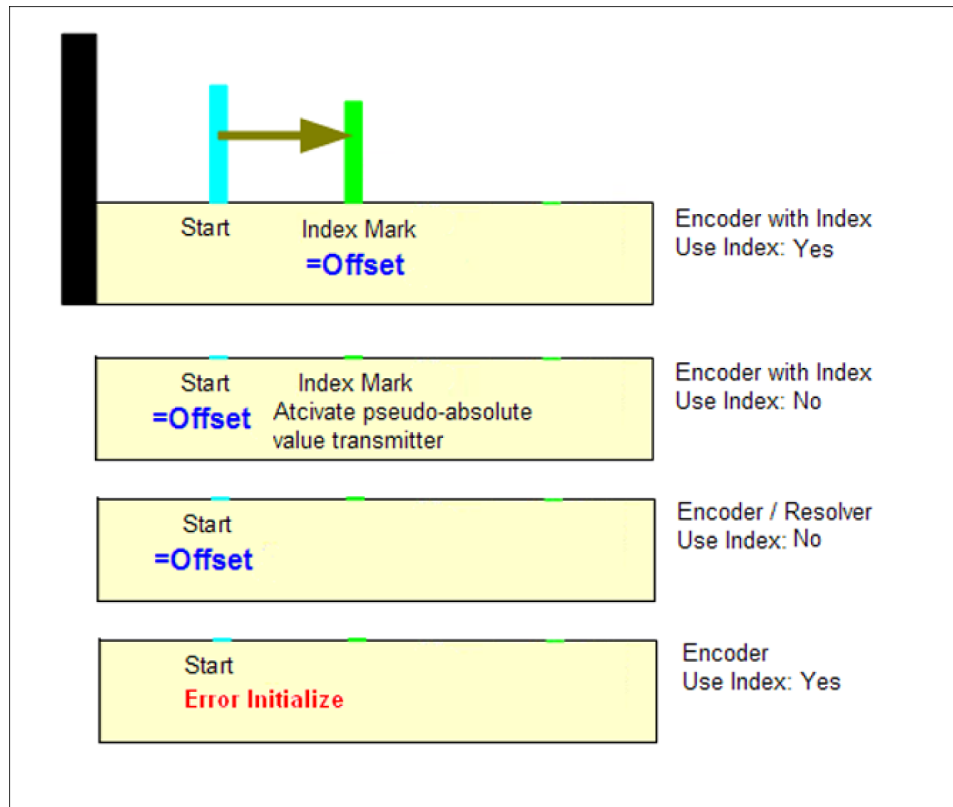
#### 5.3.4.2 Usage index

The parameter shows whether the index track of the encoder is analyzed during referencing.

Code:	<b>0x7D42</b>
Access rights, read - write:	USER - PROFI
Data type:	BOOL
Parameter value, min. - max.:	false - true



*Referencing of "none" with activated index track*



Referencing of "none"

## NOTE

Should be referenced with index pulse. If the index pulse is in a poor position, it can happen that the positions differ by one motor rotation each after repeated referencing. Remedy: Shift the reference mark slightly. This applies to all reference marks except "internal switch" and "external switch".

#### 5.3.4.3 Distance to the index

The parameter shows the distance from the reference event (switch flank detected) to the index pulse.

Code:	<b>0x7D43</b>
Access rights, read - write:	USER - PROFI
Data type:	UINT16
Parameter value, min. - max.:	0 - MAX_UINT16

If the parameter is set to "0", the distance is measured and stored during the next reference movement.

For subsequent reference movements, this is measured again and compared with the saved value. If the values are within a set tolerance range, then the referencing can be successfully completed, [Max. distance switch referencing](#) [► 62]. Furthermore, an index pulse at an unfavorable point (index pulse shortly before or after switch flanks) can be corrected.

If the distance between index pulse and reference event exceeded the permissible tolerance, then referencing will be aborted and "ERROR REFERENCED" displayed, [NOT REFERENCED](#) [► 94].

It is necessary to remeasure the distance from the reference event to the index pulse. To do that, the parameter "Distance to index" must be set to "0", [Distance to the index](#) [► 60].

#### 5.3.4.4 Approach 0 after referencing

The parameter shows whether the position "0" is approached after successful referencing.

Code:	<b>0x7D44</b>
Access rights, read - write:	USER - PROFI
Data type:	BOOL
Parameter value, min. - max.:	false - true

#### 5.3.4.5 Referencing max. current

The parameter shows the current setting from the nominal current of the motor [%]. The reference current does not exceed the specified value.

Code:	<b>0x7D45</b>
Access rights, read - write:	USER - PROFI
Data type:	UINT8
Parameter value, min. - max.:	0 - 200

If the maximum permissible reference current is not sufficient enough to move the module, the maximum reference current must be increased.

**5.3.4.6 Speed referencing**

The parameter shows the speed settings for reference movements.

Code: **0x7D46**  
 Access rights, read - write: USER - PROFI  
 Data type: FLOAT/INT32  
 Parameter value, min. - max.: 0 - [Max. speed](#) [► 46]

**5.3.4.7 Acceleration referencing**

The parameter shows the acceleration settings for reference movements with internal or external reference switch, and speed reference movements.

Code: **0x7D47**  
 Access rights, read - write: USER - PROFI  
 Data type: FLOAT/INT32  
 Parameter value, min. - max.: 0 - [Max. acceleration](#) [► 46]

**5.3.4.8 Offset referencing**

The parameter shows the position offset after successful referencing (zero offset).

Code: **0x7D48**  
 Access rights, read - write: USER - PROFI  
 Data type: FLOAT/INT32  
 Parameter value, min. - max.: MIN\_FLOAT/MIN\_INT32 - MAX\_FLOAT/MAX\_INT32

**5.3.4.9 Timeout referencing**

The parameter shows the time that a reference movement is permitted to last.

Code: **0x7D49**  
 Access rights, read - write: USER - PROFI  
 Data type: FLOAT/INT32  
 Parameter value, min. - max.: 0 - MAX\_FLOAT/MAX\_INT32

If the time is exceeded, the motor is de-energized and an error message appears [NOT REFERENCED](#) [► 94].

#### 5.3.4.10 Max. distance switch referencing

The parameter shows the maximum distance from the reference event (switch flack detected) to the index pulse.

Code: **0x7D4A**  
 Access rights, read - write: USER - ADVANCED  
 Data type: UINT16  
 Parameter value, min. - max.: 0 - MAX\_UINT16

If one encoder tick is specified.

### 5.3.5 Positioning

#### 5.3.5.1 Positioning serial number

The parameter shows the serial number of the position measuring system.

Code: **0x7D5F**  
 Access rights, read - write: USER - SCHUNK  
 Data type: UINT32  
 Parameter value, min. - max.: 0 - MAX\_UINT32

The parameter value can only be written if the motor is de-energized.

#### 5.3.5.2 Positioning type

The parameter shows the measuring system type.

A restart of the module is required after the writing.

Code: **0x7D60**  
 Access rights, read - write: USER - ADVANCED  
 Data type: ENUM  
 Parameter value, min. - max.: see the following table

Parameter value	Designation
0	Encoder
1	Encoder Index
2	Resolver
6	Encoder differential
7	Encoder index differential
8	Analog

The parameter value can only be written if the motor is de-energized.

After a new measuring system has been selected, a restart of the module is required.

- Encoder
  - Encoder measuring system without index track.
- Encoder index
  - Encoder measuring system with index track.  
For the reference movements, the index track is analyzed depending on the configuration, [Usage index](#) [► 58]. When using the index track, it is possible for the motor to move back and forth several times in short movements or to make small movements in the "wrong" direction during the referencing process. The modules also move with the "None" referencing type, due to the search for the next index pulse, [Pseudo-absolute encoder](#) [► 10].
- Resolver
  - Resolver system with adjustable exciting current
- Encoder differential
  - Differential encoder without index track
- Encoder index differential
  - Differential encoder with index track  
For reference movements, the index track is analyzed depending on the configuration, [Usage index](#) [► 58]. When using the index track, it is possible for the motor to move back and forth several times in short movements or to make small movements in the "wrong" direction during the referencing process. The modules also move with the "None" referencing type, due to the search for the next index pulse, [Pseudo-absolute encoder](#) [► 10].

### 5.3.5.3 Positioning installation

The parameter shows the installation position of the position measuring system.

The parameter value can only be written if the motor is de-energized.

A restart of the module is required after the writing.

Code: **0x7D61**  
 Access rights, read - write: USER - ADVANCED  
 Data type: ENUM  
 Parameter value, min. - max.: see the following table

Parameter value	Designation
0	Drive side
1	Output side
2	Between gear ratios

- Drive side
  - The position measuring system is mounted directly on the drive end.
- Output side
  - The position measuring system is mounted directly on the output end.
- Between gear ratios
  - The position measuring system is mounted in the middle of the gear.

### 5.3.5.4 Ticks per revolution

The parameter shows the ticks per revolution.

The parameter value can only be written if the motor is de-energized.

A restart of the module is required after the writing.

Code: **0x7D62**  
 Access rights, read - write: USER - ADVANCED  
 Data type: UINT16  
 Parameter value, min. - max.: 512 - MAX\_UINT16



**5.3.5.5 Exciter amplitude**

The parameter shows the amplitude of the input voltage at the exciter coil [%].

Code: **0x7D63**  
 Access rights, read - write: USER - ADVANCED  
 Data type: UINT8  
 Parameter value, min. - max.: 0 - 100

The parameter value can only be written if the motor is de-energized.

**5.3.5.6 Exciter frequency**

The parameter shows the voltage frequency at the exciter coil [kHz].

Code: **0x7D64**  
 Access rights, read - write: USER - ADVANCED  
 Data type: ENUM  
 Parameter value, min. - max.: see the following table

Parameter value	Designation
0	1 kHz
1	2 kHz
2	4 kHz
3	8 kHz

The parameter value can only be written if the motor is de-energized.

**5.3.5.7 ADC offset positioning**

The parameter shows the "centering" of the input signal at the resolver.

Code: **0x7D65**  
 Access rights, read - write: USER - ADVANCED  
 Data type: FLOAT  
 Parameter value, min. - max.: MIN\_FLOAT - MAX\_FLOAT

The parameter value can only be written if the motor is de-energized.

#### 5.3.5.8 Offset positioning

The parameter shows the twisting of the position measuring system opposite the motor phases.

Code:	<b>0x7D68</b>
Access rights, read - write:	USER - ADVANCED
Data type:	FLOAT/INT32
Parameter value, min. - max.:	MIN_FLOAT - MAX_FLOAT/ MIN_INT32 - MAX_INT32

The parameter value can only be written if the motor is de-energized.

This value may be automatically determined, [Standstill commutation](#) [► 11].

#### 5.3.5.9 Motion threshold

The parameter shows the value in percent [%] of the maximum permissible speed.

Code:	<b>0x7D69</b>
Access rights, read - write:	USER - PROFI
Data type:	FLOAT
Parameter value, min. - max.:	1 - 100

The parameter value can only be written if the motor is de-energized.

If the current speed falls below this value, the module is treated as if it is standing still. The "Module is moving" status display is off.

#### 5.3.5.10 Position waiting time reached

The parameter shows the reset time delay for the "Position reached" flag.

Code:	<b>0x7D6A</b>
Access rights, read - write:	USER - PROFI
Data type:	FLOAT/INT32
Parameter value, min. - max.:	0 - MAX_FLOAT

SCHUNK recommends setting this value somewhat larger than the PLC cycle.

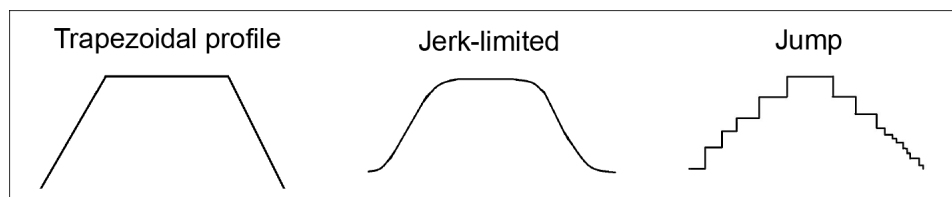
### 5.3.5.11 Position ramp

The parameter shows the ramp type for the position movement. A restart of the module is required after the writing.

Code: **0x7D6B**  
 Access rights, read - write: USER - PROFI  
 Data type: ENUM  
 Parameter value, min. - max.: see the following table

Parameter value	Designation
0	Trapezoid V2
1	Jerk-limited
2	Trapezoid V3
3	Jump

The parameter value can only be written if the motor is de-energized.



- **Trapezoid V2**  
A trapezoid is taken as the basis for the calculation of the motion profile. The traveling time is not calculated. Switching points are controlled according to the positions.
- **Jerk-limited**  
A path with jerk-limiting is calculated for the position movement. The motion parameter "Jerk" is used for this ramp type.
- **Trapezoid V3**  
A trapezoid is taken as the basis for the calculation of the motion profile. The traveling time is not calculated. Switching points are controlled according to the positions.
- **Jump**  
A path profile is not calculated here, the position jump is directly specified instead. Internal path planning is switched off. Depending on the interpolation interval of the external interpolator, it may be necessary to adapt the "KC current" control parameter, [KC current](#) [► 52].

#### 5.3.5.12 Towing error

The parameter shows the maximum permissible towing error value.

Code:	<b>0x7D6C</b>
Access rights, read - write:	USER - PROFI
Data type:	FLOAT
Parameter value, min. - max.:	0 - MAX_FLOAT

This towing error value must not be exceeded during a positioning movement. Exceeding this value results in an "ERROR TOW" error message, [ERROR TOW](#) [► 98].

The parameter value can only be written if the motor is de-energized.

#### 5.3.5.13 Motion waiting time blocked

The parameter shows the time that has to elapse in order to trigger the "Motion was blocked" status display [s].

Code:	<b>0x7D6D</b>
Access rights, read - write:	USER - PROFI
Data type:	FLOAT
Parameter value, min. - max.:	0 - MAX_FLOAT

When using the brake, this value may not be less than half the brake timeout. Otherwise the blocking detection will interfere directly after the start of a movement command.

### 5.3.6 Gear

#### 5.3.6.1 Serial number

The parameter shows the serial number of the gear.

Code:	<b>0x7D37</b>
Access rights, read - write:	USER - SCHUNK
Data type:	UINT32
Parameter value, min. - max.:	0 - MAX_UINT32

The parameter value can only be written if the motor is de-energized.

#### 5.3.6.2 Ratio 1

The parameter shows the gear ratio 1: factor of the reduction from the motor to the drive.

Code:	<b>0x7D38</b>
Access rights, read - write:	USER - PROFI
Data type:	FLOAT
Parameter value, min. - max.:	0 - MAX_FLOAT

The parameter value can only be written if the motor is de-energized.

The parameter has direct interaction with the following parameter:

- Parameter [Positioning installation](#) [► 64]

#### 5.3.6.3 Ratio 2

The parameter shows the gear ratio 2: factor of the reduction from the position measuring system to the drive.

Code:	<b>0x7D39</b>
Access rights, read - write:	USER - PROFI
Data type:	FLOAT
Parameter value, min. - max.:	0 - MAX_FLOAT

The parameter value can only be written if the motor is de-energized.

The parameter is only required if the position measuring system is installed between the gear ratios.

The parameter has direct interaction with the following parameter:

- Parameter [Positioning installation](#) [► 64]

### 5.3.7 Brake

#### 5.3.7.1 Brake serial number

The parameter shows the serial number of the brake.

Code: **0x7D3C**  
 Access rights, read - write: USER - SCHUNK  
 Data type: UINT32  
 Parameter value, min. - max.: 0 - MAX\_UINT32

The parameter value can only be written if the motor is de-energized.

#### 5.3.7.2 Brake type

The parameter shows the type of brake.

The parameter value can only be written if the motor is de-energized.

A restart of the module is required after the writing.

Code: **0x7D3D**  
 Access rights, read - write: USER - ADVANCED  
 Data type: ENUM  
 Parameter value, min. - max.: see the following table

Parameter value	Designation
0	No brake
1	Magnetic 12V
2	Magnetic 24V
3	Magnetic 48V



### **⚠ DANGER**

#### **Danger due to non-functioning brake!**

An error in configuration can lead to a non-functioning brake and cause serious injury.

- Check that the configuration is free of errors

The parameter is used to automatically determine the voltage for the brake control.

### 5.3.7.3 Brake usage

The parameter shows how the brake is used.

The parameter value can only be written if the motor is de-energized.

A restart of the module is required after the writing.

Code: **0x7D3E**  
 Access rights, read - write: USER - ADVANCED  
 Data type: ENUM  
 Parameter value, min. - max.: see the following table

Parameter value	Designation
0	Not used
1	Only in case of error
2	Normal

- **Not used (0)**
  - The brake is only applied in the event of voltage failure and is released once, immediately after starting the module.
- **Only in case of error (1)**
  - The brake is only engaged in case of error and is released with the first movement command. The motor is permanently controlled.
- **Normal (2)**
  - The brake is engaged in the event of an error and at the end of the movement.

The parameter has direct interaction with the following function:

- Function [Pseudo-absolute encoder](#) [► 10]
  - If a brake is configured and additional conditions are met, then the pseudo-absolute encoder is active.

### 5.3.7.4 Brake timeout

The parameter shows the duration of the pause from the end of a traverse movement until the brake is engaged.

Code: **0x7D3F**  
 Access rights, read - write: USER - ADVANCED  
 Data type: FLOAT/INT32  
 Parameter value, min. - max.: 0 - MAX\_FLOAT

The parameter value can only be written if the motor is de-energized.

### 5.3.8 Voltage

#### 5.3.8.1 Min. motor voltage

The parameter shows the minimum permissible motor voltage.

Code:	<b>0x7D32</b>
Access rights, read - write:	USER - ADVANCED
Data type:	FLOAT
Parameter value, min. - max.:	10 MotorMaxVolt

If the voltage falls below the parameter value, an error message occurs, [ERROR MOTOR VOLTAGE LOW](#) [► 102].

#### 5.3.8.2 Max. motor voltage

The parameter shows the maximum permissible motor voltage.

Code:	<b>0x7D33</b>
Access rights, read - write:	USER - ADVANCED
Data type:	FLOAT
Parameter value, min. - max.:	MotorMinVolt 72

If the voltage falls below the parameter value, an error message occurs, [ERROR MOTOR VOLTAGE HIGH](#) [► 103]. If this error occurs repeatedly, the module is disabled and can only be put into operation again by SCHUNK. If the error occurs frequently due to the load, an external brake chopper can be used.

#### 5.3.8.3 Min. logic voltage

The parameter shows the minimum permissible logic voltage.

Code:	<b>0x7D34</b>
Access rights, read - write:	USER - ADVANCED
Data type:	FLOAT
Parameter value, min. - max.:	5 - LogicMaxVolt

If the voltage falls below the parameter value, an error message occurs, [ERROR LOGIC LOW](#) [► 102].

#### 5.3.8.4 Max. logic voltage

The parameter shows the maximum permissible logic voltage.

Code:	<b>0x7D35</b>
Access rights, read - write:	USER - ADVANCED
Data type:	FLOAT
Parameter value, min. - max.:	MotorMinVolt 30

If the voltage falls below the parameter value, an error message occurs, [ERROR LOGIC HIGH](#) [► 102].



### 5.3.9 Communication

#### 5.3.9.1 Main communication

The parameter shows the active communication interface.

Code: **0x7D87** (deactivated under PROFIBUS)  
 Access rights, read - write: USER - USER  
 Data type: ENUM  
 Parameter value, min. - max.: see the following table

Parameter value	Designation
0	Auto
1	Serial
2	CAN
3	PROFIBUS
4	Serial without spontaneous messaging
5	CANopen
6	None
7	Anybus
8	USB

#### 5.3.9.2 Module ID

The parameter shows the current module ID.

Code: **0x7D88** (deactivated under PROFIBUS)  
 Access rights, read - write: USER - USER  
 Data type: UINT8  
 Parameter value, min. - max.: 0 - MAX\_UINT8

### 5.3.9.3 Baud rate CAN

The parameter shows the baud rate for the CAN interface.

Code: **0x7D89** (deactivated under PROFIBUS)

Access rights, read - write: USER - USER

Data type: ENUM

Parameter value, min. - max.: see the following table

Parameter value	Designation
0	50 kbit
1	100 kbit
2	125 kbit
3	250 kbit
4	500 kbit
5	1 Mbit

### 5.3.9.4 Baud rate RS232

The parameter shows the baud rate for the serial interface.

Code: **0x7D8A** (deactivated under PROFIBUS)

Access rights, read - write: USER - USER

Data type: ENUM

Parameter value, min. - max.: see the following table

Parameter value	Designation
0	1200 Baud
1	2400 Baud
2	4800 Baud
3	9600 Baud
4	19200 Baud
5	38400 Baud
6	57600 Baud
7	115200 Baud

### 5.3.9.5 Protocol mode

The parameter shows the communication protocol between the module and PLC.

The same communication protocol must be parameterized in the module and in the PLC, otherwise there is no communication.

A restart of the module is required after the writing.

Code: **0x7DF3**  
 Access rights, read - write: USER - PROFI  
 Data type: ENUM  
 Parameter value, min. - max.: see the following table

Parameter value	Designation
0	Undefined
1	SMP Single Element
2	SMP Frame
3	SDP Single Element
4	SDP Frame
5	Ethernet IP
6	CANopen

## 5.3.10 General

### 5.3.10.1 EEPROM version

The parameter shows the version of the EEPROM.

Code: **0x7D9B**  
 Access rights, read - write: USER - ROOT  
 Data type: UINT16  
 Parameter value, min. - max.: 0 - MAX\_UINT16

### 5.3.10.2 EEPROM CRC

The parameter shows the checksum for all EEPROM data.

Code: **0x7D9C**  
 Access rights, read - write: USER - ROOT  
 Data type: UINT16  
 Parameter value, min. - max.: 0 - MAX\_UINT16

#### 5.3.10.3 Data CRC

The parameter shows the checksum for all module-specific EEPROM data.

Code: **0x7D9D**  
 Access rights, read - write: USER - ROOT  
 Data type: UINT16  
 Parameter value, min. - max.: 0 - MAX\_UINT16

#### 5.3.10.4 Configuration mode

The parameter shows which module communicates with the PLC. Different output parameters are stored for each corresponding module. The parameter must be the same in the module and in the PLC, otherwise this leads to an error in the hardware parameterization.

A restart of the module is required after the writing.

Code: **0x7DF4**  
 Access rights, read - write: USER - PROFI  
 Data type: ENUM  
 Parameter value, min. - max.: see the following table

Parameter value	Designation
0	Undefined
1	ERS
2	EGN/EZN
3	Reserved
4	EGL
5	PR
6	PDU
7	PSM
8	PW
9	PEH
10	PRH
11	PRL

### 5.3.11 Info

#### 5.3.11.1 Error 0

The parameter shows the last error n.

Code:	<b>0x7DA0</b>
Access rights, read - write:	USER - ROOT
Data type:	UINT8
Parameter value, min. - max.:	0 - MAX_UINT8

For further information on the info and error messages, see [Info codes](#) [► 93] and [Error codes](#) [► 96].

#### 5.3.11.2 Error 1

The parameter shows the error n-1.

Code:	<b>0x7DA1</b>
Access rights, read - write:	USER - ROOT
Data type:	UINT8
Parameter value, min. - max.:	0 - MAX_UINT8

For further information on the info and error messages, see [Info codes](#) [► 93] and [Error codes](#) [► 96].

#### 5.3.11.3 Error 2

The parameter shows the error n-2.

Code:	<b>0x7DA2</b>
Access rights, read - write:	USER - ROOT
Data type:	UINT8
Parameter value, min. - max.:	0 - MAX_UINT8

For further information on the info and error messages, see [Info codes](#) [► 93] and [Error codes](#) [► 96].

#### 5.3.11.4 Error 3

The parameter shows the error n-3.

Code:	<b>0x7DA3</b>
Access rights, read - write:	USER - ROOT
Data type:	UINT8
Parameter value, min. - max.:	0 - MAX_UINT8

For further information on the info and error messages, see [Info codes](#) [► 93] and [Error codes](#) [► 96].

#### 5.3.11.5 Error 4

The parameter shows the error n-4.

Code:	<b>0x7DA4</b>
Access rights, read - write:	USER - ROOT
Data type:	UINT8
Parameter value, min. - max.:	0 - MAX_UINT8

For further information on the info and error messages, see [Info codes](#) [► 93] and [Error codes](#) [► 96].

#### 5.3.11.6 Error 5

The parameter shows the error n-5.

Code:	<b>0x7DA5</b>
Access rights, read - write:	USER - ROOT
Data type:	UINT8
Parameter value, min. - max.:	0 - MAX_UINT8

For further information on the info and error messages, see [Info codes](#) [► 93] and [Error codes](#) [► 96].

#### 5.3.11.7 Error 6

The parameter shows the error n-6.

Code:	<b>0x7DA6</b>
Access rights, read - write:	USER - ROOT
Data type:	UINT8
Parameter value, min. - max.:	0 - MAX_UINT8

For further information on the info and error messages, see [Info codes](#) [► 93] and [Error codes](#) [► 96].

#### 5.3.11.8 Error 7

The parameter shows the error n-7.

Code:	<b>0x7DA7</b>
Access rights, read - write:	USER - ROOT
Data type:	UINT8
Parameter value, min. - max.:	0 - MAX_UINT8

For further information on the info and error messages, see [Info codes](#) [► 93] and [Error codes](#) [► 96].

**5.3.11.9 Error 8**

The parameter shows the error n-8.

Code:	<b>0x7DA8</b>
Access rights, read - write:	USER - ROOT
Data type:	UINT8
Parameter value, min. - max.:	0 - MAX_UINT8

For further information on the info and error messages, see [Info codes](#) [► 93] and [Error codes](#) [► 96].

**5.3.11.10 Error 9**

The parameter shows the error n-9.

Code:	<b>0x7DA9</b>
Access rights, read - write:	USER - ROOT
Data type:	UINT8
Parameter value, min. - max.:	0 - MAX_UINT8

For further information on the info and error messages, see [Info codes](#) [► 93] and [Error codes](#) [► 96].

**5.3.11.11 Error 10**

The parameter shows the error n-10.

Code:	<b>0x7DAA</b>
Access rights, read - write:	USER - ROOT
Data type:	UINT8
Parameter value, min. - max.:	0 - MAX_UINT8

For further information on the info and error messages, see [Info codes](#) [► 93] and [Error codes](#) [► 96].

**5.3.11.12 Error 11**

The parameter shows the error n-11.

Code:	<b>0x7DAB</b>
Access rights, read - write:	USER - ROOT
Data type:	UINT8
Parameter value, min. - max.:	0 - MAX_UINT8

For further information on the info and error messages, see [Info codes](#) [► 93] and [Error codes](#) [► 96].

#### 5.3.11.13 Error 12

The parameter shows the error n-12.

Code:	<b>0x7DAC</b>
Access rights, read - write:	USER - ROOT
Data type:	UINT8
Parameter value, min. - max.:	0 - MAX_UINT8

For further information on the info and error messages, see [Info codes](#) [► 93] and [Error codes](#) [► 96].

#### 5.3.11.14 Error 13

The parameter shows the error n-13.

Code:	<b>0x7DAD</b>
Access rights, read - write:	USER - ROOT
Data type:	UINT8
Parameter value, min. - max.:	0 - MAX_UINT8

For further information on the info and error messages, see [Info codes](#) [► 93] and [Error codes](#) [► 96].

#### 5.3.11.15 Error 14

The parameter shows the error n-14.

Code:	<b>0x7DAE</b>
Access rights, read - write:	USER - ROOT
Data type:	UINT8
Parameter value, min. - max.:	0 - MAX_UINT8

For further information on the info and error messages, see [Info codes](#) [► 93] and [Error codes](#) [► 96].

#### 5.3.11.16 Error 15

The parameter shows the error n-15.

Code:	<b>0x7DAF</b>
Access rights, read - write:	USER - ROOT
Data type:	UINT8
Parameter value, min. - max.:	0 - MAX_UINT8

For further information on the info and error messages, see [Info codes](#) [► 93] and [Error codes](#) [► 96].



**5.3.11.17 Error 16**

The parameter shows the error n-16.

Code:	<b>0x7DB0</b>
Access rights, read - write:	USER - ROOT
Data type:	UINT8
Parameter value, min. - max.:	0 - MAX_UINT8

For further information on the info and error messages, see [Info codes](#) [► 93] and [Error codes](#) [► 96].

**5.3.11.18 Error 17**

The parameter shows the error n-17.

Code:	<b>0x7DB1</b>
Access rights, read - write:	USER - ROOT
Data type:	UINT8
Parameter value, min. - max.:	0 - MAX_UINT8

For further information on the info and error messages, see [Info codes](#) [► 93] and [Error codes](#) [► 96].

**5.3.11.19 Error 18**

The parameter shows the error n-18.

Code:	<b>0x7DB2</b>
Access rights, read - write:	USER - ROOT
Data type:	UINT8
Parameter value, min. - max.:	0 - MAX_UINT8

For further information on the info and error messages, see [Info codes](#) [► 93] and [Error codes](#) [► 96].

**5.3.11.20 Error 19**

The parameter shows the error n-19.

Code:	<b>0x7DB3</b>
Access rights, read - write:	USER - ROOT
Data type:	UINT8
Parameter value, min. - max.:	0 - MAX_UINT8

For further information on the info and error messages, see [Info codes](#) [► 93] and [Error codes](#) [► 96].

### 5.3.12 Asynchronous

#### 5.3.12.1 Current main board temperature

The parameter shows the current temperature of the main board.

Code:	<b>0x7DB9</b>
Access rights, read - write:	USER - DISABLED
Data type:	FLOAT
Parameter value, min. - max.:	MIN_FLOAT - MAX_FLOAT

#### 5.3.12.2 Current motor temperature

The parameter shows the current temperature of the motor.

Code:	<b>0x7DBA</b>
Access rights, read - write:	USER - DISABLED
Data type:	FLOAT
Parameter value, min. - max.:	MIN_FLOAT - MAX_FLOAT

#### 5.3.12.3 Current OPT temperature Comm.

The parameter shows the current temperature of the communication board.

Code:	<b>0x7DBB</b>
Access rights, read - write:	USER - DISABLED
Data type:	FLOAT
Parameter value, min. - max.:	MIN_FLOAT - MAX_FLOAT

#### 5.3.12.4 Error line

The parameter shows the last error triggered.

Code:	<b>0x7DBC</b>
Access rights, read - write:	USER - DISABLED
Data type:	UINT16
Parameter value, min. - max.:	0 - MAX_UINT16

---

#### NOTE

Important information for service purposes.

---

**5.3.12.5 Error value**

The parameter shows the detail that led to the last error.

Code:	<b>0x7DBD</b>
Access rights, read - write:	USER - DISABLED
Data type:	FLOAT
Parameter value, min. - max.:	MIN_FLOAT - MAX_FLOAT

**NOTE**

Important information for service purposes.

**5.3.12.6 Error file**

The parameter shows the name of the file in which the last triggered error was saved.

Code:	<b>0x7DBE</b>
Access rights, read - write:	USER - DISABLED
Data type:	(CHAR_ARRAY) -
Parameter value, min. - max.:	-

**NOTE**

Important information for service purposes.

**5.3.12.7 Firmware type**

The parameter shows the firmware type of the module.

The parameter value can only be written if the motor is de-energized.

A restart of the module is required after the writing.

Code:	<b>0x7DBF</b>
Access rights, read - write:	USER - DISABLED
Data type:	ENUM
Parameter value, min. - max.:	see the following table

Parameter value	Designation
0	PTA

#### 5.3.12.8 Order number

The parameter shows the order number of the module.

The parameter value can only be written if the motor is de-energized.

A restart of the module is required after the writing.

Code:	<b>0x7DC0</b>
Access rights, read - write:	USER - SCHUNK
Data type:	UINT32
Parameter value, min. - max.:	0 - MAX_UINT32

---

#### NOTE

Important information for service purposes.

---

#### 5.3.12.9 Module name

The parameter shows the name of the module.

The parameter value can only be written if the motor is de-energized.

A restart of the module is required after the writing.

Code:	<b>0x7DC1</b>
Access rights, read - write:	USER - SCHUNK
Data type:	(CHAR_ARRAY) -
Parameter value, min. - max.:	-

---

#### NOTE

Important information for service purposes.

---

#### 5.3.12.10 Date firmware on the main board

The parameter shows the compile date of the main board firmware.

Code:	<b>0x7DC2</b>
Access rights, read - write:	USER - DISABLED
Data type:	UINT32
Parameter value, min. - max.:	0 - MAX_UINT32

---

#### NOTE

Important information for service purposes.

---

**5.3.12.11 Time firmware on the main board**

The parameter shows the compile time of the main board firmware.

Code: **0x7DC3**  
 Access rights, read - write: USER - DISABLED  
 Data type: UINT32  
 Parameter value, min. - max.: 0 - MAX\_UINT32

**NOTE**

Important information for service purposes.

**5.3.12.12 Hardware version on the main board**

The parameter shows the hardware version of the main board.

Code: **0x7DC4**  
 Access rights, read - write: USER - DISABLED  
 Data type: ENUM  
 Parameter value, min. - max.: see the following table

Parameter value	Designation
530	PTA 5.3
531	PTA 5.3 with FRAM
540	PTA 5.4
541	PTA 5.4 with FRAM
611.	PTA 6.1
613.	PTA 6.1 with IO Expander
615.	PTA 6.1 with ARM Co-processor
621	PTA 6.2
623	PTA 6.2 with IO Expander
625	PTA 6.2 with ARM Co-processor
631	PTA 6.3
633	PTA 6.3 with IO Expander
635	PTA 6.3 with ARM Co-processor
801	ERB-ECU-1 for ERB 130
802	ERB-ECU-1 for ERB 150
803	ERB-ECU-1 for ERB 170
991	Unknown PTA
993	Unknown PTA with IO Expander
995	Unknown PTA with ARM Co-processor

**NOTE**

Important information for service purposes.

#### 5.3.12.13 Firmware version on the main board

The parameter shows the firmware version of the main board.

Code:	<b>0x7DC5</b>
Access rights, read - write:	USER - DISABLED
Data type:	UINT16
Parameter value, min. - max.:	0 - MAX_UINT16

---

##### NOTE

Important information for service purposes.

---

#### 5.3.12.14 Date OPT firmware. Comm.

The parameter shows the compile date of the communication board firmware.

Code:	<b>0x7DC6</b>
Access rights, read - write:	USER - SCHUNK
Data type:	UINT32
Parameter value, min. - max.:	0 - MAX_UINT32

---

##### NOTE

Important information for service purposes.

---

#### 5.3.12.15 Time OPT firmware. Comm.

The parameter shows the compile time of the communication board firmware.

Code:	<b>0x7DC7</b>
Access rights, read - write:	USER - DISABLED
Data type:	UINT32
Parameter value, min. - max.:	0 - MAX_UINT32

---

##### NOTE

Important information for service purposes.

---

**5.3.12.16 Hardware OPT version Comm.**

The parameter shows the hardware version of the communication board.

Code: **0x7DC8**  
 Access rights, read - write: USER - DISABLED  
 Data type: ENUM  
 Parameter value, min. - max.: see the following table

Parameter value	Designation	Parameter value	Designation
100	V6R PROFIBUS 1.1	130	ECM CAN-Bus 1.1
101	V6R PROFIBUS 1.2	131	ECM CAN-Bus 1.2
102	V6R PROFIBUS 1.3	132	ECM CAN-Bus 1.3
103	V6R PROFIBUS 1.4	132	ECM CAN-Bus 1.4
104	V6R PROFIBUS 1.5	133	ECM CAN-Bus 1.5
105	V6R PROFIBUS 1.6	134	ECM CAN-Bus 1.6
110	V6R CAN-Bus 1.1	135	V6R PROFINET 1.1
111	V6R CAN-Bus 1.2	140	V6R PROFINET 1.2
112	V6R CAN-Bus 1.3	142	V6R PROFINET 1.3
113	V6R CAN-Bus 1.4	143	V6R PROFINET 1.4
114	V6R CAN-Bus 1.5	144	V6R PROFINET 1.5
115	V6R CAN-Bus 1.6	145	V6R PROFINET 1.6
120	ECM PROFIBUS 1.1	150	ECM PROFINET 1.1
121	ECM PROFIBUS 1.2	151	ECM PROFINET 1.2
122	ECM PROFIBUS 1.3	152	ECM PROFINET 1.3
123	ECM PROFIBUS 1.4	153	ECM PROFINET 1.4
124	ECM PROFIBUS 1.5	154	ECM PROFINET 1.5
125	ECM PROFIBUS 1.6	155	ECM PROFINET 1.6
		255	Unknown

**NOTE**

Important information for service purposes.

#### 5.3.12.17 Firmware OPT version Comm.

Access rights, read - write:

The parameter shows the firmware version of the communication board.

Code:	<b>0x7DC9</b>
Access rights, read - write:	USER - DISABLED
Data type:	UINT16
Parameter value, min. - max.:	0 - MAX_UINT16

---

##### **NOTE**

Important information for service purposes.

---

#### 5.3.12.18 OPT serial number Comm.

The parameter shows the serial number of the communication board.

Code:	<b>0x7DCA</b>
Access rights, read - write:	USER - SCHUNK
Data type:	UINT32
Parameter value, min. - max.:	0 - MAX_UINT32

---

##### **NOTE**

Important information for service purposes.

---

#### 5.3.12.19 Date Firmware OPT 1

The parameter shows the compile date of the expansion board 1 firmware.

Code:	<b>0x7DCB</b>
Access rights, read - write:	USER - DISABLED
Data type:	UINT32
Parameter value, min. - max.:	0 - MAX_UINT32

---

##### **NOTE**

Important information for service purposes.

---



**5.3.12.20 Time firmware OPT 1**

The parameter shows the compile time of the expansion board 1 firmware.

Code:	<b>0x7DCC</b>
Access rights, read - write:	USER - DISABLED
Data type:	UINT32
Parameter value, min. - max.:	0 - MAX_UINT32

**NOTE**

Important information for service purposes.

**5.3.12.21 Hardware OPT 1 version**

The parameter shows the hardware version of the expansion board 1.

Code:	<b>0x7DCD</b>
Access rights, read - write:	USER - DISABLED
Data type:	UINT16
Parameter value, min. - max.:	0 - MAX_UINT16

**NOTE**

Important information for service purposes.

**5.3.12.22 Firmware OPT 1 version**

The parameter shows the firmware version of the expansion board 1.

Code:	<b>0x7DCE</b>
Access rights, read - write:	USER - DISABLED
Data type:	UINT16
Parameter value, min. - max.:	0 - MAX_UINT16

**NOTE**

Important information for service purposes.

**5.3.12.23 OPT 1 serial number**

The parameter shows the serial number of the expansion board 1.

Code:	<b>0x7DCF</b>
Access rights, read - write:	USER - SCHUNK
Data type:	UINT32
Parameter value, min. - max.:	0 - MAX_UINT32

**NOTE**

Important information for service purposes.

#### 5.3.12.24 Date firmware OPT 2

The parameter shows the compile date of the expansion board 2 firmware.

Code:	<b>0x7DD0</b>
Access rights, read - write:	USER - DISABLED
Data type:	UINT32
Parameter value, min. - max.:	0 - MAX_UINT32

---

#### NOTE

Important information for service purposes.

---

#### 5.3.12.25 Time firmware OPT 2

The parameter shows the compile time of the expansion board 2 firmware.

Code:	<b>0x7DD1</b>
Access rights, read - write:	USER - DISABLED
Data type:	UINT32
Parameter value, min. - max.:	0 - MAX_UINT32

---

#### NOTE

Important information for service purposes.

---

#### 5.3.12.26 Hardware OPT 2 version

The parameter shows the hardware version of the expansion board 2.

Code:	<b>0x7DD2</b>
Access rights, read - write:	USER - DISABLED
Data type:	UINT16
Parameter value, min. - max.:	0 - MAX_UINT16

---

#### NOTE

Important information for service purposes.

---

#### 5.3.12.27 Firmware OPT 2 version

The parameter shows the firmware version of the expansion board 2.

Code:	<b>0x7DD3</b>
Access rights, read - write:	USER - DISABLED
Data type:	UINT16
Parameter value, min. - max.:	0 - MAX_UINT16

---

#### NOTE

Important information for service purposes.

---

**5.3.12.28 OPT 2 serial number**

The parameter shows the serial number of the expansion board 2.

Code:	<b>0x7DD4</b>
Access rights, read - write:	USER - SCHUNK
Data type:	UINT32
Parameter value, min. - max.:	0 - MAX_UINT32

**NOTE**

Important information for service purposes.

**5.3.12.29 Protocol version**

The parameter shows the version of the protocol used.

Code:	<b>0x7DD5</b>
Access rights, read - write:	USER - DISABLED
Data type:	UINT16
Parameter value, min. - max.:	0 - MAX_UINT16

**5.3.12.30 Motor voltage**

The parameter shows the current motor voltage [V].

Code:	<b>0x7DD6</b>
Access rights, read - write:	USER - DISABLED
Data type:	FLOAT
Parameter value, min. - max.:	0 - MAX_FLOAT

**5.3.12.31 Logic voltage**

The parameter shows the current logic voltage [V].

Code:	<b>0x7DD7</b>
Access rights, read - write:	USER - DISABLED
Data type:	FLOAT
Parameter value, min. - max.:	0 - MAX_FLOAT

**5.3.12.32 Max. software end stop (temporary)**

The parameter shows the maximum value of the software stop.

Code:	<b>0x7DD8</b>
Access rights, read - write:	PROFI - PROFi
Data type:	FLOAT/INT32
Parameter value, min. - max.:	MIN_FLOAT - MAX_FLOAT

If the value is changed, this is only done temporarily without storing the change in the EEPROM.

#### 5.3.12.33 Min. software end stop (temporary)

The parameter shows the minimum value of the software stop.

Code: **0x7DD9**  
 Access rights, read - write: PROFI - PROFI  
 Data type: FLOAT/INT32  
 Parameter value, min. - max.: MIN\_FLOAT - MAX\_FLOAT

If the value is changed, this is only done temporarily without storing the change in the EEPROM.

#### 5.3.12.34 User

The parameter shows the current user, [User administration](#) [► 7].

Code: **0x7DDA**  
 Access rights, read - write: USER -  
 Data type: ENUM while reading  
 (CHAR\_ARRAY) - while writing  
 Parameter value, min. - max.: see the following table

Parameter value	Designation
0	User
1	Diag
2	Profi
3	Advanced
4	Root
5	SCHUNK

The current user is reported in the case of read-only access.

## 6 Info and error messages



*Error message*

In the event of an error, D-Len always has exactly the value "2" in the data frame from the module to the control unit.

Either the error command is in the command byte or one of the following "error commands", [Troubleshooting](#) [► 35].

The parameter byte contains information on the cause of the error.

### 6.1 Detailed error information

Information about the last error that occurred can be queried using the following parameters:

- Error Detail
- [Error value](#) [► 83]
  - Error Line
- [Error line](#) [► 82]
  - Error File
- [Error file](#) [► 83]

### 6.2 Info codes

#### 6.2.1 INFO BOOT

The module has booted successfully.

Code: **0x0001**

This is triggered after the module has been switched on and undergone a successful initialization. If this message occurs during operation, the logic voltage must be checked. It is also possible that there is a defective power drive.

#### 6.2.2 INFO NO RIGHTS

The appropriate rights to execute the command are missing.

Code: **0x03**

### 6.2.3 INFO UNKNOWN COMMAND

The sent command is not recognized.

Code: **0x04**

### 6.2.4 INFO FAILED

The command has failed.

Code: **0x05**

All of the parameters are correct, but the execution of the command is not possible at this time due to other reasons, e. g. the module is in emergency stop mode.

### 6.2.5 NOT REFERENCED

The module is not referenced and can therefore not execute the command.

Code: **0x06**

A referencing process is necessary in order to carry out a positioning movement.

### 6.2.6 INFO SEARCH SINE VECTOR

Code:: 0x0007

A search is running for the space vector for the sine commutation. 60% of the maximum current is used for the phases.

Code: **0x0007**

If the standstill commutation is not active and the commutation type is set to PMSM, a movement command for the space vector is carried out once after the power up and before the execution, [Standstill commutation](#) [► 11] and [Motor Type](#) [► 44].

### 6.2.7 INFO NO ERROR

There are no other error messages pending.

Code: **0x0008**

If no error or warning is present, or if the module has moved out of the software limit stops, this info message is displayed.

### 6.2.8 INFO COMMUNICATION ERROR

An error has occurred in communication.

Code: **0x09**

The connection of the communication cable and external influences on the communication cable must be checked.

**6.2.9 INFO TIMEOUT**

A timeout occurred during communication.

Code: **0x10**

The data could not be sent, or more data was anticipated and was not received on time.

**6.2.10 INFO WRONG DATA TYPE**

The datatype does not match the parameter.

Code: **0x12**

**6.2.11 INFO RESTART**

The module or control unit was restarted.

Code: **0x13**

**6.2.12 INFO CHECKSUM**

The checksum is incorrect, the data is invalid.

Code: **0x19**

**6.2.13 INFO VALUE LIMIT MAX**

The specified value exceeds the maximum permitted set value.

Code: **0x1B**

**6.2.14 INFO VALUE LIMIT MIN**

The specified value falls below the minimum permitted set value.

Code: **0x1C**

**6.2.15 INFO MESSAGE LENGTH**

The length command does not match the data received.

Code: **0x1D**

**6.2.16 INFO WRONG PARAMETER**

One of the specified parameter values is outside of the permitted range.

Code: **0x1E**

If a parameter value is found to be not permitted, all old parameter values will be retained, even if the other parameter values should lie in the valid range.

### **6.2.17 INFO UNKNOWN PARAMETER**

The parameter requested is unknown.

Code: **0x23**

## **6.3 Error codes**

### **6.3.1 ERROR FILE NOT FOUND**

The file to be edited is not on the USB stick or the USB stick is defective.

Code: **0x60**

### **6.3.2 ERROR FILE IS CORRUPT**

The file to be edited on the USB stick is corrupted.

Code: **0x61**

### **6.3.3 ERROR FILE TYPE WRONG**

The data type is not correct.

Code: **0x62**

- Create a new file.

### **6.3.4 ERROR FILE SYSTEM WRONG**

The file system from the USB stick has an error.

Code: **0x64**

- Check whether the USB drive is formatted with FAT16 or FAT32.

### **6.3.5 ERROR FILE READ**

A reading error has occurred during the reading of the file.

Code: **0x65**

### **6.3.6 ERROR FILE IS NOT CREATED**

No file could be created.

Code: **0x66**

- Check whether the USB stick is defective or write-protected.

### **6.3.7 ERROR FILE WRITE**

A writing error has occurred during the writing of the file.

Code: **0x67**



### 6.3.8 ERROR REBOOT

A parameter was written which is needed by a reboot.

Code: **0x7C**

- Switch the module off and on.

### 6.3.9 ERROR MOTOR PHASE

One motor phase is not properly connected.

Code: **0x7D**

### 6.3.10 ERROR WRONG RAMP TYPE

No valid motion profile has been selected for the positioning movement.

Code: **0xC8**

### 6.3.11 ERROR WRONG DIRECTION

The module moves in the wrong direction for the check of the pseudo-absolute encoder

Code: **0xD1**

- Check the sine pointer.

### 6.3.12 ERROR CONFIG MEMORY

The configuration range is incorrect. Data could not be written to EEPROM or EEPROM is defective.

Code: **0xD2**

### 6.3.13 ERROR SOFT LOW

The module has exceeded the lower software limit range.

Code: **0xD5**

- If necessary, acknowledge the error and move the module out of the software end stop with a movement command.

### 6.3.14 ERROR SOFT HIGH

The module has exceeded the upper software limit range.

Code: **0xD6**

- If necessary, acknowledge the error and move the module out of the software end stop with a movement command.

#### 6.3.15 ERROR SERVICE

An error has occurred that can only be remedied by SCHUNK.

Code: **0xD8**

The detailed error information can be used by SCHUNK Service to localize the error precisely, [Detailed error information](#) [► 93].

Contact SCHUNK's service and provide the following data:

- Module type
- Serial number of the module
- Description of how the error occurred

#### 6.3.16 ERROR FAST STOP

A fast stop was triggered.

Code: **0xD9**

#### 6.3.17 ERROR TOW

A towing error has occurred.

Code: **0xDA**

- Reduce the load.
- Check the "towing error" parameter.

#### 6.3.18 ERROR VPC3

The controller works incorrectly or is faulty.

Code: **0xDB**

#### 6.3.19 ERROR FRAGMENTATION

An error has occurred in the fragmentation protocol. Data packets have been lost.

Code: **0xDC**

### 6.3.20 ERROR COMMUTATION

The motor cannot commute.

Code: **0xDD**

If this error occurs, the commutation mode is selected incorrectly. In case of block commutation the hall sensors are defective or not connected. With regard to sine commutation, there is an error in the position measuring system.

### 6.3.21 ERROR I<sup>2</sup>T

An I<sup>2</sup>T error has occurred.

Code: **0xDF**

- Reduce load of motor.

### 6.3.22 ERROR CURRENT

The maximum current was exceeded.

Code: **0xDE**

- Reduce load of motor.

### 6.3.23 ERROR TOO FAST

The maximum speed was exceeded.

Code: **0xE4**

### 6.3.24 ERROR POS SYSTEM

The position measuring system is not working correctly.

Code: **0xE5**

- Check configuration of the module.

### 6.3.25 ERROR RESOLVER CHECK FAILED

A parameter for the resolver setting is incorrect.

Code: **0xEB**

#### 6.3.26 ERROR MATH

A mathematical error has occurred, e. g. division by zero.

Code: **0xEC**

Usually, a configuration parameter is incorrect, resulting in the value range being exceeded. In most cases, a controller parameter is set incorrectly.

The detailed error information can be used by SCHUNK Service to localize the error precisely, [Detailed error information](#) [► 93].

#### 6.3.27 ERROR CALIB CURRENT

The measured values of the current sensors are beyond the tolerance limits.

Code: **0xEE**

- Calibrate the module.
  - If the error occurs again, the current measuring is defective.

#### 6.3.28 ERROR INITIALIZE

The module could not be properly initialized.

Code: **0xE0**

- Check configuration parameters.

The detailed error information can be used by SCHUNK Service to localize the error precisely, [Detailed error information](#) [► 93].

#### 6.3.29 ERROR INTERNAL

An internal error has occurred.

Code: **0xE1**

The firmware is in an undefined status.

Contact SCHUNK's service and provide the following data:

- Module type
- Serial number of the module
- Description of how the error occurred

#### 6.3.30 ERROR CONNECTION TEMP LOW

The temperature of the communication board dropped below the minimal permissible level.

Code: **0x6A**

- Warm up the module.

### 6.3.31 ERROR CONNECTION TEMP HIGH

The maximum permissible temperature of the communication board was exceeded.

Code: **0x6B**

- Allow the module to cool down.
- Reduce the load.

### 6.3.32 ERROR MOTOR TEMP LOW

The temperature of the motor dropped below the minimal permissible level.

Code: **0x6C**

- Warm up the module.

### 6.3.33 ERROR MOTOR TEMP HIGH

Code: 0x6D

The temperature of the motor has exceeded the maximum permissible level.

Code: **0x6D**

- Allow the module to cool down.
- Reduce the load.

### 6.3.34 ERROR TEMP LOW OPTION

The temperature of the option board has fallen below the permissible temperature range.

Code: **0x6E**

- Warm up the module.

### 6.3.35 ERROR TEMP HIGH OPTION

The temperature of the option board has exceeded the permissible temperature range.

Code: **0x6F**

- Allow the module to cool down.
- Reduce the load.

#### **6.3.36 ERROR TEMP LOW**

The temperature value for the main board dropped below the minimal permissible temperature range.

Code: **0x70**

- Warm up the module.

#### **6.3.37 ERROR TEMP HIGH**

The temperature value for the main board exceeded the maximum permissible temperature range.

Code: **0x71**

- Allow the module to cool down.
- Reduce the load.

#### **6.3.38 ERROR LOGIC LOW**

The logic voltage is below the limit values.

Code: **0x72**

- Check the logic voltage.

#### **6.3.39 ERROR LOGIC HIGH**

The logic voltage is above the limit values.

Code: **0x73**

- Check the logic voltage.

#### **6.3.40 ERROR MOTOR VOLTAGE LOW**

The motor voltage is under the limit values.

Code: **0x74**

- Check the motor voltage.
  - If necessary, the power supply unit for the motor voltage might be underdimensioned or the voltage supply cables to the module are not dimensioned correctly.

---

#### **NOTE**

MotorVoltageLow is a fatal error when the module is moved.

---

#### 6.3.41 ERROR MOTOR VOLTAGE HIGH

The motor voltage is above the limit values.

Code: **0x75**

---

##### NOTE

If this error occurs repeatedly, the module is disabled and can only be put into operation again by SCHUNK.

- Check the motor voltage.
  - If necessary, an external brake chopper might have to be used.

#### 6.3.42 ERROR CABLE BREAK

Communication to the control was interrupted.

Code: **0x76**

---

##### NOTE

The error is only displayed once communication has been re-established.

- Check communication cables
  - The communication cable is defective.

#### 6.3.43 ERROR LIFE SIGN

The internal communication has failed. The module must be re-started.

Code: **0x7A**

#### 6.3.44 ERROR CUSTOM DEFINED

An error has occurred in a customer-defined function.

Code: **0x7B**

#### 6.3.45 ERROR OVERSHOOT

The module has overshoot the target position.

Code: **0x82**

- Increase the specified current.
  - The current needed for deceleration is too low.
- Check the controller parameters.

#### **6.3.46 ERROR HARDWARE VERSION**

The hardware of the different components do not match. One of the files saved on the USB stick can not be edited with the available hardware.

Code: **0x83**

#### **6.3.47 ERROR SOFTWARE VERSION**

The software of the different components does not match. One of the files saved on the USB stick can not be edited with the available software version.

Code: **0x84**