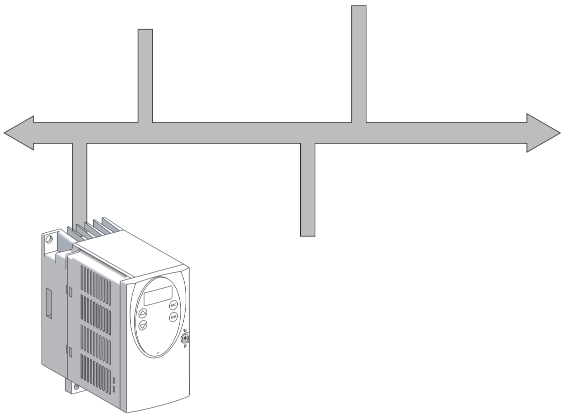
SD328A CANopen DSP402

Fieldbus interface

Fieldbus manual

V2.01, 11.2008



0198441113718, V2.01, 11.2008

BERGER LAHR

www.schneider-electric.com

Schneider Electric

Important information

This manual is part of the product.

Carefully read this manual and observe all instructions.

Keep this manual for future reference.

Hand this manual and all other pertinent product documentation over to all users of the product.

Carefully read and observe all safety instructions and the chapter "Before you begin - safety information".

0198441113718, V2.01, 11.2008

Table of Contents

0198441113718, V2.01, 11.2008

0198441113718, V2.01, 11.2008

Writing conventions and symbols

Work steps

If work steps must be performed consecutively, this sequence of steps is represented as follows:

■ Special prerequisites for the following work steps

▶ Step 1

◁ Specific response to this work step

▶ Step 2

If a response to a work step is indicated, this allows you to verify that the work step has been performed correctly.

Unless otherwise stated, the individual steps must be performed in the specified sequence.

Bulleted lists

The items in bulleted lists are sorted alphanumerically or by priority. Bulleted lists are structured as follows:

- Item 1 of bulleted list
- Item 2 of bulleted list
  - Subitem for 2
  - Subitem for 2
- Item 3 of bulleted list

Making work easier

Information on making work easier is highlighted by this symbol:

Sections highlighted this way provide supplementary information on making work easier.

SI units

SI units are the original values. Converted units are shown in brackets behind the original value; they may be rounded.

Example:
Minimum conductor cross section: 1.5 mm^2 (AWG 14)

0198441113718, V2.01, 11.2008

0198441113718, V2.01, 11.2008

1 Introduction

1.1 CAN bus

The CAN bus (**C**ontroller **A**rea **N**etwork) was originally developed for fast, economical data transmission in the automotive industry. Today, the CAN bus is also used in industrial automation technology and has been further developed for communication at fieldbus level.

Features of the CAN bus

The CAN bus is a standardized, open bus enabling communication between devices, sensors and actuators from different manufacturers. The features of the CAN bus comprise

- Multimaster capability

  Each device in the fieldbus can transmit and receive data independently without depending on an "ordering" master functionality.

- Message-oriented communication

  Devices can be integrated into a running network without reconfiguration of the entire system. The address of a new device does not need to be specified on the network.

- Prioritization of messages

  Messages with higher priority are sent first for time-critical applications.

- Residual error probability

  Various security features in the network reduce the probability of undetected incorrect data transmission to less than 10^{-11}.

Transmission technology

In the CAN bus, multiple devices are connected via a bus cable. Each network device can transmit and receive messages. Data between network devices are transmitted serially.

Network devices

Examples of CAN bus devices are

- Automation devices, e.g. PLCs
- PCs
- Input/output modules
- Drives
- Analysis devices
- Sensors and actuators

0198441113718, V2.01, 11.2008

1.2 CANopen technology

1.2.1 CANopen description language

CANopen is a device- and manufacturer-independent description language for communication via the CAN bus. CANopen provides a common basis for interchanging commands and data between CAN bus devices.

1.2.2 Communication layers

CANopen uses the CAN bus technology for data communication.

CANopen is based on the basic network services for data communication as per the ISO-OSI model model. 3 layers enable data communication via the CAN bus.

- Physical Layer
- Data Link Layer
- Application Layer

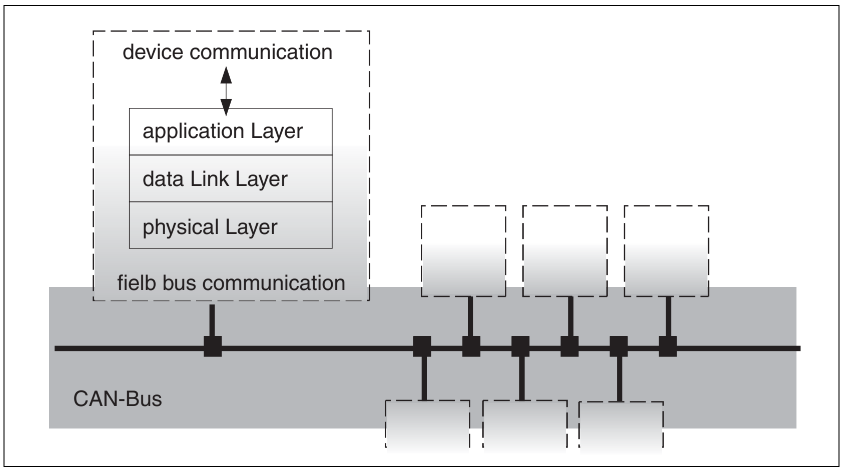


Figure 1.1 CANopen layer model

Physical Layer The physical layer defines the electrical properties of the CAN bus such as connectors, cable length and cable properties such as bit-coding and bit-timing.

Data Link Layer The data link layer connects the network devices. It assigns priorities to individual data packets and monitors and corrects errors.

Application Layer The application layer uses communication objects (COB) to exchange data between the various devices. Communication objects are elementary components for creating a CANopen application.

1.2.3 Objects

All processes under CANopen are executed via objects. Objects carry out different tasks; they act as communication objects for data transport to the fieldbus, control the process of establishing a connection or monitor the network devices. If objects are directly linked to the device (device-specific objects), the device functions can be used and changed via these objects.

The product provides corresponding parameters for CANopen object groups 3000_h and 6000_h.
The names of the parameters and the data type of the parameters may be different from the DSP402 definition for object group 6000_h. In this case, enter the data type according to the DSP402.
A detailed description of all parameters can be found in the product manual in the Parameters chapter.

Object dictionary The object dictionary of each network device allows for communication between the devices. Other devices find all objects with which they can communicate in this dictionary.

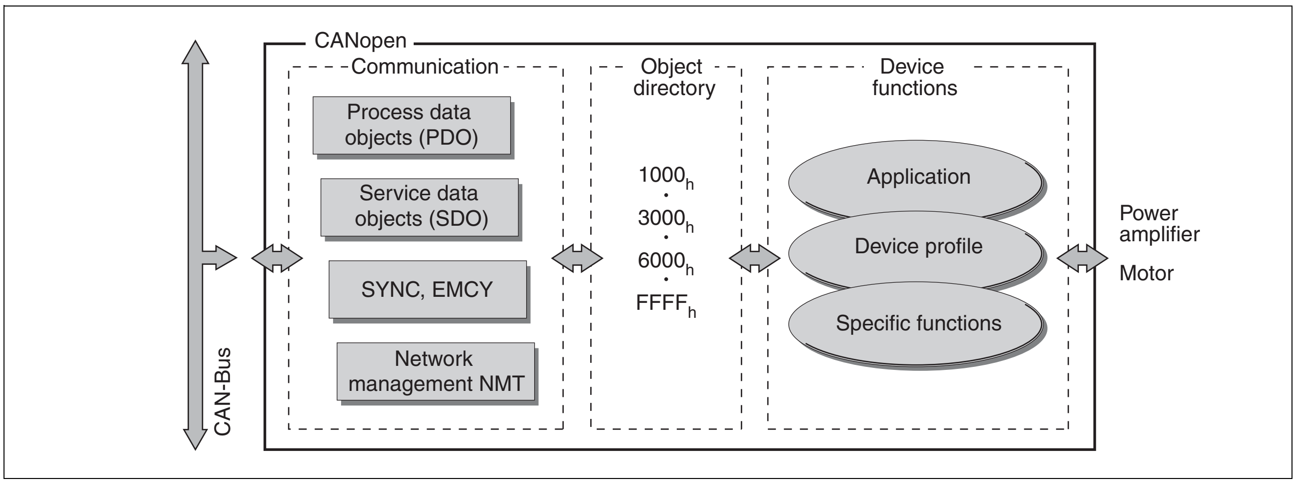


Figure 1.2 Device model with object dictionary

Objects for describing the data types and executing the communication tasks and device functions under CANopen are included in the object dictionary.

Object index Each object is addressed by means of a 16 bit index, which is represented as a four-digit hexadecimal number. The objects are arranged in groups in the object dictionary. The following table shows an overview of the object dictionary as per the CANopen.

Index range (hex)	Object groups
1000_h-$2FFF_h$	Communication profile
3000_h-$5FFF_h$	Vendor-specific objects
6000_h-$9FFF_h$	Standardized device profiles
$A000_h$-$FFFF_h$	Reserved

For a list of all CANopen objects see chapter 8 "Object directory".

0198441113718, V2.01, 11.2008

1.2.4 CANopen profiles

Standardized profiles

Standardized profiles describe objects that are used with different devices without additional configuration. The users and manufacturers organization CAN in Automation has standardized various profiles. These include:

- DS301 communication profile
- DSP402 device profile

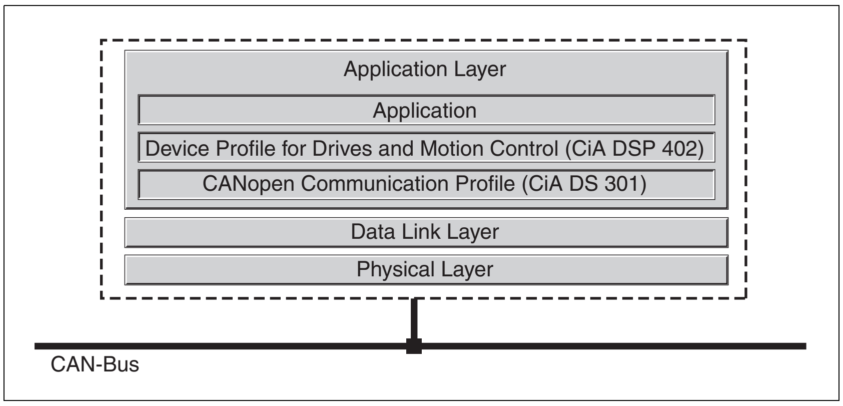

Application Layer
Application
Device Profile for Drives and Motion Control (CiA DSP 402)
CANopen Communication Profile (CiA DS 301)
Data Link Layer
Physical Layer
CAN-Bus

Figure 1.3 CANopen reference model

DS301 communication profile

The DS301 communication profile is the interface between device profiles and CAN bus. It was specified in 1995 under the name DS301 and defines uniform standards for common data exchange between different device types under CANopen.

The objects of the communication profile in the device carry out the tasks of data exchange and parameter exchange with other network devices and initialize, control and monitor the device in the network.

DSP402 device profile

The DSP402 device profile describes standardized objects for positioning, monitoring and settings of drives. The tasks of the objects include:

- Device monitoring and status monitoring (Device Control)
- Standardized parameterization
- Changing, monitoring and execution of operating modes

Vendor-specific profiles

The basic functions of a device can be used with objects of standardized device profiles standardized. Only vendor-specific device profiles offer the complete range of functions. The objects with which the special functions of a device can be used under CANopen are defined in these vendor-specific device profiles.

1.3 Documentation and literature references

CAN users and manufacturers organization	CiA - CAN in Automation Am Weichselgarten 26 D-91058 Erlangen http://www.can-cia.org/
CANopen standards	• CiA Standard 301 (DS301) CANopen application layer and communication profile V4.02, February 2002 • CiA Standard 402 (DSP402) Device profile for drives and motion control V2.0, July 2002 • ISO/DIS 11898: Controller Area Network (CAN) for high speed communication;1993 • EN 50325-4: Industrial communications subsystem based on ISO 11898 for controller device interfaces (CANopen); 2002
Literature	Controller Area Network Konrad Etschberger, Carl Hanser Verlag ISBN 3-446-19431-2
Manuals	In addition to this fieldbus manual, the following manuals also belong to the product: • **Product manual**, describes the technical data, installation, commissioning and all operating modes and functions. • **Motor manual**, describes the technical characteristics of the motors, including correct installation and commissioning.

0198441113718, V2.01, 11.2008

2 Before you begin - safety information

The information provided in this manual supplements the product manual. Carefully read the product manual before you begin.

3 Basics

3.1 Communication profile

CANopen manages communication between the network devices with object dictionaries and objects. A network device can use process data objects (PDO) and service data objects (SDO) to request the object data from the object dictionary of another device and, if permissible, write back modified values.

The following can be done by accessing the objects of the network devices

- Exchange parameter values
- Start motion functions of individual CAN bus devices
- Request status information

3.1.1 Object dictionary

Each CANopen device manages an object dictionary which contains all objects for communication.

Index, subindex The objects are addressed in the object dictionary via a 16 bit index. One or more 8 bit subindex entries for each object specify individual data fields in the object. Index and subindex are shown in hexadecimal notation with a subscript "$_h$".

Example The following table shows index and subindex entries using the example of the object `software position limit (607D`$_h$`)` for specifying the positions of software limit switches.

Index	Subindex	Name	Meaning
607D$_h$	00$_h$	-	Number of data fields
607D$_h$	01$_h$	min. position limit	Lower limit switch
607D$_h$	02$_h$	max. position limit	Upper limit switch

Table 3.1 Example of index and subindex entries

Object descriptions in the manual For CAN programming of a device, the objects of the following object groups are described in detail:

- 1xxx$_h$ objects: Communication objects in this chapter
- 3xxx$_h$ objects: Vendor-specific objects required to control the device in chapter 6 "Operation".
- 6xxx$_h$ objects: Standardized objects of the device profile in chapter 6 "Operation"

Standardized objects Standardized objects allow you to use the same application program for different network devices of the same device type. This requires these objects to be contained in the object dictionary of the network devices. Standardized objects are defined in the DS301 communication profile and the DSP402 device profile.

3.1.2 Communication objects

Overview The communication objects are standardized with the DS301 CANopen communication profile. The objects can be classified into 4 groups according to their tasks.

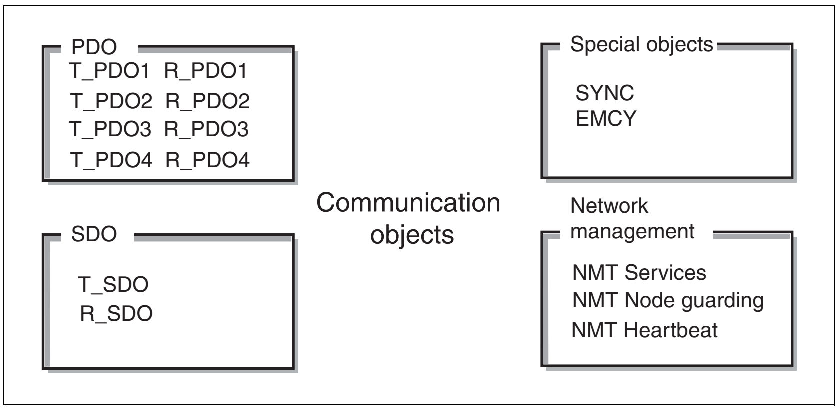


Figure 3.1 Communication objects; the following applies to the perspective of the network device: T_..: "Transmit", R_..: "Receive"

- PDOs (process data objects) for real-time transmission of process data
- SDOs (service data object) for read and write access to the object dictionary
- Objects for controlling CAN messages:
  - SYNC object (synchronization object) for synchronization of network devices
  - EMCY object (emergency object), for signaling errors of a device or its peripherals.
- Network management services:
  - NMT services for initialization and network control (NMT: network management)
  - NMT Node Guarding for monitoring the network devices
  - NMT Heartbeat for monitoring the network devices

CAN message

Data is exchanged via the CAN bus in the form of CAN messages. A CAN message transmits the communication object and a variety of administration and control information.

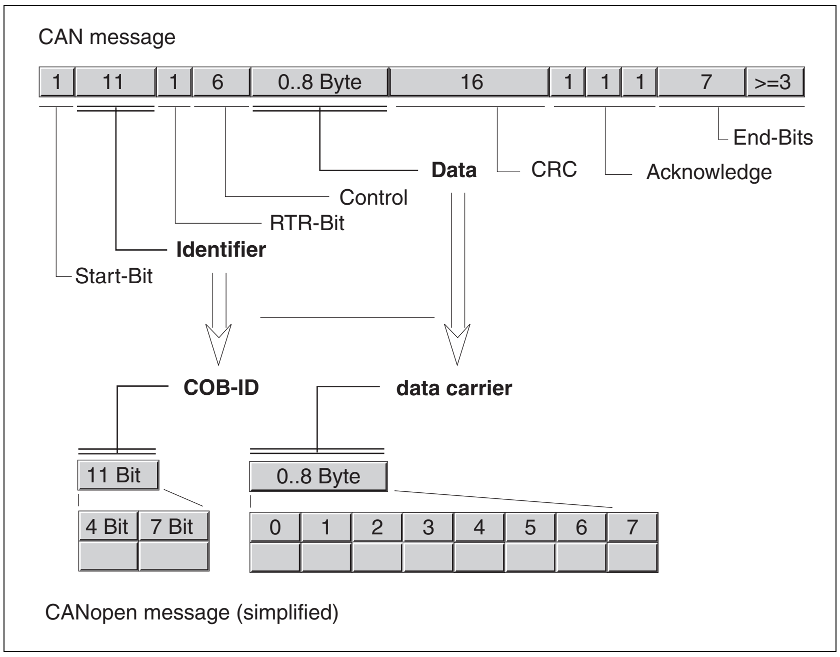


Figure 3.2 CAN message and simplified representation of CANopen message

CANopen message

For work with CANopen objects and for data exchange, the CAN message can be represented in simplified form because most of the bits are used for error correction. These bits are automatically removed from the receive message by the data link layer of the OSI model, and added to a message before it is transmitted.

The two bit fields "Identifier" and "Data" form the simplified CANopen message. The "Identifier" corresponds to the "COB ID" and the "Data" field to the data frame (maximum length 8 bytes) of a CANopen message.

COB ID

The COB ID (**C**ommunication **OB**ject **Id**entifier) has 2 tasks as far as controlling communication objects is concerned:

- Bus arbitration: Specification of transmission priorities
- Identification of communication objects

An 11 bit COB identifier as per the CAN 3.0A specification is defined for CAN communication; it comprises 2 parts

- Function code, 4 bits
- Node address (node ID), 7 bits.

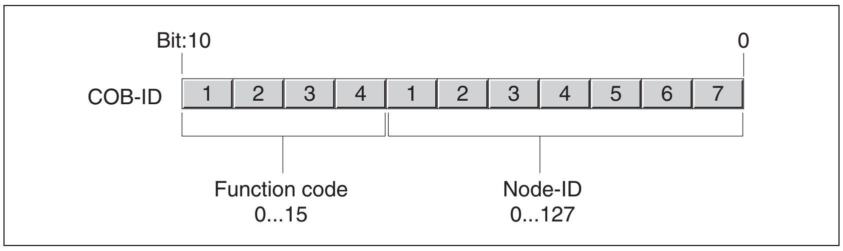


Figure 3.3 COB ID with function code and node address

COB IDs of the communication objects

The following table shows the COB IDs of all communication objects with the factory settings. The column "Index of object parameters" shows the index of special objects with which the settings of the communication objects can be read or modified via an SDO.

Communication object	Function code	Node address, node ID [1...127]	COB ID decimal (hexadecimal)	Index of object parameters
NMT Start/Stop Service	0 0 0 0	0 0 0 0 0 0 0	0 (0_h)	-
SYNC object	0 0 0 1	0 0 0 0 0 0 0	128 (80_h)	1005_h....1007_h
EMCY object	0 0 0 1	x x x x x x x	128 (80_h) + node ID	1014_h, 1015_h
T_PDO1 1)	0 0 1 1	x x x x x x x	384 (180_h) + node ID	1800_h
R_PDO1 1)	0 1 0 0	x x x x x x x	512 (200_h) + node ID	1400_h
T_PDO2 1)	0 1 0 1	x x x x x x x	640 (280_h) + node ID	1801_h
R_PDO2 1)	0 1 1 0	x x x x x x x	768 (300_h) + node ID	1401_h
T_PDO3 1)	0 1 1 1	x x x x x x x	896 (380_h) + node ID	1802_h
R_PDO3 1)	1 0 0 0	x x x x x x x	1024 (400_h) + node ID	1402_h
T_PDO4	1 0 0 1	x x x x x x x	1152 (480_h) + node ID	1803_h
R_PDO4	1 0 1 0	x x x x x x x	1280 (500_h) + node ID	1403_h
T_SDO	1 0 1 1	x x x x x x x	1408 (580_h) + node ID	-
R_SDO	1 1 0 0	x x x x x x x	1536 (600_h) + node ID	-
NMT error control	1 1 1 0	x x x x x x x	1792 (700_h) + node ID	
LMT Services 1)	1 1 1 1	1 1 0 0 1 0 x	2020 ($7E4_h$), 2021 ($7E5_h$)	
NMT Identify Service 1)	1 1 1 1	1 1 0 0 1 1 0	2022 ($7E6_h$)	
DBT Services 1)	1 1 1 1	1 1 0 0 x x x	2023 ($7E7_h$), 2024 ($7F8_h$)	
NMT Services 1)	1 1 1 1	1 1 0 1 0 0 x	2025 ($7E9_h$), 2026 ($7EA_h$)	

1) Not supported by the device

Table 3.2 COB IDs of all communication objects

COB IDs of PDOs can be changed as required. The assignment pattern for COB IDs only specifies a basic setting.

Function code The function code classifies the communication objects. Since the bits of the function code in the COB ID are more significant, the function code simultaneously controls the transmission priorities: Objects with a lower function code are transmitted with higher priority. For example, an object with function code "1" is transmitted prior to an object with function code "3" in the case of simultaneous bus access.

Node address Every network device is configured before it is operated on the network. The device is assigned a 7 bit node address (node ID) between 1 (01_h) and 127 ($7F_h$). The device address "0" is reserved for "broadcast" transmissions which are used to send messages to all devices simultaneously.

Example Selection of a COB ID

For a device with the node address 5, the COB ID of the communication object T_PDO1 is:

384+node ID = 384 (180_h) + 5 = 389 (185_h).

Data frame The data frame of the CANopen message can hold up to 8 bytes of data. In addition to the data frame for SDOs and PDOs, special frame types are specified in the CANopen profile:

- Error data frame
- Remote data frame for requesting a message

The data frames contain the respective communication objects.

3.1.3 Communication relationships

CANopen uses 3 relationships for communication between network devices:

- Master-slave relationship
- Client-server relationship
- Producer-consumer relationship

Master-slave relationship

A network master controls the message traffic. A slave only responds when it is addressed by the master.

The master-slave relationship is used with network management objects for a controlled network start and to monitor the connection of devices.

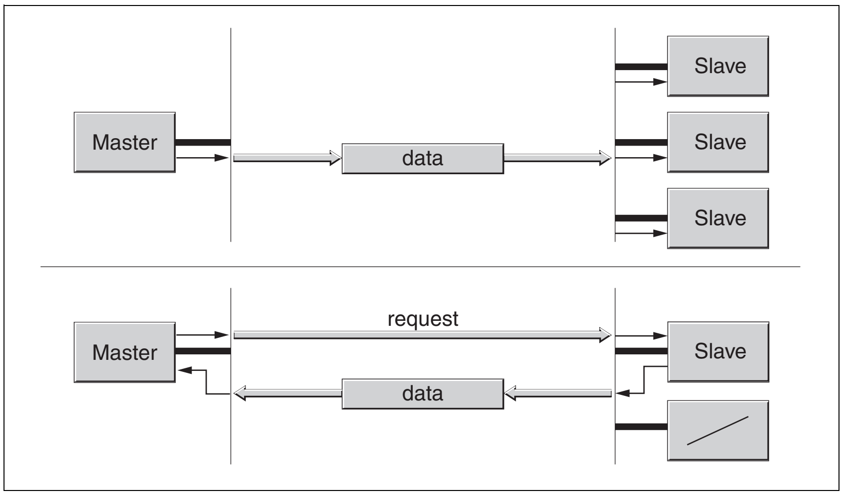


Figure 3.4 Master - slave relationships

Messages can be interchanged with and without confirmation. If the master sends an unconfirmed CAN message, it can be received by a single or by several slaves or by no slave.

To confirm the message, the master requests a message from a specific slave, which then responds with the desired data.

Client-server relationship

A client-server relationship is established between 2 devices. The "server" is the device whose object dictionary is used during data exchange. The "client" addresses and starts the exchange of messages and waits for a confirmation from the server.

A client-server relationship with SDOs is used to send configuration data and long messages.

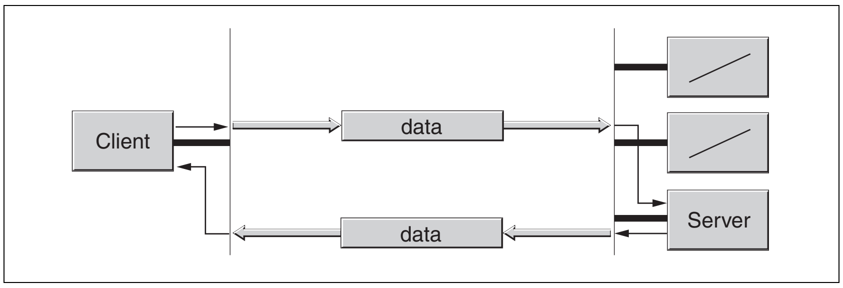


Figure 3.5 Client-server relationship

The client addresses and sends a CAN message to a server. The server evaluates the message and sends the response data as an acknowledgement.

Producer-consumer relationship

The producer-consumer relationship is used for exchanging messages with process data, because this relationship enables fast data exchange without administration data.

A "Producer" sends data, a "Consumer" receives data.

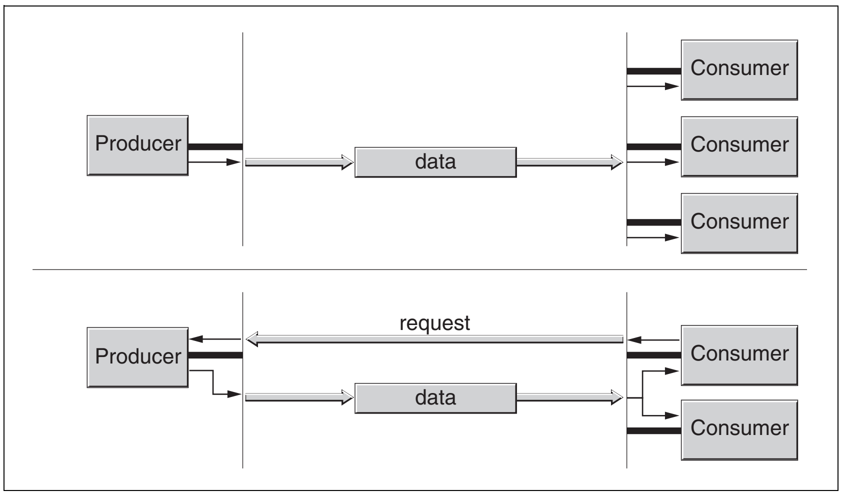


Figure 3.6 Producer-consumer relationships

The producer sends a message that can be received by one or more network devices. The producer does not receive an acknowledgement to the effect that the message was received. The message transmission can be triggered

- by an internal event, e.g. "target position reached"
- by the synchronization object SYNC
- a request of a consumer

For details on the function of the producer-consumer relationship and on requesting messages see chapter 3.3 "Process data communication".

3.2 Service data communication

3.2.1 Overview

Service Data Object(SDO: **S**ervice **D**ata **O**bject) can be used to access the entries of an object dictionary via index and subindex. The values of the objects can be read and, if permissible, also be changed.

Every network device has at least one server SDO to be able to respond to read and write requests from a different device. A client SDO is only required to request SDO messages from the object dictionary of a different device or to change them there.

The T_SDO of an SDO client is used to send the request for data exchange; the R_SDO is used to receive. The data frame of an SDO consist of 8 bytes.

SDOs have a higher COB ID than PDOs and therefore are sent over the CAN bus at a lower priority.

3.2.2 SDO data exchange

A service data object (SDO) sends parameter data between two devices. The data exchange conforms to the client-server relationship. The server is the device to whose object dictionary an SDO message refers.

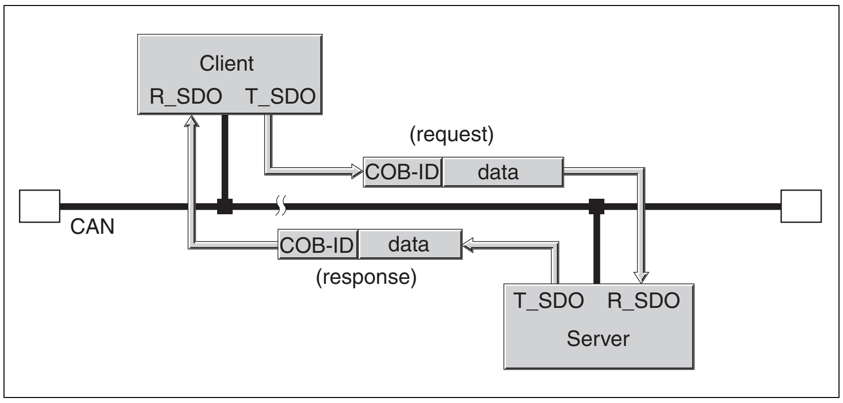


Figure 3.7 SDO message exchange with request and response

Message types Client-server communication is triggered by the client to send parameter values to the server or to get them from the server. In both cases, the client starts the communication with a request and receives a response from the server.

0198441113718, V2.01, 11.2008

3.2.3 SDO message

Put simply, an SDO message consists of the COB ID and the SDO data frame, in which up to 4 bytes of data can be sent. Longer data sequences are distributed over multiple SDO messages with a special protocol.

The device sends SDOs of up to 4 bytes data length (data). Greater amounts of data such as 8 byte values of the data type "Visible String 8" can be distributed over multiple SDOs and are transmitted successively in 7 byte blocks.

Example The following illustration shows an example of an SDO message.

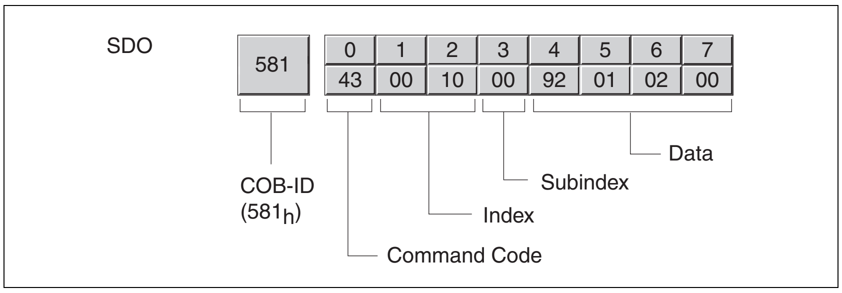


Figure 3.8 SDO message, example

COB ID and data frame R_SDO and T_SDO have different COB IDs.
The data frame of an SDO messages consists of:

- Command code (ccd) in which the SDO message type and the data length of the transmitted value are encrypted
- Index and subindex which point to the object whose data is transported with the SDO message
- Data of up to 4 bytes

Evaluation of numeric values Index and data are transmitted left-aligned in Intel format. If the SDO contains numerical values of more than 1 byte in length, the data must be rearranged byte-by-byte before and after a transmission.

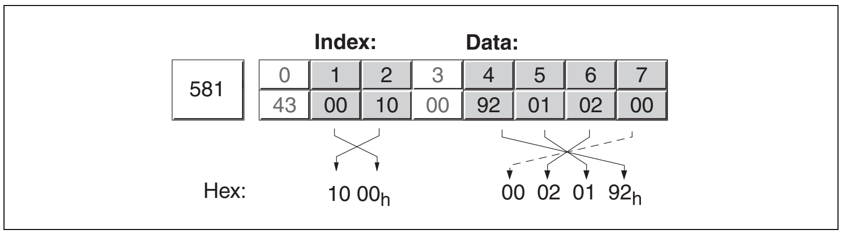


Figure 3.9 Rearranging numeric values greater than 1 byte

0198441113718, V2.01, 11.2008

3.2.4 Reading and writing data

Writing data

The client starts a write request by sending index, subindex, data length and value.

The server sends a confirmation indicating whether the data was correctly processed. The confirmation contains the same index and subindex, but no data.

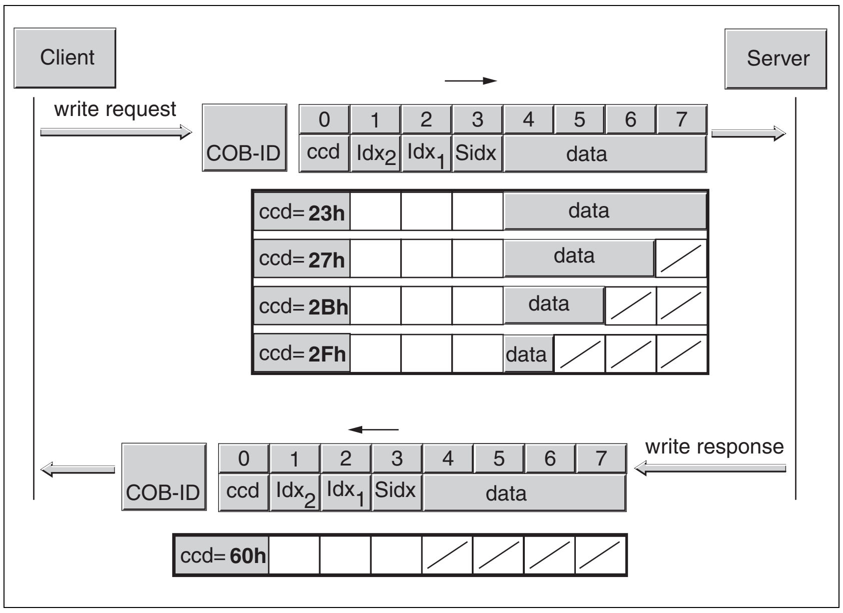


Figure 3.10 Writing parameter values

Unused bytes in the data field are shown with a slash in the graphic. The content of these data fields is not defined.

ccd coding

The table below shows the command code for writing parameter values. It depends on the message type and the transmitted data length.

Message type	Data length used				
	4 bytes	3 bytes	2 bytes	1 byte	
Write request	23_h	27_h	$2B_h$	$2F_h$	Transmitting parameters
Write response	60_h	60_h	60_h	60_h	Confirmation
Error response	80_h	80_h	80_h	80_h	Error

Table 3.3 Command code for writing parameter values

Reading data The client starts a read request by sending the index and subindex that point to the object or the object value whose value it wants to read.

The server confirms the request by sending the desired data. The SDO response contains the same index and subindex. The length of the response data is specified in the command code "ccd".

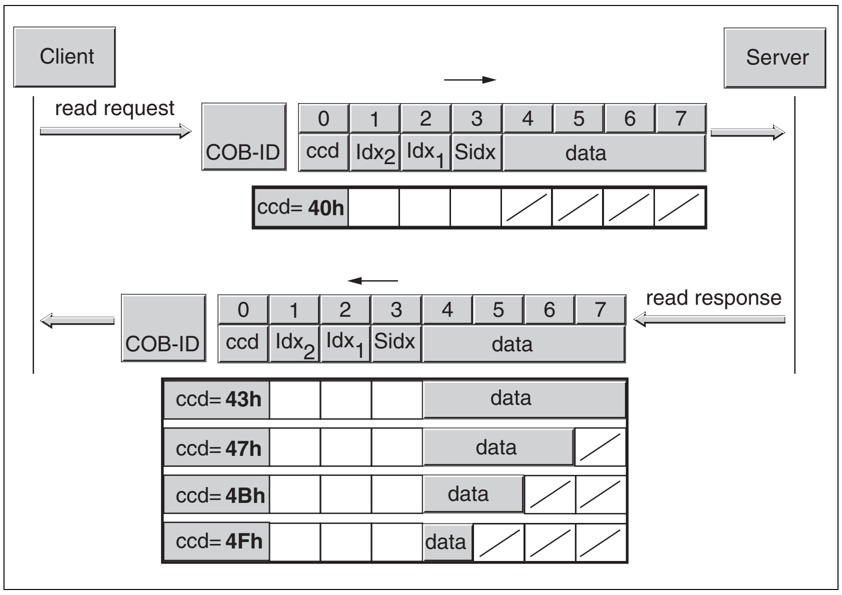


Figure 3.11 Reading a parameter value

Unused bytes in the data field are shown with a slash in the graphic. The content of these data fields is not defined.

ccd coding The table below shows the command code for transmitting a read value. It depends on the message type and the transmitted data length.

Message type	Data length used				
	4 bytes	3 bytes	2 bytes	1 byte	
read request	40_h	40_h	40_h	40_h	Request read value
Read response	43_h	47_h	$4B_h$	$4F_h$	Return read value
Error response	80_h	80_h	80_h	80_h	Error

Table 3.4 Command code for transmitting a read value

Error response If a message could not be evaluated without errors, the server sends an error message. For details on the evaluation of the error message see chapter 7.3.3 "SDO error message ABORT".

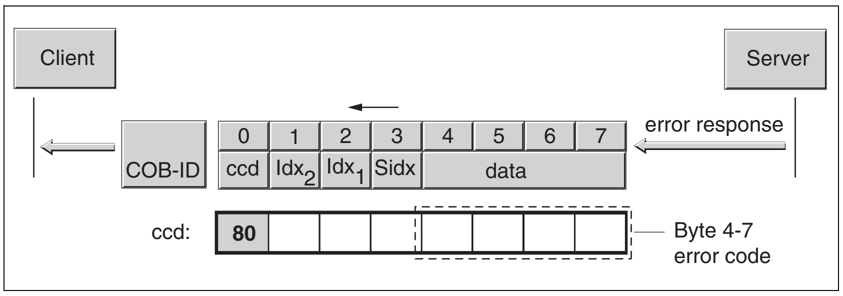


Figure 3.12 Response with error message (error response)

3.3 Process data communication

3.3.1 Overview

Process data objects (PDO: **P**rocess **D**ata **O**bject) are used for real-time data exchange of process data such as actual and reference or operating state of the device. Transmission is very fast because the data is sent without additional administration data and a response from the recipient is not required.

The flexible data length of a PDO message also increases the data throughput. A PDO message can transmit up to 8 bytes of data. If only 2 bytes are assigned, only 2 data bytes are sent.

The length of a PDO message and the assignment of the data fields are specified by PDO mapping. For more information see chapter 3.3.4 "PDO mapping".

PDO messages can be exchanged between devices that generate or process process data.

3.3.2 PDO data exchange

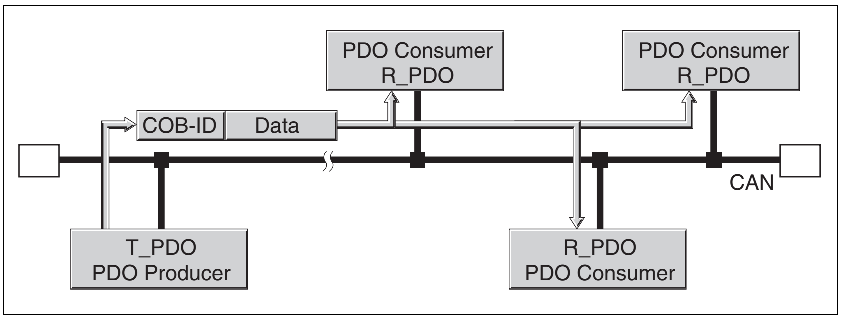


Figure 3.13 PDO data exchange

Data exchange with PDOs follows to the producer-consumer relationship and can be triggered in 3 ways

- Synchronized
- Event-driven, asynchronous
- On request of a consumer, asynchronous

The SYNC object controls synchronized data processing. Synchronous PDO messages are transmitted immediately like the standard PDO messages, but are only evaluated on the next SYNC. For example, several drives can be started simultaneously via synchronized data exchange.

The device immediately evaluates PDO messages that are called on request or in an event-driven way.

The transmission type can be specified separately for each PDO with subindex 02_h (transmission type) of the PDO communication parameter. The objects are listed in Table 3.5.

3.3.3 PDO message

T_PDO, R_PDO

One PDO each is available for sending and receiving a PDO message:

- T_PDO to transmit PDO messages (T: Transmit),
- R_PDO to receive PDO messages (R: Receive).

The following settings for PDOs correspond to the defaults for the device, unless otherwise specified. They can be read and set via objects of the communication profile.

The device uses 8 PDOs, 4 receive PDOs and 4 transmit PDOs. By default, all PDOs are evaluated or transmitted in an event-driven way.

PDO settings

The PDO settings can be read and changed with 8 communication objects:

Object	Meaning
1st receive PDO parameter (1400_h)	Settings for R_PDO1
2nd receive PDO parameter (1401_h)	Settings for R_PDO2
3rd receive PDO parameter (1402_h)	Settings for R_PDO3
4th receive PDO parameter (1403_h)	Settings for R_PDO4
1st transmit PDO parameter (1800_h)	Settings for T_PDO1
2nd transmit PDO parameter (1801_h)	Settings for T_PDO2
3rd transmit PDO parameter (1802_h)	Settings for T_PDO3
4th transmit PDO parameter (1803_h)	Settings for T_PDO4

Table 3.5 Communication objects for PDO

Activating PDOs

With the default PDO settings, R_PDO1 and T_PDO1 are activated. The other PDOs must be activated first.

A PDO is activated with bit 31 (valid bit) in subindex 01_h of the respective communication object:

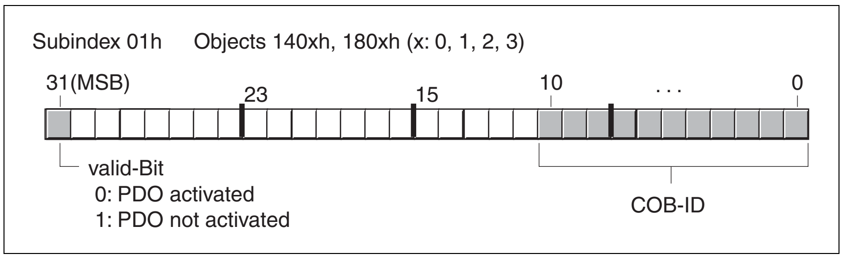


Figure 3.14 Activating PDOs via subindex 01_h, bit 31

Example

Setting for R_PDO3 in object 1402_h

- Subindex 01_h = 8000 $04xx_h$: R_PDO3 not activated
- Subindex 01_h = 0000 $04xx_h$: R_PDO3 activated.

Values for "x" in the example depend on the COB ID setting.

0198441113718, V2.01, 11.2008

PDO time intervals

The time intervals "inhibit time" and "event timer" can be set for each transmit PDO.

- The time interval "inhibit time" can be used to reduce the load on the CAN bus, which can be the result of continuous transmission of T_PDOs. If an inhibit time not equal to zero is entered, a transmitted PDO will only be re-transmitted after the inhibit time has elapsed. The time is set with subindex 03_h.
- The time interval "event timer" cyclically triggers an event message. After the time intervals has elapsed, the device transmits the event-controlled T_PDO. The time is set with subindex 05_h.

Receive PDOs

The objects for R_PDO1, R_PDO2 and R_PDO3 are permanently set. The object that is mapped to PDO R_PDO4 can be modified by PDO mapping.

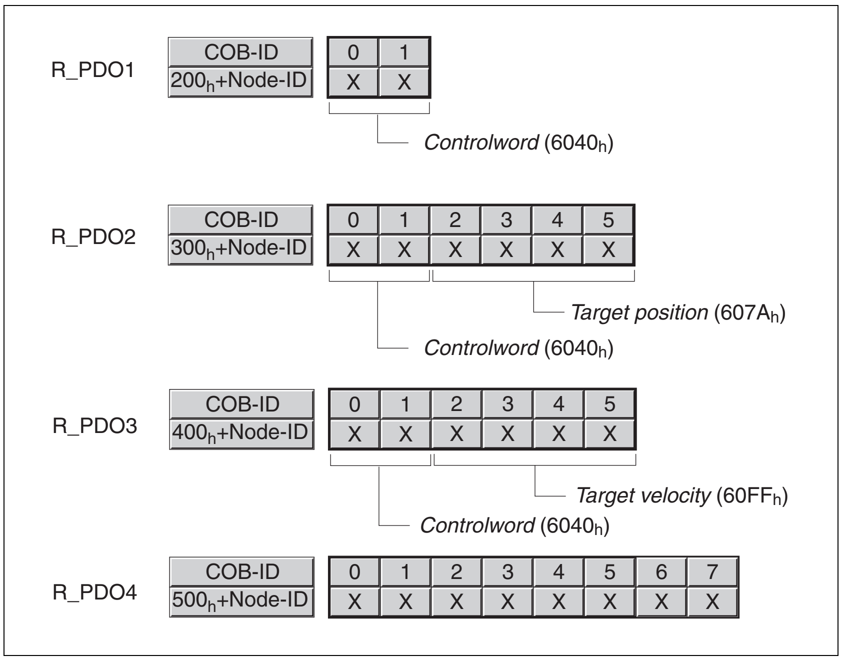


Figure 3.15 Receive PDOs

R_PDO1

In the R_PDO1, the control word, object `controlword (6040h)`, of the state machine is mapped which can be used to set the operating state of the device.

R_PDO1 is evaluated asynchronously, i.e. it is event-driven. R_PDO1 is permanently set.

R_PDO2

With R_PDO2, the control word and the target position of a motion command, object `target position (607Ah)`, is received for positioning in "Profile Position" operating mode.

R_PDO2 is evaluated asynchronously, i.e. it is event-driven. R_PDO2 is permanently set.

For details on the SYNC object see chapter 3.4 "Synchronization".

R_PDO3 In R_PDO3, the control word and the reference speed, object `Target velocity (60FF`$_h$`)`, are mapped for the velocity profile in "Profile Velocity" operating mode.

R_PDO3 is evaluated asynchronously, i.e. it is event-driven. R_PDO3 is permanently set.

R_PDO4 R_PDO4 is used to transmit vendor-specific object values. By default, R_PDO4 is empty.

R_PDO4 is evaluated asynchronously, i.e. it is event-driven. R_PDO4 can be used to map various vendor-specific objects by means of PDO mapping.

Transmit PDOs The objects for T_PDO1, T_PDO2 and T_PDO3 are permanently set. The object that is mapped to PDO T_PDO4 can be modified by PDO mapping.

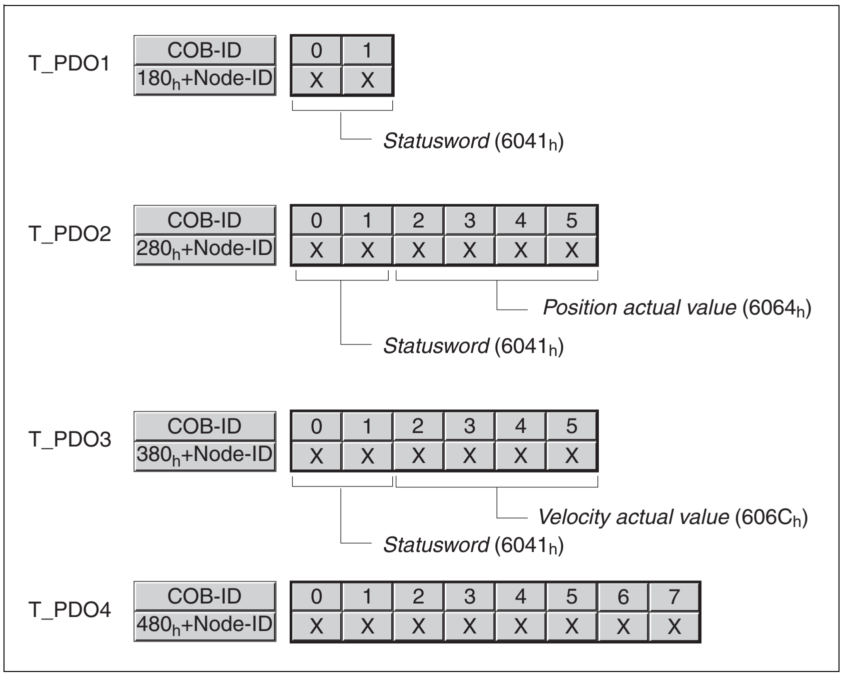


Figure 3.16 Transmit PDOs

T_PDO1 In T_PDO1, the status word, object `statusword (6041`$_h$`)`, of the state machine is mapped.

T_PDO1 is transmitted asynchronously and in an event-driven way whenever the status information changes. No other objects can be mapped with T_PDO1.

T_PDO2 In T_PDO2, the status word and the current position of the motor, object `Position actual value (6064`$_h$`)`, are mapped to monitor positioning in "Profile Position" operating mode.

T_PDO2 is transmitted after receipt of a SYNC object and in an event-driven way. No other objects can be mapped to T_PDO2.

T_PDO3 In T_PDO3, the status word and the current speed, object `Velocity actual value (606C`$_h$`)`, are mapped for monitoring the velocity profile in "Profile Velocity" operating mode.

T_PDO3 is transmitted asynchronously and in an event-driven way whenever the status information changes. No other objects can be mapped with T_PDO3.

T_PDO4 Vendor-specific object values (for monitoring) are transmitted with T_PDO4. By default, T_PDO4 is empty.

T_PDO4 is transmitted asynchronously and in an event-driven way whenever the data changes. The parameter `CANpdo4Event` is used to specify the objects which are to trigger an event. With the default setting of the parameter, all mapped objects trigger an event.

T_PDO4 can be used to map various vendor-specific objects via PDO mapping.

Parameter Name **HMI menu**	**Description**	**Unit** **Minimum value** **Default value** **Maximum value**	**Data type** **R/W** **persistent** **Expert**	**Parameter address via fieldbus**
CANpdo4Event - -	PDO4 event mask Changes of values in the object trigger an event: Bit 0 = 1: first PDO4 object Bit 1 = 1: second PDO4 object Bit 2 = 1: third PDO4 object Bit 3 = 1: fourth PDO4 object Bit 4..15 : reserved	- 0 15 15	UINT16 UINT16 R/W - -	CANopen 3017:5$_h$ Modbus 5898

3.3.4 PDO mapping

Up to 8 bytes of data from different areas of the object dictionary can be transmitted with a PDO message. Mapping of data to a PDO message is referred to as PDO mapping.

Chapter 8 "Object directory" contains a list of vendor-specific objects that are available for PDO mapping.

The picture below shows the data exchange between PDOs and object dictionary on the basis of two examples of objects in T_PDO4 and R_PDO4 of the PDOs.

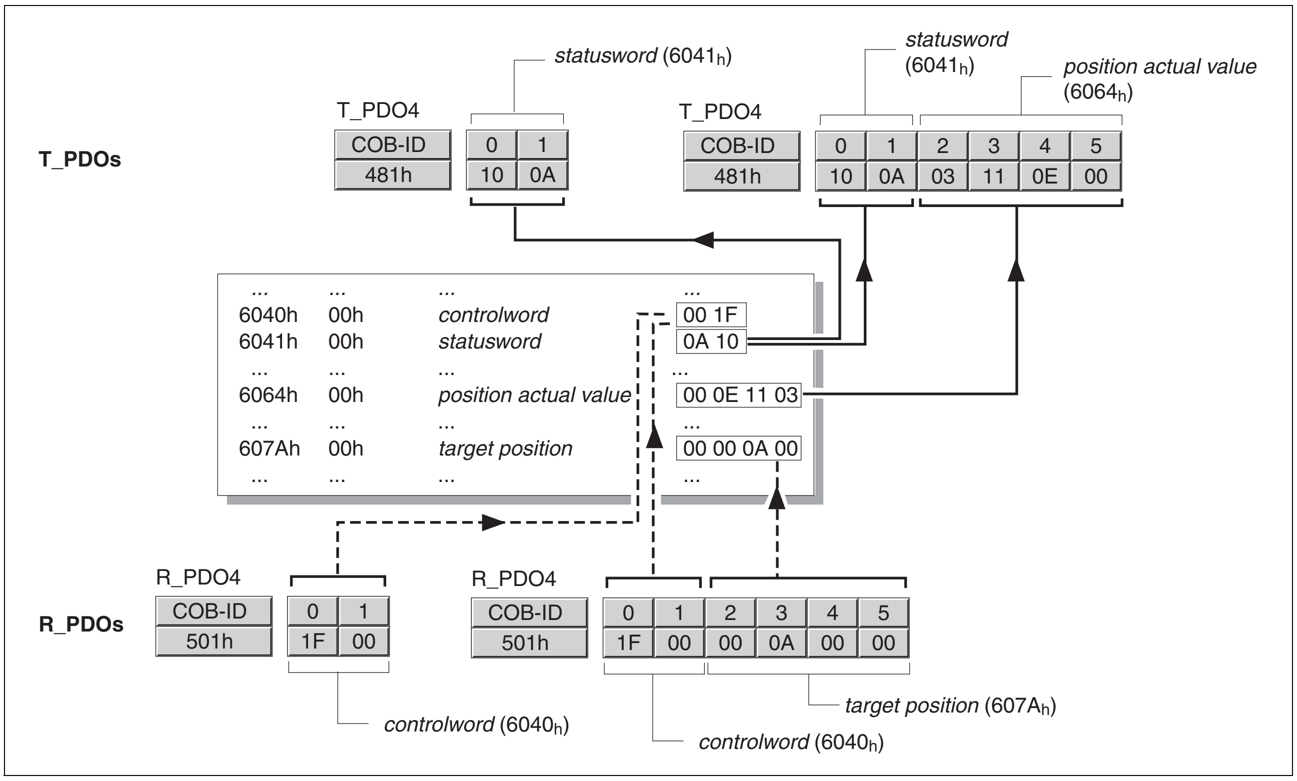


Figure 3.17 PDO mapping, in this case for a device with node address 1

Static PDO mapping

The device uses static and dynamic PDO mapping. Static PDO mapping means that all objects are mapped in accordance with a fixed setting in the corresponding PDO.

The settings for PDO mapping are defined in an assigned communication object for each PDO.

Object	**PDO mapping for**	**type**
1st receive PDO mapping (1600_h)	R_PDO1	static
2nd receive PDO mapping (1601_h)	R_PDO2	static
3rd receive PDO mapping (1602_h)	R_PDO3	static
4th receive PDO mapping (1603_h)	R_PDO4	dynamic
1st transmit PDO mapping ($1A00_h$)	T_PDO1	static
2nd transmit PDO mapping ($1A01_h$)	T_PDO2	static
3rd transmit PDO mapping ($1A02_h$)	T_PDO3	static
4th transmit PDO mapping ($1A03_h$)	T_PDO4	dynamic

Structure of entries

Up to 8 bytes of 8 different objects can be mapped in a PDO. Each communication object for setting the PDO mapping provides 4 subindex entries. A subindex entry contains 3 pieces of information on the object: the index, the subindex and the number of bits that the object uses in the PDO.

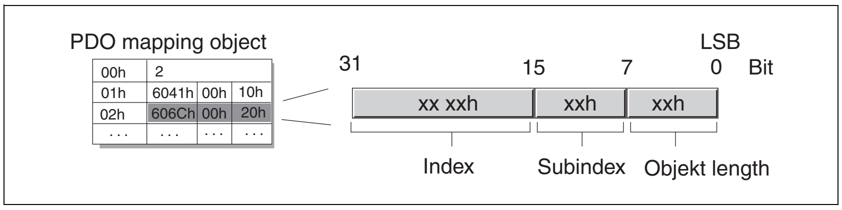


Figure 3.18 Structure of entries for PDO mapping

Subindex 00_h of the communication object contains the number of valid subindex entries.

PDO mapping objects

Object (Index:Subindex)	PDO	Data type
_IO_act (3008:1_h)	T_PDO	UINT16
ANA1_act (3009:1_h)	T_PDO	INT16
ANA2_act (3009:5_h)	T_PDO	INT16
JOGactivate (301B:9_h)	R_PDO	UINT16
_actionStatus (301C:4_h)	T_PDO	UINT16
_p_actRAMPusr (301F:2_h)	T_PDO	INT32
CUR_I_target (3020:4_h)	R_PDO	INT16
SPEEDn_target (3021:4_h)	R_PDO	INT16
GEARdenom (3026:3_h)	R_PDO	INT32
GEARnum (3026:4_h)	R_PDO	INT32
controlword (6040_h)	R_PDO	UINT16
Status word (6041_h)	T_PDO	UINT16
Position actual value (6064_h)	T_PDO	INT32
Velocity actual value ($606C_h$)	T_PDO	INT32
Target position ($607A_h$)	R_PDO	INT32
profile velocity (6081_h)	R_PDO	UINT32
Target velocity ($60FF_h$)	R_PDO	INT32

3.4 Synchronization

The synchronization object SYNC controls the synchronous exchange of messages between network devices for purposes such as the simultaneous start of multiple drives.

The data exchange conforms to the producer-consumer relationship. The SYNC object is transmitted to all devices by a network device and can be evaluated by all devices that support synchronous PDOs.

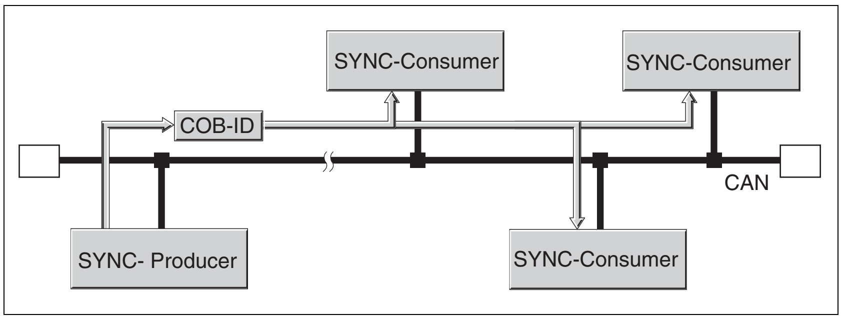


Figure 3.19 SYNC message

Time values for synchronization

Two time values define the behavior of synchronous data transmission:

- The cycle time specifies the time intervals between 2 SYNC messages. It is set with the object `Communication cycle period(1006`$_h$`)`.
- The synchronous time window specifies the time span during which the synchronous PDO messages must be received and trnasmitted. The time window is defined with the object `Synchronous window length (1007`$_h$`)`.

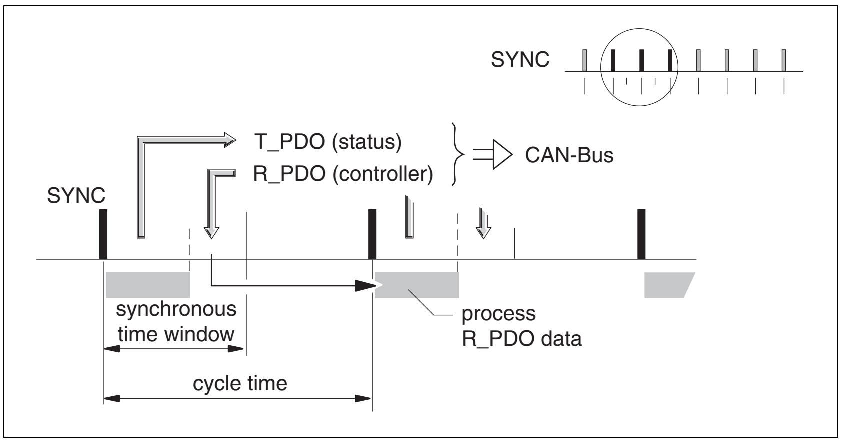


Figure 3.20 Synchronization times

Synchronous data transmission

From the perspective of a SYNC recipient, in one time window the status data is transmitted first in a T_PDO, then new control data is received via an R_PDO. However, the control data is only processed when the next SYNC message is received. The SYNC object itself does not transmit data.

Cyclic ad acyclic data transmission

Synchronous exchange of messages can be cyclic or acyclic.

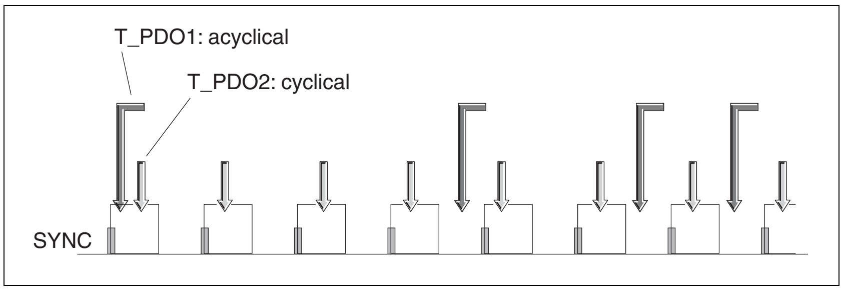


Figure 3.21 Cyclic and acyclic transmission

In the case of cyclic transmission, PDO messages are exchanged continuously in a specified cycle, e.g. with every SYNC message.

If a synchronous PDO message is transmitted acyclically, it can be transmitted or received at any time; however, it will not be valid until the next SYNC message.

Cyclic or acyclic behavior of a PDO is specified in the subindex `transmission type` (02_h) of the corresponding PDO parameter, e.g. in the object `1st receive PDO parameter` ($1400_h:02_h$) for R_PDO1.

COB ID, SYNC object

For fast transmission, the SYNC object is transmitted unconfirmed and with high priority.

The COB ID of the SYNC object is set to the value 128 (80_h) by default. The value can be changed after initialization of the network with the object `COB-ID SYNC Message` (1005_h) .

"Start" PDO

With the default settings of the PDOs, R_PDO2/T_PDO2 and R_PDO3/ T_PDO3 are received and transmitted synchronously. Both PDOs are used for starting and monitoring operating modes. The synchronization allows an operating mode to be started simultaneously on multiple devices so that, for example, the feed of a portal drive with several motors can be synchronized.

3.5 Emergency service

The Emergency Service signals internal device errors via the CAN bus. The error is transmitted to all network devices with an EMCY object according to the Consumer-Producer relationship.

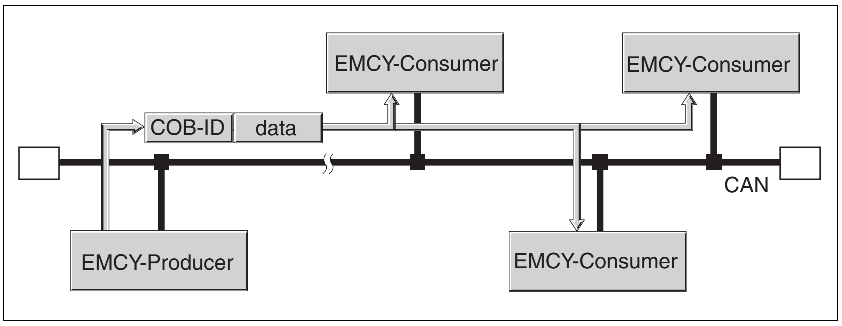


Figure 3.22 Error message via EMCY objects

Boot-up message The communication profile DS301, version 3.0, defines an additional task for the EMCY object: sending a boot-up message. A boot-up message informs all network devices that the device that transmitted the message is ready for operation in the CAN network.

The boot-up message is transmitted with the COB ID 700h + node ID and one data byte (00h).

3.5.1 Error evaluation and handling

EMCY message If an internal device error occurs, the device switches to the fault state as per the CANopen state machine. At the same time, it transmits an EMCY message with error register and error code.

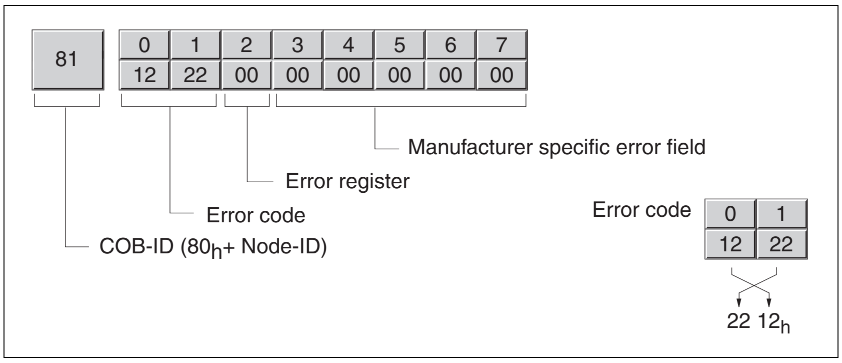


Figure 3.23 EMCY message

Bytes 0,1 - Error code, value is also saved in the object `Error code` (`603F`$_h$)

Byte 2 - Error register, value is also saved in the object `Error register` (`1001`$_h$), see 7.3.1 "Error register".

Bytes 3, 4 - Vendor-specific error code of the mapped object

Bytes 5, 6 - Index of the mapped object

Byte 7 - Subindex of the mapped object

COB ID The COB ID for each device on the network supporting an EMCY object is determined on the basis of the node address:

COB ID = Function code EMCY object (80_h) + node ID

The function code of the COB ID can be changed with the object `COB-ID emergency`(1014_h).

Error register and error code The error code indicates the fault state of the device in a bit-coded form. Bit 0 remains set as long as an error is active. The remaining bits identify the error type. The precise cause of error can be determined on the basis of the error code. The error code is transmitted in Intel format as a 2 byte value; the bytes must be reversed for evaluation.

A list of all error messages and error responses by the device as well as remedies can be found in chapter 7 "Diagnostics and troubleshooting".

Error memory The device saves the error register in the object `Error register` (1001_h) and the last error that occurred in the object Error`Error code` ($603F_h$) . The last 20 error messages are stored in the object `FLT_err_num` ($303C:1_h$) in the order in which the errors occurred. `FLT_MemReset` ($303B:5_h$) resets the read pointer of the error memory to the oldest error.

3.6 Network management services

Network management (NMT) is part of the CANopen communication profile; it is used to initialize the network and the network devices and to start, stop and monitor the network devices in network mode.

NMT services are executed in a master-slave relationship. The NMT master addresses individual NMT slaves via their node address. A message with node address "0" is directed to all NMT slaves simultaneously.

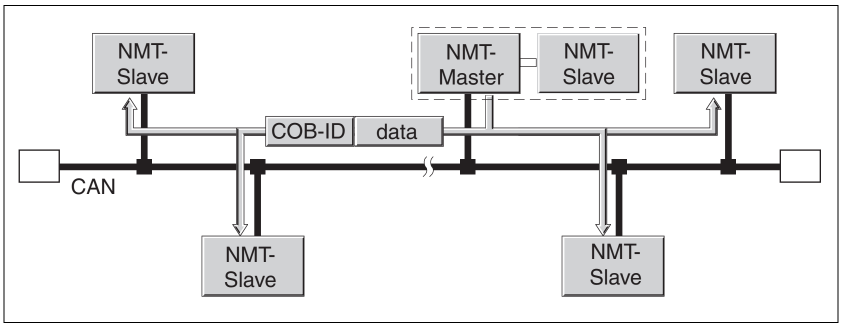


Figure 3.24 NMT services via the master-slave relationship

The device can only take on the function of an NMT slave.

NMT services NMT services can be divided into two groups:

- Services for device control, to initialize devices for CANopen communication and to control the behavior of devices in network mode
- Services:for connection monitoring

3.6.1 NMT services for device control

NMT state machine The NMT state machine describes the initialization and states of an NMT slave in mains operation.

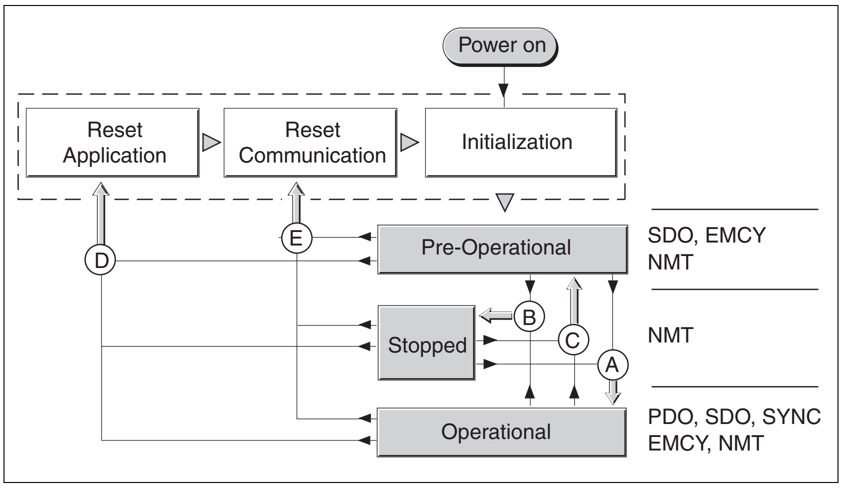


Figure 3.25 NMT state machine and available communication objects

To the right, the graphic shows all communication objects that can be used in the specific network state.

Initialization

An NMT slave automatically runs through an initialization phase after the supply voltage is switched on (power on) to prepare it for CAN bus operation. On completion of the initialization, the slave switches to the state "Pre-operational" and sends a boot-up message. From now on, an NMT master can control the operational behavior of an NMT slave in the network via 5 NMT services, represented in the above illustration by the letters A to E.

NMT service	Transition	Meaning
Start remote node (Start network node)	A	Transition to state "Operational" Start normal network mode with all network devices
Stop remote node (Stop network node)	B	Transition to state "Stopped" Stops communication of the network device in the network. If connection monitoring is active, it remains on. If the power stage is active (state "Operation Enabled" or "QuickStop"), an error of error class 2 is triggered. The drive is stopped and switched off.
Enter Pre-Operational (Transition to "Pre-Operational")	C	Transition to "Pre-Operational" All communication objects except for PDOs can be used. The state "Pre-Operational" can be used for configuration by SDOs: - PDO mapping - Start of synchronization - Start of connection monitoring
Reset node (Reset node)	D	Transition to state "Reset application" Load stored data of the device profiles and automatically transition to "Pre-operational" via "Reset communication".
Reset communication (Reset communication data)	E	Transition to state "Reset communication" Load stored data of the communication profile and automatically switch to the state "Pre-Operational.". If the power stage is active (state "Operation Enabled" or "QuickStop"), an error of error class 2 is triggered. The drive is stopped and switched off.

Persistent data memory

When the supply voltage is switched on (power on), the device loads the saved object data from the non-volatile EEPROM for persistent data to the RAM.

NMT message

The NMT services for device control are transmitted as unconfirmed messages with the COB ID = 0 . By default, they have the highest priority on the CAN bus.

The data frame of the NMT device service consists of 2 bytes.

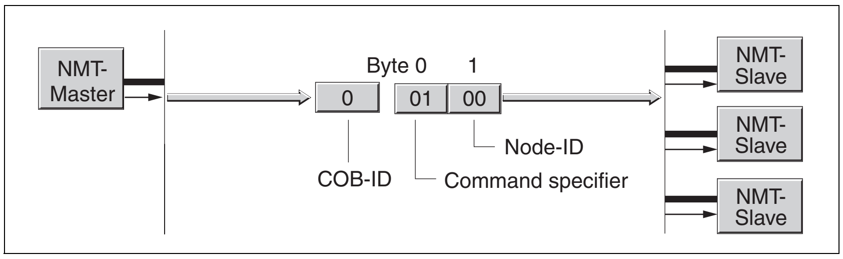


Figure 3.26 NMT message

The first byte, the "Command specifier", indicates the NMT service used.

Command Specifier	NMT service	Transition
1 (01_h)	Start remote node	A
2 (02_h)	Stop remote node	B
128 (80_h)	Enter Pre-Operational	C
129 (81_h)	Reset node	D
130 (82_h)	Reset communication	E

The second byte addresses the recipient of an NMT message with a node address between 1 and 127 ($7F_h$). A message with the node address "0" is directed to all NMT slaves.

3.6.2 NMT services for connection monitoring

Connection monitoring monitors the communication status of network devices, so a response to the failure of a device or an interruption in the network is possible.

3 NMT services for connection monitoring are available:

- "Node guarding" for monitoring the connection of an NMT slave
- "Life guarding" for monitoring the connection of an NMT master
- "Heartbeat" for unconfirmed connection messages from network devices.

3.6.2.1 Node guarding/life guarding

COB ID

Communication object `NMT error control (700`$_h$`+node-Id)` is used for connection monitoring. The COB ID for each NMT slave is determined on the basis of the node address:

COB ID = function code `NMTerror control (700`$_h$`) + node-Id`.

Structure of the NMT message

After a request from the NMT master, the NMT slave responds with one data byte.

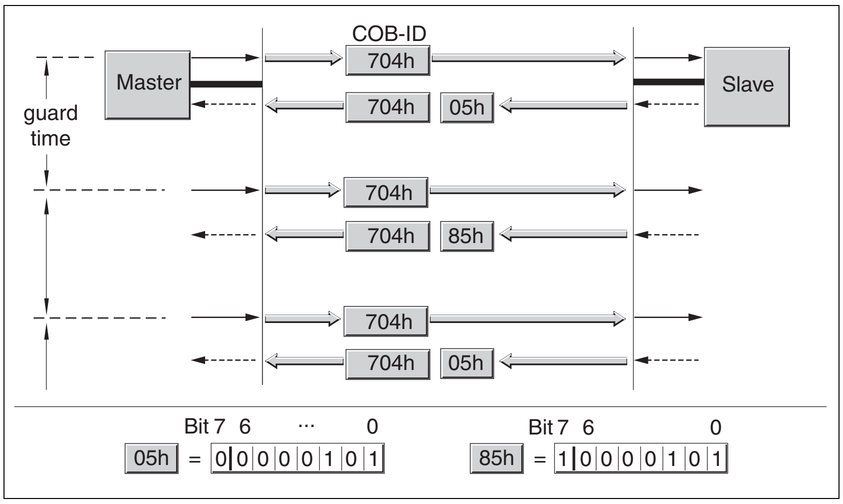


Figure 3.27 Acknowledgement of the NMT slave

Bits 0 to 6 identify the NMT state of the slave:

- 4 (04_h): "Stopped"
- 5 (05_h): "Operational"
- 127 ($7F_h$): "Pre-Operational"

After each "guard time" interval, bit 7 switches toggles between "0" and "1", so the NMT master can detect and ignore a second response within the "guard time" interval. The first request when connection monitoring is started begins with bit 7 = 0.

Connection monitoring must not be active during the initialization phase of a device. The status of bit 7 is reset as soon as the device runs though the NMT state "Reset communication".

Connection monitoring remains active in the NMT state "Stopped".

Configuration

Node guarding/life guarding is configured via:

- Guard time ($100C_h$)
- Life time factor ($100D_h$)

Connection error

The NMT master signals a connection error to the master program if:

- the slave does not respond within the "guard time" period
- the NMT state of the slave has changed without a request by the NMT master.

Figure 3.28 shows an error message after the end of the third cycle because of a missing response from an NMT slave.

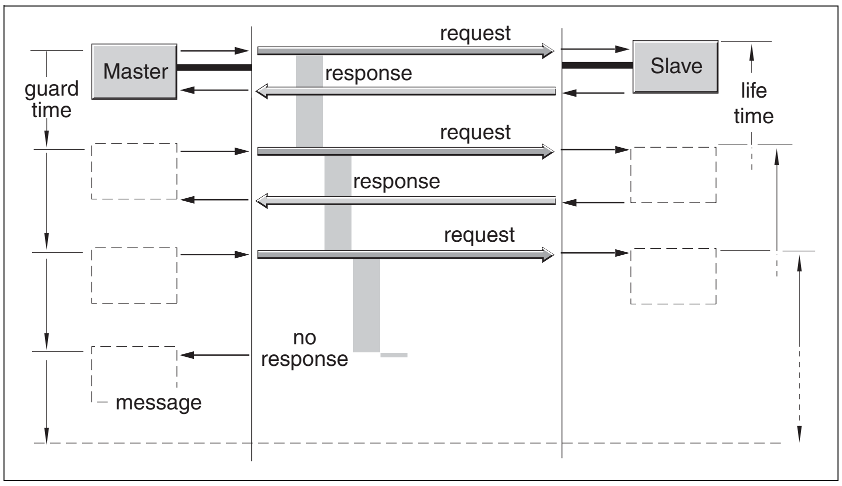


Figure 3.28 "Node Guarding" and "Life Guarding" with time intervals

3.6.2.2 Heartbeat

The optional Heartbeat protocol replaces the node guarding/life guarding protocol. It is recommended for new device versions.

A heartbeat producer transmits a heartbeat message cyclically at the frequency defined in the object `Producer heartbeat time` (`1017`$_h$). One or several consumers can receive this message. `Producer heartbeat time (1016`$_h$`) = 0` deactivates heartbeat monitoring.

The relationship between producer and consumer can be configured with objects. If a consumer does not receive a signal within the period of time set with `Consumer heartbeat time (1016`$_h$`)`, it generates an error message (heartbeat event). `Consumer heartbeat time (1016`$_h$`) = 0` deactivates monitoring by a consumer.

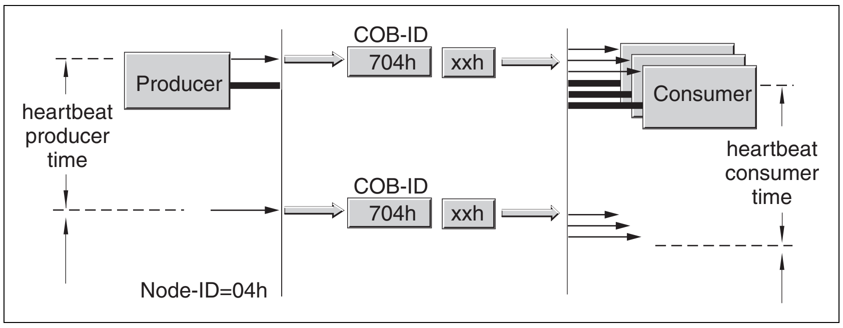


Figure 3.29 "Heartbeat" monitoring

Data byte for NMT state evaluation of the "Heartbeat" producer:

- 0 (00$_h$): "Boot-Up"
- 4 (04$_h$): "Stopped"
- 5 (05$_h$): "Operational"
- 127 (7F$_h$): "Pre-Operational"

Time intervals The time intervals are set in increments of 1 ms steps; the values for the consumer must not be less than the values for the producer. Whenever the "Heartbeat" message is received, the time interval of the producer is restarted.

Start of monitoring "Heartbeat" monitoring starts as soon as the time interval of the producer is greater than zero. If "Heartbeat" monitoring is already active during the NMT state transition to "Pre-Operational", "Heartbeat" monitoring starts with sending of the boot-up message. The boot-up message is a Heartbeat message with one data byte 00$_h$.

Devices can monitor each other via "Heartbeat" messages. They assume the function of consumer and producer at the same time.

4 Installation

⚠ WARNING

LOSS OF CONTROL

- The designer of any control scheme must consider the potential failure modes of control paths and, for certain critical functions, provide a means to achieve a safe state during and after a path failure. Examples of critical control functions are EMERGENCY STOP, overtravel stop, power outage and restart.
- Separate or redundant control paths must be provided for critical functions.
- System control paths may include communication links. Consideration must be given to the implication of unanticipated transmission delays or failures of the link.
- Observe the accident prevention regulations and local safety guidelines. 1)
- Each implementation of the product must be individually and thoroughly tested for proper operation before being placed into service.

Failure to follow these instructions can result in death or serious injury.

1) For USA: Additional information, refer to NEMA ICS 1.1 (latest edition), Safety Guidelines for the Application, Installation, and Maintenance of Solid State Control and to NEMA ICS 7.1 (latest edition), Safety Standards for Construction and Guide for Selection, Installation for Construction and Operation of Adjustable-Speed Drive Systems.

⚠ WARNING

SIGNAL AND DEVICE INTERFERENCE

Signal interference can cause unexpected responses of device.

- Install the wiring in accordance with the EMC requirements.
- Verify compliance with the EMC requirements.

Failure to follow these instructions can result in death, serious injury or equipment damage.

For information on installation of the device and connecting the device to the fieldbus see the product manual.

0198441113718, V2.01, 11.2008

5 Commissioning

5.1 Commissioning the device

For installation in the network, the device must first be properly installed (mechanically and electrically) and commissioned.

Commission the device as per product manual. This prepares the device for operation in the network.

5.2 Address and baud rate

Up to 32 devices can be addressed in one CAN bus network branch and up to 127 devices in the extended network. Each device is identified by a unique address. The default node address for a device is 127.

The default baud rate is 125 kbaud.

Each device must be assigned its own node address, i.e. any given node address may be assigned only once in the network.

Setting address and baud rate

The address is set directly at the device via parameter `canAddr` and the baud rate via parameter `canBaud`.

The baud rate must be the same for all devices in the fieldbus.

5.3 SyCon CANopen configuration software

The CANopen network can be configured with the "SyCon" configuration software. An additional EDS file is included in the SYCON subdirectory on the product CD.

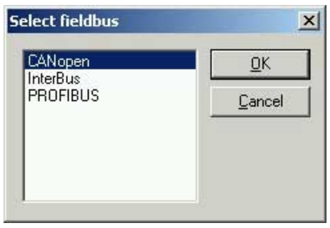
▶ Procedure:

5.3.1 Creating a new network

Create a new network via the menu item "File - New".

▶ Select CANopen as the fieldbus network.

▶ Confirm your selection with "OK".

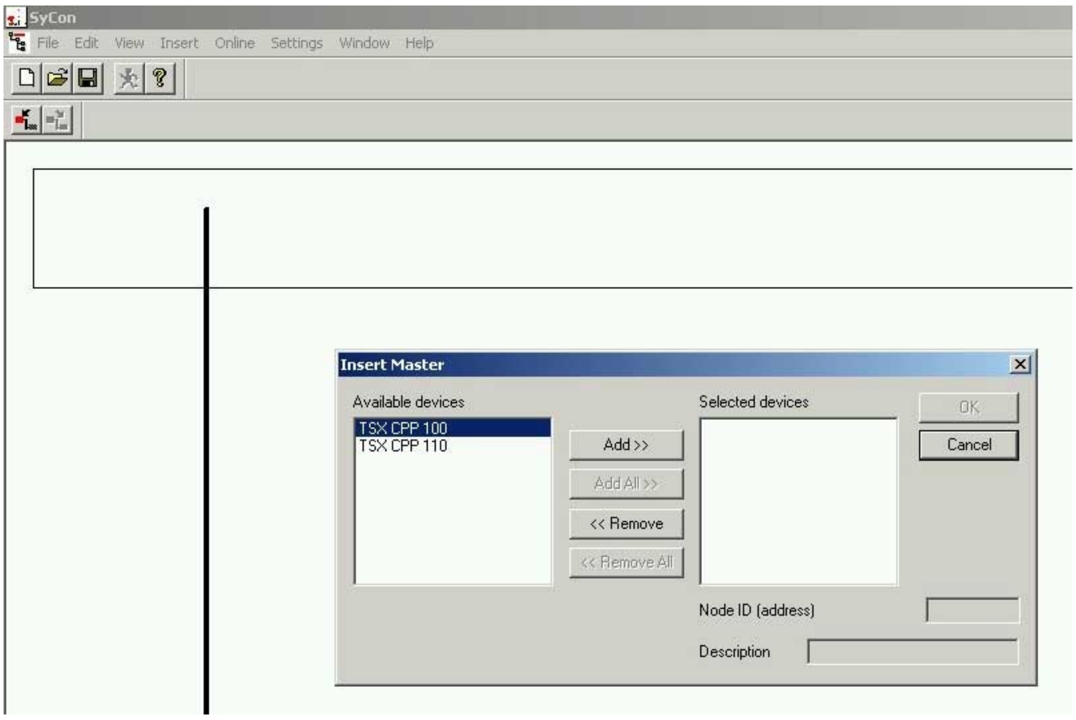


5.3.2 Selecting the CANopen master

Use the menu item "Insert - Master" to select the network master. The screenshot shows the example of a TSX CCP 110 board of a Premium PLC.

The node ID and a brief description can be entered in the appropriate fields.

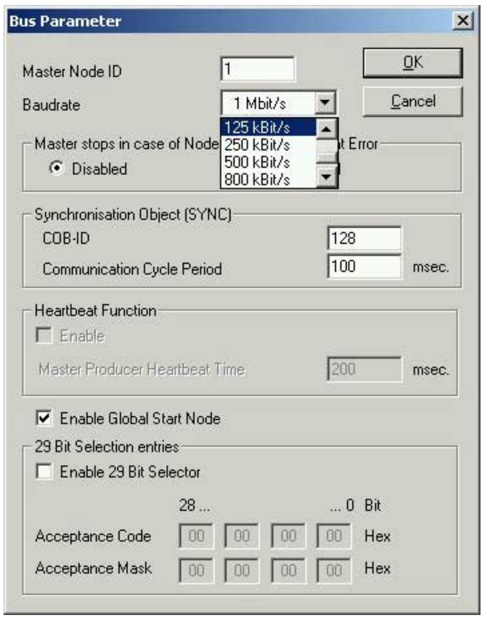
▶ Confirm your selection with "OK".



5.3.3 Setting the bus parameters

The menu item "Settings - Bus Parameter..." allows you to set the CANopen communication parameters. Please also consult the operating instructions of the SyCon configuration software.

▶ Confirm your selection with "OK".



Master Node ID: 1

Baudrate: 1 Mbit/s

Master stops in case of Node ... t Error: [✓ Disabled]

Synchronisation Object (SYNC)
COB-ID: 128
Communication Cycle Period: 100 msec.

Heartbeat Function
☐ Enable
Master Producer Heartbeat Time: 200 msec.

☑ Enable Global Start Node

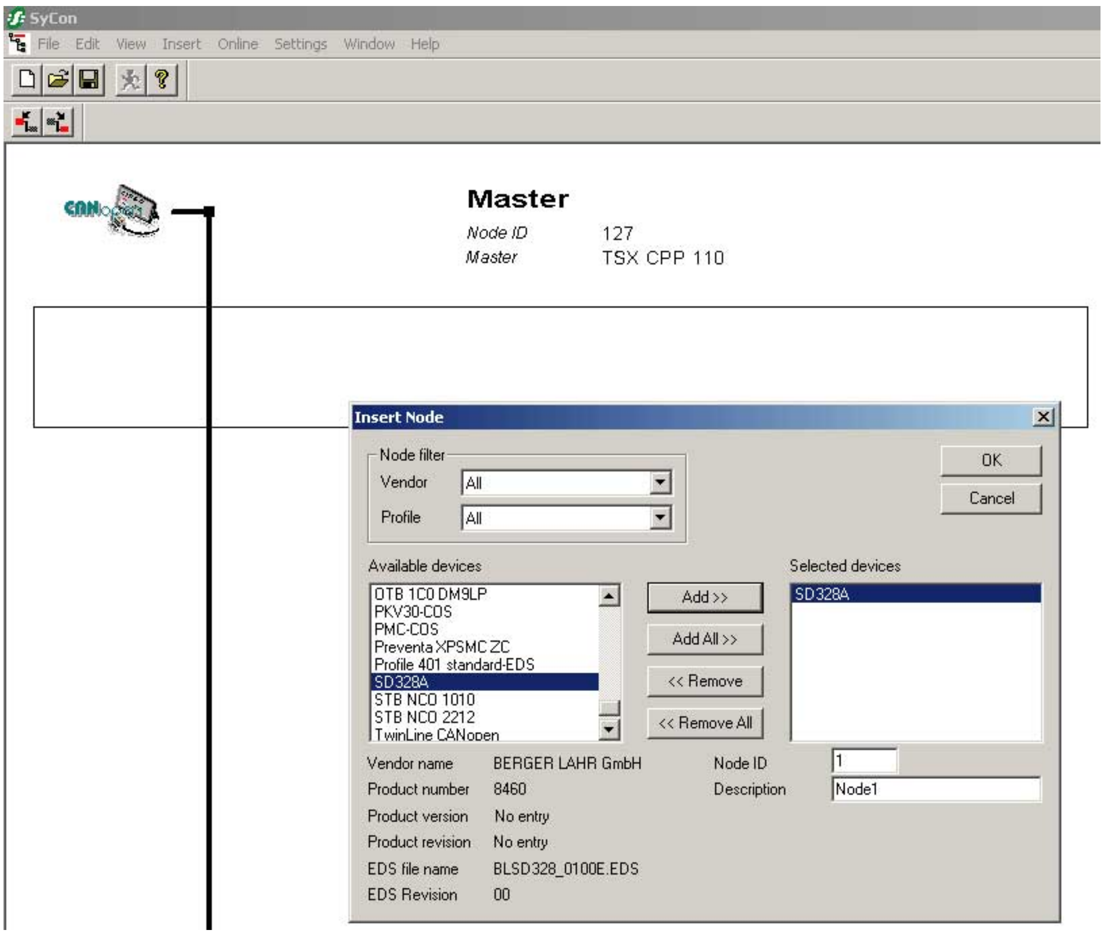
29 Bit Selection entries
☐ Enable 29 Bit Selector

	28 ...			... 0	Bit
Acceptance Code	00	00	00	00	Hex
Acceptance Mask	00	00	00	00	Hex

5.3.4 Selecting and inserting nodes

Use the menu item "Insert - Node" to select the network nodes. The example shows a SD328A.

▶ Confirm your selection with "OK".



Master
Node ID 127
Master TSX CPP 110

Node filter
Vendor All
Profile All

Available devices
OTB 1C0 DM9LP
PKV30-COS
PMC-COS
Preventa XPSMC ZC
Profile 401 standard-EDS
SD328A
STB NCO 1010
STB NCO 2212
TwinLine CANopen

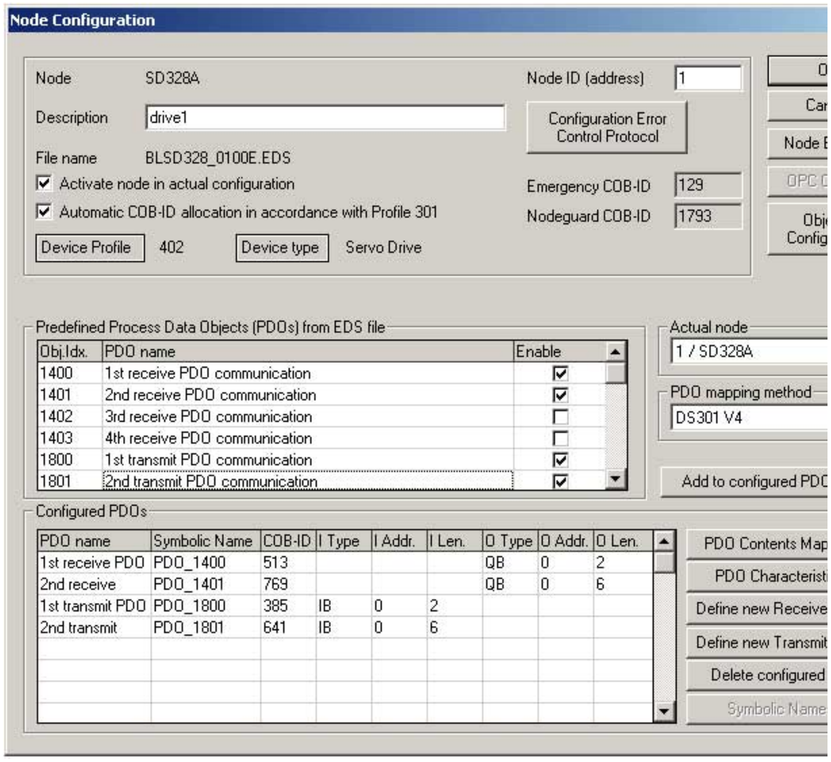
Selected devices
SD328A

Vendor name	BERGER LAHR GmbH
Product number	8460
Product version	No entry
Product revision	No entry
EDS file name	BLSD328_0100E.EDS
EDS Revision	00
Node ID	1
Description	Node1

5.3.5 Configuring the network node

Double-click the network node to open the Node Configuration dialog. This dialog lets you to set the communication properties of the selected node.

This primarily relates to the PDO characteristics of the configurable PDO4.



Node SD328A
Description drive1
File name BLSD328_0100E.EDS
☑ Activate node in actual configuration
☑ Automatic COB-ID allocation in accordance with Profile 301
Device Profile 402 Device type Servo Drive
Node ID (address) 1
Emergency COB-ID 129
Nodeguard COB-ID 1793

Predefined Process Data Objects (PDOs) from EDS file

Obj.Idx.	PDO name	Enable
1400	1st receive PDO communication	☑
1401	2nd receive PDO communication	☑
1402	3rd receive PDO communication	☐
1403	4th receive PDO communication	☐
1800	1st transmit PDO communication	☑
1801	2nd transmit PDO communication	☑

Actual node 1 / SD328A
PDO mapping method DS301 V4

Configured PDOs

PDO name	Symbolic Name	COB-ID	I Type	I Addr.	I Len.	O Type	O Addr.	O Len.
1st receive PDO	PDO_1400	513				QB	0	2
2nd receive	PDO_1401	769				QB	0	6
1st transmit PDO	PDO_1800	385	IB	0	2			
2nd transmit	PDO_1801	641	IB	0	6			

The error response of the node can be set with the "Configuration Error Control Protocol" button. You can select whether to monitor the node with the node guarding protocol or the heartbeat protocol.

▶ Confirm your selection with "OK".

Double-click the object 1403 "4th receive PDO communication" or the object 1803 "4th transmit PDO communication" to open a dialog box in which the transmission properties of the PDO can be set. The default values can be used without changes.

▶ Confirm with "OK".

Use the "PDO Contents Mapping" button to set the mapping of the PDO4.

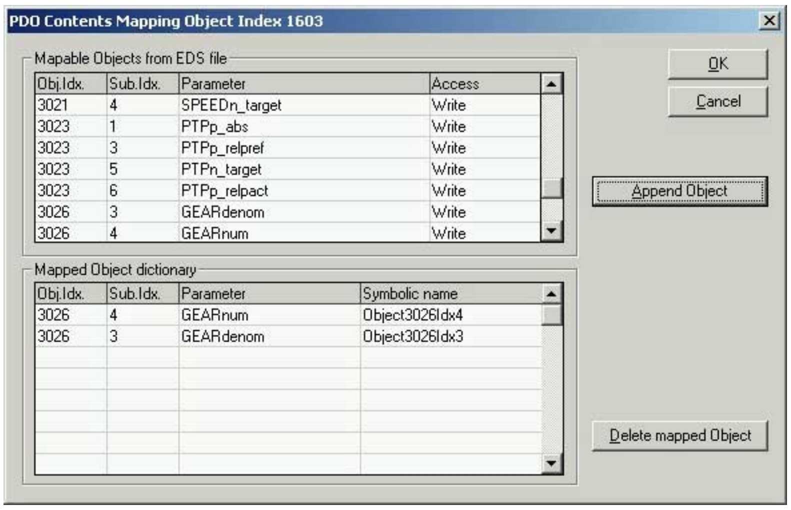
0198441113718, V2.01, 11.2008

5.3.6 Setting the mapping of the PDO4

Up to 4 objects each can be parameterized for the receive PDO4 and the transmit PDO4 via the "Append Object" button. Please note that the maximum number of 8 must not be exceeded.

In the example, a numerator and a denominator for the gear ratio (32 bits each) are assigned to PDO4.

▶ Confirm your selection with "OK".



Mapable Objects from EDS file

Obj.Idx.	Sub.Idx.	Parameter	Access
3021	4	SPEEDn_target	Write
3023	1	PTPp_abs	Write
3023	3	PTPp_relpref	Write
3023	5	PTPn_target	Write
3023	6	PTPp_relpact	Write
3026	3	GEARdenom	Write
3026	4	GEARnum	Write

Mapped Object dictionary

Obj.Idx.	Sub.Idx.	Parameter	Symbolic name
3026	4	GEARnum	Object3026Idx4
3026	3	GEARdenom	Object3026Idx3

Save the configuration via the menu item "File - Save as..".

The PLC programming software "Unity" or "PL7" can continue to use the configuration after parameterization.

0198441113718, V2.01, 11.2008

6 Operation

⚠ WARNING

UNINTENDED OPERATION

- Do not write values to reserved parameters.
- Do not write values to parameters unless you fully understand the function. For more information see the product manual.
- Run initial tests without coupled loads.
- Verify that the system is free and ready for the movement before changing parameters.
- Verify the use of the bits with fieldbus communication: bit 0 is far right (least significant). Bit 15 is far left (most significant).
- Verify the use of the word sequence with fieldbus communication.
- Do not establish a fieldbus connection unless you have fully understood all communications principles.

Failure to follow these instructions can result in death, serious injury or equipment damage.

6.1 Operating modes

Local control mode and fieldbus control mode

In a local control mode, the reference values for movements are supplied in the form of analog signals (±10V) or RS422 signals (e.g. pulse/direction).

In fieldbus control mode, the reference values are supplied via fieldbus commands.

For detailed information on setting the control mode, see the chapter Commissioning in the product manual.

Standardized operating modes

The device operates in 3 standardized operating modes. They can be started and monitored with the objects of the CANopen device profile DSP402.

- Profile position
  via the objects of object group `Profile position mode`
- Profile Velocity
  via the objects of object group `Profile velocity mode`
- Homing
  via the objects of object group `Homing mode`

Vendor-specific operating modes

The device operates in 3 manufacturer-specific operating modes. The operating modes use the full functionality of the devices. The operating modes are started and monitored with manufacturer-specific objects.

- Oscillator mode
- Electronic gear
- Jog

6.2 Standardized operating modes

6.2.1 Operating mode Profile Position

PDO2 must be activated for the Profile Position operating mode. After the activation, motion parameters such as ramps and speeds can be set.

Example Node address 1

Work step **COB ID / data**	**Object** **Value**
▶ Enable R_PDO2	1401:1$_h$
`601 / 23 01 14 01 01 03 00 04`	0400 0301$_h$
◁ `581 / 60 01 14 01 00 00 00 00`	
▶ Enable T_PDO2	1801:1$_h$
`601 / 23 01 18 01 81 02 00 04`	0400 0281$_h$
◁ `581 / 60 01 18 01 00 00 00 00`	
▶ Set acceleration ramp to 2000 min^{-1}*s	6083$_h$
`601 / 23 83 60 00 D0 07 00 00`	0000 07D0$_h$
◁ `581 / 60 83 60 00 00 00 00 00`	
▶ Set deceleration ramp to 4000 min^{-1}*s	6084$_h$
`601 / 23 84 60 00 A0 0F 00 00`	0000 0FA0$_h$
◁ `581 / 60 84 60 00 00 00 00 00`	
▶ Limit reference speed of rotation to 6000 min^{-1}	607F$_h$
`601 / 23 7F 60 00 70 17 00 00`	0000 1770$_h$
◁ `581 / 60 7F 60 00 00 00 00 00`	
▶ Set reference speed of rotation to 4000 min^{-1}	6081$_h$
`601 / 23 81 60 00 A0 0F 00 00`	0000 0FA0$_h$
◁ `581 / 60 81 60 00 00 00 00 00`	
▶ NMT Start remote node	
`0 / 01 00`	
◁ T_PDO1 with status word	
`181 / 31 66`	
▶ Enable power stage with PDO1	
`201 / 00 00`	
`201 / 06 00`	
`201 / 0F 00`	
◁ T_PDO1 (state: Operation enabled)	
`181 / 37 46`	
▶ Start operating mode	6060$_h$
`601 / 2F 60 60 00 01 00 00 00`	01$_h$
◁ `581 / 60 60 60 00 00 00 00 00`	
▶ Check operating state [1]	6061$_h$
`601 / 40 61 60 00 00 00 00 00`	
◁ Operating mode active	
`581 / 4F 61 60 00 01 00 01 00`	01$_h$

Work step COB ID / data	Object Value
▶ PDO2: Set relative position with NewSetpoint=1 `301 / 5F 00 30 75 00 00`	
◁ T_PDO2 with status word and position actual value `281 / 37 56 00 00 00 00`	
◁ Position reached `281 / 37 56 30 75 00 00`	
▶ PDO2: NewSetpoint=0 `301 / 4F 00 30 75 00 00`	

1) The operating state must be checked until the device has activated the specified operating mode.

6.2.2 Operating mode Profile velocity

PDO3 must be activated for the Profile Velocity operating mode.

Example Node address 1

Work step COB ID / data	Object Value
▶ Activate R_PDO3 `601 / 23 02 14 01 01 04 00 04`	1402:1$_h$ 0400 0401$_h$
◁ `581 / 60 02 14 01 00 00 00 00`	
▶ Activate T_PDO3 `601 / 23 02 18 01 81 03 00 04`	1802:1$_h$ 0400 0381$_h$
◁ `581 / 60 02 18 01 00 00 00 00`	
▶ Set acceleration ramp to 2000 min^{-1}*s `601 / 23 83 60 00 D0 07 00 00`	6083$_h$ 0000 07D0$_h$
◁ `581 / 60 83 60 00 00 00 00 00`	
▶ Set deceleration ramp to 10000 min^{-1}*s `601 / 23 84 60 00 10 27 00 00`	6084$_h$ 0000 2710$_h$
◁ `581 / 60 84 60 00 00 00 00 00`	
▶ Limit reference speed of rotation to 10000 min^{-1} `601 / 23 7F 60 00 10 27 00 00`	607F$_h$ 0000 2710$_h$
◁ `581 / 60 7F 60 00 00 00 00 00`	
▶ NMT Start remote node `0 / 01 00`	
◁ T_PDO1 with status word `181 / 31 66`	

▶ Enable power stage with PDO1
`201 / 00 00`
`201 / 06 00`
`201 / 0F 00`	

◁ T_PDO1 (state: Operation enabled) `181 / 37 46`	
▶ Start operating mode `601 / 2F 60 60 00 03 00 00 00`	6060$_h$ 03$_h$
◁ `581 / 60 60 60 00 00 00 00 00`	
▶ Check operating state [1] `601 / 40 61 60 00 00 00 00 00`	6061$_h$
◁ Operating mode active `581 / 4F 61 60 00 03 00 01 00`	03$_h$
▶ PDO3: Transmit reference speed 1000 min^{-1} `401 / 0F 00 E8 03 00 00`	
◁ T_PDO2 with status word and velocity actual value `381 / 37 02 00 00 00 00`	
◁ Reference speed reached `381 / 37 06 E8 03 00 00`	

1) The operating state must be checked until the device has activated the specified operating mode.

6.2.3 Operating mode Homing

The Homing operating mode is parameterized with SDOs and activated with PDO1.

Example Node address 1

Work step **COB ID / data**	**Object** **Value**
▶ Reference speed for movement to limit switch 100 min^{-1} `601 / 23 99 60 01 64 00 00 00`	$6099{:}1_h$ $0000\ 0064_h$
◁ `581 / 60 99 60 01 00 00 00 00`	
▶ Reference speed for free movement 10 min^{-1} `601 / 23 99 60 02 0A 00 00 00`	$6099{:}2_h$ $0000\ 000A_h$
◁ `581 / 60 99 60 02 00 00 00 00`	
▶ NMT Start remote node `  0 / 01 00`	
◁ T_PDO1 with status word `181 / 31 66`	

▶ Enable power stage with PDO1
`201 / 00 00`
`201 / 06 00`
`201 / 0F 00`	

◁ T_PDO1 (state: Operation enabled) `181 / 37 46`	
▶ Start operating mode `601 / 2F 60 60 00 06 00 00 00`	6060_h 06_h
◁ `581 / 60 60 60 00 00 00 00 00`	
▶ Check operating state [1)] `601 / 40 61 60 00 00 00 00 00`	6061_h
◁ Operating mode active `581 / 4F 61 60 00 06 00 01 00`	06_h
▶ Select reference movement method, LimN (17) `601 / 2F 98 60 00 11 00 00 00`	6098_h 11_h
◁ `581 / 60 98 60 00 00 00 00 00`	
▶ Reference movement with PDO1 (homing operation start) `201 / 1F 00`	
◁ TPDO1 Reference movement active `181 / 37 02`	
◁ TPDO1 Reference movement complete `181 / 37 D6`	

1) The operating state must be checked until the device has activated the specified operating mode.

0198441113718, V2.01, 11.2008

6.3 Vendor-specific operating modes

6.3.1 Operating mode Oscillator

Example Node address 1.

Work step COB ID / data	Object Value
▶ R_PDO4 Mapping: Number of mapped objects = 0 `601 / 2F 03 16 00 00 00 00 00` ◁ `581 / 60 03 16 00 00 00 00 00`	1603:0_h 00_h
▶ R_PDO4 first parameter = SPEEDn_target (3021:4_h) `601 / 23 03 16 01 10 04 21 30` ◁ `581 / 60 03 16 01 00 00 00 00`	1603:1_h 3021 0410_h
▶ R_PDO4 Number of mapped objects = 1 `601 / 2F 03 16 00 01 00 00 00` ◁ `581 / 60 03 16 00 00 00 00 00`	1603:0_h 01_h
▶ T_PDO4 Mapping: Number of mapped objects = 0 `601 / 2F 03 1A 00 00 00 00 00` ◁ `581 / 60 03 1A 00 00 00 00 00`	1A03:00_h 00_h
▶ T_PDO4 first parameter = _p_actusr (6064:0) `601 / 23 03 1A 01 20 00 64 60` ◁ `581 / 60 03 1A 01 00 00 00 00`	1A03:1_h 6064 0020_h
▶ T_PDO4 Number of mapped objects = 1 `601 / 2F 03 1A 00 01 00 00 00` ◁ `581 / 60 03 1A 00 00 00 00 00`	1A03:00_h 01_h
▶ Enable R_PDO4 (COB ID) `601 / 23 03 14 01 01 05 00 04` ◁ `581 / 60 03 14 01 00 00 00 00`	1403:1_h 0400 0501_h
▶ Enable T_PDO4 (COB ID) `601 / 23 03 18 01 81 04 00 04` ◁ `581 / 60 03 18 01 00 00 00 00`	1803:1_h 0400 0481_h

▶ NMT Start remote node
`  0 / 01 00`
◁ T_PDO1 with status word
`181 / 31 66`	

▶ Enable power stage with PDO1
`201 / 00 00`
`201 / 06 00`
`201 / 0F 00`
◁ T_PDO1 (state: Operation enabled)
`181 / 37 46`	

▶ Start operating mode `601 / 2F 60 60 00 FC 00 00 00` ◁ `581 / 60 60 60 00 00 00 00 00`	6060_h -04_h

0198441113718, V2.01, 11.2008

Work step COB ID / data	Object Value
▶ Check operating state [1)] `601 / 40 61 60 00 00 00 00 00`	6061$_h$
◁ Operating mode active `581 / 4F 61 60 00 FC 00 01 00`	-04$_h$
▶ Set reference value via parameter `601 / 2B 1B 30 11 02 00 00 00`	301B:11$_h$ 02$_h$
◁ `581 / 60 1B 30 11 00 00 00 00`	
▶ PDO4 transmit reference speed 1000 min^{-1} `501 / E8 03`	
◁ T_PDO4 with current position `481 / 6E 97 04 00`	

1) The operating state must be checked until the device has activated the specified operating mode.

6.3.2 Operating mode Electronic Gear

Example Node address 1

Work step **COB ID / data**	**Object** **Value**
▶ R_PDO4 Mapping: Number of mapped objects = 0 `601 / 2F 03 16 00 00 00 00 00`	$1603:0_h$ 00_h
◁ `581 / 60 03 16 00 00 00 00 00`	
▶ R_PDO4 first parameter = GEARnum ($3026:4_h$) `601 / 23 03 16 01 20 04 26 30`	$1603:1_h$ $3026\ 0420_h$
◁ `581 / 60 03 16 01 00 00 00 00`	
▶ R_PDO4 first parameter = GEARdenom ($3026:3_h$) `601 / 23 03 16 02 20 03 26 30`	$1603:2_h$ $3026\ 0320_h$
◁ `581 / 60 03 16 02 00 00 00 00`	
▶ R_PDO4 Number of mapped objects = 2 `601 / 2F 03 16 00 02 00 00 00`	$1603:0_h$ 02_h
◁ `581 / 60 03 16 00 00 00 00 00`	
▶ T_PDO4 Mapping: Number of mapped objects = 0 `601 / 2F 03 1A 00 00 00 00 00`	$1A03:00_h$ 00_h
◁ `581 / 60 03 1A 00 00 00 00 00`	
▶ T_PDO4 first parameter = _p_actusr (6064:0) `601 / 23 03 1A 01 20 00 64 60`	$1A03:1_h$ $6064\ 0020_h$
◁ `581 / 60 03 1A 01 00 00 00 00`	
▶ T_PDO4 Number of mapped objects = 1 `601 / 2F 03 1A 00 01 00 00 00`	$1A03:00_h$ 01_h
◁ `581 / 60 03 1A 00 00 00 00 00`	
▶ Enable R_PDO4 (COB ID) `601 / 23 03 14 01 01 05 00 04`	$1403:1_h$ $0400\ 0501_h$
◁ `581 / 60 03 14 01 00 00 00 00`	
▶ Enable T_PDO4 (COB ID) `601 / 23 03 18 01 81 04 00 04`	$1803:1_h$ $0400\ 0481_h$
◁ `581 / 30 03 18 01 00 00 00 00`	
▶ Signal selection position interface `601 / 2B 05 30 02 01 00 00 00`	$3005:2_h$ 01_h
◁ `581 / 60 05 30 02 00 00 00 00`	
▶ Activate gear with immediate synchronization `601 / 2B 1B 30 12 01 00 00 00`	$301B:12_h$ 01_h
◁ `581 / 60 1B 30 12 00 00 00 00`	
▶ NMT Start remote node `  0 / 01 00`	
◁ T_PDO1 with status word `181 / 31 66`	

0198441113718, V2.01, 11.2008

Work step **COB ID / data**	**Object** **Value**

▶ Enable power stage with PDO1
`201 / 00 00`
`201 / 06 00`
`201 / 0F 00`	

◁ T_PDO1 (state: Operation enabled) `181 / 37 46`	
▶ Start operating mode `601 / 2F 60 60 00 FE 00 00 00`	6060_h -02_h
◁ `581 / 60 60 60 00 00 00 00 00`	
▶ Check operating state 1) `601 / 40 61 60 00 00 00 00 00`	6061_h
◁ Operating mode active `581 / 4F 61 60 00 FE 00 01 00`	-02_h
▶ PDO4 transmit gear ratio 2/3 `501 / 02 00 00 00 03 00 00 00`	
◁ T_PDO4 with current position `481 / 53 76 01 00`	

1) The operating state must be checked until the device has activated the specified operating mode.

6.3.3 Operating mode Jog

Example Node address 1

Work step COB ID / data	Object Value
▶ Speed of rotation slow movement to 100 min^{-1} `601 / 2B 29 30 04 64 00 00 00`	3029:4_h 0064_h
◁ `581 / 60 29 30 04 00 00 00 00`	
▶ Speed of rotation fast movement to 250 min^{-1} `601 / 2B 29 30 05 FA 00 00 00`	3029:5_h $00FA_h$
◁ `581 / 60 29 30 05 00 00 00 00`	
▶ NMT Start remote node `0 / 01 00`	
◁ T_PDO1 with status word `181 / 31 66`	

▶ Enable power stage with PDO1
`201 / 00 00`
`201 / 06 00`
`201 / 0F 00`	

◁ T_PDO1 (state: Operation enabled) `181 / 37 46`	
▶ Start operating mode `601 / 2F 60 60 00 FF 00 00 00`	6060_h -01_h
◁ `581 / 60 60 60 00 00 00 00 00`	
▶ Check operating state [1)] `601 / 40 61 60 00 00 00 00 00`	6061_h
◁ Operating mode active `581 / 4F 61 60 00 FF 00 01 00`	-01_h
▶ Jog (clockwise rotation, slow) `601 / 2B 1B 30 09 01 00 00 00`	301B:9_h 01_h
◁ `581 / 60 1B 30 09 00 00 00 00`	
◁ T_PDO1 with status word`181 / 37 02`	
▶ Jog (clockwise rotation, fast) `601 / 2B 1B 30 09 05 00 00 00`	301B:9_h 05_h
◁ `581 / 60 1B 30 09 00 00 00 00`	
◁ T_PDO1 with status word `181 / 37 42`	

1) The operating state must be checked until the device has activated the specified operating mode.

6.4 Functions

6.4.1 Ramp function

The device controls the acceleration and deceleration behavior of the motor with ramp functions. Steepness and shape of the ramp constitute the ramp function. The ramp steepness determines the motor's change of speed, the shape of the ramp the acceleration over time.

For details on the ramp function see the chapter on the functions in the product manual.

Object (Index:Subindex)	Meaning
Profile acceleration (6083$_h$)	Acceleration [usr]
Profile deceleration (6084$_h$)	Deceleration [usr]
Max profile velocity (607F$_h$)	Limitation of reference speed of rotation

6.4.2 Quick Stop function

Function principle

"Quick Stop" is a quick brake function which stops the motor as a result of an error of error classes 1 and 2 or as a result of a software stop.

In the event of an error response to an error of error class 1, the power stage remains enabled. In the case of error class 2, the power stage is disabled after the drive has come to a standstill.

For details on the Quick Stop function see the chapter on the functions in the product manual.

The motor can be decelerated via the Quick Stop current, object `LIM_I_maxQSTP (3011:5`$_h$`)`, via the Halt current `LIM_I_maxHalt (3011:6`$_h$`)` or via the deceleration ramp of the motion profile, object `Profile deceleration (6084`$_h$`)`.

Object (Index:Subindex)	Meaning
LIM_I_maxQSTP (3011:5$_h$)	Current limitation for Quick Stop [0.01A]
LIM_I_maxHalt (3011:6$_h$)	Current limitation for Halt [0.01A]
Profile deceleration (6084$_h$)	Deceleration ramp of the motion profile

6.4.3 Motor stop

The drive can be stopped via the fieldbus during execution of a motion command. If bit 8 in object `Controlword (6040`$_h$`)` changes to "1", the device decelerates the motor with the deceleration ramp set for the motion command. The movement and position data is retained.

The execution of the motion command is continued as soon as bit 8 changes back to "0".

6.4.4 Standstill window

If the motor is held at zero speed of rotation with the controller active, minimum variations of the speed of rotation interfere with proper detection of the motor standstill. If the motor remains in the standstill window for an adjustable period, the controller signals motor standstill. Bit 10 in the status word, object `Statusword` (6041_h), is set.

For details on the standstill window see the chapter on the functions in the product manual.

Object (Index:Subindex)	Meaning
Position window (6067_h)	Standstill window, permissible control deviation
Position window time (6068_h)	Period of time during which control deviations must be in the standstill window for standstill to be signaled [ms]

6.4.5 Reversal of direction of rotation

The direction of rotation of the drive can be reversed with the object `POSdirOfRotat` ($3006:12_h$). The limit switch connections must be reversed as well.

For details on reversing the direction of rotation see the chapter on the functions in the product manual.

Object (Index:Subindex)	Meaning
POSdirOfRotat ($3006:12_h$)	Reversal of direction of rotation

6.4.6 Monitoring functions

Positioning limits

The motor can be moved to any point on the axis within the positioning range by means of absolute positioning.

The movement range of the axis is specified in internal units in the range from -2^{28} to $+2^{28}$ increments. The internal unit used is the resolution of the motor encoder in increments.

In the case of a position overrun, bit 15 (REF_OK) of the object `Statusword` (6041_h) is set to 0.

Software limit switch

The software limit switch position is specified with the object `Max position limit (607D`$_h$`)` of the DSP402 device profile.

The determining factor for position monitoring of the software limit switch range is the reference position of the position controller. Therefore, depending on the controller settings, the motor may stop before it reaches the limit switch position. Bit 2 of the object `_SigLatched (301C:8`$_h$`)` signals overtraveling of the limit switch position.

Object (Index:Subindex)	Meaning
Min position limit (607D:1$_h$)	Negative position limit for software limit switch
Max position limit (607D:2$_h$)	Positive position limit for software limit switch
_SigLatched (301C:8$_h$)	Monitoring signal bit 2 SW_LimP / SW_LimN

Limit switch signal

During movements the two limit switches are monitored via the input signals $\overline{\mathtt{LIMN}}$ and $\overline{\mathtt{LIMP}}$. When the limit switch is overtraveled the motor is stopped. Bit 1 of the object `_SigLatched (301C:8`$_h$`)` signals overtraveling of the limit switches.

Object (Index:Subindex)	Meaning
IOsigLimN (3006:F$_h$)	0: Inactive 1: Normally closed contact 2: Normally open contact
IOsigLimN (3006:10$_h$)	0: Inactive 1: Normally closed contact 2: Normally open contact
_SigLatched (301C:8$_h$)	Bit 1: $\overline{\mathtt{LimP}}$ / $\overline{\mathtt{LimN}}$

Tracking error monitoring

Tracking error monitoring checks for deviations of the actual motor position from the reference motor position. If the difference exceeds a limit value, the device signals an error. The limit value for the position deviation is adjustable.

Object (Index:Subindex)	Meaning
Tracking error window (following error window) (6065$_h$)	Maximum permissible position deviation of the position controller [Inc]

Monitoring parameters

The device status and operating state can be monitored by means of various objects.

Object (Index:Subindex)	Meaning
_SigActive (301C:7$_h$)	Current status of monitoring signals
_SigLatched (301C:8$_h$)	Saved status of monitoring signals
_WarnActive (301C:B$_h$)	Active warnings, bit-coded
_WarnLatched (301C:C$_h$)	Saved warnings, bit-coded
_actionStatus (301C:4$_h$)	Action word
Error code (603F$_h$)	Cause of last stop

6.4.7 Monitoring inputs and outputs of the device

The analog signal and the digital signals of the device can be monitored via the fieldbus.

The analog input ANA1 is monitored via the object `ANA1_act` (`3009:1`$_h$). The digital inputs are monitored via the object `_IO_act` (`3008:1`$_h$). For example, this allows you to monitor the start of a jog movement via interface signals via the fieldbus.

Object (Index:Subindex)	Meaning
ANA1_act (3009:1$_h$)	Monitoring analog input ANA1
_IO_act (3008:1$_h$)	Monitoring digital inputs and outputs [mV]

6.4.7.1 Saving and restoring object data

The device copies persistent object data to the RAM memory after the device is switched on. The device works with the data in RAM during operation.

In order to avoid data loss caused by power outage, the data must be saved to the persistent memory with the object `PAReeprSave` (`3004:1`$_h$).

User-specific object settings can be reset with the object `PARusrReset` (`3004:4`$_h$).

If resetting is triggered with object `Restore Default Parameters 1011`$_h$, object `PARusrReset (3004:4`$_h$`)` assumes the value `1`. As soon as resetting is complete, the value changes back to `0`.

Object (Index:Subindex)	Meaning
PAReeprSave (3004:1$_h$)	Saving object settings to the EEPROM
PARusrReset (3004:8$_h$)	Restoring object settings

7 Diagnostics and troubleshooting

7.1 Fieldbus communication error diagnostics

A properly operating fieldbus is essential for evaluating operating and error messages.

Connections for fieldbus mode

If the product cannot be addressed via the fieldbus, first check the connections. The product manual contains the technical data of the device and information on network and device installation. Check the following:

- $24V_{DC}$ power supply
- Power connections to the device
- Fieldbus cable and fieldbus wiring
- Network connection to the device

You can also use the commissioning software for troubleshooting.

Baud rate and address

If it is impossible to connect to a device, check the baud rate and node address.

- The baud rate must be the same for all devices in the network.
- The node address of each device must be between 1 and 127 and unique for each device.

To set the baud rate and node address see chapter 5.2 "Address and baud rate".

Fieldbus function test

After correct configuration of the transmission data, test fieldbus mode. This requires installation of a CAN configuration tool that displays CAN messages. Feedback from the product is indicated by a boot-up message:

- Switch the power supply off and on again.
- Observe the network messages after switching on. After initialization of the bus, the device sends a boot-up message (COB ID 700_h + node ID and 1 data byte with the content 00_h).
- With the factory setting 127 ($7F_h$) for the node address, the boot-up message is sent via the bus . The device can then be put into operation via NMT services.

If network operation cannot be started, the network function of the device must be checked by your local representative. Contact your local sales representative.

7.2 Error diagnostics via fieldbus

7.2.1 Message objects

A number of objects provide information on the operating and error state:

- Object `Statusword (6041`$_h$`)`
  Operating states, see product manual
- Object `EMCY (80`$_h$`+ Node-ID)`
  Error message from a device with fault state and error code, see chapter 3.5 "Emergency service"
- Object `Error register (1001`$_h$`)`
  Fault state
- Object `Error code (603F`$_h$`)`
  Error code of the most recent error
- Devices use the special SDO error message ABORT to signal errors in exchanging messages by SDO.

7.2.2 Messages on the device status

Synchronous and asynchronous errors are distinguished in the evaluation and handling of errors.

Synchronous errors The device signals a synchronous error directly as a response to a message that cannot be evaluated. Possible causes comprise transmission errors or invalid data. For a list of synchronous errors see chapter 7.3.1 "Error register".

Asynchronous errors Asynchronous errors are signaled by the monitoring units in the device as soon as a device fault occurs. An asynchronous error is signal via bit 3, "Fault", of the object `statusword (6041`$_h$`)`. In the case of errors that cause a an interruption of the movement, the device transmits an EMCY message.

Asynchronous errors are also reported via bits 5..7 of the object `driveStat (2041`$_h$`)`.

7.3 CANopen error messages

CANopen error messages are signaled in the form of EMCY messages. They are evaluated via the objects `Error register (1001`$_h$`)` and `Error code (603F`$_h$`)`. For information on the object `EMCY` see chapter 3.5 "Emergency service".

CANopen signals errors that occur during data exchange via SDO with the special SDO error message ABORT.

7.3.1 Error register

The object `Error register(1001`$_h$`)` indicates the error state of a device in bit-coded form. The exact cause of error must be determined with the error code table. Bit 0 is set as soon as an error occurs.

Bit	Message	Meaning
0	Generic error	An error has occurred
1	-	reserved
2	-	reserved
3	-	reserved
4	Communication	Network communication error
5	Device profile-specific	Error in execution as per device profile
6	-	reserved
7	Manufacturer-specific	Vendor-specific error message

7.3.2 Error code table

The error code is evaluated with the object `error code (603F`$_h$`)`, an object of the DSP402 device profile, and output as a four-digit hexadecimal value. The error code indicates the cause of the last interruption of movement. See the Troubleshooting chapter of the product manual for the meaning of the error code.

7.3.3 SDO error message ABORT

An SDO error message is generated as a response to an SDO transmission error. The cause of error is contained in `error code`, byte 4 to byte 7.

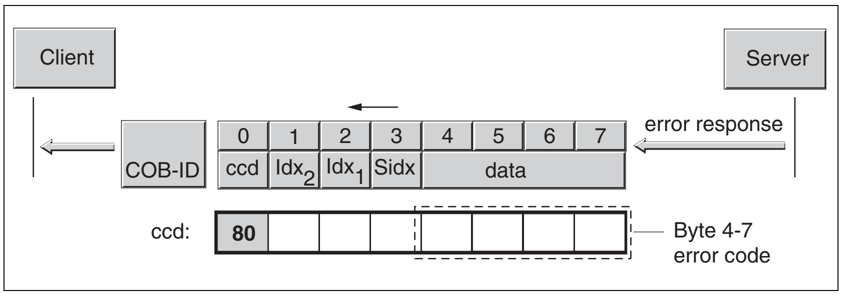


Figure 7.1 SDO error message as a response to an SDO message

The table below shows all error messages that may occur during data exchange with the product.

Error code	Meaning
0503 0000_h	Toggle bit not toggled
0504 0000_h	Time-out during SDO transfer
0504 0001_h	Command specifier CS incorrect or unknown
0504 0005_h	No memory available
0601 0000_h	Access to object impossible
0601 0001_h	No read access, because write-only object (wo)
0601 0002_h	No write access, because read object (ro)
0602 0000_h	Object does not exist in object dictionary
0604 0041_h	Object does not support PDO mapping
0604 0042_h	PDO mapping: number or length of objects exceed the byte length of the PDO
0604 0043_h	Parameters are incompatible
0604 0047_h	Device detects internal incompatibility
0606 0000_h	Hardware error, access denied
0607 0010_h	Data type and parameter length do not match
0607 0012_h	Data type does not match, parameter too long
0607 0013_h	Data type does not match, parameter too short
0609 0011_h	Subindex not supported
0609 0030_h	Value range of parameter too large (relevant only for write access)
0609 0031_h	Parameter values too great
0609 0032_h	Parameter values too small
0609 0036_h	Upper value is less than lower value
0800 0000_h	General error
0800 0020_h	Data can neither be transferred nor saved to the application.
0800 0021_h	Device control is local, data cannot be transmitted or saved.
0800 0022_h	Data cannot be transmitted or saved in this device state.
0800 0023_h	Object dictionary does not exist or cannot be generated, for example, if data error occurs during generation from file.
0800 $xxxx_h$	Manufacturer-specific error, xxxx corresponds to the error number of the device. It is listed in the error code table of the device manual.

0198441113718, V2.01, 11.2008

8 Object directory

8.1 Specifications for the objects

Index The index specifies the position of the object in the object dictionary. The index value is specified as a hexadecimal value.

Object code The object code specifies the data structure of the object.

Object code	Meaning	Coding
VAR	A simple value, for example of the type Integer8, Unsigned32 or Visible String8.	7
ARR (ARRAY)	A data field in which every entry is of the same data type.	8
REC (RECORD)	A data field that contains entries that are a combination of simple data types.	9

Data type	Value range	Data length	DS301 coding
Boolean	0 = false, 1 = true	1 byte	0001
Integer8	-128 ... +127	1 byte	0002
Integer16	-32768 ... +32767	2 bytes	0003
Integer32	-2147483648 ... +2147483647	4 bytes	0004
Unsigned8	0 ... 255	1 byte	0005
Unsigned16	0 ... 65535	2 bytes	0006
Unsigned32	0 ... 4294967295	4 bytes	0007
Visible String8	ASCII characters	8 byte	0009
Visible String16	ASCII characters	16 byte	0010

RO/RW Indicates read and/or write values
RO: values can only be read
RW: values can be read and written.

PDO R_PDO: Mapping for R_PDO possible
T_PDO: Mapping for T_PDO possible
No specification: PDO mapping not possible with the object

Min/max values Specifies the permissible range in which the object value is defined and valid.

Default value Factory setting.

Persistent "per." indicates whether the value of the parameter is persistent, i.e. whether it remains in the memory after the device is switched off . When changing a value via commissioning software or fieldbus, the user must explicitly store the changed value in the persistent memory.

8.2 Overview of object group 1000_h

Index	Subi ndex	Name	Obj. code	Data type	Acce ss	PDO	Description	Page
1000_h		Device type	VAR	Unsigned32	ro		Device type and profile	80
1001_h		Error register	VAR	Unsigned8	ro		Error register	80
1003_h		Predefined error field	ARR		rw		Error history, memory for error messages	81
1003_h	00_h	Number of errors	VAR	Unsigned8	rw		Number of error entries	81
1003_h	01_h	Error field	VAR	Unsigned32	ro		Error number	81
1005_h		COB ID SYNC	VAR	Unsigned32	rw		Identifier of the synchronization object	82
1008_h		Manufacturer device name	VAR	Visible String8	ro		User device name	82
1009_h		Manufacturer hardware version	VAR	Visible String8	ro		Hardware version	83
$100A_h$		Manufacturer software version	VAR	Visible String8	ro		Software version	83
$100C_h$		Guard time	VAR	Unsigned16	rw		Time span for node guarding [ms]	84
$100D_h$		Life time factor	VAR	Unsigned8	rw		Repeat factor for the node guarding protocol	84
1010_h		Save parameters	ARR	Unsigned32	rw		Saves parameters:	85
1010_h	01_h	Save all parameters	VAR	Unsigned32	rw		Saves all parameters	85
1010_h	02_h	Save communication parameters	VAR	Unsigned32	rw		Saves communication parameters	85
1010_h	03_h	Save application parameters	VAR	Unsigned32	rw		Saves application parameters	85
1011_h		restore default of parameters	ARR	Unsigned32	rw		Resets parameter values to the default setting	86
1011_h	01_h	Restore default of all parameters	VAR	Unsigned32	rw		Resets all parameter values to the default setting	86
1011_h	02_h	Restore default of communication parameters	VAR	Unsigned32	rw		Resets communication parameter values to default	86
1011_h	03_h	Restore default of communication parameters	VAR	Unsigned32	rw		Resets application parameter values to default	86
1014_h		COB ID EMCY	VAR	Unsigned32	rw		Unsigned16	88
1015_h		Inhibit time EMCY	VAR	Unsigned16	rw		Unsigned16	89
1016_h		Consumer Heartbeat Time	ARR	Unsigned32	rw		Unsigned16	89
1016_h	01_h	Consumer Heartbeat Time	VAR	Unsigned32	rw		Time interval and node ID of the "Heartbeat" recipient	89
1017_h		Producer Heartbeat Time	VAR	Unsigned16	rw		Time interval for producer "Heartbeat"	90
1018_h		Identity Object	REC	Identity	ro		Identification object:	90
1018_h	01_h	Vendor ID	VAR	Unsigned32	ro		Vendor ID	90
1018_h	02_h	Product code	VAR	Unsigned32	ro		Product code	90
1018_h	03_h	Revision number	VAR	Unsigned32	ro		Revision number	90

0198441113718, V2.01, 11.2008

Index	Subindex	Name	Obj. code	Data type	Access	PDO	Description	Page
1018_h	04_h	Serial number	VAR	Unsigned32	ro		Serial number	90
1020_h		Verify configuration	ARR	Unsigned32	rw		Checks data for configuration	92
1020_h	01_h	Configuration date	VAR	Unsigned32	rw		Date of configuration	92
1020_h	02_h	Configuration time	VAR	Unsigned32	rw		Time of configuration	92
1029_h		Number of elements	ARR	Unsigned8	ro		Number of values for the object	92
1029_h	01_h	Communication error	ARR	Unsigned8	rw		Communication error	92
1200_h		1st server SDO parameter	REC	SDO server param.	ro		First server SDO, settings	94
1200_h	01_h	COB ID client -> server	VAR	Unsigned32	ro		Identifier client -> server	94
1200_h	02_h	COB ID server -> client	VAR	Unsigned32	ro		Identifier server -> client	94
1201_h		2nd server SDO parameter	REC	SDO server param.	rw		Second server SDO, settings	95
1201_h	01_h	COB ID client -> server	VAR	Unsigned32	rw		Identifier client -> server	95
1201_h	02_h	COB ID server -> client	VAR	Unsigned32	rw		Identifier server -> client	95
1201_h	03_h	Node ID SDO client	VAR	Unsigned32	rw		Node ID SDO client	95
1400_h		1st receive PDO parameter	REC	PDO comm. param.	rw		First receive PDO (R_PDO1), settings	96
1400_h	01_h	COB ID R_PDO1	VAR	Unsigned32	rw		Identifier of the R_PDO1	96
1400_h	02_h	Transmission type R_PDO1	VAR	Unsigned8	rw		Transmission type	96
1401_h		2nd receive PDO parameter	REC	PDO comm. param.	rw		Second receive PDO (R_PDO2), settings	98
1401_h	01_h	COB ID R_PDO2	VAR	Unsigned32	rw		Identifier of the R_PDO2	98
1401_h	02_h	Transmission type R_PDO2	VAR	Unsigned8	rw		Transmission type	98
1402_h		3rd receive PDO parameter	REC	PDO comm. param.	rw		Third receive PDO (R_PDO3), settings	99
1402_h	01_h	COB ID R_PDO3	VAR	Unsigned32	rw		Identifier of the R_PDO3	99
1402_h	02_h	Transmission type R_PDO3	VAR	Unsigned8	rw		Transmission type	99
1403_h		4th receive PDO parameter	REC	PDO comm. param.	rw		Fourth receive PDO (R_PDO4), settings	100
1403_h	01_h	COB ID R_PDO4	VAR	Unsigned32	rw		Identifier of the R_PDO4	100
1403_h	02_h	Transmission type R_PDO4	VAR	Unsigned8	rw		Transmission type	100
1600_h		1st receive PDO mapping	REC	PDO mapping	ro		PDO mapping for R_PDO1, settings	101
1600_h	01_h	1st mapped object R_PDO1	VAR	Unsigned32	ro		First object for mapping in R_PDO1	101
1601_h		2nd receive PDO mapping	REC	PDO mapping	ro		PDO mapping for R_PDO2, settings	102
1601_h	01_h	1st mapped object R_PDO2	VAR	Unsigned32	ro		First object for mapping in R_PDO2	102

Index	Subi ndex	Name	Obj. code	Data type	Acce ss	PDO	Description	Page
1601_h	02_h	2nd mapped object R_PDO2	VAR	Unsigned32	ro		Second object for mapping in R_PDO2	102
1602_h		3rd receive PDO mapping	REC	PDO mapping	ro		PDO mapping for R_PDO3, settings	103
1602_h	01_h	1st mapped object R_PDO3	VAR	Unsigned32	ro		First object for mapping in R_PDO3	103
1602_h	02_h	2nd mapped object R_PDO3	VAR	Unsigned32	ro		Second object for mapping in R_PDO3	103
1603_h		4th receive PDO mapping	REC	PDO mapping	rw		PDO mapping for R_PDO3, settings	104
1603_h	01_h	1st mapped object R_PDO4	VAR	Unsigned32	rw		First object for mapping in R_PDO4	104
1603_h	02_h	2nd mapped object R_PDO4	VAR	Unsigned32	rw		Second object for mapping in R_PDO4	104
1603_h	03_h	3rd mapped object R_PDO4	VAR	Unsigned32	rw		Third object for mapping in R_PDO4	104
1800_h		1st transmit PDO parameter	REC	PDO comm. param.	rw		First transmit PDO (T_PDO1), settings	105
1800_h	01_h	COB ID T_PDO1	VAR	Unsigned32	rw		Identifier of the T_PDO1	105
1800_h	02_h	Transmission type T_PDO1	VAR	Unsigned8	rw		Transmission type	105
1800_h	03_h	Inhibit time T_PDO1	VAR	Unsigned16	rw		Inhibit time for locking bus access (1=100µs)	105
1800_h	04_h	Reserved T_PDO1	VAR	Unsigned8	rw		Priority for CAN bus arbitration ([0-7]).	105
1800_h	05_h	Event timer T_PDO1	VAR	Unsigned16	rw		Time span for event triggering (1=1 ms)	105
1801_h		2nd transmit PDO parameter	REC	PDO comm. param.	rw		Second transmit PDO (T_PDO2), settings	106
1801_h	01_h	COB ID T_PDO2	VAR	Unsigned32	rw		Identifier of the T_PDO2	106
1801_h	02_h	Transmission type T_PDO2	VAR	Unsigned8	rw		Transmission type	106
1801_h	03_h	Inhibit time T_PDO2	VAR	Unsigned16	rw		Inhibit time for locking bus access (1=100µs)	106
1801_h	04_h	Reserved T_PDO2	VAR	Unsigned8	rw		Reserved	106
1801_h	05_h	Event timer T_PDO2	VAR	Unsigned16	rw		Time span for event triggering (1=1 ms)	106
1802_h		3rd transmit PDO parameter	REC	PDO comm. param.	rw		Third transmit PDO (T_PDO3), settings	108
1802_h	01_h	COB ID T_PDO3	VAR	Unsigned32	rw		Identifier of the T_PDO3	108
1802_h	02_h	Transmission type T_PDO3	VAR	Unsigned8	rw		Transmission type	108
1802_h	03_h	Inhibit time T_PDO3	VAR	Unsigned16	rw		Inhibit time for locking bus access (1=100µs)	108
1802_h	04_h	Reserved T_PDO3	VAR	Unsigned8	rw		Reserved	108
1802_h	05_h	Event timer T_PDO3	VAR	Unsigned16	rw		Time span for event triggering (1=1 ms)	108

Index	Subi ndex	Name	Obj. code	Data type	Acce ss	PDO	Description	Page
1803_h		4th transmit PDO parameter	REC	PDO comm. param.	rw		Fourth transmit PDO (T_PDO4), settings	110
1803_h	01_h	COB ID T_PDO4	VAR	Unsigned32	rw		Identifier of the T_PDO4	110
1803_h	02_h	Transmission type T_PDO4	VAR	Unsigned8	rw		Transmission type	110
1803_h	03_h	Inhibit time T_PDO4	VAR	Unsigned16	rw		Inhibit time for locking bus access (1=100µs)	110
1803_h	04_h	Reserved T_PDO4	VAR	Unsigned8	ro		Reserved	110
1803_h	05_h	Event timer T_PDO4	VAR	Unsigned16	rw		Time span for event triggering (1=1 ms)	110
$1A00_h$		1st transmit PDO mapping	REC	PDO mapping	rw		PDO mapping for T_PDO1, settings	111
$1A00_h$	01_h	1st mapped object T_PDO1	VAR	Unsigned32	ro		First object for the mapping in T_PDO1	111
$1A01_h$		2nd transmit PDO mapping	REC	PDO mapping	rw		PDO mapping for T_PDO2, settings	112
$1A01_h$	01_h	1st mapped object T_PDO2	VAR	Unsigned32	ro		First object for the mapping in T_PDO2	112
$1A01_h$	02_h	2nd mapped object T_PDO2	VAR	Unsigned32	ro		Second object for the mapping in T_PDO2	112
$1A02_h$		3rd transmit PDO mapping	REC	PDO mapping	rw		PDO mapping for T_PDO3, settings	113
$1A02_h$	01_h	1st mapped object T_PDO3	VAR	Unsigned32	ro		First object for the mapping in T_PDO3	113
$1A02_h$	02_h	2nd mapped object T_PDO3	VAR	Unsigned32	ro		Second object for the mapping in T_PDO3	113
$1A03_h$		4th transmit PDO mapping	REC	PDO mapping	rw		PDO mapping for T_PDO4, settings	114
$1A03_h$	01_h	1st mapped object T_PDO4	VAR	Unsigned32	rw		First object for the mapping in T_PDO4	114
$1A03_h$	02_h	2nd mapped object T_PDO4	VAR	Unsigned32	rw		Second object for the mapping in T_PDO4	114
$1A03_h$	03_h	3rd mapped object T_PDO4	VAR	Unsigned32	rw		Third object for the mapping in T_PDO4	114
$1A03_h$	04_h	4th mapped object T_PDO4	VAR	Unsigned32	rw		Fourth object for the mapping in T_PDO4	114

8.3 Assignment object group 6000_h

The product provides corresponding parameters for CANopen object groups 3000_h and 6000_h.
The names of the parameters and the data type of the parameters may be different from the DSP402 definition for object group 6000_h. In this case, enter the data type according to the DSP402.
A detailed description of all parameters can be found in the product manual in the Parameters chapter.

Index	DSP402 object name	DSP402 data type	Parameter name
$603F{:}0_h$	Error code	UINT16	_StopFault
$6040{:}0_h$	Control word	UINT16	DCOMcontrol
$6041{:}0_h$	Status word	UINT16	DCOMstatus
$6060{:}0_h$	Operating modes	INT8	DCOMopmode
$6061{:}0_h$	Modes of operation display	INT8	_DCOMopmd_act
$6063{:}0_h$	Position actual value int	INT32	_p_act
$6064{:}0_h$	Position actual value	INT32	_p_actusr
$6065{:}0_h$	Tracking error window	UINT32	SPV_p_maxDiff
$6067{:}0_h$	Position window	UINT32	STANDp_win
$6068{:}0_h$	Position window time	UINT16	STANDpwinTime
$606B{:}0_h$	Velocity demand value	INT32	_n_actRAMP
$606C{:}0_h$	Velocity actual value	INT32	_n_act
$607A{:}0_h$	Target position	INT32	PPp_targetusr
$607D{:}1_h$	Min position limit	INT32	SPVswLimNusr
$607D{:}2_h$	Max position limit	INT32	SPVswLimPusr
$607F{:}0_h$	Max profile velocity	UINT32	RAMPn_max
$6081{:}0_h$	Profile velocity	UINT32	PPn_target
$6083{:}0_h$	Profile acceleration	UINT32	RAMPacc
$6084{:}0_h$	Profile deceleration	UINT32	RAMPdecel
$6086{:}0_h$	Motion profile type	INT16	ProfileType
$6098{:}0_h$	Homing method	INT8	HMmethod
$6099{:}1_h$	Homing speed during search for switch	UINT32	HMn
$6099{:}2_h$	Homing speed during search for zero	UINT32	HMn_out
$60F2{:}00_h$	Position Option Code	UINT16	PPoption
$60F4{:}00_h$	Tracking error actual value	INT32	_p_dif
$60FF{:}0_h$	Target velocity	INT32	PVn_target
$6502{:}0_h$	Supported drive modes	UINT32	SuppDriveModes

0198441113718, V2.01, 11.2008

8.4 Objects for PDO mapping

Object	Index:Subindex	PDO	Data type
_IO_act (3008:1_h)	_IO_act (3008:1_h)	T_PDO	UINT16
ANA1_act (3009:1_h)	ANA1_act (3009:1_h)	T_PDO	INT16
ANA2_act (3009:5_h)	ANA2_act (3009:5_h)	T_PDO	INT16
JOGactivate (301B:9_h)	JOGactivate (301B:9_h)	R_PDO	UINT16
_actionStatus (301C:4_h)	_actionStatus (301C:4_h)	T_PDO	UINT16
_p_actRAMPusr (301F:2_h)	_p_actRAMPusr (301F:2_h)	T_PDO	INT32
CUR_I_target (3020:4_h)	CUR_I_target (3020:4_h)	R_PDO	INT16
SPEEDn_target (3021:4_h)	SPEEDn_target (3021:4_h)	R_PDO	INT16
GEARdenom (3026:3_h)	GEARdenom (3026:3_h)	R_PDO	INT32
GEARnum (3026:4_h)	GEARnum (3026:4_h)	R_PDO	INT32
controlword (6040_h)	controlword (6040_h)	R_PDO	UINT16
Status word (6041_h)	Status word (6041_h)	T_PDO	UINT16
Position actual value (6064_h)	Position actual value (6064_h)	T_PDO	INT32
Velocity actual value ($606C_h$)	Velocity actual value ($606C_h$)	T_PDO	INT32
Target position ($607A_h$)	Target position ($607A_h$)	R_PDO	INT32
profile velocity (6081_h)	profile velocity (6081_h)	R_PDO	UINT32
Target velocity ($60FF_h$)	Target velocity ($60FF_h$)	R_PDO	INT32

8.5 Details of object group 1000h

8.5.1 1000_h Device type

The object specifies the device profile used as well as the device type.

Object description

Index	1000_h
Object name	Device type
Object code	VAR
Data type	Unsigned32

Value description

Subindex	00_h, device type
Meaning	Device type and profile
Access	read-only
PDO mapping	–
Value range	–
Default value	0044 0192_h
Can be saved	–

Bit coding, subindex 00_h

Bit	Access	Value	Meaning
31-24	ro	00_h	not used
23-16	ro	44_h	Bit18=1: Stepper motor drive
15-0	ro	0192_h	Device profile DS-402 (192_h)

8.5.2 1001_h Error register

The object specifies the error state of the device. The detailed cause of error can be determined with the object `predefined error field` (`1003`$_h$) and - for reasons of compatibility with devices with other fieldbus profiles - the object `error code (603F`$_h$`)`.

Errors are signaled by an EMCY message as soon as they occur.

Object description

Index	1001_h
Object name	Error register
Object code	VAR
Data type	Unsigned8

Value description

Subindex	00_h, error register
Meaning	Error register
Access	read-only
PDO mapping	–
Value range	–
Default value	–
Can be saved	–

Bit coding, subindex 00_h

Bit	Access	Value	Meaning
0	ro	–	Error! (generic error)
1	ro	–	Reserved
2	ro	–	Reserved
3	ro	–	Reserved
4	ro	–	Communication profile (communication error)
5	ro	–	Device profile (device profile error)
6	ro	–	Reserved
7	ro	–	Manufacturer-specific

8.5.3 1003_h Predefined error field

The object saves the latest error messages that were shown as EMCY messages.

- The subindex 00_h entry contains the number of saved error messages.
- The current error message is stored at subindex 01_h, older messages are moved to higher subindex entries.
- Writing 0 to subindex 00_h resets the error list.

Object description

Index	1003_h
Object name	Predefined error field
Object code	ARRAY
Data type	Unsigned32

Value description

Subindex	00_h, number of errors
Meaning	Number of error entries
Access	read-write
PDO mapping	–
Value range	0...1
Default value	1
Can be saved	–

Subindex	01_h, error field
Meaning	Error number
Access	read-only
PDO mapping	–
Value range	–
Default value	0
Can be saved	–

Bit coding, subindex 00_h..05_h

Bytes 0..15: Error code. Bytes 16..31 additional error information, not assigned in the device.

8.5.4 1005_h COB ID SYNC message

The object specifies the COB ID of the SYNC object and determines whether a device sends or receives SYNC messages.

The device can only receive SYNC messages.

For synchronization, a device in the network must send SYNC objects.

The COB ID can be changed in the NMT state "Pre-Operational"

Object description

Index	1005_h
Object name	COB ID SYNC
Object code	VAR
Data type	Unsigned32

Value description

Subindex	00_h, COB ID SYNC
Meaning	Identifier of the synchronization object
Access	read-write
PDO mapping	–
Value range	0...4294967295
Default value	$8000\ 0080_h$
Can be saved	Yes

Bit coding, subindex 00_h

Bit	Access	Value	Meaning
31	ro	0_b	1: Device can receive SYNC messages (SYNC consumer)
30	ro	1_b	1: Device can send SYNC messages (SYNC producer)
29	ro	0_b	0: 11 bit identifier (CAN 3.0A) 1: 29.bit identifier (CAN 3.0B)
28-11	ro	0000_h	Only relevant if bit 29=1, not used by the device.
10-7	rw	0001_b	Function code, bit 10..7 of the COB ID
6-0	ro	$7F_h$	Node address, bit 6..0 of the COB ID

8.5.5 1008_h Manufacturer device name

The object specifies the device name of the manufacturer.

Object description

Index	1008_h
Object name	Manufacturer device name
Object code	VAR
Data type	Visible String8

0198441113718, V2.01, 11.2008

Value description

Subindex	00_h, manufacturer device name
Meaning	User device name
Access	read-only
PDO mapping	–
Value range	–
Default value	–
Can be saved	–

The following objects contain additional information on the device:- Objects 6404_h, 6410_h: Motor data

8.5.6 1009_h Manufacturer hardware version

The object specifies the version of the device hardware.

Object description

Index	1009_h
Object name	Manufacturer hardware version
Object code	VAR
Data type	Visible String8

Value description

Subindex	00_h, manufacturer hardware version
Meaning	Hardware version
Access	read-only
PDO mapping	–
Value range	–
Default value	–
Can be saved	–

8.5.7 $100A_h$ Manufacturer software version

The object specifies the version of the device software.

Object description

Index	$100A_h$
Object name	Manufacturer software version
Object code	VAR
Data type	Visible String8

Value description

Subindex	00_h, manufacturer software version
Meaning	Software version
Access	read-only
PDO mapping	–
Value range	–
Default value	–
Can be saved	–

8.5.8 100C$_h$ Guard time

The object specifies the time span for connection monitoring (node guarding) of an NMT slave.

The time span for connection monitoring of an NMT master results from the time span "guard time" multiplied by the factor "life time", object `Life time factor(100Dh)`.

The time span can be changed in the NMT state "Pre-Operational".

Object description

Index	100C$_h$
Object name	Guard time
Object code	VAR
Data type	Unsigned16

Value description

Subindex	00$_h$, guard time
Meaning	Time span for node guarding [ms]
Access	read-write
PDO mapping	–
Value range	0...65535
Default value	0
Can be saved	Yes

8.5.9 100D$_h$ Life time factor

The object specifies the factor that, together with the time span "guard time", results in the time interval for connection monitoring of an NMT master. Within this period, the NMT slave device expects a monitoring request via node guarding from the NMT master.

life time = guard time * life time factor

The value "0" deactivates monitoring of the NMT master.

If there is no connection monitoring through the NMT master during the time interval "life time", signals an error and switches to fault state.

The time factor can be changed in the NMT state "Pre-Operational".

The time span "guard time" is set with the object `Guard time (100Ch)`.

Object description

Index	100D$_h$
Object name	Life time factor
Object code	VAR
Data type	Unsigned8

Value description

Subindex	00_h, life time factor
Meaning	Repeat factor for the node guarding protocol.
Access	read-write
PDO mapping	–
Value range	0...255
Default value	0
Can be saved	Yes

8.5.10 1010_h Save Parameters

The object is used to save parameters.

- Subindex 01_h, all parameters
- Subindex 02_h, communication parameters
- Subindex 03_h, application parameters

Object description

Index	1010_h
Object name	Save parameters
Object code	ARRAY
Data type	Unsigned32

Value description

Subindex	00_h, number of elements
Meaning	Number of values for the object
Access	read-only
PDO mapping	–
Value range	–
Default value	3
Can be saved	–

Subindex	01_h, save all parameters
Meaning	saves all parameters
Access	read-write
PDO mapping	–
Value range	–
Default value	1
Can be saved	–

Subindex	02_h, save communication parameters
Meaning	Saves communication parameters
Access	read-write
PDO mapping	–
Value range	–
Default value	1
Can be saved	–

Subindex	03_h, save application parameters
Meaning	Saves application parameters
Access	read-write
PDO mapping	–
Value range	–
Default value	1
Can be saved	–

8.5.11 1011_h Restore default parameters

The object is used to restore the default parameters.

Switching off the product

NOTE: If you use the object to reset application parameters or user parameter or both, the device also saves the data to the EEPROM. If the product is switched off while the data is saved, the product signals a CRC error the next time it is switched on. To avoid this, proceed as follows:

▶ Read the object `PAReeprSave 3004:01`$_h$ during the reset (see 6.4.7.1 "Saving and restoring object data").

▶ Do not switch off the product unless the object `PAReeprSave 3004:01`$_h$ has assumed the value 0.

- Subindex 01_h, all parameters
- Subindex 02_h, communication parameters
- Subindex 03_h, application parameters

Object description

Index	1011_h
Object name	Restore Default Parameters
Object code	ARRAY
Data type	Unsigned32

0198441113718, V2.01, 11.2008

Value description

Subindex	00_h, number of elements
Meaning	Number of values for the object
Access	read-only
PDO mapping	–
Value range	–
Default value	3
Can be saved	–

Subindex	01_h, restore default of all parameters
Meaning	Resets all parameter values to the default setting
Access	read-write
PDO mapping	–
Value range	–
Default value	1
Can be saved	–

Subindex	02_h, restore default of communication parameters
Meaning	Resets communication parameter values to default
Access	read-write
PDO mapping	–
Value range	–
Default value	1
Can be saved	–

Subindex	03_h, restore default of application parameters
Meaning	Restores parameter settings for the application to default
Access	read-write
PDO mapping	–
Value range	–
Default value	1
Can be saved	–

8.5.12 1014$_h$ COB ID-Emergency message

The object specifies the COB ID of the emergency object "EMCY".

Object description

Index	1014$_h$
Object name	COB ID EMCY
Object code	VAR
Data type	Unsigned32

Value description

Subindex	00$_h$, COB ID EMCY
Meaning	Identifier of the emergency object
Access	read-write
PDO mapping	–
Value range	0...4294967295
Default value	4000 0080$_h$ + node ID
Can be saved	Yes

Bit coding, subindex 00$_h$

Bit	Access	Value	Meaning
31, 30	ro	0$_b$	Reserved
29	ro	0$_b$	0: 11 bit identifier (CAN 3.0A) 1: 29.bit identifier (CAN 3.0B)
28-11	ro	0000$_h$	Only relevant if bit 29=1 is not used by the device.
10-7	rw	0001$_b$	Function code, bits 10-7 of the COB ID
6-0	ro	–	Node address, bit 6-0 of the COB ID

The COB ID can be changed in the NMT state "Pre-Operational".

8.5.13 1015_h Inhibit time emergency message

The object specifies the waiting time for the repeated transmission of EMCY messages as a multiple of 100µs.

Object description

Index	1015_h
Object name	Inhibit time EMCY
Object code	VAR
Data type	Unsigned16

Value description

Subindex	00_h, inhibit time EMCY
Meaning	Waiting time for repeated transmission of an EMCY
Access	read-write
PDO mapping	–
Value range	0...65535
Default value	0
Can be saved	Yes

8.5.14 1016_h Consumer Heartbeat Time

The object contains the settings of the "Heartbeat Consumers" for NMT monitoring by mans of "Heartbeat" connection message.

Object description

Index	1016_h
Object name	Consumer Heartbeat Time
Object code	ARRAY
Data type	Unsigned32

Value description

Subindex	00_h, number of elements
Meaning	Number of values for the object
Access	read-only
PDO mapping	–
Value range	–
Default value	3
Can be saved	–

Subindex	01_h, Consumer Heartbeat Time
Meaning	Time interval and node ID of the "Heartbeat" recipient
Access	read-write
PDO mapping	–
Value range	0...4294967295
Default value	0
Can be saved	Yes

Bit coding subindex $01_h..03_h$

Bit	Meaning
31..24	Reserved
23..16	Node ID
15..0	Time interval for "Heartbeat" message

The time interval is specified as a multiple of 1 ms and must be greater than the producer "heartbeat" time, object `Producer Heartbeat Time (1017h)`. If the time interval is zero, the device specified via the node ID is not monitored.

8.5.15 1017_h Producer Heartbeat Time

The object contains the time interval of the "Heartbeat" producer for NMT monitoring by means of "Heartbeat" connection message as a multiple of 1 ms.

The producer "Heartbeat" time must be less than the time interval of the "Heartbeat" consumer, object `Consumer Heartbeat Time (1016h)`. A time interval of zero deactivates monitoring.

Object description

Index	1017_h
Object name	Producer Heartbeat Time
Object code	VAR
Data type	Unsigned16

Value description

Subindex	00_h, Producer Heartbeat Time
Meaning	Time interval for producer "Heartbeat"
Access	read-write
PDO mapping	–
Value range	0...65535
Default value	0
Can be saved	Yes

8.5.16 1018_h Identity Object

The object provides information on the product.

- Subindex 01_h (vendor ID) contains the manufacturer ID
- Subindex 02_h (product ID) contains the manufacturer-specific product code
- Subindex 03_h (revision number) identifies special CANopen properties for the device
- Subindex 04_h (serial number) contains the serial number

Object description

Index	1018_h
Object name	Identity Object
Object code	RECORD
Data type	Identity

0198441113718, V2.01, 11.2008

Value description

Subindex	00_h, number of elements
Meaning	Number of values for the object
Access	read-only
PDO mapping	–
Value range	–
Default value	4
Can be saved	–

Subindex	01_h, vendor ID
Meaning	Vendor ID
Access	read-only
PDO mapping	–
Value range	–
Default value	$0800\ 0054_h$
Can be saved	–

Subindex	02_h, product code
Meaning	Product code
Access	read-only
PDO mapping	–
Value range	–
Default value	8401
Can be saved	–

Subindex	03_h, revision number
Meaning	Revision number
Access	read-only
PDO mapping	–
Value range	–
Default value	1
Can be saved	–

Subindex	04_h, serial number
Meaning	Serial number
Access	read-only
PDO mapping	–
Value range	–
Default value	0
Can be saved	–

8.5.17 1020_h data on configuration

The object is used to verify the configuration.

- Subindex 01_h, date of configuration
- Subindex 02_h, time of configuration

Object description

Index	1020_h
Object name	
Object code	RECORD
Data type	Identity

Value description

Subindex	00_h, verify configuration
Meaning	Checks data for configuration
Access	read-only
PDO mapping	–
Value range	–
Default value	2
Can be saved	–

Subindex	01_h, configuration date
Meaning	Date of configuration
Access	read-write
PDO mapping	–
Value range	–
Default value	
Can be saved	Yes

Subindex	02_h, configuration time
Meaning	Time of configuration
Access	read-write
PDO mapping	–
Value range	–
Default value	
Can be saved	Yes

8.5.18 1029_h error behavior

The object specifies the behavior of the NMT state machine in the event of a communication error.

Object description

Index	1029_h
Object name	Error behavior
Object code	ARRAY
Data type	Unsigned8

Value description

Subindex	00_h, number of elements
Meaning	Number of values for the object
Access	read-only
PDO mapping	–
Value range	–
Default value	1
Can be saved	–

Subindex	01_h, Communication Error
Meaning	Communication error
Access	read-write
PDO mapping	–
Value range	0...2
Default value	0
Can be saved	Yes

Settings, subindex 01_h

Value	Meaning
0	Pre-operational (with operational state only)
1	No state transition
2	stopped

8.5.19 1200_h 1st server SDO parameter

The object contains the settings for the first server SDO.

Object description

Index	1200_h
Object name	1st server SDO parameter
Object code	RECORD
Data type	SDO server parameter

Value description

Subindex	00_h, number of elements
Meaning	Number of values for the object
Access	read-only
PDO mapping	–
Value range	–
Default value	2
Can be saved	–

Subindex	01_h, COB ID client -> server
Meaning	Identifier client -> server
Access	read-only
PDO mapping	–
Value range	0...4294967295
Default value	1536 + node ID
Can be saved	Yes

Subindex	02_h, COB ID server -> client
Meaning	Identifier server -> client
Access	read-only
PDO mapping	–
Value range	0...4294967295
Default value	1408 + node ID
Can be saved	Yes

8.5.20 1201_h 2nd server SDO parameter

The object contains the settings for the second server SDO.

Object description

Index	1201_h
Object name	2nd server SDO parameter
Object code	RECORD
Data type	SDO server parameter

Value description

Subindex	00_h, number of elements
Meaning	Number of values for the object
Access	read-only
PDO mapping	–
Value range	–
Default value	3
Can be saved	–

Subindex	01_h, COB ID client -> server
Meaning	Identifier client -> server
Access	read-write
PDO mapping	–
Value range	0...4294967295
Default value	8000 0000_h
Can be saved	Yes

Subindex	02_h, COB ID server -> client
Meaning	Identifier server -> client
Access	read-write
PDO mapping	–
Value range	0...4294967295
Default value	8000 0000_h
Can be saved	Yes

Subindex	03_h, node ID SDO client
Meaning	Node ID SDO client
Access	read-write
PDO mapping	–
Value range	1...127
Default value	–
Can be saved	Yes

8.5.21 1400_h 1st receive PDO parameter

The object contains the settings for the first receive PDO R_PDO1.

Object description

Index	1400_h
Object name	1st receive PDO parameter
Object code	RECORD
Data type	PDO Communication Parameter

Value description

Subindex	00_h, number of entries
Meaning	Number of values for the object
Access	read-only
PDO mapping	–
Value range	–
Default value	2
Can be saved	–

Subindex	01_h, COB ID used by PDO
Meaning	Identifier of the R_PDO1
Access	read-write
PDO mapping	–
Value range	0...4294967295
Default value	0200_h + node ID
Can be saved	Yes

Subindex	02_h, transmission type = asynchronous
Meaning	Transmission type
Access	read-write
PDO mapping	–
Value range	0...255
Default value	255
Can be saved	Yes

Bit assignment subindex 01_h

Bit	Access	Value	Meaning
31	rw	0_b	0: PDO is enabled 1: PDO is disabled
30	ro	0_b	0: RTR (see below) is possible 1: RTR not permitted
29	ro	0_b	0: 11 bit identifier (CAN 3.0A) 1: 29.bit identifier (CAN 3.0B)
28-11	ro	0000_h	Only relevant if bit 29=1 is not used by the device.
10-7	rw	0100_b	Function code, bits 10-7 of the COB ID
6-0	ro	–	Node address, bit 6-0 of the COB ID

Bit 31 A R_PDO can only be used if bit 31="0".

Bit 30: RTR Bit

If a device supports R_PDOs with RTR (remote transmission request), it can request a PDO from a PDO producer with RTR = "0" in accordance with the producer-consumer relationship.

The device cannot request PDOs, but it can respond to the request for a PDO, see RTR bit for T_PDO1 settings (1800$_h$).

Bit coding, subindex 02$_h$

The control for evaluating R_PDO data is specified via subindex 02$_h$. The values 241..251 are reserved.

Transmission type	**cyclic**	**acyclic**	**synchro-nous**	**asynchro-nous**	**RTR-controlled**
0	–	X	X	–	–
1-240	X	–	X	–	–
252	–	–	X	–	X
253	–	–	–	X	X
254	–	–	–	X	–
255	–	–	–	X	–

If an R_PDO is transmitted synchronously (transmission type=0..252), the product evaluates the received data depending on the SYNC object.

- In the case of acyclic transmission (transmission type=0), the evaluation depends on the SYNC object, but not the transmission of the PDO. A received PDO message is evaluated with the following SYNC.

  A value between 1 and 240 specifies the number of SYNC cycles after which a received PDO is evaluated.

The values 252 to 254 are relevant for updating T_PDOs, but not for sending them.

- 252: Updating of transmit data with receipt of the next SYNC
- 253: Updating of transmit data with receipt of a request from a PDO consumer
- 254: Updating of data in an event-driven way, the triggering event is specified in a manufacturer-specific way

R_PDOs with the value 255 are updated immediately upon receipt of the PDOs. The triggering event is the data that is transmitted corresponding to the definition of the DSP402 device profile in the PDO.

Settings

R_PDO1 is processed asynchronously and in an event-driven way.

The byte assignment of the R_PDO1 is specified via PDO mapping with the object `1st receive PDO mapping (1600`$_h$`)`. The following default assignment is used for R_PDO1:

- Bytes 0..1: Control word `controlword (6040`$_h$`)`.

The COB ID of the object can be changed in the NMT state "Pre-Operational".

8.5.22 1401_h 2nd receive PDO_parameter

The object contains settings for the second receive PDO R_PDO2.

Object description

Index	1401_h
Object name	2nd receive PDO parameter
Object code	RECORD
Data type	PDO Communication Parameter

Value description

Subindex	00_h, largest subindex supported
Meaning	Largest subindex supported
Access	read-only
PDO mapping	–
Value range	–
Default value	2
Can be saved	–

Subindex	01_h, COB ID R_PDO2
Meaning	Identifier of the R_PDO2
Access	read-write
PDO mapping	–
Value range	0...4294967295
Default value	8000 0300_h + node ID
Can be saved	Yes

Subindex	02_h, transmission type
Meaning	Transmission type
Access	read-write
PDO mapping	–
Value range	0...255
Default value	255
Can be saved	Yes

The meaning of the bit states and subindex values is described with the object 1st receive PDO parameters (1400_h).

Settings

R_PDO2 is processed synchronously, acyclically and in an event-driven way and must be activated with bit 31=1 in subindex 01_h before it can be used.

The byte assignment of R_PDO2 is specified via PDO mapping with the object 2nd Receive PDO mapping (1601_h). The following defaults are set for "Profile Position" operating mode:

- Bytes 0..1: Control word `controlword` (6040_h).
- Bytes 2..5: Target position of the motion command `target position` ($607A_h$)

The COB ID of the object can be changed in the NMT state "Pre-Operational".

The transmission type for the receive PDO can have 3 value ranges:

0	For an asynchronous cycle
1 to 240	Instructs the receive PDO to become active only if a SYNC object is received
255	Specifies that the PDO is executed when it is received

8.5.23 1402$_h$ 3rd receive PDO-Parameter

The object contains settings for the third receive PDO R_PDO3.

Object description

Index	1402$_h$
Object name	3rd receive PDO parameter
Object code	RECORD
Data type	PDO Communication Parameter

Value description

Subindex	00$_h$, largest subindex supported
Meaning	Largest subindex supported
Access	read-only
PDO mapping	–
Value range	–
Default value	2
Can be saved	–

Subindex	01$_h$, COB ID used by PDO
Meaning	Identifier of the R_PDO3
Access	read-write
PDO mapping	–
Value range	0...4294967295
Default value	8000 0400$_h$ + node ID
Can be saved	Yes

Subindex	02$_h$, transmission type
Meaning	Transmission type
Access	read-write
PDO mapping	–
Value range	0...255
Default value	255
Can be saved	Yes

The meaning of the bit states and subindex values is described with the object `1st receive PDO-parameters (1400`$_h$`)`.

Settings

R_PDO3 is processed synchronously, acyclically and in an event-driven way and must be activated with bit 31=1 in subindex 01_h before it can be used.

The byte assignment of the R_PDO3 is specified via PDO mapping with the object `3rd Receive PDO mapping (1602`$_h$`)`. The following defaults are set for "Profile Velocity" operating mode:

- Byte 0..1: Control word `controlword (6040`$_h$`)`
- Bytes 2..5: reference speed of the motion command `Target velocity (60FF`$_h$`)`

The COB ID of the object can be changed in the NMT state "Pre-Operational".

The transmission type for the receive PDO can have 3 value ranges:

0	For an asynchronous cycle
1 to 240	Instructs the receive PDO to become active only if a SYNC object is received
255	Specifies that the PDO is executed when it is received

8.5.24 1403_h 4th receive PDO parameter

The object stores settings for the fourth receive PDO R_PDO4.

Object description

Index	1403_h
Object name	4th receive PDO parameter
Object code	RECORD
Data type	PDO Communication Parameter

Value description

Subindex	00_h, largest subindex supported
Meaning	Largest subindex supported
Access	read-only
PDO mapping	–
Value range	–
Default value	2
Can be saved	–

Subindex	01_h, COB ID used by PDO
Meaning	Identifier of the R_PDO4
Access	read-write
PDO mapping	–
Value range	0...4294967295
Default value	8000 0500_h + node ID
Can be saved	Yes

Subindex	02_h, transmission type
Meaning	Transmission type
Access	read-only
PDO mapping	–
Value range	–
Default value	254
Can be saved	Yes

The meaning of the bit states and subindex values is described under object `1st receive PDO-parameters (1400`$_h$`)`.

PDO settings

R_PDO4 is processed asynchronously and in an event-driven way and must be activated with bit 31=1 in subindex 01_h before it can be used.

The COB ID of the object can be changed in the NMT state "Pre-Operational".

8.5.25 1600_h 1st receive PDO mapping

The object specifies the objects mapped in R_PDO1 and transmitted with the PDO. When the object is read, subindex 00_h, the number of mapped objects is read.

Object description

Index	1600_h
Object name	1st receive PDO mapping
Object code	RECORD
Data type	PDO mapping

Value description

Subindex	00_h, number of mapped objects
Meaning	Number of values for the object
Access	read-only
PDO mapping	–
Value range	1...8
Default value	1
Can be saved	–

Subindex	01_h, CMD: Control word
Meaning	First object for mapping in R_PDO1
Access	read-only
PDO mapping	–
Value range	0...4294967295
Default value	6040 0010_h
Can be saved	–

Bit coding starting at subindex 01_h

Each subindex entry from subindex 01_h on specifies the object and the bit length of the object. The object is identified via the index and the subindex, which refer to the object dictionary of the device.

Bit	Meaning
31..16	Index
15..8	Subindex
7..0	Object length in bit

Settings

The PDO assignment for R_PDO1 cannot be modified. The following default assignment is used:

- Subindex 01_h: PDO mapping of the control word, object `controlword` (6040_h).

8.5.26 1601_h 2nd receive PDO mapping

The object specifies the objects mapped in R_PDO2 and transmitted with the PDO. When the object is read, subindex 00_h, the number of mapped objects is read.

Object description

Index	1601_h
Object name	2nd receive PDO mapping
Object code	RECORD
Data type	PDO mapping

Value description

Subindex	00_h, number of mapped application objects in PDO
Meaning	Number of values for the object
Access	read-only
PDO mapping	–
Value range	1...8
Default value	2
Can be saved	–

Subindex	01_h, PDO mapping for the first application object to be mapped (control word)
Meaning	First object for mapping in R_PDO2
Access	read-only
PDO mapping	–
Value range	0...4294967295
Default value	$6040\ 0010_h$
Can be saved	–

Subindex	02_h, PDO mapping for the second application object to be mapped (target position)
Meaning	Second object for mapping in R_PDO2
Access	read-only
PDO mapping	–
Value range	0...4294967295
Default value	607A 0020_h
Can be saved	–

The meaning of the bit states is described with the object `1st receive PDO-mapping` (1600_h).

Settings The PDO assignment for R_PDO2 cannot be modified. The following defaults are set for "Profile Position" operating mode:

- Subindex 01_h: PDO mapping of the control word, object `controlword` (6040_h).
- Subindex 02_h: target position of the motion command, object `target position` ($607A_h$).

8.5.27 1602_h 3rd receive PDO mapping

The object specifies the objects mapped in R_PDO3 and transmitted with the PDO. When the object is read, subindex 00_h, the number of mapped objects is read.

Object description

Index	1602_h
Object name	3rd receive PDO mapping
Object code	RECORD
Data type	PDO mapping

Value description

Subindex	00_h, number of mapped application objects in PDO
Meaning	Number of values for the object
Access	read-only
PDO mapping	–
Value range	1...8
Default value	2
Can be saved	–

Subindex	01_h, PDO mapping for the first application object to be mapped (control word)
Meaning	First object for mapping in R_PDO3
Access	read-only
PDO mapping	–
Value range	0...4294967295
Default value	6040 0010_h
Can be saved	–

Subindex	02_h, PDO mapping for the second application object to be mapped (target velocity)
Meaning	Second object for mapping in R_PDO3
Access	read-only
PDO mapping	–
Value range	0...4294967295
Default value	60FF 0020_h
Can be saved	–

The meaning of the bit states is described with the object `1st receive PDO-mapping (1600`$_h$`)`.

Settings The PDO assignment for R_PDO3 cannot be modified. The following defaults are set for "Profile Velocity" operating mode:

- Subindex 01_h: PDO mapping of the control word, object `controlword (6040`$_h$`)`.
- Bytes 2..5: reference speed of the motion command `Target velocity (60FF`$_h$`)`.

8.5.28 1603_h 4th receive PDO mapping

The object specifies the objects mapped in R_PDO4 and transmitted with the PDO. When the object is read, subindex 00_h, the number of mapped objects is read.

Object description

Index	1603_h
Object name	4th receive PDO mapping
Object code	RECORD
Data type	PDO mapping

Value description

Subindex	00_h, number of elements
Meaning	Number of values for the object
Access	read-write
PDO mapping	–
Value range	0...4
Default value	0
Can be saved	Yes

The meaning of the bit states is described with the object `1st receive PDO mapping (1600`$_h$`)`.

Settings The PDO assignment for R_PDO4 can be modified.

8.5.29 1800_h 1st transmit PDO parameter

The object contains settings for the first transmit PDO T_PDO1.

Object description

Index	1800_h
Object name	1st transmit PDO parameter
Object code	RECORD
Data type	PDO Communication Parameter

Value description

Subindex	00_h, number of entries
Meaning	Number of values for the object
Access	read-only
PDO mapping	–
Value range	–
Default value	5
Can be saved	–

Subindex	01_h, COB ID used by PDO
Meaning	Identifier of the T_PDO1
Access	read-write
PDO mapping	–
Value range	0...4294967295
Default value	0180_h + node ID
Can be saved	Yes

Subindex	02_h, transmission type = asynchronous
Meaning	Transmission type
Access	read-write
PDO mapping	–
Value range	0...255
Default value	255
Can be saved	Yes

Subindex	03_h, inhibit time
Meaning	Inhibit time for locking bus access (1=100µs)
Access	read-write
PDO mapping	–
Value range	0...65535
Default value	0
Can be saved	Yes

Subindex	04_h, reserved
Meaning	Reserved
Access	–
PDO mapping	–
Value range	0...255
Default value	–
Can be saved	–

Subindex	05_h, event timer
Meaning	Time span for event triggering (1=1 ms)
Access	read-write
PDO mapping	–
Value range	0...65535
Default value	0
Can be saved	Yes

The meaning of the bit states and subindex values is described with the object `1st receive PDO-parameters` (`1400`$_h$).

Settings

T_PDO1 is transmitted asynchronously and in an event-driven way whenever the PDO data changes.

The byte assignment of the T_PDO1 is specified via PDO mapping with the object `1st transmit PDO mapping` (`1A00`$_h$). The following default assignment is used:

- Bytes 0..1: Status word `statusword` (`6041`$_h$).

The COB ID of the object can be changed in the NMT state "Pre-Operational".

8.5.30 1801_h 2nd transmit PDO parameter

The object contains settings for the second transmit PDO T_PDO2.

Object description

Index	1801_h
Object name	2nd transmit PDO parameter
Object code	RECORD
Data type	PDO Communication Parameter

Value description

Subindex	00_h, largest subindex supported
Meaning	Largest subindex supported
Access	read-only
PDO mapping	–
Value range	–
Default value	5
Can be saved	–

Subindex	01$_h$, COB ID used by PDO
Meaning	Identifier of the T_PDO2
Access	read-write
PDO mapping	–
Value range	0...4294967295
Default value	C000 0280$_h$ + node ID
Can be saved	Yes

Subindex	02$_h$, transmission type
Meaning	Transmission type
Access	read-write
PDO mapping	–
Value range	0...255
Default value	255
Can be saved	Yes

Subindex	03$_h$, inhibit time
Meaning	Inhibit time for locking bus access (1=100µs)
Access	read-write
PDO mapping	–
Value range	0...65535
Default value	0
Can be saved	Yes

Subindex	04$_h$, reserved
Meaning	Reserved
Access	–
PDO mapping	–
Value range	0...255
Default value	–
Can be saved	–

Subindex	05$_h$, event timer
Meaning	Time span for event triggering (1=1 ms)
Access	read-write
PDO mapping	–
Value range	0...65535
Default value	100
Can be saved	Yes

The meaning of the bit states and subindex values is described with the object `1st receive PDO-parameters (1400`$_h$`)`.

Settings T_PDO2 is transmitted synchronously and acyclically.

The byte assignment of the T_PDO2 is specified via PDO mapping with the object `2nd transmit PDO mapping (1A01h)`. The following defaults are set for "Profile Position" operating mode:

- Bytes 0..1: Status word `statusword (6041h)`.
- Bytes 2..5: Current position `position actual value (6064h)`.

The COB ID of the object can be changed in the NMT state "Pre-Operational".

8.5.31 1802_h 3rd transmit PDO parameter

The object contains settings for the third transmit PDO T_PDO3.

Object description

Index	1802_h
Object name	3rd transmit PDO parameter
Object code	RECORD
Data type	PDO Communication Parameter

Value description

Subindex	00_h, largest subindex supported
Meaning	Largest subindex supported
Access	read-only
PDO mapping	–
Value range	–
Default value	5
Can be saved	–

Subindex	01_h, COB ID used by PDO
Meaning	Identifier of the T_PDO3
Access	read-write
PDO mapping	–
Value range	0...4294967295
Default value	C000 0380_h + node ID
Can be saved	Yes

Subindex	02_h, transmission type
Meaning	Transmission type
Access	read-write
PDO mapping	–
Value range	0...255
Default value	255
Can be saved	Yes

Subindex	03_h, inhibit time
Meaning	Inhibit time for locking bus access (1=100µs)
Access	read-write
PDO mapping	–
Value range	0...65535
Default value	0
Can be saved	Yes

Subindex	04_h, reserved
Meaning	Reserved
Access	–
PDO mapping	–
Value range	0...255
Default value	–
Can be saved	–

Subindex	05_h, event timer
Meaning	Time span for event triggering (1=1 ms)
Access	read-write
PDO mapping	–
Value range	0...65535
Default value	100
Can be saved	Yes

The meaning of the bit states and subindex values is described with the object `1st receive PDO-parameters` (`1400`$_h$).

Settings T_PDO3 is transmitted synchronously and acyclically.

The byte assignment of the T_PDO3 is specified via PDO mapping with the object `3rd transmit PDO mapping` (`1A02`$_h$). The following defaults are set for "Profile Velocity" operating mode:

- Bytes 0..1: Status word `statusword` (`6041`$_h$).
- Byte 2..5: Current Speed `velocity actual value` (`606C`$_h$).

The COB ID of the object can be changed in the NMT state "Pre-Operational".

8.5.32 1803_h 4th transmit PDO parameter

The object contains settings for the fourth transmit PDO T_PDO4.

Object description

Index	1803_h
Object name	4th transmit PDO parameter
Object code	RECORD
Data type	PDO Communication Parameter

Value description

Subindex	00_h, largest subindex supported
Meaning	Largest subindex supported
Access	read-only
PDO mapping	–
Value range	–
Default value	5
Can be saved	–

Subindex	01_h, COB ID used by PDO
Meaning	Identifier of the T_PDO4
Access	read-write
PDO mapping	–
Value range	0...4294967295
Default value	C000 0480_h + node ID
Can be saved	Yes

Subindex	02_h, transmission type
Meaning	Transmission type
Access	read-only
PDO mapping	–
Value range	0...255
Default value	254
Can be saved	Yes

Subindex	03_h, inhibit time
Meaning	Inhibit time for locking bus access (1=100µs)
Access	read-write
PDO mapping	–
Value range	0...65535
Default value	0
Can be saved	Yes

Subindex	04_h, reserved
Meaning	Reserved
Access	–
PDO mapping	–
Value range	0...255
Default value	–
Can be saved	–

Subindex	05_h, event timer
Meaning	Time span for event triggering (1=1 ms)
Access	read-write
PDO mapping	–
Value range	0...65535
Default value	0
Can be saved	Yes

The meaning of the bit states and subindex values is described with the object `1st receive PDO-parameters (1400`$_h$`)`.

Settings R_PDO4 is transmitted asynchronously and in an event-driven way.

The COB ID of the object can be changed in the NMT state "Pre-Operational".

8.5.33 1A00$_h$ 1st transmit PDO mapping

The object specifies the objects mapped in T_PDO1 and transmitted with the PDO. When the object is read, subindex 00_h, the number of mapped objects is read.

Object description

Index	$1A00_h$
Object name	1st transmit PDO mapping
Object code	RECORD
Data type	PDO mapping

Value description

Subindex	00_h, number of mapped objects
Meaning	Number of values for the object
Access	read-only
PDO mapping	–
Value range	1...8
Default value	1
Can be saved	–

Subindex	01_h, ETA: status word
Meaning	First object for the mapping in T_PDO1
Access	read-only
PDO mapping	–
Value range	0...4294967295
Default value	6041 0010_h
Can be saved	–

The meaning of the bit states is described with the object `1st receive PDO mapping (1600`$_h$`)`.

Settings The PDO assignment for T_PDO1 cannot be modified. The following default assignment is used:

- Subindex 1: PDO mapping of the status word, object `statusword (6041`$_h$`)`

8.5.34 1A01$_h$ 2nd transmit PDO mapping

The object specifies the objects mapped in T_PDO2 and transmitted with the PDO. When the object is read, subindex 00_h, the number of mapped objects is read.

Object description

Index	$1A01_h$
Object name	2nd transmit PDO mapping
Object code	RECORD
Data type	PDO mapping

Value description

Subindex	00_h, number of mapped application objects in PDO
Meaning	Number of values for the object
Access	read-only
PDO mapping	–
Value range	1...8
Default value	2
Can be saved	–

Subindex	01_h, PDO mapping for the first application object to be mapped (status word)
Meaning	First object for the mapping in T_PDO2
Access	read-only
PDO mapping	–
Value range	0...4294967295
Default value	6041 0010_h
Can be saved	–

Subindex	02$_h$, PDO mapping for the second application object to be mapped (actual position)
Meaning	Second object for the mapping in T_PDO2
Access	read-only
PDO mapping	–
Value range	0...4294967295
Default value	6064 0020$_h$
Can be saved	–

The meaning of the bit states is described with the object `1st receive PDO-mapping (1600`$_h$`)`.

Settings The PDO assignment for T_PDO2 cannot be modified. The following defaults are set for "Profile Position" operating mode:

- Subindex 1: PDO mapping of the status word, object `statusword (6041`$_h$`)`
- Subindex 2: PDO mapping of the current position, object `position actual value (6064`$_h$`)`.

8.5.35 1A02$_h$ 3rd transmit PDO mapping

The object specifies the objects mapped in T_PDO3 and transmitted with the PDO. When the object is read, subindex 00$_h$, the number of mapped objects is read.

Object description

Index	1A02$_h$
Object name	3rd transmit PDO mapping
Object code	RECORD
Data type	PDO mapping

Value description

Subindex	00$_h$, number of mapped application objects in PDO
Meaning	Number of values for the object
Access	read-only
PDO mapping	–
Value range	1...8
Default value	2
Can be saved	–

Subindex	01$_h$, PDO mapping for the first application object to be mapped (status word)
Meaning	First object for the mapping in T_PDO3
Access	read-only
PDO mapping	–
Value range	0...4294967295
Default value	6041 0010$_h$
Can be saved	–

Subindex	02_h, PDO mapping for the second application object to be mapped (actual velocity)
Meaning	Second object for the mapping in T_PDO3
Access	read-only
PDO mapping	–
Value range	0...4294967295
Default value	606C 0020_h
Can be saved	–

The meaning of the bit states is described with the object `1st receive PDO-mapping` (1600_h).

Settings The PDO assignment for T_PDO3 cannot be modified. The following defaults are set for "Profile Velocity" operating mode:

- Bytes 0..1: Status word `statusword` (6041_h).
- Byte 2..5: Current Speed `velocity actual value` ($606C_h$).

8.5.36 $1A03_h$ 4th transmit PDO mapping

The object specifies the objects mapped in T_PDO4 and transmitted with the PDO. When the object is read, subindex 00_h, the number of mapped objects is read.

Object description

Index	$1A03_h$
Object name	4th transmit PDO mapping
Object code	RECORD
Data type	PDO mapping

Value description

Subindex	00_h, number of elements
Meaning	Number of values for the object
Access	read-write
PDO mapping	–
Value range	0...4
Default value	0
Can be saved	Yes

The meaning of the bit states is described under object `1st receive PDO mapping` (1600_h) .

Settings The PDO assignment for T_PDO4 cannot be modified.

9 Glossary

9.1 Units and conversion tables

The value in the specified unit (left column) is calculated for the desired unit (top row) with the formula (in the field).

Example: conversion of 5 meters [m] to yards [yd]
5 m / 0.9144 = 5.468 yd

9.1.1 Length

	in	ft	yd	m	cm	mm
in	-	/ 12	/ 36	* 0.0254	* 2.54	* 25.4
ft	* 12	-	/ 3	* 0.30479	* 30.479	* 304.79
yd	* 36	* 3	-	* 0.9144	* 91.44	* 914.4
m	/ 0.0254	/ 0.30479	/ 0.9144	-	* 100	* 1000
cm	/ 2.54	/ 30.479	/ 91.44	/ 100	-	* 10
mm	/ 25.4	/ 304.79	/ 914.4	/ 1000	/ 10	-

9.1.2 Mass

	lb	oz	slug	kg	g
lb	-	* 16	* 0.03108095	* 0.4535924	* 453.5924
oz	/ 16	-	* 1.942559*10^{-3}	* 0.02834952	* 28.34952
slug	/ 0.03108095	/ 1.942559*10^{-3}	-	* 14.5939	* 14593.9
kg	/ 0.45359237	/ 0.02834952	/ 14.5939	-	* 1000
g	/ 453.59237	/ 28.34952	/ 14593.9	/ 1000	-

9.1.3 Force

	lb	oz	p	dyne	N
lb	-	* 16	* 453.55358	* 444822.2	* 4.448222
oz	/ 16	-	* 28.349524	* 27801	* 0.27801
p	/ 453.55358	/ 28.349524	-	* 980.7	* 9.807*10^{-3}
dyne	/ 444822.2	/ 27801	/ 980.7	-	/ 100*10^{3}
N	/ 4.448222	/ 0.27801	/ 9.807*10^{-3}	* 100*10^{3}	-

9.1.4 Power

	HP	W
HP	-	* 746
W	/ 746	-

9.1.5 Rotation

	min^{-1} (RPM)	rad/s	deg./s
min^{-1} (RPM)	-	* π / 30	* 6
rad/s	* 30 / π	-	* 57.295
deg./s	/ 6	/ 57.295	-

9.1.6 Torque

	lb·in	lb·ft	oz·in	Nm	kp·m	kp·cm	dyne·cm
lb·in	-	/ 12	* 16	* 0.112985	* 0.011521	* 1.1521	* 1.129*10^6
lb·ft	* 12	-	* 192	* 1.355822	* 0.138255	* 13.8255	* 13.558*10^6
oz·in	/ 16	/ 192	-	* 7.0616*10^{-3}	* 720.07*10^{-6}	* 72.007*10^{-3}	* 70615.5
Nm	/ 0.112985	/ 1.355822	/ 7.0616*10^{-3}	-	* 0.101972	* 10.1972	* 10*10^6
kp·m	/ 0.011521	/ 0.138255	/ 720.07*10^{-6}	/ 0.101972	-	* 100	* 98.066*10^6
kp·cm	/ 1.1521	/ 13.8255	/ 72.007*10^{-3}	/ 10.1972	/ 100	-	* 0.9806*10^6
dyne·cm	/ 1.129*10^6	/ 13.558*10^6	/ 70615.5	/ 10*10^6	/ 98.066*10^6	/ 0.9806*10^6	-

9.1.7 Moment of inertia

	lb·in^2	lb·ft^2	kg·m^2	kg·cm^2	kp·cm·s^2	oz·in^2
lb·in^2	-	/ 144	/ 3417.16	/ 0.341716	/ 335.109	* 16
lb·ft^2	* 144	-	* 0.04214	* 421.4	* 0.429711	* 2304
kg·m^2	* 3417.16	/ 0.04214	-	* 10*10^3	* 10.1972	* 54674
kg·cm^2	* 0.341716	/ 421.4	/ 10*10^3	-	/ 980.665	* 5.46
kp·cm·s^2	* 335.109	/ 0.429711	/ 10.1972	* 980.665	-	* 5361.74
oz·in^2	/ 16	/ 2304	/ 54674	/ 5.46	/ 5361.74	-

9.1.8 Temperature

	°F	°C	K
°F	-	(°F - 32) * 5/9	(°F - 32) * 5/9 + 273.15
°C	°C * 9/5 + 32	-	°C + 273.15
K	(K - 273.15) * 9/5 + 32	K - 273.15	-

9.1.9 Conductor cross section

AWG	1	2	3	4	5	6	7	8	9	10	11	12	13
mm^2	42.4	33.6	26.7	21.2	16.8	13.3	10.5	8.4	6.6	5.3	4.2	3.3	2.6

AWG	14	15	16	17	18	19	20	21	22	23	24	25	26
mm^2	2.1	1.7	1.3	1.0	0.82	0.65	0.52	0.41	0.33	0.26	0.20	0.16	0.13

0198441113718, V2.01, 11.2008

9.2 Terms and Abbreviations

AC	Alternating current
CAN	(**C**ontroller **A**rea **N**etwork), standardized open fieldbus as per ISO 11898, allows drives and other devices from different manufacturers to communicate.
CANopen	Device- and manufacturer-independent description language for communication via the CAN bus
CiA	**C**AN **i**n **A**utomation, CAN interest group, standardization group for CAN and CANopen.
COB	**C**ommunication **OB**ject, transport unit in a CAN network.
COB ID	**C**ommunication **OB**ject **ID**entifier; uniquely identifies each communication object in a CAN network
DC	Direct current
Default value	Factory setting.
DriveCom	Specification of the DSP402 state machine was created in accordance with the DriveCom specification.
DS301	Standardizes the CANopen communication profile
DSP402	Standardizes the CANopen device profile for drives
E	Encoder
EDS	(**E**lectronic **D**ata **S**heet); contains the specific properties of a product.
Electronic gear	Calculation of a new output speed for the motor movement based on the input speed and the values of an adjustable gear ratio; calculated by the drive system.
EMCY object	Emergency Object
EMC	Electromagnetic compatibility
Encoder	Sensor for detection of the angular position of a rotating component. Installed in a motor, the encoder shows the angular position of the rotor.
Error	Discrepancy between a computed, observed or measured value or condition and the specified or theoretically correct value or condition.
Error class	Classification of errors into groups. The different error classes allow for specific responses to faults, for example by severity.
Fatal error	In the case of fatal error, the product is not longer able to control the motor, so that an immediate deactivation of the power stage is necessary.
Fault	Operating state of the drive caused as a result of a discrepancy between a detected (computed, measured or signaled) value or condition and the specified or theoretically correct value or condition.
Fault reset	A function used to restore the drive to an operational state after a detected error is cleared by removing the cause of the error so that the error is no longer active (transition from operating state "Fault" to state "Operation Enable").
Heartbeat	Used for unconfirmed connection acknowledgement messages from network devices.
HMI	Human Machine Interface: hand-held operating device.

I/O	Inputs/outputs
Input device	A device that can be connected via the RS232 interface; either the handheld HMI device or a PC with commissioning software.
Life guarding	For monitoring the connection of an NMT master
Limit switch	Switches that signal overtravel of the permissible range of travel.
Mapping	Assignment of object dictionary entries to PDOs
Node ID	Node address assigned to a device on the network.
NMT	Network Management (NMT), part of the CANopen communication profile; tasks include initialization of the network and devices, starting, stopping and monitoring of devices
Node guarding	Monitoring of the connection to the slave at an interface for cyclic data traffic.
Object dictionary	List of all parameters, values and functions available in the device. Each entry is uniquely referenced via index (16 bit) and subindex (8 bit).
Parameter	Device data and values that can be set by the user.
PDO	Process Data Object
Persistent	Indicates whether the value of the parameter remains in the memory after the device is switched off.
Power stage	The power stage controls the motor. The power stage generates current for controlling the motor on the basis of the positioning signals from the controller.
Quick Stop	Function used to enable fast deceleration of the motor via a command or in the event of an error.
R_PDO	Receive PDO
SDO	Service Data Object
SYNC object	Synchronization object
T_PDO	Transmit PDO
Warning	If the term is used outside the context of safety instructions, a warning alerts to a potential problem that was detected by a monitoring function. A warning is not an error and does not cause a transition of the operating state.

10 Index

0198441113718, V2.01, 11.2008

0198441113718, V2.01, 11.2008