



STÖBER

CANopen®

Operation manual

Fundamentals

Communication

Parameters

V 5.6-H or later

09/2013

en

Table of contents

| | | |
|----------|---|-----------|
| 1 | Introduction | 5 |
| 1.1 | Purpose of the manual | 5 |
| 1.2 | Readers | 5 |
| 1.3 | Other manuals | 5 |
| 1.4 | Further support | 6 |
| 2 | Notes on Safety | 7 |
| 2.1 | Component part of the product | 7 |
| 2.2 | Operation in accordance with its intended use | 7 |
| 2.3 | Qualified personnel | 8 |
| 2.4 | Transportation and storage | 8 |
| 2.5 | Installation and connection | 9 |
| 2.6 | Service | 9 |
| 2.7 | Disposal | 9 |
| 2.8 | Presentation of notes on safety | 10 |
| 3 | Mounting | 11 |
| 3.1 | Install in MDS 5000 or SDS 5000 | 11 |
| 3.2 | Installation in the FDS 5000 | 14 |
| 4 | Electrical installation | 17 |
| 4.1 | Setup | 17 |
| 4.2 | Connection | 18 |
| 4.2.1 | Cable specifications | 18 |
| 4.2.2 | Shielding | 18 |
| 4.2.3 | Terminating resistance | 19 |
| 5 | Fundamentals of CAN Bus | 20 |
| 5.1 | CANopen® | 20 |
| 5.2 | NMT state machine | 20 |
| 5.3 | Parameters in the CANopen® system of the 5th generation of STÖBER inverters | 21 |
| 6 | User interface of the CAN 5000 | 22 |

| | | |
|----------|---|-----------|
| 7 | Setup of communication | 24 |
| 7.1 | Telegram Service | 24 |
| 7.1.1 | Identifier | 25 |
| 7.2 | Pre-defined Connection Set | 26 |
| 7.3 | Dynamic Distribution | 28 |
| 7.3.1 | Procedure | 28 |
| 7.3.2 | Example 1 | 29 |
| 7.3.3 | Example 2 | 29 |
| 7.3.4 | Example 3 | 29 |
| 8 | Data transmission with PDO and SDO | 30 |
| 8.1 | Process data transmission with PDO | 31 |
| 8.1.1 | Process data mapping | 31 |
| 8.1.2 | Transmission parameters | 33 |
| 8.1.3 | Parameters of the PDO channels | 35 |
| 8.2 | Parameter transmission with SDO | 39 |
| 8.2.1 | SDO channels | 40 |
| 8.2.2 | Expedited Transfer | 40 |
| 8.2.3 | Segmented Transfer | 42 |
| 8.2.4 | Error codes for SDO services | 47 |
| 9 | Further communication objects | 48 |
| 9.1 | NMT | 48 |
| 9.2 | SYNC | 49 |
| 9.3 | Emergency | 50 |
| 9.4 | Error control protocols | 52 |
| 9.4.1 | BOOT-UP | 52 |
| 9.4.2 | Node-Guarding | 53 |
| 9.4.3 | Heartbeat | 54 |



| | |
|---|----|
| 10 Simple commissioning | 55 |
| 11 Parameter list | 57 |
| 11.1 List of the CANopen® objects | 57 |
| 11.2 List of the drive parameters | 63 |
| 12 List of literature | 64 |



1 Introduction

1.1 Purpose of the manual

This manual gives you information on the connection of the 5th generation of STÖBER inverters to the CANopen[®] fieldbus system. The structure of CANopen[®] and the principal procedures are also discussed.

Goals of the manual:

- Provide you with a basic knowledge of CANopen[®] communication.
- Offer you support when you design an application and configure communication.

1.2 Readers

Users who are familiar with the control of drive systems and have a knowledge of commissioning inverter systems are the target group of this manual.

1.3 Other manuals

The documentation of the MDS 5000 includes the following manuals:

| Manual | Contents | ID | Latest version ^{a)} |
|----------------------------|--|--------|------------------------------|
| Commissioning Instructions | Reinstallation, replacement, function test | 442297 | V 5.6-H |
| Projecting manual | Installation and connection | 442273 | V 5.6-H |
| Operating manual | Set up the inverter | 442285 | V 5.6-H |

a) At the time of publication. You can find all versions at www.stoeber.de > Products > Doc Center.

The documentation of the FDS 5000 includes the following manuals:

| Manual | Contents | ID | Latest version ^{a)} |
|----------------------------|--|--------|------------------------------|
| Commissioning Instructions | Reinstallation, replacement, function test | 442293 | V 5.6-H |
| Projecting manual | Installation and connection | 442269 | V 5.6-H |
| Operating manual | Set up the inverter | 442281 | V 5.6-H |

a) At the time of publication. You can find all versions at www.stoeber.de > Products > Doc Center.

The documentation of the SDS 5000 includes the following manuals:

| Manual | Contents | ID | Latest version ^{a)} |
|----------------------------|--|--------|------------------------------|
| Commissioning Instructions | Reinstallation, replacement, function test | 442301 | V 5.6-H |
| Projecting manual | Installation and connection | 442277 | V 5.6-H |
| Operating manual | Set up the inverter | 442289 | V 5.6-H |

a) At the time of publication. You can find all versions at www.stoeber.de > Products > Doc Center.



You can find information on the POSITool software in the following manuals:

| Manual | Contents | ID | Latest version ^{a)} |
|---------------------------|--|--------|------------------------------|
| POSITool operating manual | Information on the basic functions of POSITool | 442233 | V 5.6-H |
| Programming manual | Information on programming with POSITool | 441693 | V 5.6-H |

a) At the time of publication. You can find all versions at www.stoeber.de > Products > Doc Center.

Note that the programming functionality of POSITool can only be used after training by STÖBER ANTRIEBSTECHNIK. You can find information on training at www.stoeber.de

1.4 Further support

If you have technical questions that are not answered by this document, please contact:

- Phone: +49 7231 582-3060
- E-mail: applications@stoeber.de

If you have questions about the documentation, please contact:

- E-mail: electronics@stoeber.de

If you have questions about training sessions, please contact:

- E-mail: training@stoeber.de



2 Notes on Safety

The devices may cause risks. For these reasons, comply with the following:

- The safety notes listed in the following sections and points
- The technical rules and regulations.

In addition, always read the appropriate documentation. STÖBER ANTRIEBSTECHNIK GmbH + Co. KG accepts no liability for damages caused by non-adherence to the instructions or applicable regulations. Subject to technical changes to improve the devices without prior notice. This documentation is purely a product description. It does not represent promised properties in the sense of warranty law.

2.1 Component part of the product

The technical documentation is a component part of a product.

- Since the technical documentation contains important information, always keep it handy in the vicinity of the device until the machine is disposed of.
- If the product is sold, disposed of, or rented out, always include the technical documentation with the product.

2.2 Operation in accordance with its intended use

The CAN 5000 accessory is only intended for establishing communication between devices from the 5th generation of STÖBER inverters and a CANopen® network.

Improper use includes integration in other communication networks.



2.3 Qualified personnel

Since the devices may harbor residual risks, all configuration, transportation, installation and commissioning tasks including operation and disposal may only be performed by trained personnel who are aware of the possible risks. Personnel must have the qualifications required for the job. The following table lists examples of occupational qualifications for the jobs:

| Activity | Possible occupational qualifications |
|--|--|
| Transportation and storage | Worker skilled in storage logistics or comparable training |
| Configuration | - Graduate engineer (electro-technology or electrical power technology) - Technician (m/f) (electro-technology) |
| Installation and connection | Electronics technician (m/f) |
| Commissioning (of a standard application) | - Technician (m/f) (electro-technology) - Master electro technician (m/f) |
| Programming | Graduate engineer (electro-technology or electrical power technology) |
| Operation | - Technician (m/f) (electro-technology) - Master electro technician (m/f) |
| Disposal | Electronics technician (m/f) |

Tab. 2-1: examples of occupational qualifications

In addition, the valid regulations, the legal requirements, the reference books, this technical documentation and, in particular, the safety information contained therein must be carefully

- read
- understood and
- complied with

2.4 Transportation and storage

Inspect the delivery for any transport damage immediately after you receive it. Notify any damage to the transport company immediately. Do not operate the product if damaged. Store the device in a dry and dust-free room if you do not install it immediately

2.5 Installation and connection

The accessory installation instructions allow the following actions during the installation of accessories:

- The housing in the upper slot can be opened.

Opening the housing in another place or for other purposes is not permitted.

Installation and connection work are only permitted after the device has been isolated from the power!

Apply the 5 safety rules in the order stated before performing any work on the machine:

1. Enable. Also enable the auxiliary circuits.
2. Secure against restart.
3. Check that voltage is not present.
4. Earth and short circuit.
5. Cover adjacent live parts.



Information

Note that the discharge time of the DC link capacitors is 5 minutes. You can only determine the absence of voltage after this time period.

Afterwards you can carry out the work.

2.6 Service

Repairs must only be performed by STÖBER ANTRIEBSTECHNIK GmbH + Co.

KG. Send faulty devices with a fault description to:

STÖBER ANTRIEBSTECHNIK GmbH + Co. KG

Abteilung VS-EL

Kieselbronner Str.12

75177 Pforzheim, Germany

GERMANY

2.7 Disposal

Please comply with the latest national and regional regulations! Dispose of the individual parts separately depending on their nature and currently valid regulations such as, for example:

- Electronic scrap (PCBs)
- Plastic
- Sheet metal
- Copper
- Aluminum

2.8 Presentation of notes on safety

NOTICE

Notice

means that property damage may occur

- ▶ if the stated precautionary measures are not taken.
-



CAUTION!

Caution

with warning triangle means that minor injury may occur

- ▶ if the stated precautionary measures are not taken.
-



WARNING!

Warning

means that there may be a serious danger of death

- ▶ if the stated precautionary measures are not taken.
-



DANGER!

Danger

means that serious danger of death exists

- ▶ if the stated precautionary measures are not taken.
-



Information

indicates important information about the product or a highlighted portion of the documentation which requires special attention.

3 Mounting

The fieldbus module CANopen[®] DS-301 (CAN 5000) must first be installed before the inverter of the 5th generation of STÖBER inverters can be integrated in a CAN bus system.

We recommend ordering the option card when you order the inverter since the card will already have been installed by STÖBER ANTRIEBSTECHNIK before delivery.

If you install the option card yourself, proceed as described in the following sections.

3.1 Install in MDS 5000 or SDS 5000



WARNING!

Danger of injury/death and property damage due to electric shock!

- ▶ Before installing accessories, turn off all voltage supplies! Then wait 5 minutes for the DC link capacitors to discharge. Never begin with accessory installation until after this!



CAUTION!

Danger of property damage due to electrostatic discharge, among others!

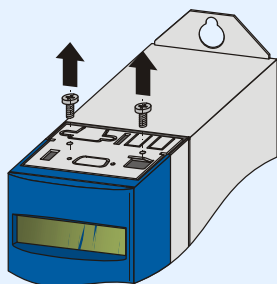
- ▶ Provide suitable protective measures while handling open PCBs (e.g., ESD clothing, environment free of dirt and grease).
- ▶ Do not touch the contact surfaces.

You will need the following tools to install the CAN 5000:

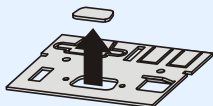
- TX10 Torx screwdriver
- Pliers
- Hexagon socket wrench, 4.5 mm

Installing the CAN 5000

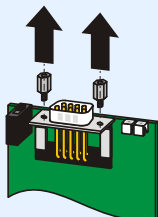
1. Unscrew the mounting screws and remove the cover plate:



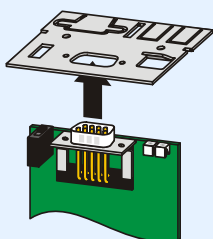
2. Remove the punched out metal section with the pliers:



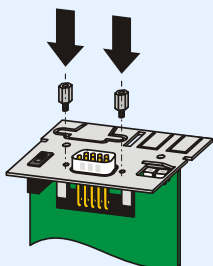
3. Remove the screws from the option PCB:



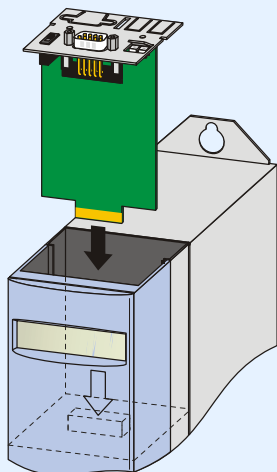
4. Stick the sub D plug connector of the PCB through the plate from below:



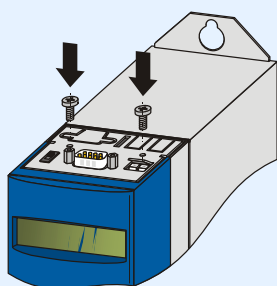
5. Secure the PCB to the plate with the screw in step 3:



6. Insert the option PCB in the inverter so that the gold contacts slide into the black terminal block:



7. Secure the plate to the inverter with the mounting screws:



⇒ You have installed the accessory.

3.2 Installation in the FDS 5000



WARNING!

Danger of injury/death and property damage due to electric shock!

- ▶ Before installing accessories, turn off all voltage supplies! Then wait 5 minutes for the DC link capacitors to discharge. Never begin with accessory installation until after this!



CAUTION!

Danger of property damage due to electrostatic discharge, among others!

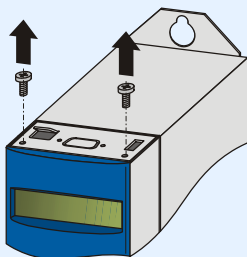
- ▶ Provide suitable protective measures while handling open PCBs (e.g., ESD clothing, environment free of dirt and grease).
- ▶ Do not touch the contact surfaces.

You will need the following tools to install the CAN 5000:

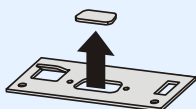
- TX10 Torx screwdriver
- Pliers
- Hexagon socket wrench, 4.5 mm

Installing the CAN 5000 in an FDS 5000

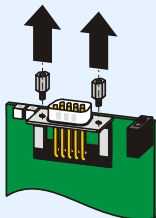
1. Unscrew the mounting screws and remove the cover plate:



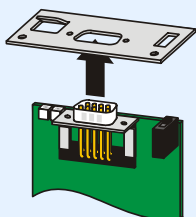
2. Remove the punched out metal section with the pliers:



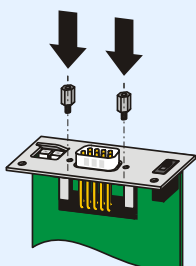
3. Remove the screws from the option PCB:



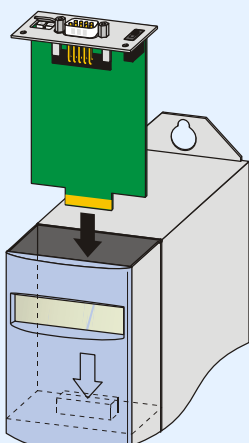
4. Stick the sub D plug connector of the PCB through the plate from below:



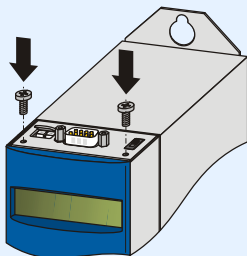
5. Secure the PCB to the plate with the screws from step 3:



6. Stick the option PCB in the inverter so that the gold contacts slide into the black terminal block:



7. Secure the plate to the inverter with the mounting screws:



⇒ You have installed the accessory.

4 Electrical installation

4.1 Setup

To set up a CAN bus, all stations (nodes) must be connected together via the lines CAN_Low and CAN_High. This means that every device which is not at the end of the bus has an incoming and an outgoing bus cable. For the device at the end of the CAN bus to which only one bus cable is connected, a terminal resistance of 124 Ohm must be connected between CAN_Low and CAN_High instead of the second bus cable..

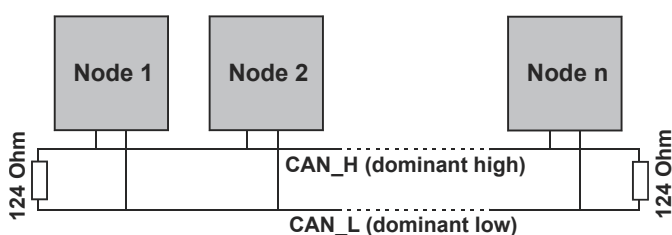


Fig. 4-1 Organization of a CAN bus system


All stations on the CAN bus must be set to the same transmission speed. This is done with parameter *A82 CAN baudrate*. The maximum line length depends on the baud rate being used. The following table shows the maximum permissible lengths over the entire extent of the bus.

| A82 CAN baudrate | Maximum Length | |
|------------------|----------------|---|
| 0:10 kbit/s | 5000 m | |
| 0:20 kbit/s | 2500 m | |
| 0:50 kbit/s | 1000 m | |
| 0:100 kbit/s | 800 m | |
| 0:125 kbit/s | 500 m | |
| 0:250 kbit/s | 250 m | |
| 0:500 kbit/s | 100 m | |
| 0:800 kbit/s | < 30 m! | Only with special bus cable with ± 60 nF/km |
| 0:1000 kbit/s | < 10 m! | |

4.2 Connection

The following table contains the allocation of the sub D plug connector for connection of a CAN bus system to the inverter.

Terminal description X200

| Pin | Designation | Function |
|---|-------------|----------|
|  plug | 1 | nc |
| | 2 | CAN-low |
| | 3 | GND |
| | 4 | nc |
| | 5 | nc |
| | 6 | CAN-low |
| | 7 | CAN-high |
| | 8 | nc |
| | 9 | CAN-high |

4.2.1 Cable specifications

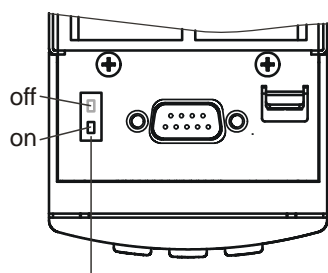
We recommend a shielded cable made especially for CAN bus communication (as per ISO 11898) since only a suitable fieldbus cable offers the necessary technical prerequisites such as, for example, ripple resistance and a sufficiently small cable capacity (approx. 60 nF/km) for correct operation particularly at high baud rates.

4.2.2 Shielding

Use of a suitable T plug connector (e.g., with screw-type terminals) makes it easy to connect the individual cores of the bus cable. The shield of the bus cable is mounted under the pull-relief of the plug connector. The shield is then connected correctly to the inverter via the plug housing and the sub D plug connector.

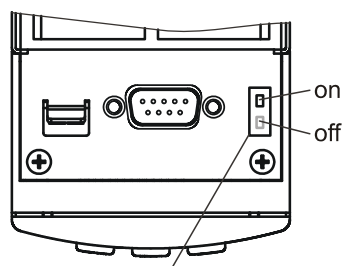
4.2.3 Terminating resistance

The terminating resistance installed on the option board makes implementation of the bus termination very easy. The bus termination is activated by moving the sliding switch in the "On" direction on the last device. Fig. 4-2 and Fig. 4-3 show the top of the inverter with an installed CAN5000 board. The sliding switch for activating the terminating resistance is also shown. Please note the differences between MDS 5000 and SDS 5000 and the FDS 5000.



Internal terminating resistance 120 Ω can be activated

Fig. 4-2 MDS 5000 and SDS 5000



internal terminating resistance 120 Ω can be activated

Fig. 4-3 FDS 5000



5 Fundamentals of CAN Bus

The serial bus system CAN (Controller Area Network) is a serial, multi-master communication protocol. It was originally developed for networking in the automobile. Since it is very fast, flexible and robust, its use in industrial applications is increasing. The actual CAN protocol corresponds to the "Data Link Layer" (layer 2) of the ISO/OSI reference model. Simple or manufacturer-specific CAN networks can be established at this level. The disadvantages of a layer-2 solution are the lack of standardized network management and the way the message IDs are distributed to all stations of the CAN network.

5.1 CANopen®

The user and manufacturer association "CAN in Automation (CiA®)" have defined as open standards the higher communication layers "CAN Application Layer (CAL)" and "CANopen®" which is based on this. These have established themselves as the industrial standard. Services and protocols for network initialization, monitoring and configuration and for process data and parameter communication are defined here.

STÖBER ANTRIEBSTECHNIK GmbH + Co. KG offers a way to link the 5th generation of STÖBER inverters to the CAN bus via the CANopen® profile CiA®/DS-301 and all CiA® specifications below (*see list of literature*).

The inverter is a logical CANopen® slave which is controlled and parameterized by a logical CANopen® master (controller, PLC). The manuals and commissioning instructions of all related components (CANopen® master/controller, additional slaves, and so on) must be adhered to when a CAN bus system is commissioned. If more information on the CAN bus or CANopen® profile is needed, extensive literature can be accessed through the CiA® organization (www.can-cia.org) or obtained from the CiA®.

5.2 NMT state machine

All CANopen® slaves offer the same routine via which the CAN interface can be initialized at power-on and controlled with commands.

This routine is shown by the Network Management state machine (see Fig. 5-1). For detailed explanations, see chapter 9.1.

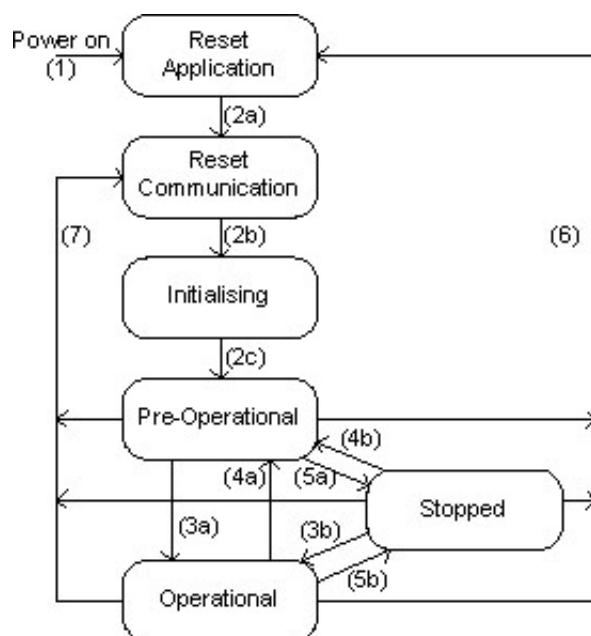


Fig. 5-1 Network management state machine

The following tables describe the states. The current state is indicated by the green LED on the option board (see chapter 6).

| State | Description | green LED |
|---------------------|---|--------------|
| Reset Application | Inverter configuration starts. Stored values are loaded. | Off |
| Reset Communication | Communication parameters are set to start values. | Off |
| Initialising | CAN bus interface is initialized. | Off |
| Pre-Operational | Inverter is ready for parameterization in preparation for actual operation. | Flashes |
| Operational | CAN bus interface is in operation with all services. | On |
| Stopped | Almost all communication activity has been stopped. | Single flash |

5.3 Parameters in the CANopen® system of the 5th generation of STÖBER inverters

Parameters in the CANopen® system have two addresses. They can be specified with the CANopen® address or with the STÖBER address. Both use the same parameters! The parameter tables in the next few chapters include both addresses.

6 User interface of the CAN 5000

The state of bus communication can be queried on the option board. The CAN5000 option board is equipped with a green and a red LED. The LEDs indicate communication-specific states as per CiA[®] DR-303-3. This makes diagnosis of the CAN state on the device simple and fast.

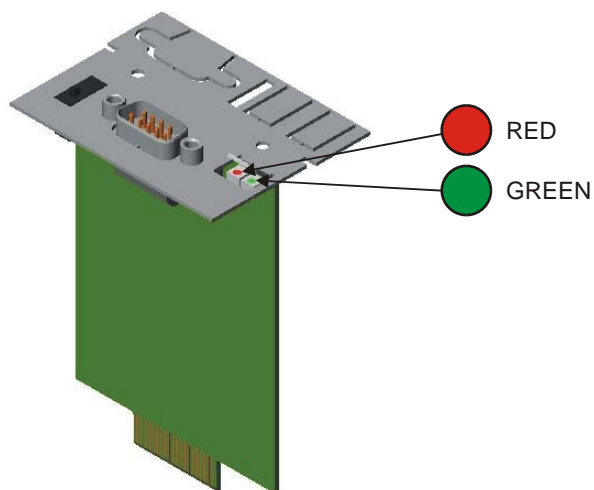


Fig. 6-1 LED arrangement

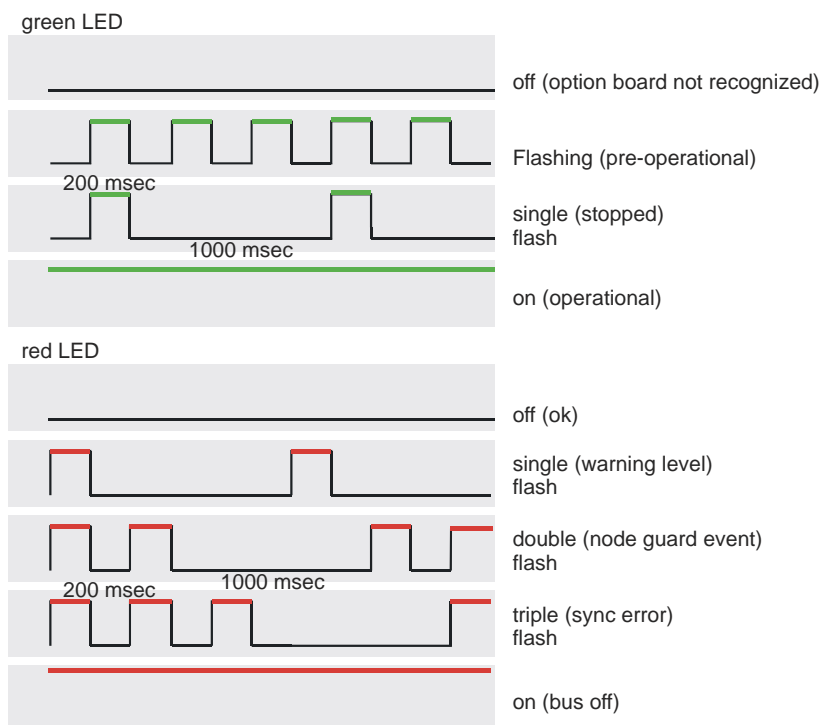


Fig. 6-2 LED response message signals



Table for the red LED:

| LED State | Meaning | Measure |
|--------------|--|--|
| Off | No error, no warning | None. Everything is okay. |
| Single flash | At least one of the error counters (Transmit or Receive) on the CAN controller has been incremented up to the warning level. | Check bus cabling and CAN bus bit timing of all stations. The warning resets itself automatically when only stations with identically set timing are communicating. |
| Double flash | Node guarding event has occurred. | Please check node guarding on the CANopen [®] master. |
| Triple flash | Sync error | A sync message has not been received in object 1006 within the set timeout time. Set timeout time correctly or check whether the sync in sync master is being sent reliably. |
| On | Bus off determined by the CAN controller. | The device is no longer participating in CAN communication. Check CAN baud rate and bus cabling, and turn off and on again. |

7 Setup of communication

The telegrams of the CAN bus consist of 0 to 8 bytes of user data which are combined into a telegram frame. The exact setup of the CAN telegram is not important for the application. This is handled by intelligent CAN controllers. (For detailed information, see the Internet site of the "CAN in Automation" association www.can-cia.de.)

7.1 Telegram Service



Information

The TIME STAMP object is not available on the 5th generation of STÖBER inverters!

Several telegram services are offered by a CANopen[®] DS301 system:

- **NMT:** NMT telegrams affect the state machines of the CANopen[®] slaves (s. chap. 9.1 "NMT State Machine"). The commands which are sent in the NMT telegram give the slaves a chance to change their states.
- **SYNC:** The SYNC object is usually used to create a time pattern. Telegrams can be received or sent in relation to the SYNC object.
- **TIME STAMP:** The TIME STAMP object supplies a precise time stamp. This function is currently not supported by the 5th generation of STÖBER inverters.
- **EMERGENCY:** Emergency telegrams are triggered by an internal device error. They are sent once when the error occurs and once when the error departs. The event is specified in the telegram.
- **PDO:** A PDO telegram supplies process data. The parameter information to be sent is defined on the receiver and the sender. Only the data of these parameters are sent in the actual telegram. Up to four PDO channels in the sending and receiving direction can be set up for each inverter.
- **SDO:** Parameters are directly addressed and queried with an SDO telegram. Each inverter has four channels in the receiving and sending direction.
- **ERROR CONTROL:** The CAN network is monitored by ERROR CONTROL objects. These include BOOT-UP, NODE and LIFE GUARDING telegrams.

NMT, SYNC and TIME STAMP are broadcast objects which are sent simultaneously by the master to all slaves. EMERGENCY, PDO, SDO and ERROR CONTROL are peer-to-peer objects which are sent by the master to a slave or vice versa.



The following table of communication services indicates which objects are active in which NMT state.

| Communication Service | Reset App, Reset Com. Initialising | Pre-Operational | Operational | Stopped |
|-------------------------------------|------------------------------------|-----------------|-------------|---------|
| PDO | – | – | Active | – |
| SDO | – | Active | Active | – |
| SYNC | – | Active | Active | – |
| Emergency | – | Active | Active | Active |
| Boot-Up | Active | – | – | – |
| Network Management (NMT, Heartbeat) | – | Active | Active | Active |

7.1.1 Identifier

Each telegram has a unique identifier so that the various objects and stations on the bus can be distinguished between. In CANopen®, the identifier is called COB-ID. In addition to the differentiation between communication services and stations, the COB-ID specifies the priority of a telegram.

There are two ways to assign the COB-ID:

1. "Pre-defined connection set": When a new device is purchased, the communication parameters are set in accordance with the principle of the "Pre-defined connection set." This offers simple commissioning of a "normal" CAN network with one NMT master and up to 127 NMT slaves (inverter and additional devices). When SDO channels 2, 3 and 4 are activated, up to 31 NMT slaves can be connected. This method is recommended for most applications!
2. "Dynamic distribution": The principle of "dynamic distribution" offers many ways to optimize a complicated CAN network with different stations and tasks.

7.2 Pre-defined Connection Set

A COB ID consists of 11 bits. The setup is described in the Figure 7 1. Bits 7 to 10 specify the "Function-Code" and bits 0 to 6 the "Node-ID". A function code is defined for each type of CAN message. With peer-to-peer objects, the Node-ID (*A83 bus address*) is added to the function code. The COB-ID is created from this for each individual message of each station on the CAN bus.

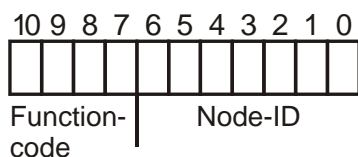


Fig. 7-1 Setup of the COB-ID

Remember: COB-ID = Function-Code + Node-ID

The following tables describe the COB-IDs of the different communication objects based on the "Pre-defined connection set". The Function-Code, the resulting COB-ID, the communication parameter and the priority of the message are specified for the object.

Table of broadcast objects

| Object | Function code (binary) | Resulting COB-ID Hexadecimal / decimal | | Index of the Assigned Communication Parameters | Priority |
|--------|------------------------|--|-----|--|----------|
| NMT | 0000 | 0 _{hex} | 0 | – | highest |
| SYNC | 0001 | 80 _{hex} | 128 | 1005 | |
| TIME | 0010 | 100 _{hex} | 256 | – | |

Table of peer-to-peer objects (here, the node ID of the affected slave is important. It must be between 1 and 31, and if the SDO channels are deactivated it must be 2, 3, 4 up to 127).

| Object | Function code (binary) | Resulting COB-ID | | | Index of the communication parameter | | Priority |
|-----------|------------------------|------------------------------|---|-------------|---|----------|----------|
| | | Formula | Hexadecimal | Decimal | CAN open [®] | Inverter | |
| Emergency | 0001 | 80 _{hex} + Node-ID | 81 _{hex} - FF _{hex} | 129 - 255 | 1014 _{hex} , 1015 _{hex} | A207 | high |
| PDO1 (tx) | 0011 | 180 _{hex} + Node-ID | 181 _{hex} - 1FF _{hex} | 385 - 511 | 1800 _{hex} | A229.0 | |
| PDO1 (rx) | 0100 | 200 _{hex} + Node-ID | 201 _{hex} - 27F _{hex} | 513 - 639 | 1400 _{hex} | A221.0 | |
| PDO2 (tx) | 0101 | 280 _{hex} + Node-ID | 281 _{hex} - 2FF _{hex} | 641 - 767 | 1801 _{hex} | A230.0 | |
| PDO2 (rx) | 0110 | 300 _{hex} + Node-ID | 301 _{hex} - 37F _{hex} | 769 - 895 | 1401 _{hex} | A222.0 | |
| PDO3 (tx) | 0111 | 380 _{hex} + Node-ID | 381 _{hex} - 3FF _{hex} | 897 - 1023 | 1802 _{hex} | A231.0 | |
| PDO3 (rx) | 1000 | 400 _{hex} + Node-ID | 401 _{hex} - 47F _{hex} | 1025 - 1151 | 1402 _{hex} | A223.0 | |

| Object | Function code (binary) | Resulting COB-ID | | | Index of the communication parameter | | Priority |
|---------------|------------------------|--|---|-------------|--|----------|----------|
| | | Formula | Hexadecimal | Decimal | CAN open® | Inverter | |
| PDO4 (tx) | 1001 | $480_{\text{hex}} + \text{Node-ID}$ | $481_{\text{hex}} - 4\text{FF}_{\text{hex}}$ | 1153 - 1279 | 1803_{hex} | A232.0 | |
| PDO4 (rx) | 1010 | $500_{\text{hex}} + \text{Node-ID}$ | $501_{\text{hex}} - 5\text{FF}_{\text{hex}}$ | 1281 - 1407 | 1403_{hex} | A224.0 | |
| SDO1 (tx) | 1011 | $580_{\text{hex}} + \text{Node-ID}$ | $581_{\text{hex}} - 59\text{F}_{\text{hex}}$ | 1409 - 1439 | 1200_{hex} | – | |
| SDO1 (rx) | 1100 | $600_{\text{hex}} + \text{Node-ID}$ | $601_{\text{hex}} - 61\text{F}_{\text{hex}}$ | 1537 - 1567 | 1200_{hex} | – | |
| SDO2 (tx) | 1011 | $5\text{A}0_{\text{hex}} + \text{Node-ID}$ | $5\text{A}1_{\text{hex}} - 5\text{BF}_{\text{hex}}$ | 1441 - 1471 | 1201_{hex} | A218.1 | |
| SDO2 (rx) | 1100 | $620_{\text{hex}} + \text{Node-ID}$ | $621_{\text{hex}} - 63\text{F}_{\text{hex}}$ | 1539 - 1599 | 1201_{hex} | A218.0 | |
| SDO3 (tx) | 1011 | $5\text{C}0_{\text{hex}} + \text{Node-ID}$ | $5\text{C}1_{\text{hex}} - 5\text{DF}_{\text{hex}}$ | 1473 - 1503 | 1202_{hex} | A219.1 | |
| SDO3 (rx) | 1100 | $640_{\text{hex}} + \text{Node-ID}$ | $641_{\text{hex}} - 65\text{F}_{\text{hex}}$ | 1601 - 1631 | 1202_{hex} | A219.0 | |
| SDO4 (tx) | 1011 | $5\text{E}0_{\text{hex}} + \text{Node-ID}$ | $5\text{E}1_{\text{hex}} - 5\text{FF}_{\text{hex}}$ | 1505 - 1535 | 1203_{hex} | A220.1 | |
| SDO4 (rx) | 1100 | $660_{\text{hex}} + \text{Node-ID}$ | $661_{\text{hex}} - 67\text{F}_{\text{hex}}$ | 1633 - 1663 | 1203_{hex} | A220.0 | |
| ERROR CONTROL | 1110 | $700_{\text{hex}} + \text{Node-ID}$ | $701_{\text{hex}} - 77\text{F}_{\text{hex}}$ | 1793 - 1919 | $1016_{\text{hex}}, 1017_{\text{hex}}$ | – | low |

The Pre-defined connection set is active when the value of the function code is entered in the communication parameters. The related Node-ID is then automatically added to the Function-Code.

Explanation of the abbreviations:

| | |
|--------------|---|
| 1-4 | This abbreviation identifies the channel. Devices of the 5 th generation of STÖBER inverters can simultaneously transmit four PDOs and four SDOs separately. |
| (rx) | This identifier specifies the objects which the inverter receives from the NMT master. |
| (tx) | This identifier specifies the objects which the inverter (NMT slave) sends to the master. |
| SDO2, -3, -4 | This abbreviation identifies the channel. Devices of the 5 th generation of STÖBER inverters can simultaneously transmit four PDOs and four SDOs separately. |

7.3 Dynamic Distribution

In special cases, it may be necessary to change the above described allocation between COB-ID and the objects. The change is possible with the following objects.:

- SYNC
- EMERGENCY
- PDO1, 2, 3 and 4 in both sending directions
- SDO2, 3 and 4 in both sending directions

The dynamic distribution principle is introduced by changing the allocated communication parameters. When these parameters do not contain the standard value (Function-Code from the above table), the addition of the Node-ID to create the resulting COB-ID is not performed. The "Pre- defined Connection Set" routine is switched off for the applicable message.

7.3.1 Procedure

The NMT master overwrites the ID when the inverter (NMT slave) is in the "Pre-operational" state. When parameters are changed and the functionality is then activated (PDOs are activated by NMT start and SDOs are reactivated by NMT reset communication, for example), the new COB IDs become valid.

When these changes are also to remain effective after the power is turned off, the action "save values" must be triggered (via STÖBER parameter *A00.0* or via CANopen® object "1010 Store parameters"). During the next device startup, the inverter checks to determine whether the values in the allocated communication parameters are standard values (Function-Code in bits 7 to 10). If this is the case, the value of the Node-ID (STÖBER parameter *A83 Bus address*) is added in accordance with the "Pre-defined Connection Set" standard principle. If no standard values are determined, the dynamic distribution principle applies.



7.3.2 Example 1

You want to activate the second SDO channel and connect fewer than 64 NMT slaves to the CAN bus.

After the first startup of the inverter with the bus address 1, only SDO channel 1 is active. SDO1 (rx) has COB ID 601_{hex} and SDO1 (tx) has COB ID 581_{hex}.

Using the SDO service, you change the communication parameter with index/subindex 1201/1 2nd server SDO parameter / COB ID client -> server to the value 80000641_{hex}, and parameter 1201/2 2nd server SDO parameter / COB ID server -> to the value 800005C1_{hex}. With a change to the NMT states *reset-*

communication or stop and then a change to the state *preoperational*, both receive and transmit directions of the SDO channel are activated. At the same time, the COB IDs are specified as 641_{hex} for rx and 5C1_{hex} for tx. With "save values," this SDO channel is immediately activated after the next power-on.

These identifiers would be used as standard by another station with the node ID 65 (41_{hex}) for SDO channel 1. This must be prevented by ensuring that no station receives a node ID of 65 or higher.

7.3.3 Example 2

You want to give the PDO (tx) objects of the three PDO channels of station 1 a higher priority than the PDO objects of other stations. Make the following parameter settings:

- Parameter 1800/1_{hex} 1st transmit PDO parameter / COB-ID set the identifier of PDO1 (tx) to the value F0_{hex} and
- Parameter 1801/1_{hex} 1st transmit PDO parameter / COB-ID set the identifier of PDO2 (tx) to the value F1_{hex} and
- Parameter 1802/1_{hex} 1st transmit PDO parameter / COB-ID set the identifier of PDO3 (tx) to the value F2_{hex}.

To prevent the COB-IDs from colliding with EMERGENCY signals of other stations, no station may use the Node-ID 112, 113 or 114 with the Pre-defined connection set.

7.3.4 Example 3

Each received CAN message adds to the computing load of the station. When a system frequently sends the SYNC broadcast message in a CAN network and some of the stations do not need this, receipt and processing for these stations can be disabled. Set the communication parameter 1005_{hex} COB-ID SYNC to a value which is not one of the identifiers used in the network. The value 7FF_{hex} is recommended since it is not used with CANopen®.

8 Data transmission with PDO and SDO

Since the exchange of user data is the main task of a CAN system, we will start by describing the communication services PDO and SDO. The objects NMT, SYNC, EMERGENCY and ERROR CONTROL are discussed in chapter 7.1.

All parameters of the inverter can be read or changed by the SDP service (SDO = Service Data Object) in the parameter channel. The desired parameter (communication object) can be addressed by index and subindex within an SDO telegram.

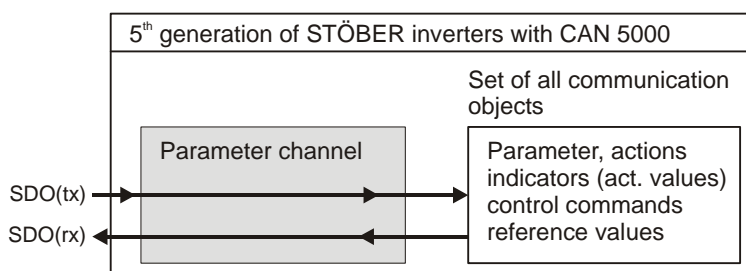


Fig. 8-1 Communication via SDO channel

A PDO telegram transmits data which are used to control and monitor the running process and which require short transmission times. The contents of the previously selected parameters are sent directly and no objects are addressed in the telegram.

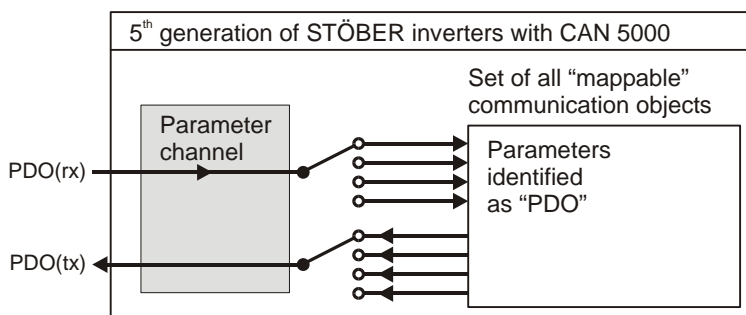


Fig. 8-2 Communication via PDO channel



8.1 Process data transmission with PDO

Process data are transferred with PDO telegrams. As seen by the inverter, each PDO channel has a receiving direction (rx) and a sending direction (tx).

PDO channels can be run at the same time. For example, when the axis switch POSISwitch® AX 5000 is used, one channel each can be used for one axis. This permits operation of four mechanical axes in succession on one inverter.

Allocation of one PDO channel to one axis makes handling easy and logical.

8.1.1 Process data mapping

Process data mapping specifies which parameters (communication objects) will be transmitted via the process data channel (PDO service). Devices of the 5th generation of STÖBER inverters support flexible mapping of the communication objects on the PDO channels. This routine is called PDO mapping.



Information

The total length of the mapped parameters may not exceed 8 bytes per channel! Make sure that the total length does not contain more than 8 bytes.

One parameter with six sub-elements exists for each PDO channel for each sending direction. The addresses of the parameters whose contents are transferred via the PDO channel are entered in the sub-elements. Depending on the number and size of the entered objects, the inverter expects a corresponding number of bytes in the PDO telegram. If more bytes are received, the excess data are ignored. If too few bytes arrive, the target objects which were not completely written remain unchanged.

There are two ways to set the mapping:

- In a CANopen® system with an SDO telegram
- In the parameter list with the POSITool software

With both methods, the mapping parameter or one of its sub-elements is addressed. It is addressed in the SDO telegram with the CANopen® address or, with POSITool, with the STÖBER address.

A sub-element of the mapping parameter has a length of four bytes. It is described by the SDO telegram with index, subindex and, optionally, the bit length of the parameter to be mapped. The bit length of the mapped parameter must not be specified when sent to the inverter. During read-accesses, it is supplied from the inverter.

The example below shows the selection of the 2808hex / 0 parameter (parameter *E08 n-Motor* with 16 bits of data) as it is transmitted in an SDO telegram. If the parameter is to be mapped first on the 1st PDO (rx) channel, the telegram must be addressed in the parameter with index 1600hex and subindex 1.



| LSB 1 st byte | 2 nd byte | MSW | |
|-----------------------------|----------------------|-----------------------------|-----------------------------|
| Length | Subindex | 3 rd byte LSB | 4 th byte MSB |
| In bits | | Index | |

| Length | Subindex | Index | |
|-------------------|-------------------|-------------------|-------------------|
| 10 _{hex} | 00 _{hex} | 08 _{hex} | 28 _{hex} |

With the software POSiTool, mapping is set in the parameter list. The parameter belonging to the particular channel must be called in the parameter list of the global area (in our screen: A225). The coordinates of the parameters which are to be mapped on the 1st PDO (rx) telegram are entered in parameter elements 0 to 5. The figure below shows the parameter A180 as the first mapped parameter.

| Coordinates | Label | Value | Default | Min | Max | Unit | Data type | Status | r/w Access level |
|-------------|------------------------|-------|---------|-----|--------|------|-------------|--------|------------------|
| A225 | 1. rec. PDO Mapping Rx | 0 | 0 | 0 | 0 | | PDO-Mapping | | 3 / 3 |
| A225.0 | 1. mapped Parameter | A180 | A180 | | 1.A0.0 | | U32 | | 1 / 1 |
| A225.1 | 2. mapped Parameter | D230 | D230 | | 1.A0.0 | | U32 | | 1 / 1 |
| A225.2 | 3. mapped Parameter | C230 | C230 | | 1.A0.0 | | U32 | | 1 / 1 |
| A225.3 | 4. mapped Parameter | | | | 1.A0.0 | | U32 | | 1 / 1 |
| A225.4 | 5. mapped Parameter | | | | 1.A0.0 | | U32 | | 1 / 1 |
| A225.5 | 6. mapped Parameter | | | | 1.A0.0 | | U32 | | 1 / 1 |

Fig. 8-3 Mapping in the parameter list (POSiTool)

When parameters from one axis are mapped, the prefix must be specified:

2.C230 (torque limit, parameter of the second axis)

The total length of the mapped parameters may not exceed 8 bytes! The mapping parameters of the channels and their sub-elements are specified in chapter 7.3 Dynamic Distribution.

The PDO mapping cannot be used for all parameters. The mappable parameters are identified in the parameter list by the abbreviation "PDO" for fieldbus notes.



8.1.2 Transmission parameters

Transmission of PDO telegrams is defined by the additional parameters listed below.

- Inhibit time (only for PDO (tx))
- Event timer (only for PDO (tx))
- Transmission type

The inhibit time only applies to sending channels (as seen by the inverter). It specifies the minimum time which must pass between two telegrams on the channel. It is defined separately for each sending channel. Its unit is specified in 0.1 ms.

When a transmission type is set for a PDO (tx) channel which permits independent triggering by SYNC, the event timer can be used. When the set time in ms has expired, the telegram is triggered. The event timer only applies to asynchronous and asynchronous-cyclic transmission types (see below). One event timer exists for each PDO (tx) channel. The timer is deactivated when the entered value is zero.

The transmission type specifies the relationship of a PDO telegram to the SYNC object. There are three types:

- Asynchronous: Asynchronous PDO telegrams have no relationship to the SYNC object. A PDO (rx) is accepted immediately after receipt. A PDO (tx) is sent immediately after it is triggered. The event timer or a sending request triggers this.
- Synchronous: Synchronous PDO telegrams are processed with a time relationship to the SYNC object. The transmission type is used to set the number of SYNCs after which processing takes place. For instance, a PDO (rx) can be accepted for every fifth SYNC.
- Acyclic synchronous: Acyclic synchronous PDO telegrams have a conditional relationship to SYNC. The following applies to PDO (rx) telegrams: they are received without a relationship to SYNC but are not processed until after the next SYNC. Triggering a PDO (tx) telegram is event-controlled but it is not transmitted until the next SYNC.

The following figures illustrate this relationship.

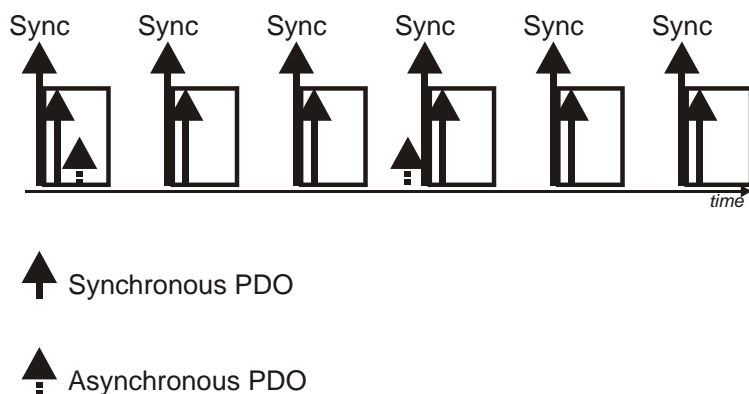
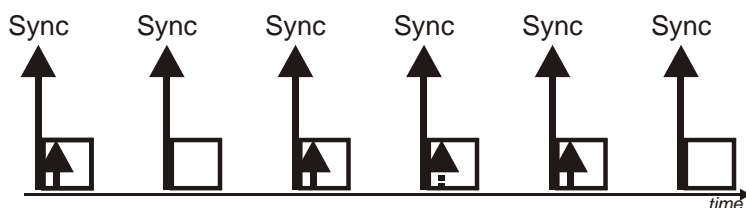


Fig. 8-4 Synchronous and asynchronous PDO



↑ Synchronous cyclic PDO

▲ Synchronous acyclic PDO

Fig. 8-5 Synchronous cyclic PDO and synchronous acyclic PDO

The type of transmission is specified by the entry of a number in the transmission type parameter. Each channel and each sending direction has a transmission type parameter. For the numbers which must be entered, see the following table:

| Transmission type | Cyclic | Acyclic | Synchronous | Asynchronous | RTR | Description |
|-------------------|--------|---------|-------------|--------------|-----|--|
| 0 | – | yes | yes | – | – | No cyclic transmission. Event-controlled but with relationship to SYNC. Receipt: Regular PDO-rx are not expected. If one has been received, it is accepted after arrival of the next SYNC. Sending: When the event is waiting to be sent, it waits until the next SYNC and then the event is sent. |
| 1 - 240 | yes | – | yes | – | – | Cyclic transmission of the PDOs with time relationship to SYNC. The number value specifies the number of transferred SYNCs for which receipt is expected and accepted for PDO-rx and for which a PDO-tx is sent. |
| 241 - 251 | | | | | | Reserved. Not supported! |
| 252 - 253 | – | – | X | X | yes | Currently not supported. |
| 254 | – | – | – | yes | – | Event-controlled sending of PDO without any relationship to SYNC. The inverter accepts every PDO-rx immediately when it is received and sends PDO-tx. |
| 255 | – | – | – | yes | – | Reserved by device profile. Not supported! |

Normal examples of transmission types



| No. | Parameter setting | Description |
|-----|--|--|
| 1. | 1400 _{hex} / 2 1. Rec PDO transmission type = 11800 _{hex} / 2 1. Tra PDO transmission type = 1 | After each received SYNC, the inverter accepts a PDO telegram with reference values and sends a PDO telegram with actual values of PDO channel 1. |
| 2. | 1400 _{hex} / 2 1. Rec PDO transmission type = 2541800 _{hex} / 2 1. Tra PDO transmission type = 254 | The inverter accepts every received PDO telegram with reference values immediately and sends in reply a PDO telegram with actual values (also PDO channel 1 here). |

8.1.3 Parameters of the PDO channels

This chapter covers the parameters for each PDO channel. Both sending directions are summarized in a table. The user can quickly find important parameters with the descriptions. This includes the CANopen[®] and STÖBER addresses of the parameters and the default values. The parameters are also listed in the parameter list at the end of this manual.

PDO channel 1

| CANopen [®] parameter with index / subindex | STÖBER parameter | Description | Default value (hex) |
|---|-------------------|---|---------------------|
| 1400 _{hex} / 1 1. rec.PDO para / COB-ID | A221.0 | Specifies the CAN identifier (COB-ID) of the (rx) channel. | 200 ^{a)} |
| 1400 _{hex} / 2 1. rec.PDO para / trans. type | A221.1 | Specifies the transmission type. | FE |
| 1600 _{hex} / 1 1. rec.PDO Mapping Rx / 1. mapped object ... 1600 _{hex} / 6 1. rec.PDO Mapping Rx / 6. mapped object | A225.0 ... A225.5 | Up to 6 parameters to which the received CAN data are written are selected here. | 0 ... 0 |
| <i>Not available with CANopen![®]</i> | A237 | Display parameter which indicates in bytes the size of the expected receive telegram of the 1st PDO channel for the current parameterization. | |
| 1800 _{hex} / 1 1. tra.PDO para / COB-ID | A229.0 | Specifies the CAN identifier (COB-ID) of the (tx) channel. | 180 ^{a)} |
| 1800 _{hex} / 2 1. tra.PDO para / trans. type | A229.1 | Specifies the transmission type. | FE |
| 1800 _{hex} / 3 1. tra.PDO para / inhibit time | A229.2 | Specifies the minimum time in 0.1 ms between 2 sending telegrams. | 0 |

Data transmission with PDO and SDO

Operation manual



| CANopen [®] parameter with index / subindex | STÖBER parameter | Description | Default value (hex) |
|---|---------------------|---|---------------------|
| 1800 _{hex} / 5 1. tra.PDO para / event timer | A229.3 | When inverter sending is event-controlled, this time applies in ms after which a message is sent cyclically. If 0 is set here, the timer is not active. | 0 |
| 1A00 _{hex} / 1 1. trans. PDO Mapping Tx / 1. mapped object ... 1A00 _{hex} / 6 1. trans. PDO Mapping Tx / 6. mapped object | A233.0 ...A233.5 | Up to 6 parameters from which the CAN data to be sent are copied are selected here. | 0 ... 0 |
| <i>Not available with CANopen![®]</i> | A241 | Display parameter which indicates in bytes the size of the expected receive telegram of the 1st PDO channel for the current parameterization. | |

a) The predefined connection set is then active when the value of the function code is entered in the communication parameters. The associated node ID is then automatically added to the function code.

PDO channel 2

| CANopen [®] parameter with index / subindex | STÖBER Parameter | Description | Default value (hex) |
|---|---------------------|---|---------------------|
| 1401 _{hex} / 1 2. rec.PDO para / COB-ID | A222.0 | Specifies the CAN identifier (COB-ID) of the (rx) channel. | 200 ^{a)} |
| 1401 _{hex} / 2 2. rec.PDO para / trans. type | A222.1 | Specifies the transmission type. | FE |
| 1601 _{hex} / 1 2. rec.PDO Mapping Rx / 1. mapped object ... 1601 _{hex} / 6 2. rec.PDO Mapping Rx / 6. mapped object | A226.0 ...A226.5 | Up to 6 parameters to which the received CAN data are written are selected here. | 0 ... 0 |
| <i>Not available with CANopen![®]</i> | A238 | Display parameter which indicates in bytes the size of the expected receive telegram of the 2nd PDO channel for the current parameterization. | |
| 1801 _{hex} / 1 2. tra.PDO para / COB-ID | A230.0 | Specifies the CAN identifier (COB-ID) of the (tx) channel. | 180 ^{a)} |
| 1801 _{hex} / 2 2. tra.PDO para / trans. type | A230.1 | Specifies the transmission type. | FE |
| 1801 _{hex} / 3 2. tra.PDO para / inhibit time | A230.2 | Specifies the minimum time in 0.1 ms between 2 sending telegrams. | 0 |
| 1801 _{hex} / 5 2. tra.PDO para / event timer | A230.3 | When inverter sending is event-controlled, this time applies in ms after which a message is sent cyclically. If 0 is set here, the timer is not active. | 0 |

WE KEEP THINGS MOVING



| CANopen [®] parameter with index / subindex | STÖBER Parameter | Description | Default value (hex) |
|---|---------------------|---|---------------------|
| 1A01 _{hex} / 1 2. trans. PDO Mapping Tx / 1. mapped object ... | A234.0 ...A234.5 | Up to 6 parameters from which the CAN data to be sent are copied are selected here. | 0 ... 0 |
| <i>Not available with CANopen![®]</i> | A242 | Display parameter which indicates in bytes the size of the expected receive telegram of the 2nd PDO channel for the current parameterization. | |

a) The predefined connection set is then active when the value of the function code is entered in the communication parameters. The associated node ID is then automatically added to the function code.

PDO channel 3

| CANopen [®] parameter with index / subindex | STÖBER parameter | Description | Default value (hex) |
|---|---------------------|---|---------------------|
| 1402 _{hex} / 1 3. rec.PDO para / COB-ID | A223.0 | Specifies the CAN identifier (COB-ID) of the (rx) channel. | 200 ^{a)} |
| 1402 _{hex} / 2 3. rec.PDO para / trans. type | A223.1 | Specifies the transmission type. | FE |
| 1602 _{hex} / 1 3. rec.PDO Mapping Rx / 1. mapped object ... 1602 _{hex} / 6 3. rec.PDO Mapping Rx / 6. mapped object | A227.0 ...A227.5 | Up to 6 parameters to which the received CAN data are written are selected here. | 0 ... 0 |
| <i>Not available with CANopen![®]</i> | A239 | Display parameter which indicates in bytes the size of the expected receive telegram of the 3rd PDO channel for the current parameterization. | |
| 1802 _{hex} / 1 3. tra.PDO para / COB-ID | A231.0 | Specifies the CAN identifier (COB-ID) of the (tx) channel. | 180 ^{a)} |
| 1802 _{hex} / 2 3. tra.PDO para / trans. type | A231.1 | Specifies the transmission type. | FE |
| 1802 _{hex} / 3 3. tra.PDO para / inhibit time | A231.2 | Specifies the minimum time in 0.1 ms between 2 sending telegrams. | 0 |
| 1802 _{hex} / 5 3. tra.PDO para / event timer | A231.3 | When inverter sending is event-controlled, this time applies in ms after which a message is sent cyclically. If 0 is set here, the timer is not active. | 0 |
| 1A02 _{hex} / 1 3. trans. PDO Mapping Tx / 1. mapped object ... 1A02 _{hex} / 6 3. trans. PDO Mapping Tx / 6. mapped object | A235.0 ...A235.5 | Up to 6 parameters from which the CAN data to be sent are copied are selected here. | 0 ... 0 |

Data transmission with PDO and SDO

Operation manual



| CANopen [®] parameter with index / subindex | STÖBER parameter | Description | Default value (hex) |
|--|------------------|---|---------------------|
| <i>Not available with CANopen![®]</i> | A243 | Display parameter which indicates in bytes the size of the expected receive telegram of the 3rd PDO channel for the current parameterization. | |

a) The predefined connection set is then active when the value of the function code is entered in the communication parameters. The associated node ID is then automatically added to the function code.

PDO channel 4

| CANopen [®] parameter with index / subindex | STÖBER parameter | Description | Default value (hex) |
|---|---------------------|---|---------------------|
| 1403 _{hex} / 1 4. rec.PDO para / COB-ID | A224.0 | Specifies the CAN identifier (COB-ID) of the (rx) channel. | 200 ^{a)} |
| 1403 _{hex} / 2 4. rec.PDO para / trans. type | A224.1 | Specifies the transmission type. | FE |
| 1603 _{hex} / 1 4. rec.PDO Mapping Rx / 1. mapped object ... 1603 _{hex} / 6 4. rec.PDO Mapping Rx / 6. mapped object | A228.0 ...A228.5 | Up to 6 parameters to which the received CAN data are written are selected here. | 0 ... 0 |
| <i>Not available with CANopen![®]</i> | A240 | Display parameter which indicates in bytes the size of the expected receive telegram of the 4th PDO channel for the current parameterization. | |
| 1803 _{hex} / 1 4. tra.PDO para / COB-ID | A232.0 | Specifies the CAN identifier (COB-ID) of the (tx) channel. | 180 ^{a)} |
| 1803 _{hex} / 2 4. tra.PDO para / trans. type | A232.1 | Specifies the transmission type. | FE |
| 1803 _{hex} / 3 4. tra.PDO para / inhibit time | A232.2 | Specifies the minimum time in 0.1 ms between 2 sending telegrams. | 0 |
| 1803 _{hex} / 5 4. tra.PDO para / event timer | A232.3 | When inverter sending is event-controlled, this time applies in ms after which a message is sent cyclically. If 0 is set here, the timer is not active. | 0 |
| 1A03 _{hex} / 1 4. trans. PDO Mapping Tx / 1. mapped object ... 1A03 _{hex} / 6 4. trans. PDO Mapping Tx / 6. mapped object | A236.0 ...A236.5 | Up to 6 parameters from which the CAN data to be sent are copied are selected here. | 0 ... 0 |
| <i>Not available with CANopen![®]</i> | A244 | Display parameter which indicates in bytes the size of the expected receive telegram of the 4th PDO channel for the current parameterization. | |

a) The predefined connection set is then active when the value of the function code is entered in the communication parameters. The associated node ID is then automatically added to the function code.

WE KEEP THINGS MOVING

**Examples:**

Parameterization of the 1st PDO-tx so that the inverter with node ID = 1 automatically sends once every 100 ms.

| Step | CAN object | Contents: index / subindex, name | Explanation |
|------|------------|--|---|
| 1. | Send SDO | 1800 _{hex} / 1 1.tra.PDO para / set COB-IDCAN telegram looks like this:601 _{hex} / 23 _{hex} 00 _{hex} 18 _{hex} 01 _{hex} 80 _{hex} 01 _{hex} 00 _{hex} 00 _{hex} | Enter standard value 180 _{hex} . Then the inverter counts the bus address COB-ID = 180 _{hex} . |
| 2. | Send SDO | 1800 _{hex} / 2 A229.1 1.tra.PDO para / transmission typeCAN telegram looks like this:601 _{hex} / 23 _{hex} 00 _{hex} 18 _{hex} 02 _{hex} FE _{hex} 00 _{hex} 00 _{hex} 00 _{hex} | Set event-oriented sending to value 254. The inverter then needs no SYNC. |
| 3. | Send SDO | 1800 _{hex} / 5 A229.3 1.tra.PDO para / event timerCAN telegram look like this:601 _{hex} / 23 _{hex} 00 _{hex} 18 _{hex} 03 _{hex} 64 _{hex} 00 _{hex} 00 _{hex} 00 _{hex} | Event timer should trigger every 100 ms. |
| 4. | Send NMT | NMT command start nodeCAN telegram looks like this:000 _{hex} 01 _{hex} 00 _{hex} | PDO is now initialized and started. The inverter now automatically begins with the sending of PDO-tx on the COB-ID 181 _{hex} . |

8.2 Parameter transmission with SDO

The SDO service permits all parameters of the inverter to be read and written. The client (usually the CANopen[®] master) starts a job with an SDO (rx) message. It selects a communication object (parameter) index and subindex in this message. On the server (inverter) the object directory with the list of all available parameters based on this size is searched and the service is processed. The server then responds with the corresponding SDP (tx) message.



8.2.1 SDO channels

Four separate SDO channels are simultaneously available. The first SDO channel is always active. Its COB-IDs $600_{\text{hex}} + \text{Node-ID}$ and $580_{\text{hex}} + \text{Node-ID}$ cannot be changed.

The other SDO channels can be set by parameterizing to different COB-IDs or deactivated. With the devices of the 5th generation of STÖBER inverters, the default setting is "deactivated."

Since only 2 bytes for the index and only 1 byte for the subindex are available for addressing via SDO and the devices of the 5th generation of STÖBER inverters use its G5 addresses to cover the address area of 4 bytes, a suitable combination had to be introduced. Together with the POSISwitch[®], the inverter permits the operation of up to four axes. This means that exactly one SDO channel is used for each axis. Addressing of the axes is not handled via index and subindex. Instead there is an element of parameter *A11* axis to be edited for each SDO channel

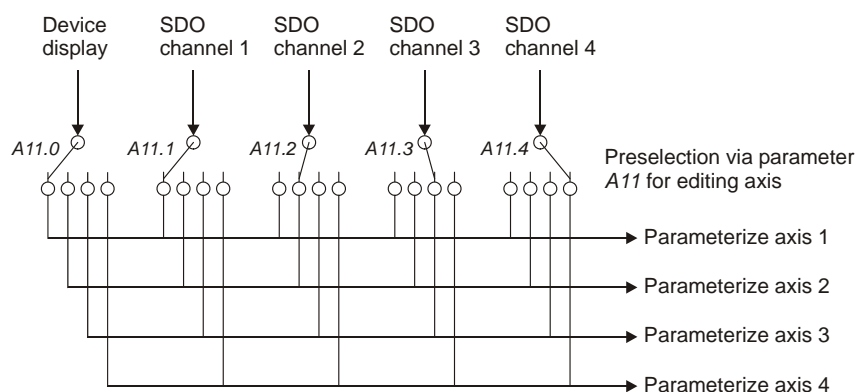


Fig. 8-6 Allocation of one SDO channel each to an axis

8.2.2 Expedited Transfer

This simplified (expedited) transmission type is used for SDO communication for all normal parameters which have a data type of up to 4 bytes. All four data bytes fit into one telegram here. The data arrangement on the bus uses the Intel format. More-significant byte/word is located at the more-significant address in memory or is sent later to the bus = Little Endian.

The entire service consists of the following telegrams:



Write a parameter

- The client (CANopen® master) sends Initiate Domain Download Request.
- The server acknowledges the request with positive response with Initiate Domain Download Response.

| 1 st Byte | 2 nd Byte | 3 rd Byte | 4 th Byte | 5 th Byte | 6 th Byte | 7 th Byte | 8 th Byte |
|----------------------|----------------------|----------------------|----------------------|----------------------|----------------------|----------------------|----------------------|
| 23 _{hex} | LSB | MSB | | LSB | MSB | LSB | BSB |
| Command | Index | | Sub-index | LSW data | | MSW data | |

| 1 st Byte | 2 nd Byte | 3 rd Byte | 4 th Byte | 5 th Byte | 6 th Byte | 7 th Byte | 8 th Byte |
|----------------------|----------------------|----------------------|----------------------|----------------------|----------------------|----------------------|----------------------|
| 63 _{hex} | LSB | MSB | | 0 | 0 | 0 | 0 |
| Command | Index | | Sub-index | unused | | | |

Read a parameter, an indication

- The client (controller) sends Initiate Domain Upload Request.
- The server acknowledges the request with positive response with Initiate Domain Upload Response.

| 1 st Byte | 2 nd Byte | 3 rd Byte | 4 th Byte | 5 th Byte | 6 th Byte | 7 th Byte | 8 th Byte |
|----------------------|----------------------|----------------------|----------------------|----------------------|----------------------|----------------------|----------------------|
| 40 _{hex} | LSB | MSB | | – | – | – | – |
| Command | Index | | Sub-index | reserved | | | |

| 1 st Byte | 2 nd Byte | 3 rd Byte | 4 th Byte | 5 th Byte | 6 th Byte | 7 th Byte | 8 th Byte |
|----------------------|----------------------|----------------------|----------------------|----------------------|----------------------|----------------------|----------------------|
| 42 _{hex} | LSB | MSB | | LSB | MSB | LSB | MSB |
| Command | Index | | Sub-index | LSW data | | MSW data | |

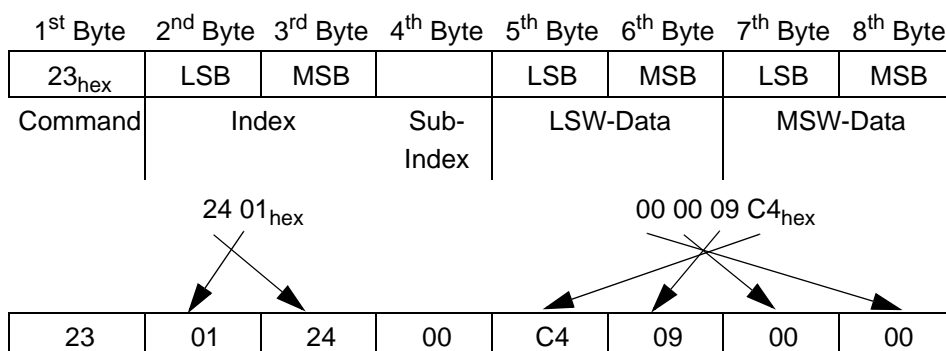
Negative response to an attempt to read or write

When an error occurs, the server responds with Upload or Download Request with Abort Domain Transfer.

| 1 st Byte | 2 nd Byte | 3 rd Byte | 4 th Byte | 5 th Byte | 6 th Byte | 7 th Byte | 8 th Byte |
|----------------------|----------------------|----------------------|----------------------|----------------------|----------------------|----------------------|----------------------|
| 80 _{hex} | LSB | MSB | | LSB | MSB | LSB | MSB |
| Command | Index | | Sub-index | Additionalcode | | Error code | Error class |

Example:

- Set parameter C01 n-Max to 2500 rpm:
- Search for index and subindex from EDS file: Index = 2401_{hex}, subindex = 0.
- Convert number value 2500 to a hexadecimal number: 09C4_{hex}
- Enter the bytes at the correct locations of Initiate Domain Download Request Service:



8.2.3 Segmented Transfer

Parameters with data types which are longer than four bytes cannot be transferred with expedited transfer. The segmented transfer is used for this. The total number of data bytes to be transferred is specified in the first initiate telegram. As many 7-data-byte telegrams as necessary follow until all bytes have been transported.

Since only a few user parameters (e.g., visible strings 1008_{hex} / 0 manufacturer device name, ...) consist of more than 4 bytes, this type of transmission is rarely needed for normal applications.

Writing of communication objects begins with the Initiate SDO Download protocol and is continued with several passes of Download SDO Segment protocol. The following figures define in detail how the telegram is set up.:

Initiate SDO Download Protocol

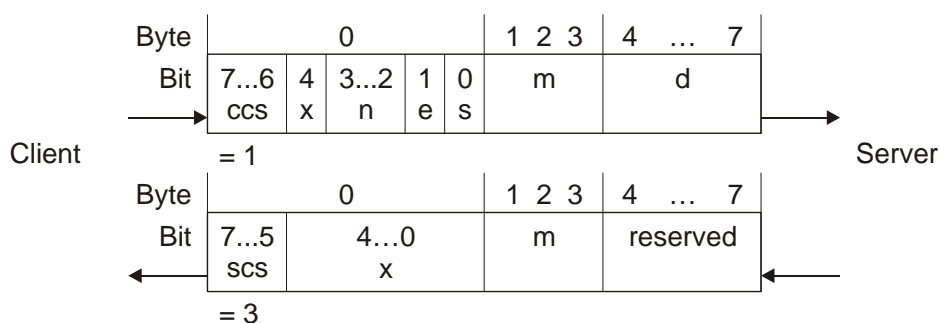


Fig. 8-7 Telegram setup "Initiate SDO Download Protocol"

| Abbrev. | Explanation | Values |
|---------|--|---|
| ccs | Client command specifier | 1 = Initiate download request |
| scs | Server command specifier | 3 = Initiate download response |
| n | Only valid when e=1 and s=1, otherwise n=0 | When valid, this value specifies the number of data bytes in d (bytes 1 to 7) which do not contain any user data. |



| Abbrev. | Explanation | Values |
|----------|----------------|---|
| e | Transfer type | 0 = Normal transfer 1 = Expedited transfer (for accelerated short transfer, see above). |
| s | Size indicator | 0 = Data record size is not indicated. 1 = Data record size is indicated. |
| m | multiplexor | Consists of index and subindex for object selection in SDO. |
| d | data | E=0, s=0: d is reserved. e=0, s=1: d contains the number of bytes to be transferred. e=1, s=1: d contains the data length 4-n. e=1, s=0: d contains a number of bytes not specified here. |
| x | unused | Reserved. Value must be 0. |
| reserved | reserved | Reserved. Value must be 0. |

Download SDO Segment Protocol

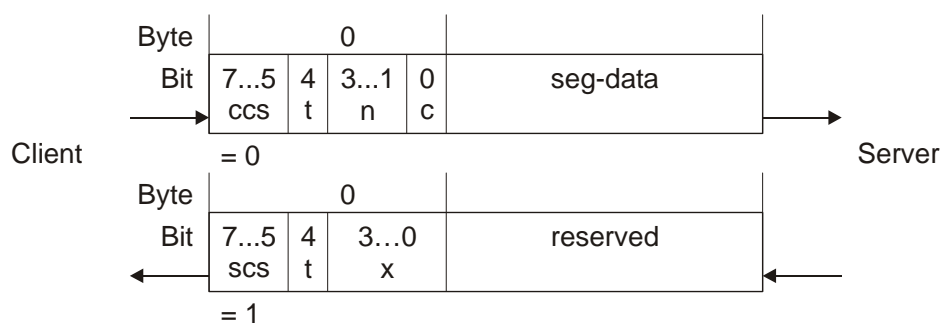


Fig. 8-8 Telegram setup "Download SDO Segment Protocol"

| Abbrev. | Explanation | Values |
|----------|--------------------------|---|
| ccs | Client command specifier | 0 = Download segment request |
| scs | Server command specifier | 1 = Download segment response |
| Seg-data | Segment data | All seven available bytes are usually filled with user data here. |
| n | Number of bytes | Number of bytes of segment data which do not contain user data: n=0: unused data not specified. |
| c | continue | Specifies whether additional segments follow. 0: More segments are to come. 1: That was the last segment. |



| Abbrev. | Explanation | Values |
|----------|-------------|---|
| t | Toggle bit | This bit must toggle for each segment. With the first segment, this bit = 0. The value in the response is identical to the that in the request. |
| x | Unused | Reserved. Value must be 0. |
| reserved | Reserved | Reserved. Value must be 0. |

Initiate SDO Upload Protocol

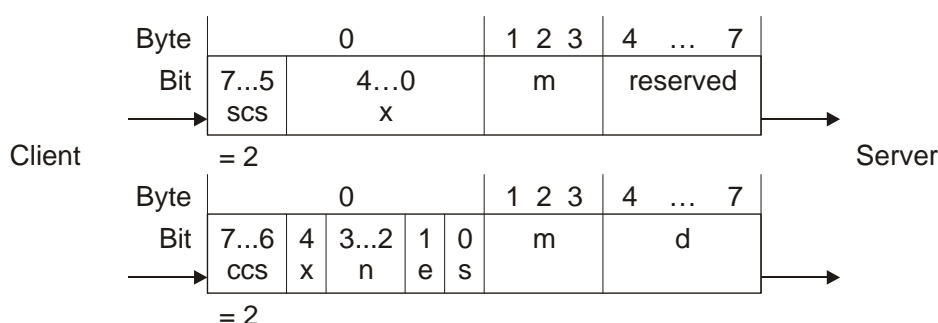


Fig. 8-9 Telegram setup "Initiate SDO Upload Protocol"

| Abbrev. | Explanation | Values |
|----------|---|--|
| ccs | Client command specifier | 2 = Initiate upload request |
| scs | Server command specifier | 2 = Initiate upload response |
| n | Nur gültig, wenn e=1 und s=1, sonst ist n=0 | When valid, this value specifies the number of data bytes in d (bytes 1 to 7) which do not contain any user data. |
| e | Transfer type | 0 = Normal transfer 1 = Expedited transfer (for accelerated short transfer, see above). |
| s | Size indicator | 0 = Data record size is not indicated. 1 = Data record size is indicated. |
| m | multiplexor | Consists of index and subindex for object selection in SDO. |
| d | data | E=0, s=0: d is reserved. e=0, s=1: d contains the number of bytes to be transferred. e=1, s=1: d contains the data length 4-n. |
| x | unused | Reserved. Value must be 0. |
| reserved | reserved | Reserved. Value must be 0. |



Upload SDO Segment Protocol

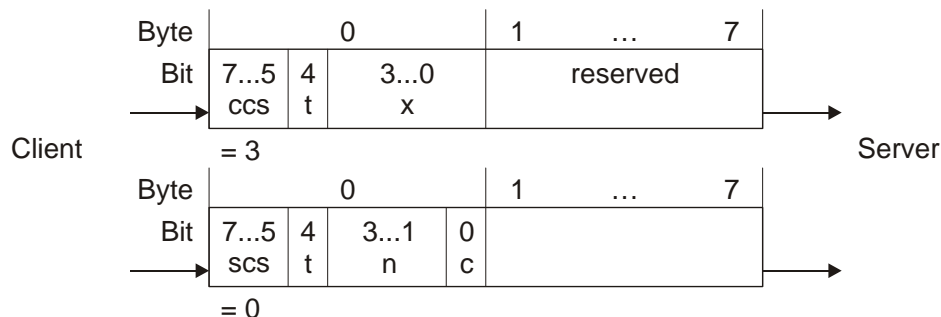


Fig. 8-10 Telegram setup "Upload SDO Segment Protocol"

| Abbrev. | Explanation | Values |
|----------|--------------------------|---|
| ccs | Client command specifier | 3 = Upload segment request |
| scs | Server command specifier | 0 = Upload segment response |
| Seg-data | Segment data | All seven available bytes are usually filled with user data here. |
| n | Number of bytes | Number of bytes of segment data which do not contain user data: n=0: unused data not specified. |
| c | continue | Specifies whether additional segments follow. 0: More segments are to come. 1: This was the last segment. |
| t | Toggle bit | This bit must be toggled for each segment. With the first segment, this bit = 0. The value in the response is identical to that in the request. |
| x | unused | Reserved. Value must be 0. |
| reserved | reserved | Reserved. Value must be 0. |

The examples below illustrate these routines:

Example 1: Segment Download with 16 bytes of data with the contents 01, 02, 03, ... 10_{hex}

| | | | | | |
|------------------|----|----------------------|---|-------------|---|
| Client: IDDRReq: | 21 | idx | x | 10 00 00 00 | (ccs=1, e=0=normal, s=1 -> data=no of bytes) |
| Server: IDDRes: | 60 | idx | x | 00 00 00 00 | |
| Client: DSegReq: | 00 | 01 02 03 04 05 06 07 | | | (ccs=0, t=0, n=0, c=0 -> all data bytes are used) |

Data transmission with PDO and SDO

Operation manual



| | | | |
|------------------|----|-------------------------|---|
| Server: DSegRes: | 20 | 00 00 00 00 00 00 00 00 | |
| Client: DSegReq: | 10 | 08 09 0A 0B 0C 0D 0E | (ccs=0, t=1, n=0, c=0 -> all data bytes are used) |
| Server: DSegRes: | 30 | 00 00 00 00 00 00 00 00 | |
| Client: DSegReq: | 0b | 0F 10 00 00 00 00 00 00 | (ccs=0, t=0, n=5, c=1 -> 5 data bytes are unused) |
| Server: DSegRes: | 20 | 00 00 00 00 00 00 00 00 | |

Example 2: Segment Upload with 16 bytes of data with the contents 01, 02, 03, .. 10_{hex}

| | | | | | |
|------------------|----|-------------------------|---|-------------|--|
| Client: IDUReq: | 40 | idx | x | 00 00 00 00 | (ccs=2, rest=0) |
| Server: IDURes: | 41 | idx | x | 10 00 00 00 | (scs=2, x=0, e=0, s=1 -> data contains no of bytes to be uploaded) |
| Client: USegReq: | 60 | 00 00 00 00 00 00 00 00 | | | (ccs=3, t=0) |
| Server: USegRes: | 00 | 01 02 03 04 05 06 07 | | | (scs=0, t=0, n=0, c=0 -> all data bytes are used) |
| Client: USegReq: | 70 | 00 00 00 00 00 00 00 00 | | | (ccs=3, t=1) |
| Server: USegRes: | 10 | 08 09 0A 0B 0C 0D 0E | | | (scs=0, t=1, n=0, c=0 -> all data bytes are used) |
| Client: USegReq: | 60 | 00 00 00 00 00 00 00 00 | | | (ccs=3, t=0) |
| Server: USegRes: | 0b | 0F 10 00 00 00 00 00 00 | | | (scs=0, t=0, n=5, c=1 -> 5 data bytes are unused) |



8.2.4 Error codes for SDO services

With the negative reply to an SDO service regardless of whether expedited or segmented, the inverter supplies in the "Abort SDO Transfer Protocol" one of the following error descriptions in the data bytes when an error occurs:

| Error Code Hexadecimal | Meaning |
|---------------------------|---|
| 0503 0000 | Toggle bit has not changed. |
| 0504 0000 | SDO protocol timeout has expired. |
| 0504 0001 | Invalid command received. |
| 0504 0005 | Not enough memory. |
| 0601 0000 | Access to object (parameter) not supported. |
| 0601 0001 | Attempt to read a write-only parameter. |
| 0601 0002 | Attempt to write a read-only parameter. |
| 0602 0000 | Object (parameter) is not listed in the object directory. |
| 0604 0041 | Object (parameter) cannot be mapped on PDO. |
| 0604 0042 | Number of length of the objects to be transferred exceeds the PDO length. |
| 0604 0043 | General parameter incompatibility. |
| 0604 0047 | General internal device incompatibility. |
| 0606 0000 | Access refused due to hardware error. |
| 0607 0010 | Wrong data type or length of the service parameter not right. |
| 0607 0012 | Wrong data type or length of the service parameter too large. |
| 0607 0013 | Wrong data type or length of the service parameter too small. |
| 0609 0011 | Subindex does not exist. |
| 0609 0030 | Invalid value of the parameter (only for write access). |
| 0609 0031 | Value of parameter too large. |
| 0609 0032 | Value of parameter too small. |
| 0609 0036 | Maximum value below minimum value. |
| 0800 0000 | General error |
| 0800 0020 | Data cannot be transferred or stored in the application. |
| 0800 0021 | Data cannot be transferred or stored in the application due to local controller. |
| 0800 0022 | Data cannot be transferred or stored in the application due to device state. |
| 0800 0023 | Dynamic generation of the object directory failed or no object directory available (does the inverter have a valid configuration?). |

9 Further communication objects

This chapter discusses the communication objects NMT, SYNC, EMERGENCY and ERROR CONTROL.

9.1 NMT

The master sends NMT telegrams as broadcast services to all slaves at the same time. NMT telegrams change the device state of the slaves. The following diagram and the table show which telegrams change the states.

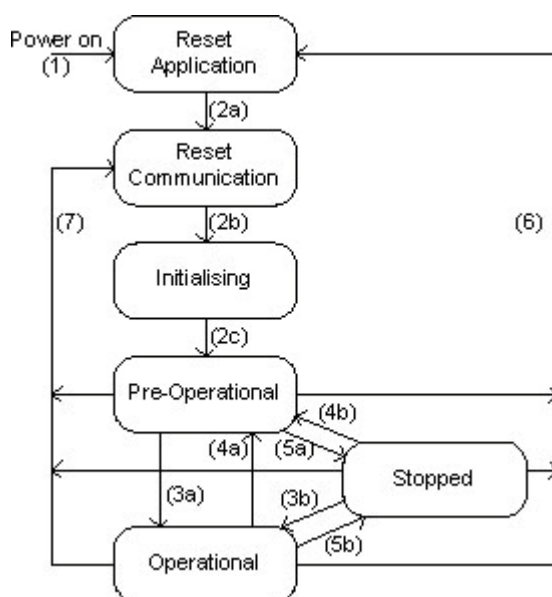


Fig. 9-1 Network management state machine

Table of state transitions

| No. | Transition | Related CAN Telegram |
|------------------|--|---|
| (1) | Turn on the power supply. | - |
| (2a), (2b), (2c) | Automatic switching after conclusion of the internal action. | - |
| (3a), (3b) | Receipt of the NMT command Start_Remote_Node. | Id = 0, 2 bytes: byte 1 = 1, byte 2 = 0 |
| (4a), (4b) | Receipt of the NMT command Enter_Pre_Operational. | Id = 0, 2 bytes: byte 1 = 128, byte 2 = 0 |
| (5a), (5b) | Receipt of the NMT command Stop_Remote_Node. | Id = 0, 2 bytes: byte 1 = 2, byte 2 = 0 |
| (6) | Receipt of the NMT command Reset_Node. | Id = 0, 2 bytes: byte 1 = 129, byte 2 = 0 |

| No. | Transition | Related CAN Telegram |
|-----|--|--|
| (7) | Receipt of the NMT command Reset_Communication. | Id = 0, 2 bytes: byte 1 = 130, byte 2 = 0 |

The following figure shows what the NMT telegram looks like with the bytes specified in the table. The command Start_Remote_Node is shown:

| COB-ID | CS | Node |
|--------|----|------|
| 0 | 1 | 0 |

The COB-ID identifies the NMT telegram. The particular command is entered in the Command Specifier (CS) byte. The Node byte defines which node is to be addressed. When the value is 0, all nodes are addressed. In addition, a single node can be addressed by entering the related node ID (*A83 bus address*).

9.2 SYNC



Information

Remember that when a fieldbus and the IGB-Motion bus are used at the same time, the fieldbus communication cannot be synchronized on the controller.

SYNC telegrams belong to the broadcast objects which are sent to all nodes simultaneously in the CANopen[®] system. Based on the "predefined connection set" principle, the SYNC identifier is set to 80_{hex}. The SYNC telegram does not contain data!



9.3 Emergency

When the service is active (bit 31 in parameter for COB-ID), the CAN bus interface of the inverter continuously monitors the device state. If the inverter changes to the "fault" state or "fault reaction active," the EMERGENCY object with one of the below error codes is sent exactly once. When the inverter leaves the "fault" state (due to acknowledgment), the EMERGENCY object with the error code "no error" is sent once. This procedure makes it possible to omit cyclic device state polling for faults by the CANopen® master. The master is automatically informed of every occurrence or departure of a fault and of the exact cause of the fault.

Within this telegram, the inverter provides information on the type of fault in three different ways:

The inverter changes after fault "temperature motor TMS":

| 1 st byte | 2 nd byte | 3 rd byte | 4 th b. | 5 th b. | 6 th b. | 7 th b. | 8 th b. |
|--|----------------------|----------------------------------|----------------------------------|----------------------|--------------------|--------------------|--------------------|
| 00 | 10 _{hex} | 01 _{hex} | 29 _{hex} | 0 | 0 | 0 | 0 |
| Emergency Error Code Temperature Drive | | Error Register Temperature | ↑ Event Temperature Motor TMS | ↑ E43 Event cause | | free | |

The inverter exits the fault:

| 1 st byte | 2 nd byte | 3 rd byte | 4 th b. | 5 th b. | 6 th b. | 7 th b. | 8 th b. |
|--|----------------------|------------------------------------|--------------------------------|--------------------|--------------------|--------------------|--------------------|
| 00 | 00 | 00 | 1E _{hex} | 0 | 0 | 0 | 0 |
| Emergency Error Code "Kein Fehler" | | Error Register "Kein Fehler" | ↑ E82 Event type = no error | | free | | |

Tab. 9-2: Setup of Emergency telegrams

The coding of error code in the first and second bytes and error register in the third byte corresponds to the specifications in the CiA/DS-301 profile and CiA DSP402. The fourth byte contains the value of the STÖBER parameter E82 Event type. The fifth byte contains the contents of E39 Event cause.

When the parameter A83 Bus address is set to 0, an EMERGENCY object is never sent since the identifier 128 which is to be used might interfere with the synchronization of the process data with the SYNC object (also identifier 128).



List of possible coding in the EMERGENCY message:

| Error Code Hex Value: Designation | Error Register Hex Value: Designation | E82 Event Code Decimal Value: Designation |
|--|--|--|
| 0 _{hex} : no error | 0: no error | 30: No event |
| 2110 _{hex} : short circuit earth | 2: current | 31: Short circuit/ground fault |
| 2230 _{hex} : intern short circuit earth | 2: current | 32: Short circuit/ground fault, internal |
| 2310 _{hex} : continous overcurrent | 2: current | 33: Overcurrent |
| 5000 _{hex} : device hardware | 1: generic error | 34: Hardware defect |
| 6010 _{hex} : software reset | 1: generic error | 35: Watchdog |
| 3110 _{hex} : mains overvoltage | 4: voltage | 36: Overvoltage |
| 7303 _{hex} : resolver 1 fault | 1: generic error | 37: n-feedback |
| 4210 _{hex} : temperature device | 8: temperature | 38: Temperature device sensor |
| 4280 _{hex} : temperature device I2t | 8: temperature | 39: TempDev.i2t |
| 6310 _{hex} : loss of parameters | 1: generic error | 40: Invalid data |
| 4310 _{hex} : temperature drive | 8: temperature | 41: Temperature motor sensor |
| 7110 _{hex} : brake chopper | 8: temperature | 42: Temperature brake resistor i2t |
| 9000 _{hex} : external error | 1: generic error | 44: External fault |
| 4380 _{hex} : temperature drive I2t | 8: temperature | 45: Excess temperature motor i2t |
| 3120 _{hex} : mains undervoltage | 4: voltage | 46: Undervoltage |
| 8311 _{hex} : excess torque | 1: generic error | 47: M-Max limit |
| 8312 _{hex} : difficult start up | 1: generic error | 48: Accelerate overload |
| 8331 _{hex} : torque fault | 1: generic error | 49: Brake overload |
| 8100 _{hex} : communication | 10: communication | 52: Communication |
| 5200 _{hex} : device hw control | 1: generic error | 55: Option board |
| 8400 _{hex} : Velocity speed control | 1: generic error | 56: Overspeed |
| 6100 _{hex} : internal software | 1: generic error | 57: Runtime load |
| 2110 _{hex} : short circuit earth | 2: current | 58: Ground fault |
| 4280 _{hex} : temperature device I2t | 8: temperature | 59: Temperature device i2t |
| 6200 _{hex} : user software | 1: generic error | 60 - 67: Application event 0 to 7 |
| 9000 _{hex} : external error | 1: generic error | 68: External fault 2 |
| 7120 _{hex} : motor | 1: generic error | 69: Motor connection |
| 6300 _{hex} : data record | 1: generic error | 70: Parameter consistency |



9.4 Error control protocols

The different protocols described below are used to monitor the CAN network and for its error detection. Devices of the 5th generation of STÖBER inverter with CAN 5000 option support all three protocols.

The following calculation is used for the identifier for all types of protocols:

$$\text{COB-ID} = 1792 + \text{Node-ID}$$

This means that there is a separate identifier starting at 1793_{hex} or 700_{hex} for each NMT slave.

9.4.1 BOOT-UP

As per CANopen[®] paper DS 301 V 4.01, the inverter sends boot-up telegram during each startup of the NMT state machine during the transition from initialization to pre-operation. This feature is always active. It is a short message on the CAN bus with one byte of data. The content of this byte is zero. This permits the NMT master to determine which stations are present after power-on.

9.4.2 Node-Guarding

Node guarding offers a protocol which permits master and slave to check each other for correct function. At regular intervals (node guard time in milliseconds) the master sends the query (request) as remote transmit telegram to each NMT slave to be monitored. Each slave recognizes the query (indication) and sends 1 byte of data as the response. Bit no. 7 of this byte has a toggle function. It must change its state each time. Bit nos. 0 to 6 contain the state of the NMT state machine. The master receives the telegram (configuration) and can then check the slave. The time of the regular telegrams is specified with the CANopen[®] parameter 100C_{hex} / 0 Guard Time. The CANopen[®] parameter 100D_{hex} / 0 Life Time Factor specifies a type of tolerance factor. If, during the time resulting from the multiplication of Guard Time and Life Time Factor, the slave fails to respond or its response is faulty this triggers the event "Node Guarding Event" on the NMT master.

In reverse, the NMT slave (inverter) monitors the regular queries of the master. If they are not received for this time period, the slave triggers the event "Life Guarding Event." For the inverter this means that the event "52:Communication" with the cause "1:Error Control" has occurred. The NMT state machine of the inverter changes to the "pre-operational" state. If one of the two parameters is zero, the service is inactive. The default values for the parameters are 0.

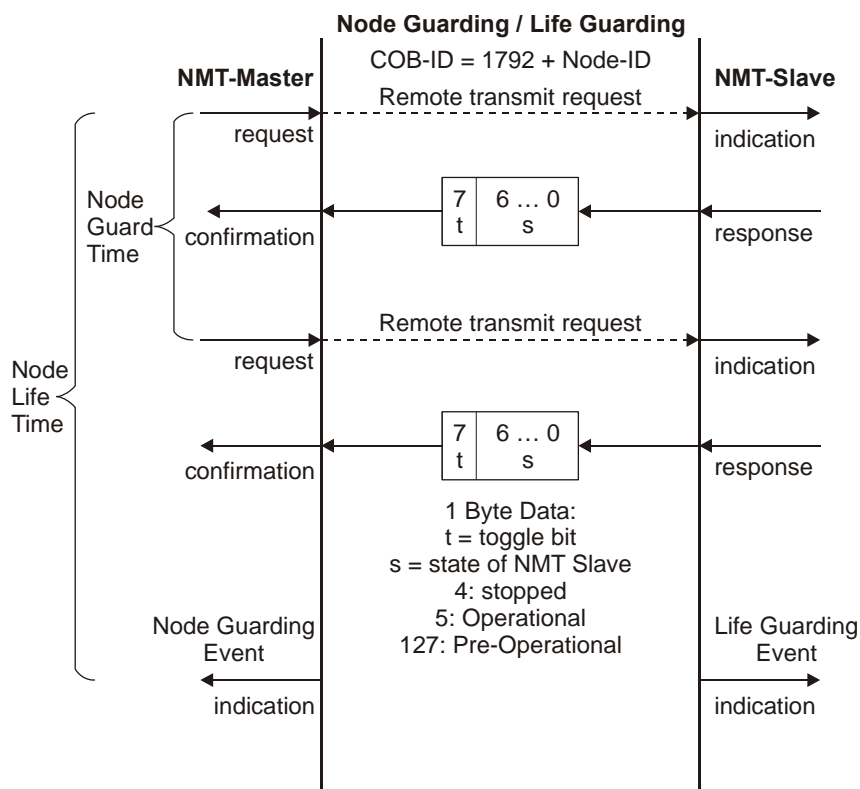


Fig. 9-3 Procedure Node Guarding

9.4.3 Heartbeat

The heartbeat protocol is another way to check the CAN network. Only Node Guard or only Heartbeat is used. The advantage of this protocol is that no request telegrams (remote frames) must be sent. This reduces the bus load of the CAN network. The heartbeat producer (usually the NMT slave) automatically regularly sends its telegrams at the time intervals entered in the CANopen® parameter 1017_{hex} Heartbeat Producer Time. It begins with this directly after the startup of the NMT state machine. If the value of this parameter is zero (as in the default setting), the service is inactive. When this is set in its CANopen® parameter Heartbeat Consumer Time, the Heartbeat Consumer expects regular telegrams from the Producer. When the telegrams are not received within the set time, it triggers the event "Heartbeat Event." The NMT master is usually set as the Heartbeat Consumer.

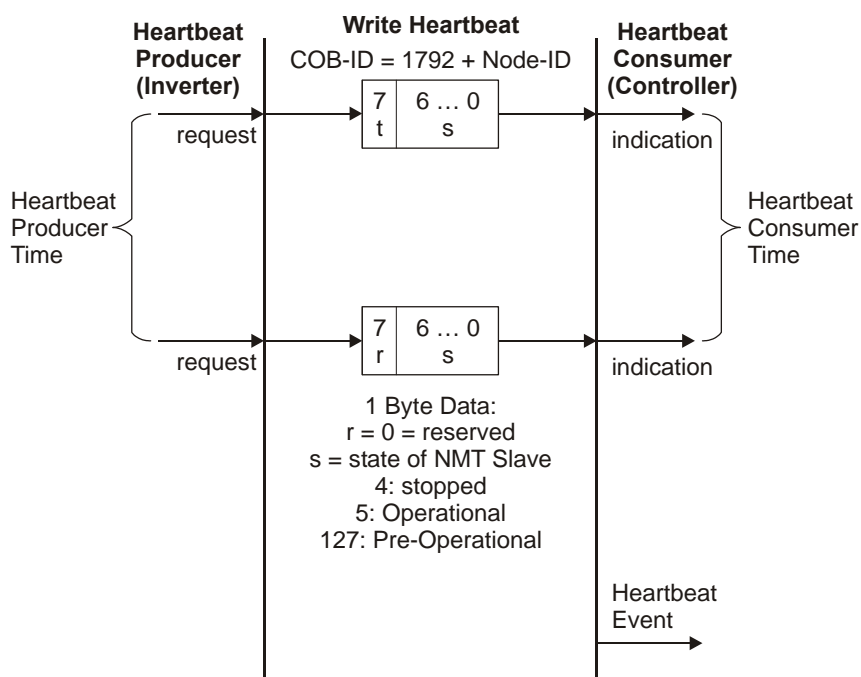


Fig. 9-4 Setup Heartbeat protocol



10 Simple commissioning

The devices of the 5th generation of STÖBER inverters with the CAN 5000 option offer numerous features for a wide variety of individual applications on the CAN bus. This chapter provides a simple introduction to this subject with step-by-step instructions for commissioning the basic functions for a popular application.

In our example, a drive is to perform quick, single-axis positioning. Process data are specified via the CAN bus with target position, positioning speed, acceleration information and a command (e.g., "absolute positioning"). Appropriate setting of the many CAN parameters makes it possible to eliminate extensive parameter assignment tasks.

Step-by-step commissioning

| Step | Task | Explanation |
|------|---|--|
| 1. | Mounting of CAN 5000 on the inverter. | See chapter 3. |
| 2. | Start new project with PC software POSITool. | This Windows software offers both easy-to-use, prompted commissioning and comprehensive access to all parameters and configurations. |
| 3. | Select type of inverter, number of axes, application and CAN 5000 option. | Processing of the configuration wizard in POSITool. In our example, the selection of the command positioning application and the CAN 5000 operation are important. |
| 4. | Check the parameters for positioning. | POSITool indicates a list of all important parameters. For adjustment of the drive to the physical conditions, the parameters <i>I07 path/motor revolution counter</i> , <i>I08 path/motor revolution denominator</i> , and <i>I10 max. speed</i> must be checked. |
| 5. | Check the 2 important parameters for CAN bus. | All CAN stations must be set to the same baud rate. At the plant, the parameter <i>A82 CAN-Baudrate</i> is set to 250 kbit/s. All CAN stations have different addresses. At the plant, the parameter <i>A83 Bus address</i> is set to 1. This permits a simple test with an inverter on a controller. |
| 6. | Download to device. | When the "Online" icon is clicked, POSITool searches on the set COM for the inverter connected via serial cable and queries whether this configuration should be downloaded to the device. After you answer with YES the download is performed and the online connection is established. After this we recommend executing the action <i>A00 Save values</i> . |
| 7. | Turn off inverter and connect CAN bus to controller with cable. | See chapter 4. Start the controller with the same baud rate as set in <i>A82</i> . Connect the voltage supply, motor and enable as described in the projecting manuals! |

| Step | Task | Explanation |
|------|----------------------------------|---|
| 8. | Turn on the inverter. | After device startup, the inverter automatically sends the boot-up telegrams on the CAN bus: identifier = 701 _{hex} . data = 1 byte with 00 _{hex} . The green LED on the CAN board of the inverter now starts flashing. If the red LED flashes or is on, the controller is not active or the bus cabling is wrong. |
| 9. | Send NMT-Start. | The controller must now send the NMT-Start telegram so that the inverter turns on its basic function of process data transmission (PDO). Sending telegram: identifier = 00 _{hex} , data: 2 bytes: byte 0 = 01 _{hex} , byte 2 = 00 _{hex} . Instead of flashing, the green LED on the CAN board of the inverter is continuously on. This means that process data can be sent. |
| 10. | Send the first Posi command. | By selecting the appropriate application with the sample circuiting in POSITool, the CAN bus process data are links with the reference values and actual values of command positioning. They can be used as shown below. Reference values are received by the inverter via the PDO1-rx telegram on identifier 201 _{hex} : Byte 0 = Corresponds to the parameterization of the process data on the inverter. Byte 1 = see byte 0 Byte 2 = see byte 0 Byte 3 = see byte 0 Byte 4 = see byte 0 Byte 5 = see byte 0 Byte 6 = see byte 0 Byte 7 = see byte 0 |
| 11. | Evaluate the response telegrams. | With the default setting of the CAN parameters, the inverter responds immediately to the execution of the command with its actual values via the PDO1-tx telegram on identifier 181 _{hex} : Byte 0 Byte 1 Byte 2 Byte 3 Byte 4 Byte 5 Byte 6 Byte 7 |

11 Parameter list

All parameters of the inverter are available as communication objects via the SDO service. Some of the parameters are described in the list in this document on the next few pages and also later in the form of an EDS file (EDS = Electronic Data Sheet).

11.1 List of the CANopen® objects

The following list of communication objects which are located in the index area between 1000_{hex} and 1FFF_{hex} have been implemented in accordance with DiA/DS-301. These objects permit every CANopen® master to read the primary features of the device on the CAN bus without having to be familiar with this documentation. Some of these objects are not in the menu of the inverter and thus have no coordinates (e.g., 1000_{hex} device type). For other communication objects there are additional manufacturer-specific parameters or indications (e.g., E51). These are entered in the "Coordinates" column.

| Index | Sub-index | G5-Adr | Name | Wertebereich | Skalierung / Einheit | r/w | Datentyp |
|---------------------|-----------|--------|---|--|----------------------|-----|------------|
| 1000 _{hex} | 0 | — | Device Type | 00020192 _{hex} | — | r | U32 |
| 1001 _{hex} | 0 | E82 | Error Register | 0, 1, 2, 4, 8 | — | r | U8 |
| 1002 _{hex} | 0 | E48 | Manufacturer Status Register | 0 – 7 | Device states | r | U32 |
| 1005 _{hex} | 0 | A200 | COB-ID SYNC Message | 1000 – 2047 | — | rw | U32 |
| 1006 _{hex} | 0 | A201 | Communication Cycle Period | 0 – 32 000 000 | — | rw | U32 |
| 1008 _{hex} | 0 | E50 | Manufac. Device Name | e.g. "MDS 5xxx" | 16 signs | r | Vis String |
| 1009 _{hex} | 0 | E149 | Manufac. Hardware Version | e.g. "MDS 5/123/3802" | 16 signs | r | Vis String |
| 100A _{hex} | 0 | E51 | Manufac. Software Version | e.g. "V 5.0 alpha 1 " | 16 signs | r | Vis String |
| 100B _{hex} | 0 | A83 | Node-Id | 0 125 | — | r | U32 |
| 100C _{hex} | 0 | A203 | Guard Time | 100C * 100D = guard time in ms | — | rw | U16 |
| 100D _{hex} | 0 | A204 | Life Time Factor | | — | rw | U8 |
| 1010 _{hex} | 0 | — | Store parameters largest subindex supported | Value =1 for only 1 additional subobject | — | r | U32 |
| 1010 _{hex} | 1 | A00.0 | store parameters.save all parameters | ASCII "save" triggers storing | Spec. SDO mechanism | rw | U32 |

| Index | Sub-index | G5-Adr | Name | Wertebereich | Skalierung / Einheit | r/w | Datentyp |
|---------------------|-----------|--------|--|---|-------------------------|-----|----------|
| 1014 _{hex} | 0 | A207 | COB-ID Emergency Object | Default: 80 _{hex} + Node-Id | — | rw | U32 |
| 1015 _{hex} | 0 | A208 | Inhibit Time EMCY | Time in 100 µs | — | rw | U32 |
| 1017 _{hex} | 0 | A210 | Producer Heartbeat Time | Time in 1 ms | — | rw | U16 |
| 1018 _{hex} | 0 | — | Identity Object: number of entries | 4 | — | r | r |
| 1018 _{hex} | 1 | — | Identity Object: Vendor ID | B9 _{hex} = no. for STÖBER Antriebstechnik GmbH | — | r | r |
| 1018 _{hex} | 2 | — | Identity Object: Product code | Rated power in 0,1 kW | — | r | r |
| 1018 _{hex} | 3 | — | Identity Object: Revision number | SW build number | — | r | r |
| 1018 _{hex} | 4 | E52 | Identity Object: Serial number | 8000000 ... | — | r | r |
| 1020 _{hex} | 0 | — | Verify Configuration.No of sup. entries | 2 | — | r | r |
| 1020 _{hex} | 1 | A211 | Verify Configuration. Configuration date | U32 | Days starts from 1.1.84 | rw | U32 |
| 1020 _{hex} | 2 | A212 | Verify Configuration. Configuration time | U32 | ms after 0:00 o'clock | rw | U32 |
| 1200 _{hex} | 0 | — | 1.Server SDO Parameter: no of elem | Constant 2 | — | r | U32 |
| 1200 _{hex} | 1 | — | 1.Server SDO Parameter: COB-ID rx | 600 _{hex} + Node-Id | Read only | r | U32 |
| 1200 _{hex} | 2 | — | 1.Server SDO Parameter: COB-ID tx | 580 _{hex} + Node-Id | Read only | r | U32 |
| 1201 _{hex} | 0 | — | 2.Server SDO Parameter: no of elem | Constant 3 | — | r | U8 |
| 1201 _{hex} | 1 | A218.0 | 2.Server SDO Parameter: COB-ID rx | 620 _{hex} + Node-Id | — | rw | U32 |
| 1201 _{hex} | 2 | A218.1 | 2.Server SDO Parameter: COB-ID tx | 5A0 _{hex} + Node-Id | — | rw | U32 |
| 1201 _{hex} | 3 | A218.2 | 2.Server SDO Para.: Client. Node-Id | 0 | — | rw | U32 |
| 1202 _{hex} | 0 | — | 3.Server SDO Parameter: no of elem | Constant 3 | — | r | U8 |

| Index | Sub-index | G5-Adr | Name | Wertebereich | Skalierung / Einheit | r/w | Datentyp |
|---------------------|-----------|--------|--------------------------------------|------------------------------|----------------------|-----|----------|
| 1202 _{hex} | 1 | A219.0 | 3.Server SDO Parameter: COB-ID rx | 640 _{hex} + Node-Id | — | rw | U32 |
| 1202 _{hex} | 2 | A219.1 | 3.Server SDO Parameter: COB-ID tx | 5C0 _{hex} + Node-Id | — | rw | U32 |
| 1202 _{hex} | 3 | A219.2 | 3.Server SDO Para.: Client. Node-Id | 0 | — | rw | U32 |
| 1203 _{hex} | 0 | — | 4.Server SDO Parameter: no of elem | Constant 3 | — | r | U8 |
| 1203 _{hex} | 1 | A220.0 | 4.Server SDO Parameter: COB-ID rx | 660 _{hex} + Node-Id | — | rw | U32 |
| 1203 _{hex} | 2 | A220.1 | 4.Server SDO Parameter: COB-ID tx | 5E0 _{hex} + Node-Id | — | rw | U32 |
| 1203 _{hex} | 3 | A220.2 | 4.Server SDO Para.: Client. Node-Id | 0 | — | rw | U32 |
| 1400 _{hex} | 0 | — | 1. rec. PDO para/largest subindex | 2 | — | r | U8 |
| 1400 _{hex} | 1 | A221.0 | 1. rec. PDO para/COB-ID | 200 _{hex} + Node-ID | — | rw | U32 |
| 1400 _{hex} | 2 | A221.1 | 1. rec. PDO para/trans. Type | 1 ... 240, 254 | — | rw | U8 |
| 1401 _{hex} | 0 | — | 2. rec. PDO para/largest subindex | 2 | — | r | U8 |
| 1401 _{hex} | 1 | A222.0 | 2. rec. PDO para/COB-ID | 300 _{hex} + Node-ID | — | rw | U32 |
| 1401 _{hex} | 2 | A222.1 | 2. rec. PDO para/trans. Type | 1 ... 240, 254 | — | rw | U8 |
| 1402 _{hex} | 0 | — | 3. rec. PDO para/largest subindex | 2 | — | r | U8 |
| 1402 _{hex} | 1 | A223.0 | 3. rec. PDO para/COB-ID | 400 _{hex} + Node-ID | — | rw | U32 |
| 1402 _{hex} | 2 | A223.1 | 3. rec. PDO para/trans. type | 1 ... 240, 254 | — | rw | U8 |
| 1403 _{hex} | 0 | — | 4. rec. PDO para/largest subindex | 2 | — | r | U8 |
| 1403 _{hex} | 1 | A224.0 | 4. rec. PDO para/COB-ID | 500 _{hex} + Node-ID | — | rw | U32 |
| 1403 _{hex} | 2 | A224.1 | 4. rec. PDO para/trans. type | 1 ... 240, 254 | — | rw | U8 |
| 1600 _{hex} | 0 | — | 1. rec. PDO Mapping Rx/no. of mapped | — | — | r | U8 |
| 1600 _{hex} | 1 | A225.0 | 1. rec. PDO Mapping Rx/1. Object | 0 (inactive) | — | rw | U32 |
| 1600 _{hex} | ... | ... | ... | ... | — | rw | U32 |

| Index | Sub-index | G5-Adr | Name | Wertebereich | Skalierung / Einheit | r/w | Datentyp |
|---------------------|-----------|--------|--------------------------------------|------------------------------|----------------------|-----|----------|
| 1600 _{hex} | 6 | A225.5 | 1. rec. PDO Mapping Rx/6. Object | 0 (inactive) | — | rw | U32 |
| 1601 _{hex} | 0 | — | 2. rec. PDO Mapping Rx/no. of mapped | — | — | r | U8 |
| 1601 _{hex} | 1 | A226.0 | 2. rec. PDO Mapping Rx/1. Object | — | — | rw | U32 |
| 1601 _{hex} | ... | ... | ... | ... | — | rw | U32 |
| 1601 _{hex} | 6 | A226.5 | 2. rec. PDO Mapping Rx/6. Object | — | — | rw | U32 |
| 1602 _{hex} | 0 | — | 3. rec. PDO Mapping Rx/no. of mapped | — | — | r | U8 |
| 1602 _{hex} | 1 | A227.0 | 3. rec. PDO Mapping Rx/1. Object | — | — | rw | U32 |
| 1602 _{hex} | ... | ... | ... | ... | — | | |
| 1602 _{hex} | 6 | A227.5 | 3. rec. PDO Mapping Rx/6. Object | — | — | rw | U32 |
| 1603 _{hex} | 1 | A228.0 | 4. rec. PDO Mapping Rx/1. Object | — | — | rw | U32 |
| 1603 _{hex} | ... | ... | ... | ... | — | | |
| 1603 _{hex} | 6 | A228.5 | 4. rec. PDO Mapping Rx/6. Object | — | — | rw | U32 |
| 1800 _{hex} | 0 | — | 1. tra. PDO para/largest subindex | 5 | — | r | U8 |
| 1800 _{hex} | 1 | A229.0 | 1. tra. PDO para/COB-ID | 180 _{hex} + Node-ID | — | rw | U32 |
| 1800 _{hex} | 2 | A229.1 | 1. tra. PDO para/trans. Type | 1 ... 240, 254 | — | rw | U8 |
| 1800 _{hex} | 3 | A229.2 | 1. tra. PDO para/inhibit time | Time in n * 100 µs | — | rw | U16 |
| 1800 _{hex} | 5 | A229.3 | 1. tra. PDO para/event timer | Time in ms | — | rw | U16 |
| 1801 _{hex} | 0 | — | 2. tra. PDO para/largest subindex | 5 | — | r | U8 |
| 1801 _{hex} | 1 | A230.0 | 2. tra. PDO para/COB-ID | 280 _{hex} + Node-ID | — | rw | U32 |
| 1801 _{hex} | 2 | A230.1 | 2. tra. PDO para/trans. Type | 1 ... 240, 254 | — | rw | U8 |
| 1801 _{hex} | 3 | A230.2 | 2. tra. PDO para/inhibit time | Time in n * 100 µs | — | rw | U16 |
| 1801 _{hex} | 5 | A230.3 | 2. tra. PDO para/event timer | Time in ms | — | rw | U16 |
| 1802 _{hex} | 0 | — | 3. tra. PDO para/largest subindex | 5 | — | r | U8 |
| 1802 _{hex} | 1 | A231.0 | 3. tra. PDO para/COB-ID | 380 _{hex} + Node-ID | — | rw | U32 |
| 1802 _{hex} | 2 | A231.1 | 3. tra. PDO para/trans. Type | 1 ... 240, 254 | — | rw | U8 |

| Index | Sub-index | G5-Adr | Name | Wertebereich | Skalierung / Einheit | r/w | Datentyp |
|---------------------|-----------|--------|--|------------------------------|----------------------|-----|----------|
| 1802 _{hex} | 3 | A231.2 | 3. tra. PDO para/inhibit time | Time in n * 100 µs | — | rw | U16 |
| 1802 _{hex} | 5 | A231.3 | 3. tra. PDO para/event timer | Time in ms | — | rw | U16 |
| 1803 _{hex} | 0 | — | 4. tra. PDO para/largest subindex | 5 | — | r | U8 |
| 1803 _{hex} | 1 | A232.0 | 4. tra. PDO para/COB-ID | 480 _{hex} + Node-ID | — | rw | U32 |
| 1803 _{hex} | 2 | A232.1 | 4. tra. PDO para/trans. Type | 1 ... 240, 254 | — | rw | U8 |
| 1803 _{hex} | 3 | A232.2 | 4. tra. PDO para/inhibit time | Time in n * 100 µs | — | rw | U16 |
| 1803 _{hex} | 5 | A232.3 | 4. tra. PDO para/event timer | Time in ms | — | rw | U16 |
| 1A00 _{hex} | 0 | — | 1. trans. PDO Mapping Tx/ no. of mapped | 6 | — | r | U8 |
| 1A00 _{hex} | 1 | A233.0 | 1. trans. PDO Mapping Tx/ 1. Object | 0 (inactive) | — | rw | U32 |
| 1A00 _{hex} | ... | ... | ... | ... | — | rw | U32 |
| 1A00 _{hex} | 6 | A233.5 | 1. trans. PDO Mapping Tx/ 6. Object | 0 (inactive) | — | rw | U32 |
| 1A01 _{hex} | 0 | — | 2. trans. PDO Mapping/no. of mapped | — | — | r | U8 |
| 1A01 _{hex} | 1 | A234.0 | 2. trans. PDO Mapping Tx/ 1. Object | — | — | rw | U32 |
| 1A01 _{hex} | ... | ... | ... | ... | — | rw | U32 |
| 1A01 _{hex} | 6 | A234.5 | 2. trans. PDO Mapping Tx/ 6. Object | — | — | rw | U32 |
| 1A02 _{hex} | 0 | — | 3. trans. PDO Mapping Tx/ no. of mapped | 6 | — | r | U8 |
| 1A02 _{hex} | 1 | A235.0 | 3. trans. PDO Mapping Tx/ 1. Object | — | — | rw | U32 |
| 1A02 _{hex} | ... | ... | ... | ... | — | rw | U32 |
| 1A02 _{hex} | 6 | A235.5 | 3. trans. PDO Mapping Tx/ 6. Object | — | — | rw | U32 |
| 1A03 _{hex} | 0 | — | 4. trans. PDO Mapping Tx/ no. of mapped | 6 | — | r | U8 |
| 1A03 _{hex} | 1 | A236.0 | 4. trans. PDO Mapping Tx/ 1. Object | — | — | rw | U32 |
| 1A03 _{hex} | ... | ... | ... | ... | — | rw | U32 |
| 1A03 _{hex} | 6 | A236.5 | 4. trans. PDO Mapping Tx/ 6. Object | — | — | rw | U32 |
| — | — | A237 | 1. rec. PDO Mapped Len | 0 ... 8 | — | r | U8 |

| Index | Sub-index | G5-Adr | Name | Wertebereich | Skalierung / Einheit | r/w | Datentyp |
|-------|-----------|--------|--------------------------|--------------|----------------------|-----|----------|
| — | — | A238 | 2. rec. PDO Mapped Len | 0 ... 8 | — | r | U8 |
| — | — | A239 | 3. rec. PDO Mapped Len | 0 ... 8 | — | r | U8 |
| — | — | A240 | 4. rec. PDO Mapped Le | 0 ... 8 | — | r | U8 |
| — | — | A241 | 1. trans. PDO Mapped Len | 0 ... 8 | — | r | U8 |
| — | — | A242 | 2. trans. PDO Mapped Len | 0 ... 8 | — | r | U8 |
| — | — | A243 | 3. trans. PDO Mapped Len | 0 ... 8 | — | r | U8 |
| — | — | A244 | 4. trans. PDO Mapped Len | 0 ... 8 | — | r | U8 |

1001 Error Register: Only one byte (the LSB) contains information:

The following values are possible:

| Hex Value | Dec. Value | Meaning / Inverter Fault |
|-------------------|------------|---|
| 00 _{hex} | 0 | Inverter is not faulty |
| 01 _{hex} | 1 | General fault |
| 02 _{hex} | 2 | Fault in connection with current: Overcurrent, short circuit/ground fault, etc. |
| 04 _{hex} | 4 | Fault in connection with voltage: Overvoltage, undervoltage, etc. |
| 08 _{hex} | 8 | Fault in connection with temperature: I ² t, motor temperature, etc. |
| 10 _{hex} | 16 | Fault in communication |
| 20 _{hex} | 32 | Reserved |
| 40 _{hex} | 64 | Reserved |
| 80 _{hex} | 128 | Reserved |



11.2 List of the drive parameters

All STÖBER-specific drive parameters are listed as communication objects in the area specified by CANopen[®] (manufacturer-specific area) from index 2000_{hex} to 5FFF_{hex}. The line number after the group are added to the start index. The subindex of a drive parameter (e.g., 0 for start, 1 for progress and 2 for result of actions) is entered in the subindex of the SDO servic.

| Area no. | Group | Start Index | End Index |
|----------|-------|---------------------|---------------------|
| 1 | A | 2000 _{hex} | 21FF _{hex} |
| 2 | B | 2200 _{hex} | 23FF _{hex} |
| 3 | C | 2400 _{hex} | 25FF _{hex} |
| 4 | D | 2600 _{hex} | 27FF _{hex} |
| 5 | E | 2800 _{hex} | 29FF _{hex} |
| 6 | F | 2A00 _{hex} | 2BFF _{hex} |
| 7 | G | 2C00 _{hex} | 2DFF _{hex} |
| 8 | H | 2E00 _{hex} | 2FFF _{hex} |
| 9 | I | 3000 _{hex} | 31FF _{hex} |
| 10 | J | 3200 _{hex} | 33FF _{hex} |
| 11 | K | 3400 _{hex} | 35FF _{hex} |
| 12 | L | 3600 _{hex} | 37FF _{hex} |
| 13 | M | 3800 _{hex} | 39FF _{hex} |
| 14 | N | 3A00 _{hex} | 3BFF _{hex} |
| 15 | O | 3C00 _{hex} | 3DFF _{hex} |
| 16 | P | 3E00 _{hex} | 3FFF _{hex} |
| 17 | Q | 4000 _{hex} | 41FF _{hex} |
| 18 | R | 4200 _{hex} | 43FF _{hex} |
| 19 | S | 4400 _{hex} | 45FF _{hex} |
| 20 | T | 4600 _{hex} | 47FF _{hex} |
| 21 | U | 4800 _{hex} | 49FF _{hex} |
| 22 | V | 4A00 _{hex} | 4BFF _{hex} |
| 23 | W | 4C00 _{hex} | 4DFF _{hex} |
| 24 | X | 4E00 _{hex} | 4FFF _{hex} |
| 25 | Y | 5000 _{hex} | 51FF _{hex} |
| 26 | Z | 5200 _{hex} | 53FF _{hex} |
| Reserved | – | 5400 _{hex} | 5FFF _{hex} |



12 List of literature

1. Projecting manuals:
 - POSIDRIVE® MDS 5000: ID 442273
 - POSIDRIVE® FDS 5000: ID 442269
 - POSIDYN® SDS 5000: ID 442277
2. Commissioning instructions:
 - POSIDRIVE® MDS 5000: ID 442297
 - POSIDRIVE® FDS 5000: ID 442293
 - POSIDYN® SDS 5000: ID 442301
3. Operating manuals:
 - POSIDRIVE® MDS 5000: ID 442285
 - POSIDRIVE® FDS 5000: ID 442281
 - POSIDYN® SDS 5000: ID 442289
4. Modules for the 5th generation of STÖBER inverters (ID 441682)
5. Programming manual for the 5th generation of STÖBER inverters (ID 441693)
6. Installation and commissioning instructions for POSIDRIVE® FAS 4000 (ID 441581)
7. Installation and commissioning instructions for POSIDRIVE® FDS 4000 (ID 441408)
8. Installation and commissioning instructions for POSIDYN® SDS 4000 (ID 441449)
9. ISO 11898, 1993 Road Vehicles, Interchange of Digital Information - Controller Area Network (CAN) for high-speed communication
10. Robert Bosch GmbH, CAN Specification 2.0 Part A+B, September 1991
11. CiA/DS-102, CAN Physical Layer for Industrial Applications, April 1994
12. CiA/DS-201, CAN in the OSI Reference Model, February 1996
13. CiA/DS-202/1, CMS Service Specification, February 1996
14. CiA/DS-202/2, CMS Protocol Specification, February 1996
15. CiA/DS-202/3, CMS Data Types and Encoding Rules, February 1996
16. CiA/DS-203/1, NMT Service Specification, February 1996
17. CiA/DS-203/2, NMT Protocol Specification, February 1996
18. CiA/DS-204/1, DBT Service Specification, February 1996
19. CiA/DS-204/2, DBT Protocol Specification, February 1996
20. CiA/DS-205/1, LMT Service Specification, February 1996
21. CiA/DS-205/2, LMT Protocol Specification, February 1996
22. CiA/DS-207, Application Layer Naming Specification, February 1996
23. CiA/DS-301, CANopen: Application Layer and Communication Profile V 4.01, June 2000
24. CiA/DSP-306, CANopen: Electronic Data Sheet Specification V1.0, 31.05.2000
25. CiA/DSP-402, CANopen: Drives and Motion Control V 2.0, August 2002
26. DRIVECOM profile drive systems no. 21, November 1991



Address registers

Always up to date on the internet: www.stober.com → contact

- Technical Offices (TB) for advice and marketing in Germany
- Global presence for advice and marketing in about 25 countries
- Service Network Germany
- Service Network International
- STÖBER Subsidiaries:

Austria

STÖBER ANTRIEBSTECHNIK GmbH
Hauptstraße 41a
4663 Laakirchen
Fon +43 7613 7600-0
Fax +43 7613 7600-2525
E-Mail: office@stoeber.at
www.stoeber.at

USA

STOBER DRIVES INC.
1781 Downing Drive
Maysville, KY 41056
Fon +1 606 7595090
Fax +1 606 7595045
eMail: sales@stober.com
www.stober.com

France

STÖBER S.a.r.l.
131, Chemin du Bac à Traille
Les Portes du Rhône
69300 Caluire et Cuire
Fon +33 4 78989180
Fax +33 4 78985901
eMail: mail@stober.fr
www.stober.fr

Switzerland

STÖBER SCHWEIZ AG
Rugghölzli 2
5453 Remetschwil
Fon +41 56 496 96 50
Fax +41 56 496 96 55
eMail: info@stoeber.ch
www.stoeber.ch

Great Britain

STOBER DRIVES LTD.
Upper Keys Business Village
Keys Park Road, Hednesford
Cannock WS12 2HA
Fon +44 1543 458 858
Fax +44 1543 448 688
E-Mail: mail@stober.co.uk
www.stober.co.uk

Italy

STÖBER TRASMISSIONI S. r. l.
Via Italo Calvino, 7
Palazzina D
20017 Rho (MI)
Fon +39 02 93909-570
Fax +39 02 93909-325
eMail: info@stoeber.it
www.stoeber.it

China

STOBER CHINA
German Centre Beijing
Unit 2010, Landmark Tower 2,
8 North Dongsanhuan Road
Chaoyang District
100004 Beijing
Fon +86 10 65907391
Fax +86 10 65907393
eMail: info@stoeber.cn
www.stoeber.cn

Japan

STOBER Japan
P.O. Box 113-002, 6 chome
15-8, Hon-komagome
Bunkyo-ku
Tokyo
Fon +81 3 5395-6788
Fax +81 3 5395-6799
eMail: mail@stober.co.jp
www.stober.co.jp

Singapore

STOBER Singapore Pte. Ltd.
50 Tagore Lane
#05-06B
Entrepreneur Centre
Singapore 787494
Fon +65 65112912
Fax +65 65112969
E-Mail: info@stober.sg
www.stober.sg

**STÖBER ANTRIEBSTECHNIK GmbH + Co. KG**

Kieselbronner Str. 12
75177 PFORZHEIM
GERMANY
Tel. +49 7231 582-0
Fax. +49 7231 582-1000
E-Mail: mail@stoeber.de

24/h service hotline +49 180 5 786 323

www.stober.com

Technische Änderungen vorbehalten
Errors and changes excepted
ID 441686.06
09/2013

