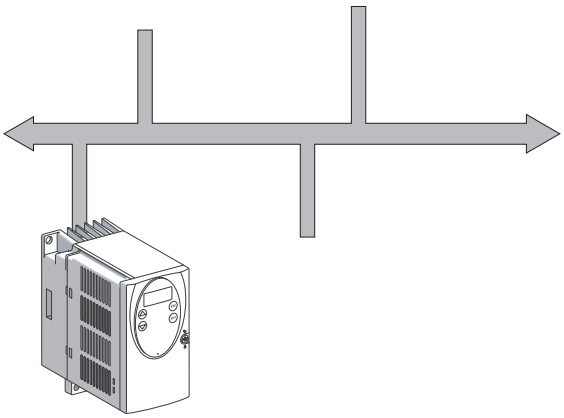
Telemecanique

Technical Documentation



Fieldbus manual

Protocol for AC servo drive

LXM05A CANopen

Document: 0198441113235

Edition: V1.04, 01.2006

a brand of Schneider Electric

Important information

The drive systems described here are products for general use that conform to the state of the art in technology and are designed to prevent any dangers. However, drives and drive controllers that are not specifically designed for safety functions are not approved for applications where the functioning of the drive could endanger persons. The possibility of unexpected or unbraked movements can never be totally excluded without additional safety equipment. For this reason personnel must never be in the danger zone of the drives unless additional suitable safety equipment prevents any personal danger. This applies to operation of the machine during production and also to all service and maintenance work on drives and the machine. The machine design must ensure personal safety. Suitable measures for prevention of property damage are also required.

See safety section for additional critical instructions.

Not all product variants are available in all countries.
Please consult the current catalogue for information on the availability of product variants.

We reserve the right to make changes during the course of technical developments.

All details provided are technical data and not promised characteristics.

In general, product names must be considered to be trademarks of the respective owners, even if not specifically identified as such.

0198441113235, V1.04, 01.2006

Table of Contents

0198441113235, V1.04, 01.2006

0198441113235, V1.04, 01.2006

0198441113235, V1.04, 01.2006

0198441113235, V1.04, 01.2006

Writing conventions and symbols

Work steps

If work steps must be carried out in sequence, they are shown as follows:

- ■ Special prerequisites for the following work steps
- ▶ Step 1
- ◁ Important response to this work step
- ▶ Step 2

If a response to a work step is specified, this will inform you that the step has been carried out correctly.

Unless otherwise stated, the individual instruction steps must be carried in the given sequence.

Lists

Lists can be sorted alphanumerically or by priority. Lists are structured as follows:

- Point 1
- Point 2
  - – Subpoint to 2
  - – Subpoint to 2
- Point 3

Making work easier

Information on making work easier can be found at this symbol:

This offers supplementary information on making work easier.
See the chapter on safety for an explanation of the safety instructions.

0198441113235, V1.04, 01.2006

0198441113235, V1.04, 01.2006

1 Introduction

1.1 CAN bus

The CAN bus (CAN:**Controller Area Network**) was originally developed for fast, economical data transmission in automotive engineering. In the meantime 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 standardised open bus, through which devices, sensors and actuators from different manufacturers communicate with each other. The features of the CAN bus are

- Multimaster capacity

  Every device in the fieldbus can send and receive data independently without being assigned to an "ordering" master function.

- Message-oriented communication

  Devices can be linked into an existing network without requiring reconfiguration of the entire system. The address of a new device does not need to be specified on the network.

- Prioritisation of messages

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

- Residual error probability

  Various backup processes in the network reduce the probability of an undetected, faulty data transfer to less than 10^{-11}. In practice, 100%-secure transmission can be assumed.

Transmission technology

In the CAN bus multiple devices are connected via a bus cable. Every network device can send 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
- drive controllers
- analysis devices
- Sensors and actuators

0198441113235, V1.04, 01.2006

1.2 CANopen technology

1.2.1 CANopen description language

CANopen is a device and manufacturer-independent description language for communication on the CAN bus. CANopen offers a unified base for exchanging commands and data between CAN bus devices.

1.2.2 Communications layers

CANopen uses the CAN bus technology for data communications.

CANopen is based on the ISO-OSI layer model on the data communications basic network service. 3 layers secure data communications in the CAN bus.

- CAN Physical Layer
- CAN Data Link Layer
- CANopen Application Layer

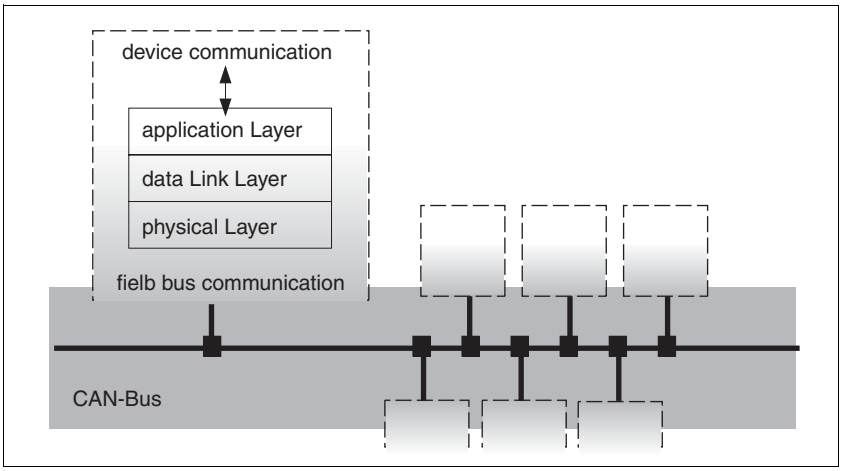


Figure 1.1 CANopen layer model

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

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

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

0198441113235, V1.04, 01.2006

1.2.3 Objects

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

The product includes 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 DSP 402 definition for object group 6000_h. In this case, the data type corresponding to DSP402 must be input.
A detailed description of all parameters can be found in the product manual in the Parameters chapter.

Object directory

The central controller connection for all objects is the object directory of every network device. Other devices find all objects here with which they can establish a connection with the device.

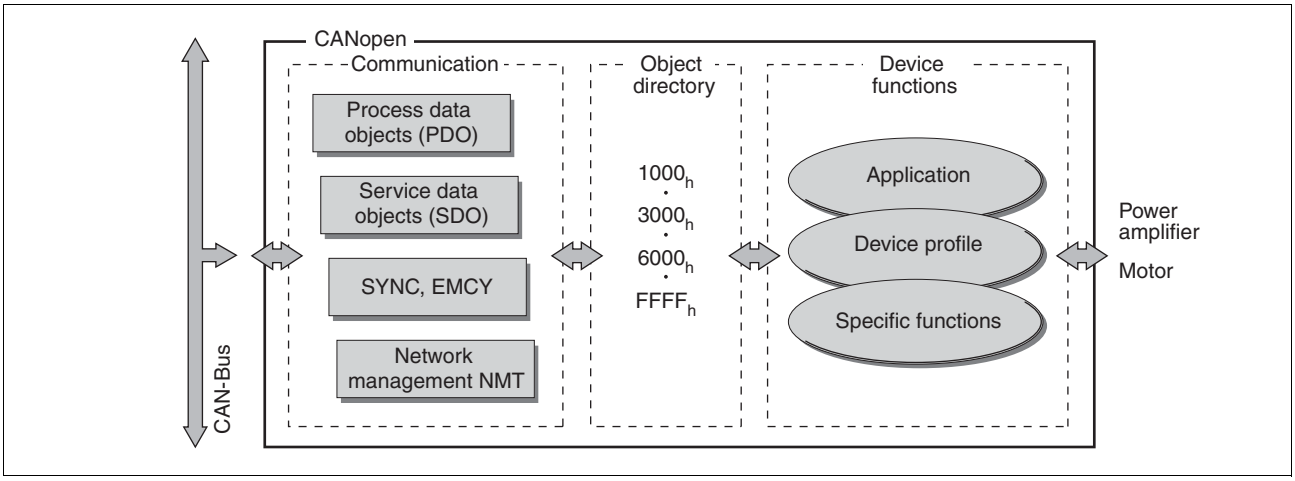


Figure 1.2 Device model with object directory

Objects for describing the data types and executing the communications tasks and device functions under CANopen are registered.

Object index

Every object is addressed over a 16-bit index, which is displayed as a four-character hexadecimal number. The objects are arranged in groups in the object directory. The following table shows an overview of the object directory as per the CANopen agreement.

Index range (hex)	Object groups
1000_h-$2FFF_h$	Communications profile
3000_h-$5FFF_h$	Manufacturer-specific objects
6000_h-$9FFF_h$	Standardised device profiles
$A000_h$-$FFFF_h$	reserved

A list of the CANopen objects can be found in chapter 9 “Object directory“.

0198441113235, V1.04, 01.2006

1.2.4 CANopen profiles

Standardised profiles

Standardised profiles describe objects that can be applied to various devices without additional configuration. The association for CAN in Automation e. V. (CiA) has standardised different profiles. They include:

- the communications profile DS 301
- the device profile DSP 402

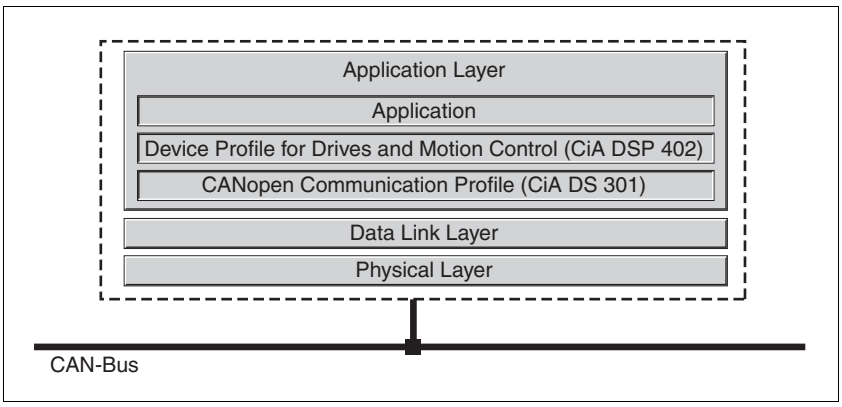


Figure 1.3 CANopen reference model

DS301 communications profile

The DS 301 communications profile forms the interface between device profiles and CAN bus. It was specified in 1995 under the name DS 301 and defines unified standards for common data exchange between different device types under CANopen.

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

DSP 402 device profile

The DSP 402 device profile describes standardised objects for positioning, monitoring and settings of drives. The tasks of the objects are:

- Device control and status monitoring (Device Control)
- Standardised parameter setting
- Switching, verification and execution of operating modes

Manufacturer-specific profiles

The basic functions of a device can be used with device profiles standardised with objects. Only manufacturer-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 them.

0198441113235, V1.04, 01.2006

1.3 Documentation and literature references

CAN interest group

CiA - CAN in Automation
Am Weichselgarten 26
D-91058 Erlangen
http://www.can-cia.org/

CANopen standards

- CiA Draft Standard 301 (DS 301)
  CANopen application layer and communication profile
  V4.02, February 2002
- CiA Draft Standart Proposal 402 (DSP 402)
  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

Documentation

- Product manual of the AC servo amplifier Lexium05
- CANopen fieldbus manual

0198441113235, V1.04, 01.2006

0198441113235, V1.04, 01.2006

2 Safety

2.1 Qualification of personnel

Only technicians who are familiar with and understand the contents of this manual and the other relevant manuals are authorised to work on and with this drive system. The technicians must be able to detect potential dangers that may be caused by setting parameters, changing parameter values and generally by the mechanical, electrical and electronic equipment.

The technicians must have sufficient technical training, knowledge and experience to recognise and avoid dangers.

The technicians must be familiar with the relevant standards, regulations and safety regulations that must be observed when working on the drive system.

2.2 Intended use

The drive systems described here are products for general use that conform to the state of the art in technology and are designed to prevent any dangers. However, drives and drive controllers that are not specifically designed for safety functions are not approved for applications where the functioning of the drive could endanger persons. The possibility of unexpected or unbraked movements can never be totally excluded without additional safety equipment. For this reason personnel must never be in the danger zone of the drives unless additional suitable safety equipment prevents any personal danger. This applies to operation of the machine during production and also to all service and maintenance work on drives and the machine. The machine design must ensure personal safety. Suitable measures for prevention of property damage are also required.

In the system configuration described the drive systems must be used in industrial applications only and must have a fixed connection only.

In all cases the applicable safety regulations and the specified operating conditions, such as environmental conditions and specified technical data, must be observed.

The drive system must not be commissioned and operated until completion of installation in accordance with the EMC regulations and the specifications in this manual.

To prevent personal injury and damage to property damaged drive systems must not be installed or operated.

Changes and modifications of the drive systems are not permitted and if made all no warranty and liability will be accepted.

The drive system must be operated only with the specified wiring and approved accessories. In general, use only original accessories and spare parts.

The drive systems must not be operated in an environment subject to explosion hazard (ex area).

0198441113235, V1.04, 01.2006

2.3 General safety instructions

⚠ DANGER

Risk of injury by complex system.

When the system is started the drives are generally out of the operator's view and cannot be visually monitored.

- Only start the system if there are no persons in the operating zone of the moving components and the system can be operated safely.

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

⚠ WARNING

Danger of injury by loss of control!

- Observe the accident prevention regulations. (For USA see also NEMA ICS1.1 and NEMA ICS7.1)
- The system manufacturer must take the potential error possibilities of the signals and the critical functions into account to ensure a safe state during and after errors. Some examples are: emergency stop, final position limitation, power failure and restart.
- The assessment of error possibilities must also include unexpected delays and the failure of signals or functions.
- Suitable redundant control paths must be in place for dangerous functions.
- Check that measures taken are effective.

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

0198441113235, V1.04, 01.2006

3 Basics

3.1 Communications profile

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

The following can be done with the access to the objects of the network devices

- exchange parameter values
- start movement functions of individual CAN bus devices
- query status information

3.1.1 Object directory

Every CANopen device administers an object directory, in which all objects for communications are listed.

Index, subindex

The objects are addressed in the object directory with a 16-bit long index. One or more 8-bit-long subindex entries to every object point to individual data fields in the object. Index and subindex are shown in hexadecimal characters, recognisable by the attached "$_h$".

Example

The following table shows index and subindex entries with the example of the object `software position limit (607D`$_h$`)` for identifying the position of the software limit switch.

Index	Subindex	Name	Meaning
$607D_h$	00_h	-	Number of data fields
$607D_h$	01_h	min. position limit	Bottom limit value switch
$607D_h$	02_h	max. position limit	Top limit value switch

Table 3.1 Example for index and subindex entries

Object descriptions in the manual

The objects of the following object groups are described to distinguish them for the CANopen programming of a device:

- $1xxx_h$ objects: Communications objects in this chapter
- $3xxx_h$ objects: Manufacturer-specific objects required for the control of the device, in chapter 6 “Operation“.
- $6xxx_h$ objects: Standardised objects of the device profile in chapter 6 “Operation“

Standardised objects

Standardised objects form the basis of applying the same applications for the various network devices of a device type. This requires the devices to list the objects in their directory. Standardised objects are defined in the DS 301 communications profile and the DSP 402 device profile.

0198441113235, V1.04, 01.2006

3.1.2 Communications objects

Overview

The communications objects are standardised with the DS301 CANopen communications profile. The objects can be classified into four groups according to their tasks.

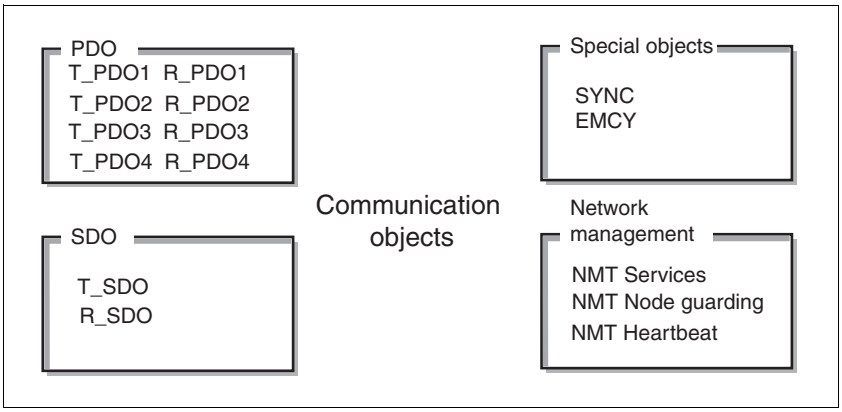


Figure 3.1 The following are considered communications objects from the point of view of the device: T_..: "Transmit", R_..: "Receive"

- PDO (process data object) for real-time transmission of process data
- SDO (service data object) for read and write access to the object directory
- Objects for controlling CAN messages:
  - SYNC object (synchronisation object) for synchronisation of network devices
  - EMCY object (emergency object) for the error display of a device or its peripheral equipment.
- Network management services:
  - NMT services for initialisation and network control (NMT: network management)
  - NMT Node Guarding for monitoring the network devices
  - NMT heartbeat for monitoring the network devices

CAN message

Data are exchanged on the CAN bus as CAN messages. A CAN message sends the communications object and a variety of administration and control information to ensure data transmission without loss and errors.

0198441113235, V1.04, 01.2006

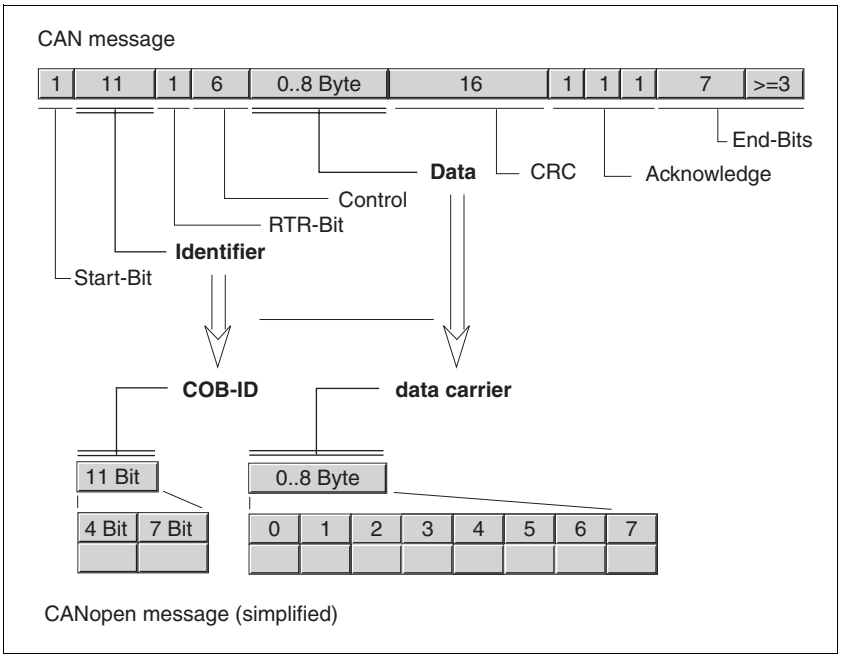


Figure 3.2 CAN message and simplified display of CANopen message

CANopen message

The CAN message can be displayed in simplified form for work with CANopen objects and for data exchange, because most of the bits are used to ensure error-free data transmission. These bits are automatically removed from the received message by the data security layer, the data link layer of the OSI layer model, and added to a message before transmission.

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 maximum 8-byte data frame of a CANopen message.

COB-ID

The COB Id (**C**ommunication **OB**ject **Id**entifier) has two tasks in the control of communications objects:

- Bus arbitration: specification of transmission priorities
- identification of communications objects

An 11-bit COB identifier as per the CAN 3.0A specification is defined for CAN communications. It comprises two parts:

- Function code, 4 bit size
- Node-ID, 7 bit size.

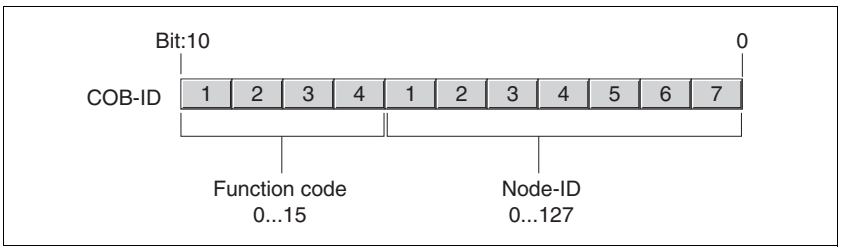


Figure 3.3 COB Id with function code and node address

COB-IDs of the communications objects

The following table shows the COB-IDs of all communications objects in the factory setting. The column "index of object parameters" shows the

0198441113235, V1.04, 01.2006

index of special objects with which the settings of the communications objects can be read or modified by SDO.

Communications object	Function code	Node address-node-Id [1...127]	COB-IDdecimal (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	0 0 1 1	x x x x x x x	384 (180_h) + node-Id	1800_h
R_PDO1	0 1 0 0	x x x x x x x	512 (200_h) + node-Id	1400_h
T_PDO2	0 1 0 1	x x x x x x x	640 (280_h) + node-Id	1801_h
R_PDO2	0 1 1 0	x x x x x x x	768 (300_h) + node-Id	1401_h
T_PDO3	0 1 1 1	x x x x x x x	896 (380_h) + node-Id	1802_h
R_PDO3	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 communications objects

COB Ids of PDOs can be changed as required. The assignment scheme for COB Ids specifies only one basic setting.

Function code The function code classifies the communications objects. Because the bits of the function code in the COB Id are significantly higher, the function code simultaneously controls the transmission priorities: Objects with a small function code are sent at high priority. For example, with simultaneous bus access an object with the function code "1" is sent before an object with the function code "3".

Node address Every network device is configured before network operation. It is given a unique, 7-bit-long 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 the messages to all devices simultaneously.

Example **Selection of a COB-Id**

For a device with the node address 5, the COB-Id of the communications object T PDO1 is:

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

0198441113235, V1.04, 01.2006

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 are described with the relevant communications objects.

3.1.3 Communications relationships

CANopen uses three relationships for communications between network devices:

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

Master-slave relationship

A "master" in the network controls the message traffic. A "slave" only responds when addressed by the master.

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

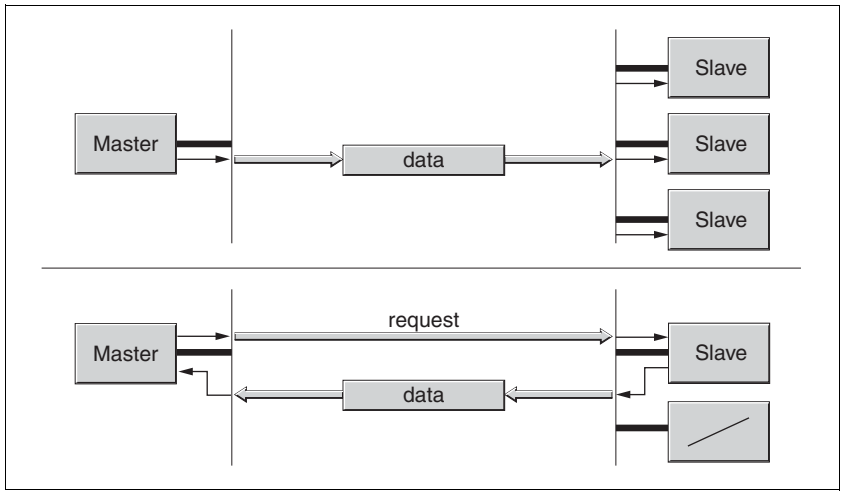


Figure 3.4 Master-slave relationships

The exchange of messages can be executed unconfirmed and confirmed. If the master sends an unconfirmed CAN message, it can be received by multiple 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 always established between two devices. The "server" is the device whose object list is used during the data exchange. The "client" addresses and starts the exchange of messages and waits for a response from the server.

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

0198441113235, V1.04, 01.2006

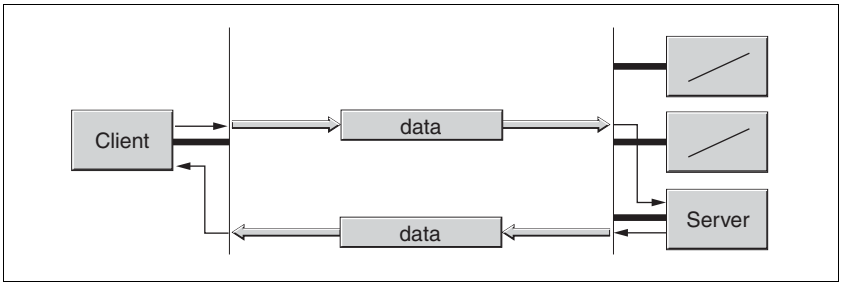


Figure 3.5 Client-server relationship

The client addresses and sends a CAN message to a server. The server evaluate the message and sends the answer data as response.

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.

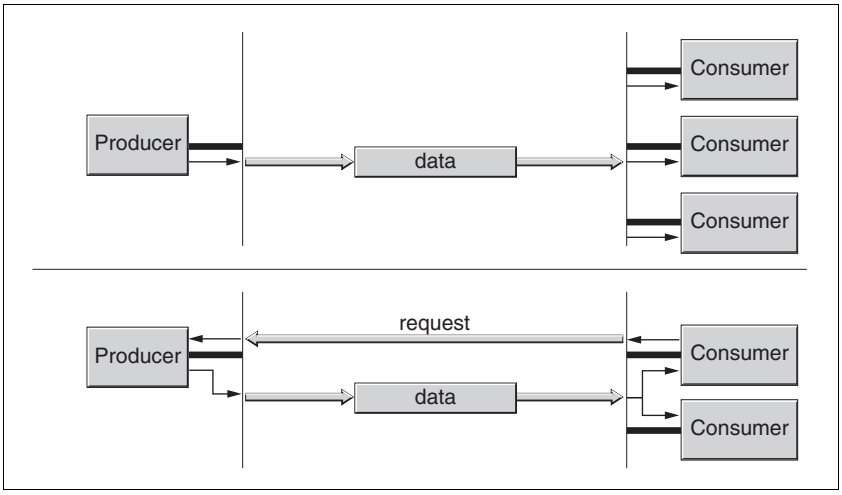


Figure 3.6 Producer-consumer relationships

The producer sends a message that can be received by one or more network devices. The producer does not receive a receipt response. The message transmission can be triggered

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

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

0198441113235, V1.04, 01.2006

3.2 Service data communication

3.2.1 Overview

Service data objects (SDO: **S**ervice **D**ata **O**bject) can be used to access the entries of an object directory 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 directory of a different device or to change them there.

The T_SDO of a SDO client can be used to send the request for data exchange and to receive with the R_SDO. The data frame of a SDO is always 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 directory a SDO message refers.

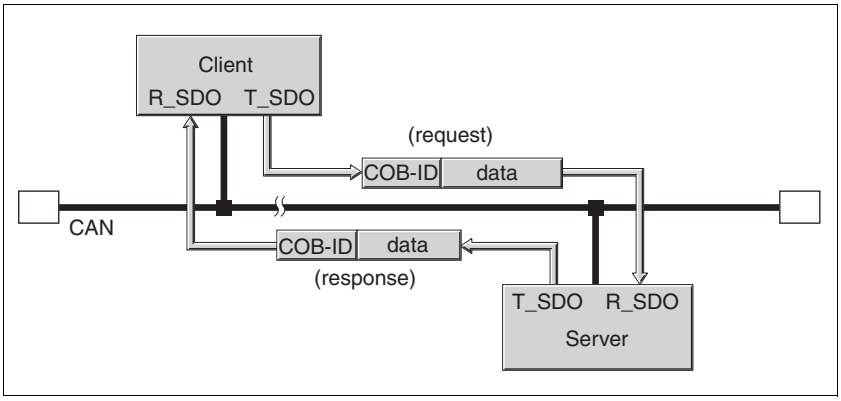


Figure 3.7 SDO message exchange with request and response

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

0198441113235, V1.04, 01.2006

3.2.3 SDO message

A SDO message in simplified form consists of the COB-Id and the SDO data frame, in which up to four bytes of data can be sent. Longer data strings are distributed over multiple SDO messages with a special protocol.

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

Example The following diagram shows an example of a SDO message.

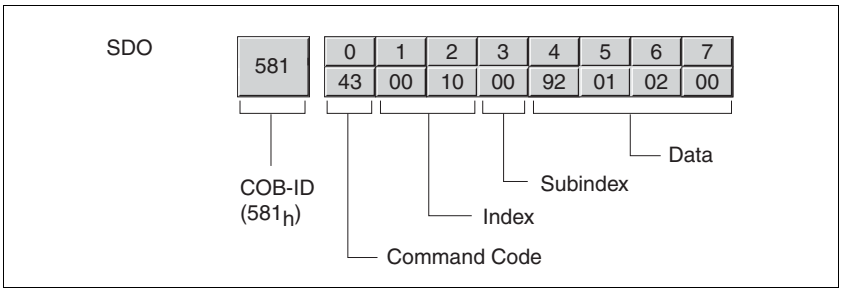


Figure 3.8 SDO message, example

COB-ID and data frame R_SDO and T_SDO have different COB-Ids.
The data frame of a SDO messages includes the following:

- Command code 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 are transported with the SDO message
- Data that comprise up to 4 bytes

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

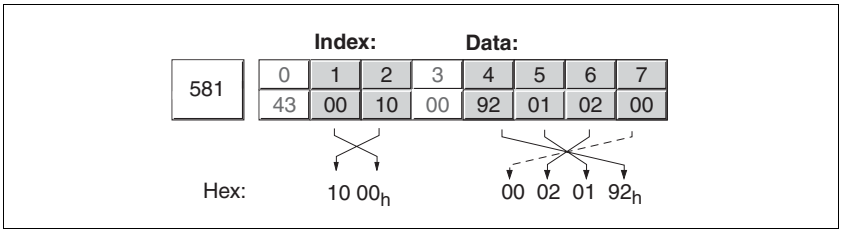


Figure 3.9 Repositioning numeric values greater than 1 byte

0198441113235, V1.04, 01.2006

3.2.4 Read and write data

Write data

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

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

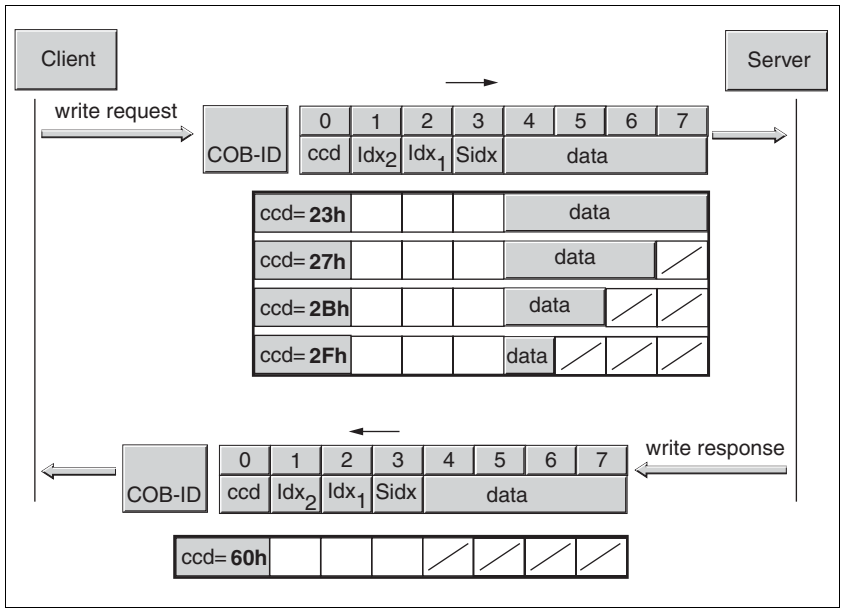


Figure 3.10 Writing parameter values

Unused bytes in the data field are shown with a slash in the graphic. The content 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 byte	**3 byte**	**2 byte**	**1 byte**	
write request	23_h	27_h	$2B_h$	$2F_h$	Send parameters
write response	60_h	60_h	60_h	60_h	response
error response	80_h	80_h	80_h	80_h	Error

Table 3.3 Command codes for writing parameter values

Read data

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

The server responds to the query with 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".

0198441113235, V1.04, 01.2006

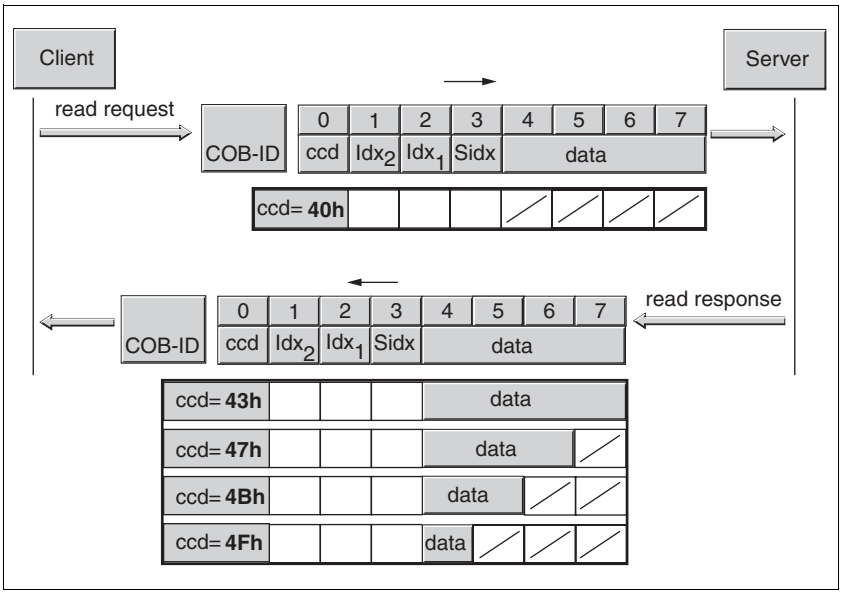


Figure 3.11 Reading parameter value

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

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

Message type	Data length used				
	4 byte	3 byte	2 byte	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 sending 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“.

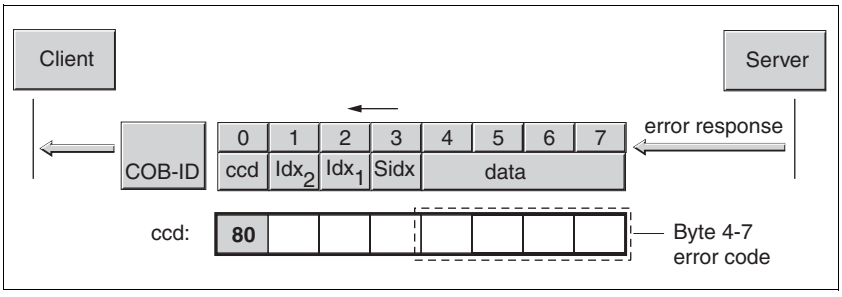


Figure 3.12 Response with error message (error response)

0198441113235, V1.04, 01.2006

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 setpoint or operating status of the device. The transmission can be executed very fast, because it is sent without additional administration data and does not require a response from the recipient.

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

The length of a PDO message and the allocation of the data fields is 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 the process data.

3.3.2 PDO data exchange

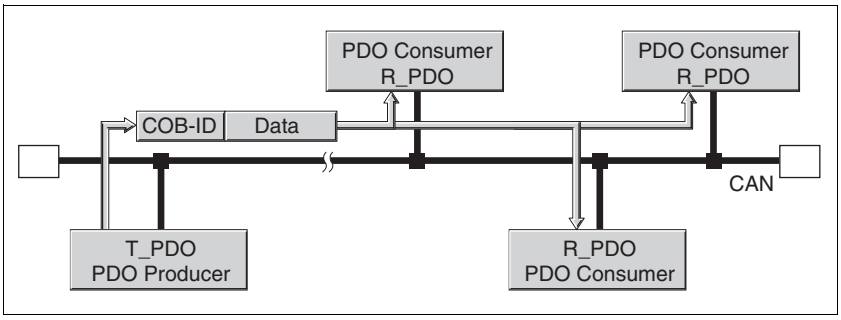


Figure 3.13 PDO data exchange

Data exchange with PDOs conforms to the producer-consumer relationship and can be triggered by three methods

- synchronised
- event-driven, asynchronous
- by request of a consumer, asynchronous

The synchronised data processing is controlled by the SYNC object. Synchronous PDO messages are sent immediately like the standard PDO messages, but are only evaluated on the next SYNC. For example, multiple drives can be started simultaneously by synchronised data exchange.

The device evaluates PDO messages that are called on request or are event-controlled immediately.

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

0198441113235, V1.04, 01.2006

3.3.3 PDO message

T_PDO, R_PDO

A PDO always is available for sending and receiving a PDO message:

- The T_PDO for sending PDO messages (T: Transmit),
- The R_PDO for receiving PDO messages (R: Receive).

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

The device uses 8 PDOs, 4 receive PDOs and 4 send PDOs. All PDOs are evaluated or transmitted event-controlled in the default setting.

PDO settings

The settings for PDOs can be red and changed with 8 communications objects:

Object	Description
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 Communications objects for PDO

Enable PDO

In the default setting of the PDOs R_PDO1 and T_PDO1 are enabled. The other PDOs must be enabled first.

A PDO is enabled with bit 31 (valid bit) in subindex 01_h of that communications object:

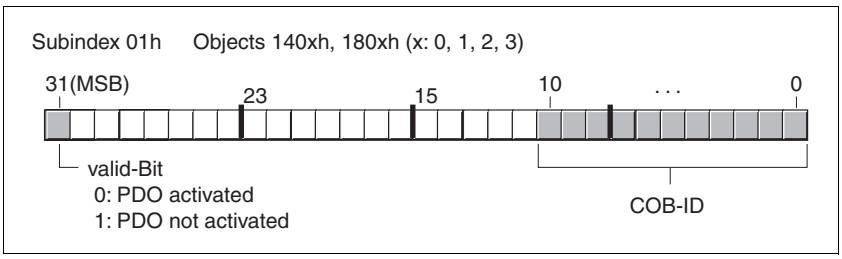


Figure 3.14 Enable PDOs with subindex 01_h, enable bit 31

Example

Setting for R_PDO3 in object 1402_h

- subindex 01_h = 8000 $04xx_h$: R_PDO3 not enabled
- subindex 01_h = 0000 $04xx_h$: R_PDO3 enabled.

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

PDO time intervals

The time intervals "inhibit time" and "event timer" can be set for every send PDO.

0198441113235, V1.04, 01.2006

- 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 interval time that is not equal to zero is entered, a sent PDO will only be sent again when the interval time expires. The time is set with subindex 03_h.
- The time interval "event timer" triggers an event message periodically. After the interval time has expired 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 specified. The object that is represented in the PDO R_PDO4 can be modified by PDO mapping.

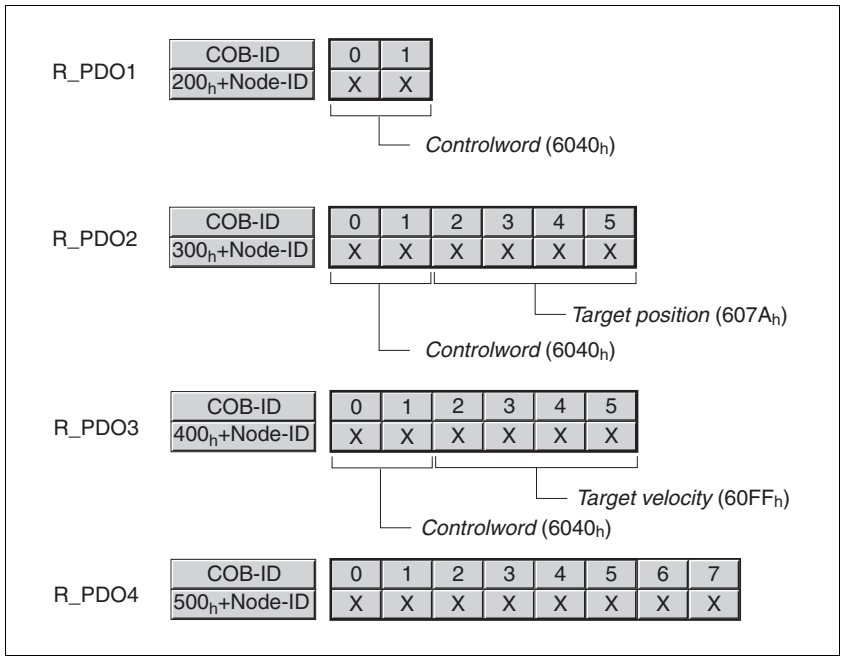


Figure 3.15 Receive PDOs

R_PDO1 In the first receive PDO the control word, object `controlword (6040h)`, of the status machine is represented, which can be used to set the operating status of the device.

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

R_PDO2 With the second receive PDO the control word and the target position of a travel command, object `target position (607Ah)`, is received for a profile positioning in the "profile position mode".

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

For details on the SYNC object see chapter 3.4 “Synchronisation“.

R_PDO3 In the third receive PDO the control word and the setpoint speed, object `Target velocity (60FFh)`, is mapped for the speed mode in the "profile velocity mode".

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

0198441113235, V1.04, 01.2006

R_PDO4 Manufacturer-specific object values are transmitted with the fourth receive PDO. R_PDO4 is empty by default.

R_PDO4 is evaluated asynchronously, i.e. is event-controlled. R_PDO4 can be used to map various manufacturer-specific objects with PDO mapping.

Transmit PDOs The objects for T_PDO1, T_PDO2 and T_PDO3 are permanently specified. The object that is represented in the PDO T_PDO4 can be modified by PDO mapping.

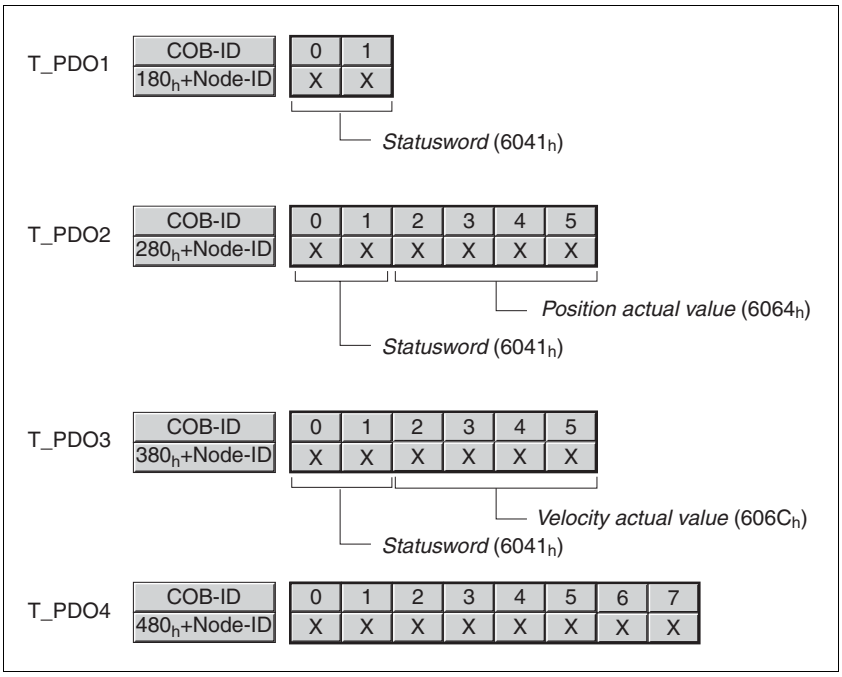


Figure 3.16 Send PDOs

T_PDO1 In the first transmit PDO the status word, object `statusword (6041h)`, of the status machine is mapped.

T_PDO1 is sent asynchronously and event-controlled at every change of the status information. No other objects can be mapped with T_PDO1.

T_PDO2 In the second send PDO the status word and the current position of the motor, object `Position actual value (6064h)`, is mapped to monitor a profile positioning in the "profile position mode"

T_PDO2 is sent after receipt of a SYNC object and event-controlled. No other objects can be mapped with T_PDO2.

T_PDO3 In the third send PDO the status word and the current speed, object `Velocity actual value (606Ch)`, is mapped for monitoring the speed mode in "profile velocity mode".

T_PDO3 is sent asynchronously and event-controlled at every change of the status information. No other objects can be mapped with T_PDO3.

T_PDO4 Manufacturer-specific object values (for monitoring) are sent with the fourth send PDO. T_PDO4 is empty by default.

0198441113235, V1.04, 01.2006

T_PDO4 is sent asynchronously and event-controlled at every change of the status information. The specification of which objects trigger an event can be set with the parameter `CANpdo4Event`. With the default setting of the parameter all mapped objects trigger an event.

T_PDO4 can be used to map various manufacturer-specific objects with PDO mapping.

Parameter Name Code HMI menu, Code	Description	Unit Minimum value Default value Maximum value	Data type R/W persistent Expert	Parameter address via fieldbus
CANpdo4Event -	PDO4 event mask() Value changes in the object trigger 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 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 directory can be sent with a PDO message. The mapping of data in a PDO message is referred to as PDO mapping.

Figure 3.17 shows data exchange between PDOs and object directory with two examples of objects in T_PDO4 and R_PDO4 of the PDOs.

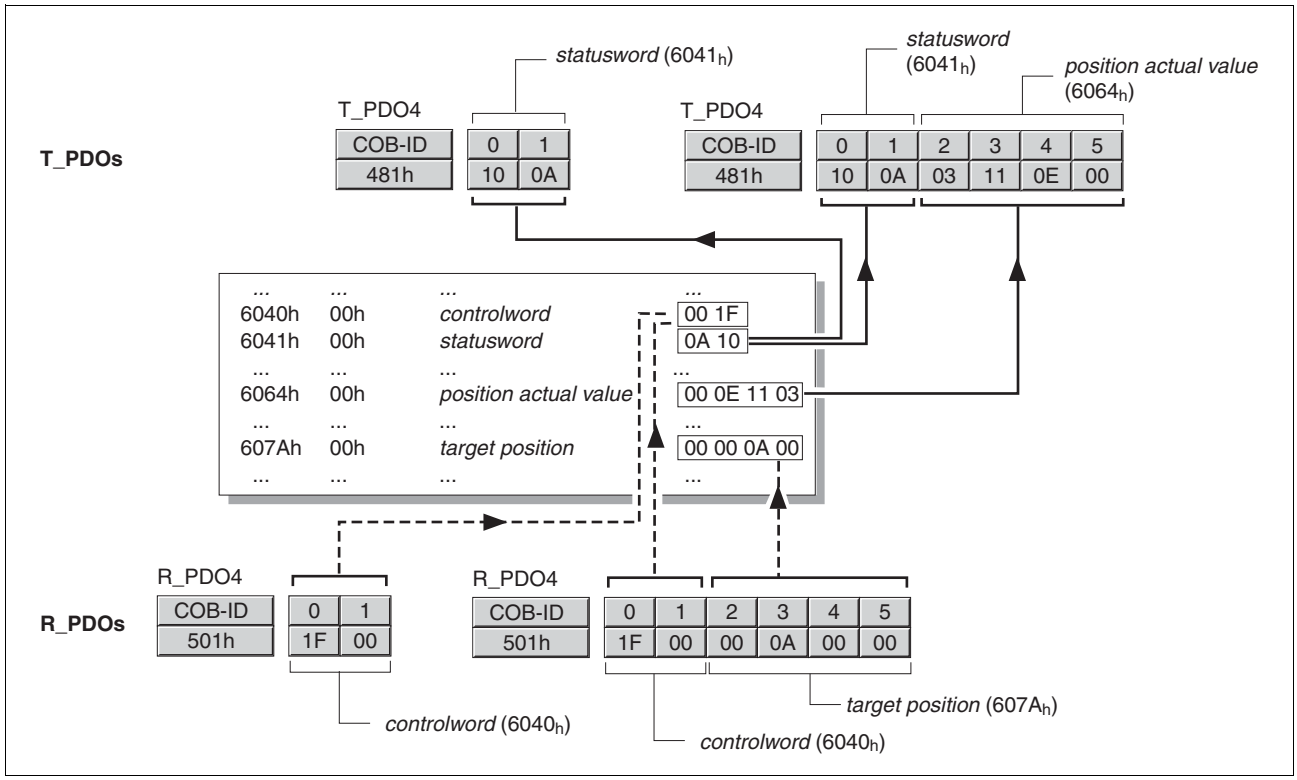


Figure 3.17 PDO mapping, in this case for a device with node address 1

0198441113235, V1.04, 01.2006

Static PDO mapping

The device uses static and dynamic PDO mapping. In static PDO mapping all objects are mapped in accordance with a fixed, non-modifiable setting in the relevant PDO.

The settings for PDO mapping are defined in an assigned communications object for every 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. Every communications 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 occupies in the PDO.

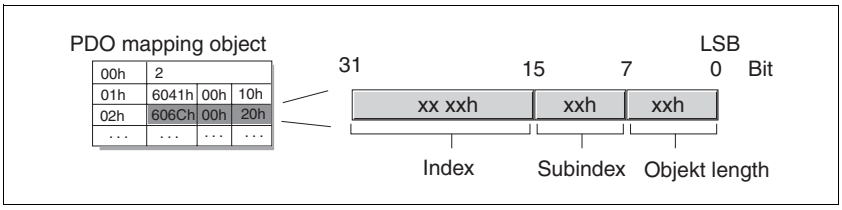


Figure 3.18 Structure of entries for the PDO mapping

The number of valid subindex entries is contained in subindex 00_h of the communications object.

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

0198441113235, V1.04, 01.2006

Object (Index:Subindex)	PDO	Data type
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

0198441113235, V1.04, 01.2006

3.4 Synchronisation

The synchronisation 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 sent to all devices by a network device and can be evaluated by all devices that support synchronous PDOs.

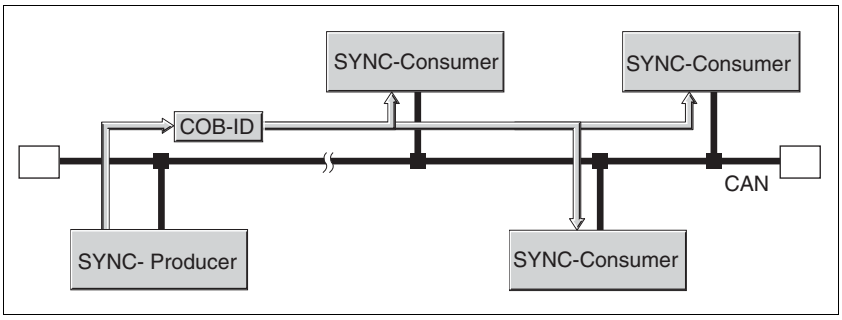


Figure 3.19 SYNC message

Time values for synchronisation

2 time values define the behaviour of synchronous data transfer:

- The cycle time specifies the time intervals between 2 SYNC messages. It is set with the object `Communication cycle period(1006h)`.
- The synchronous time window specifies the time interval in which the synchronous PDO messages must be received and sent. The time window is defined with the object `Synchronous window length (1007h)`.

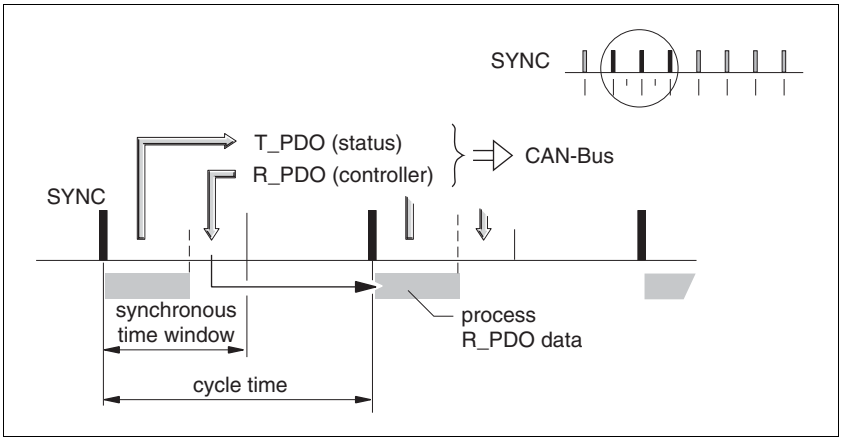


Figure 3.20 Synchronisation periods

Synchronous data transmission

From the point of view of a SYNC receiver, the status data are first sent in a T_PDO and the new control data are received via an R_PDO in one time window. However, the control data are only processed when the next SYNC message is received. The SYNC object itself does not transmit data.

Cyclic ad acyclic data transfer

Synchronous exchange of messages can be executed cyclically or acyclically.

0198441113235, V1.04, 01.2006

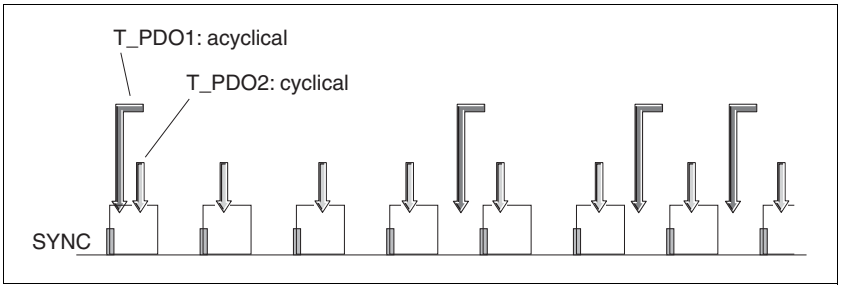


Figure 3.21 Cyclic and acyclic transmission

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

If a synchronous PDO message is sent acyclically, it can be sent or received at any time, but will only be valid with the next SYNC message.

The cyclic or acyclic behaviour of PDOs is stored in subindex `transmission type` (02_h) of the corresponding PDO parameter, e.g. for R_PDO1 in the object `1st receive PDO parameter` (1400_h:02_h).

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 initialising the network with the object `COB-ID SYNC Message` (1005_h) .

"Start" PDO

In the default setting 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 synchronisation allows an operating mode to be started simultaneously on multiple devices and, for example, synchronisation of the feed of a multi-motor portal drive.

0198441113235, V1.04, 01.2006

3.5 Emergency service

The emergency service reports internal device error over the CAN bus. The error message is sent to all devices with an EMCY object in accordance with the consumer-producer relationship.

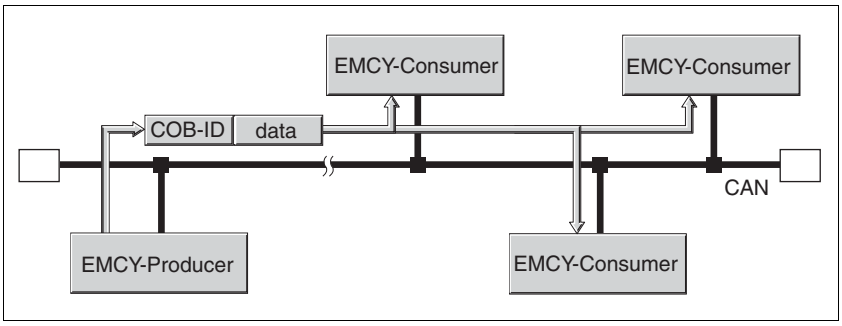


Figure 3.22 Error message via EMCY objects

Boot Up message

The communications profile DS 301, 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 sent the message is ready for operation in the CAN network.

The boot-up message is sent 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 error status as per the CANopen status machine. At the same it sends an EMCY message with error register and error code.

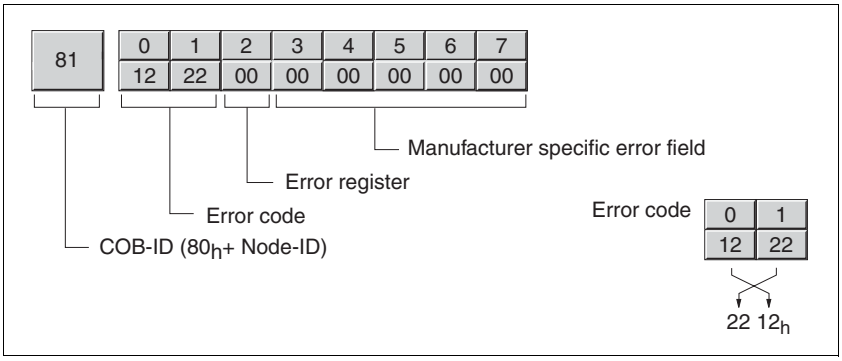


Figure 3.23 EMCY message

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

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

Byte 3, 4 - Manufacturer-specific error code of mapped object

Byte 5, 6 - Index of mapped object

Byte 7 - Subindex of mapped object

0198441113235, V1.04, 01.2006

COB-ID

The COB-Id is calculated from the node address for every device in the network that supports an EMCY object:

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 register reports the error status of the device in bit-coded form. Bit 0 remains set so long as an error is pending. The remaining bits identify the error type. The precise cause of error can be found with the error code. The error code is sent in Intel format as a 2-byte value and must be reversed by bytes for evaluation.

A list of all error messages and responses by the device and 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 code (`$603F_h$`)`. The last 20 error messages are backed up in sequence of occurrence in the object `FLT_err_num (303C:1`$_h$`)`. `FLT_MemReset (303B:5`$_h$`)` resets the read flag of the error memory to the oldest error.

0198441113235, V1.04, 01.2006

3.6 Network management services

Network management (NMT) is a component of the CANopen communications profile and is used to initialise the network and 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 through their node address. A message with node address "0" is directed to all NMT slaves simultaneously.

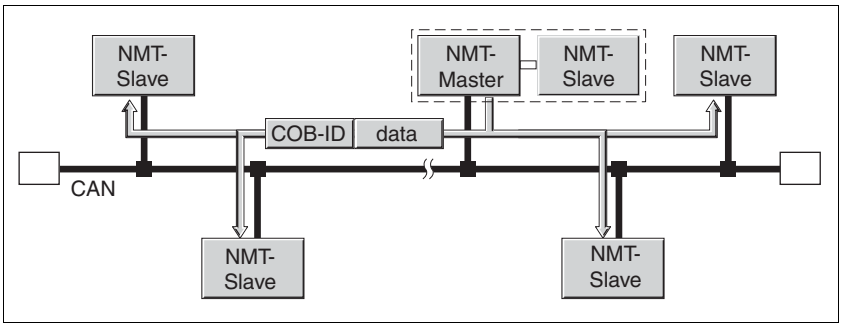


Figure 3.24 NMT services over the master-slave relationship

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

NMT services NMT services can be divided into two groups:

- Services for device control, to initialise devices for CANopen communications and to control the behaviour of devices in network operation
- Services for connection monitoring, to ensure error-free network operation

3.6.1 NMT services for device control

NMT status machine The NMT status machine describes the initialising and status of an NMT slave in mains operation.

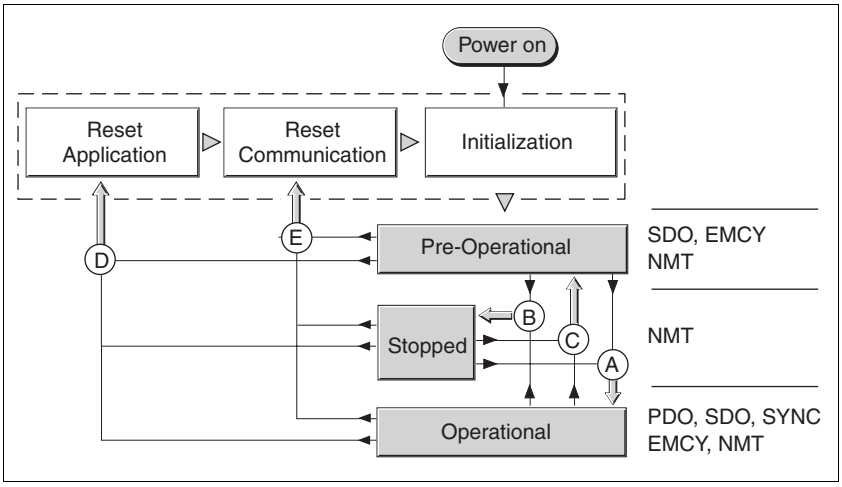


Figure 3.25 NMT status machine and available communications objects

The graphic shows on the right side all communications objects that can be used in the specific network status.

0198441113235, V1.04, 01.2006

Initialisation

A NMT slave automatically runs through an initialisation phase after the supply voltage is switched on (power on) to prepare it for CAN bus operation. On completion of the initialising process the slave switches to the "pre-operational" status and sends a boot-up message. Now a NMT master can control the operational behaviour of a NMT slave in the network with 5 NMT services, shown in the above graph with the letters A to E.

NMT service	Transition	Description
Start remote node (Start network nodes)	A	Switch to "Operational" status Start normal mains operation to all devices
Stop remote node (Stop network nodes)	B	Switch to "Stopped" status Stop communications of the device in the network. If connection monitoring is active, it remains switched on. With an active power amplifier (status "Operation Enabled" or "Quick Stop") an error of error class 2 is triggered. The drive is stopped and switched off.
Enter Pre-Operational (Switch to "Pre-Operational")	C	Switch to "Pre-Operational" status All communications objects except for PDOs can be used. The "Pre-Operational" status can be used for configuration by SDOs: - PDO mapping - start of synchronisation - start of connection monitoring
Reset node (Reset nodes)	D	Switch to "Reset application" status Load saved data of the device profiles and switch automatically to "pre-operational" via "Reset communication"
Reset communication (Reset communications data)	E	Switch to "Reset communication" status Load stored data of the communication profile and switch automatically to "Pre-Operational" status. With an active power amplifier (status "Operation Enabled" or "Quick Stop") an error of error class 2 is triggered. The drive is stopped and switched off.

Non-volatile saved data

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

NMT message

The NMT services for device control are sent as unconfirmed message with the COB-ID = 0. By default they receive top priority on the CAN bus.

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

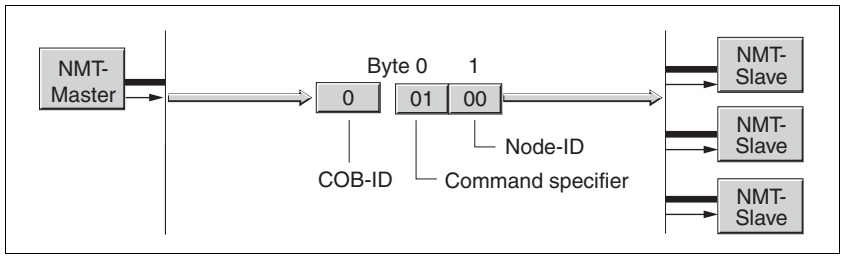


Figure 3.26 NMT message

The first byte, the "command specifier" identifies the NMT service in use.

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

0198441113235, V1.04, 01.2006

Command Specifier	NMT service	Transition
130 (82_h)	Reset communication	E

The second byte addresses the receiver of a 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 services for connection monitoring

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

Three NMT services for connection monitoring are available:

- "Node guarding" for monitoring the connection of a NMT slave
- "Life guarding" (monitoring for signs of life) for monitoring the connection of a NMT master
- "Heartbeat" for the unconfirmed connection message from network devices.

3.6.2.1 Node/Life guarding

COB-ID

Connection monitoring is executed with the communications object `NMT error control (700h+node-Id)`. The COB-ID for every NMT slave is calculated from the node address:

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

Structure of the NMT message

On request of the NMT master the NMT slave responds with one data byte.

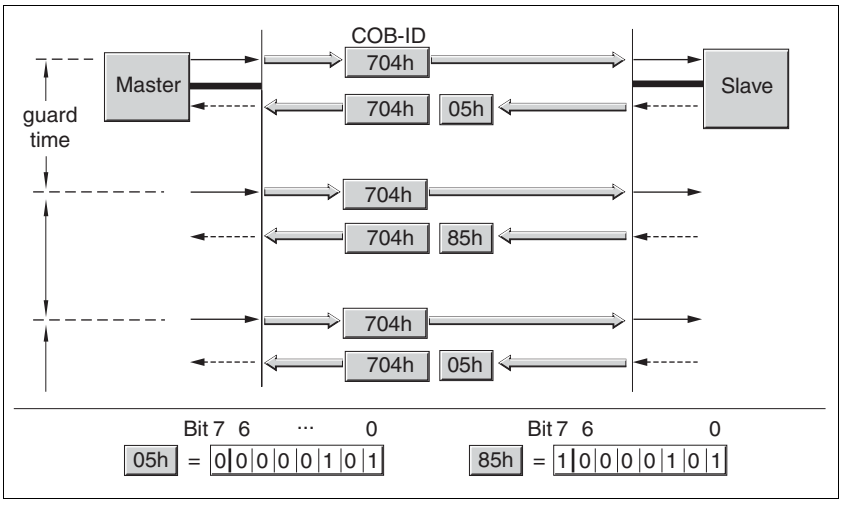


Figure 3.27 Acknowledgement of the NMT slave

Bit 0 to 6 identify the NMT status of the slave:

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

0198441113235, V1.04, 01.2006

After every interval "guard-time" bit 7 switches its status between "0" and "1", so the NMT master can detect and ignore a second acknowledgement within the "guard-time" interval time. The first request when starting connection monitoring begins with bit 7 = 0.

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

In NMT status "Stopped" the connection monitoring continues to operate.

Configuration

Node/life guarding is configured by:

- guard time ($100C_h$)
- life time factor ($100D_h$)

Connection error

The NMT master reports a connection error to the higher level master program if:

- the slave does not acknowledge within the "guard-time" period
- the NMT status of the slave has changed without the initiation of the NMT master.

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

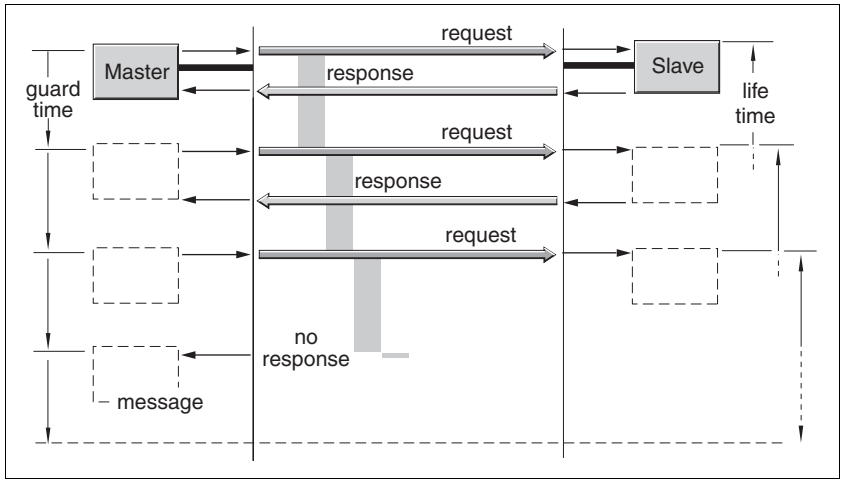


Figure 3.28 "Node guarding" and "Life guarding" with time intervals

0198441113235, V1.04, 01.2006

3.6.2.2 Heartbeat

The optional heartbeat protocol replaces the node/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 more consumers can receive this message. `Producer heartbeat time` (1016_h) `= 0` disables heartbeat monitoring.

The relationship between producer and consumer can be configured with objects. If a consumer does not receive a signal within the time interval specified in the object `Consumer heartbeat time` (1016_h), it generates an error message (heartbeat event). `Consumer heartbeat time` (1016_h) `= 0` disables the monitoring by a consumer.

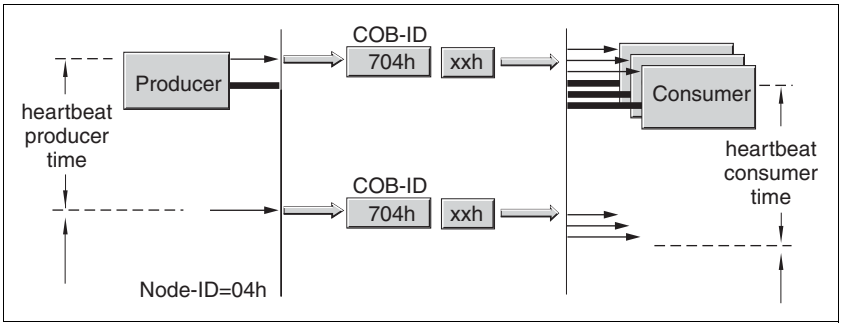


Figure 3.29 "Heartbeat" monitoring

Data byte for NMT status 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 1-ms steps and must not be set smaller for the consumer than 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 active during the NMT status change to "Pre-Operational", the "heartbeat" monitoring starts by sending the boot up message. The boot up message is a heartbeat message with one data byte 00_h.

Devices can monitor each other by "heartbeat" message. They have consumer and producer function simultaneously.

0198441113235, V1.04, 01.2006

4 Installation

⚠ WARNING

Danger of injury by loss of control!

- Observe the accident prevention regulations. (For USA see also NEMA ICS1.1 and NEMA ICS7.1)
- The system manufacturer must take the potential error possibilities of the signals and the critical functions into account to ensure a safe state during and after errors. Some examples are: emergency stop, final position limitation, power failure and restart.
- The assessment of error possibilities must also include unexpected delays and the failure of signals or functions.
- Suitable redundant control paths must be in place for dangerous functions.
- Check that measures taken are effective.

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

⚠ WARNING

Interference with signals and devices may cause injury

Distorted signals can cause unexpected device responses.

- Install the wiring in accordance with the EMC requirements.
- Check compliance with the EMC requirements, particularly in an environment subject to strong interference.

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

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

0198441113235, V1.04, 01.2006

0198441113235, V1.04, 01.2006

5 Commissioning

5.1 Setting up the device

For installation in the network the device must be mechanically and electrically installed correctly and the device must be successfully commissioned.

Set up the device following the 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. Every unit is identified by a unique address. The default node address for a unit is 127.

The baud rate is preset to 125 kbaud.

Every unit must be assigned its own node address, i.e. every node address must be assigned only once in the network.

Setting address and baud rate

The address is set locally at the unit in the parameter `canAddr` and the baud rate in the parameter `canBaud`.

The baud rate must be the same for all units in the field bus.

5.3 SyCon CANopen configuration software

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

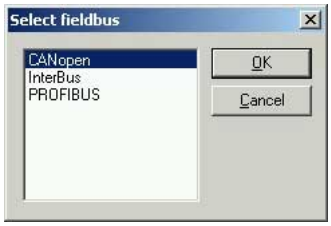
▶ Carry out the following steps:

0198441113235, V1.04, 01.2006

5.3.1 Create new network

A new network is created using the menu option "File - New".

- ▶ Select CANopen as the fieldbus network.
- ▶ Confirm your selection by clicking on "OK".

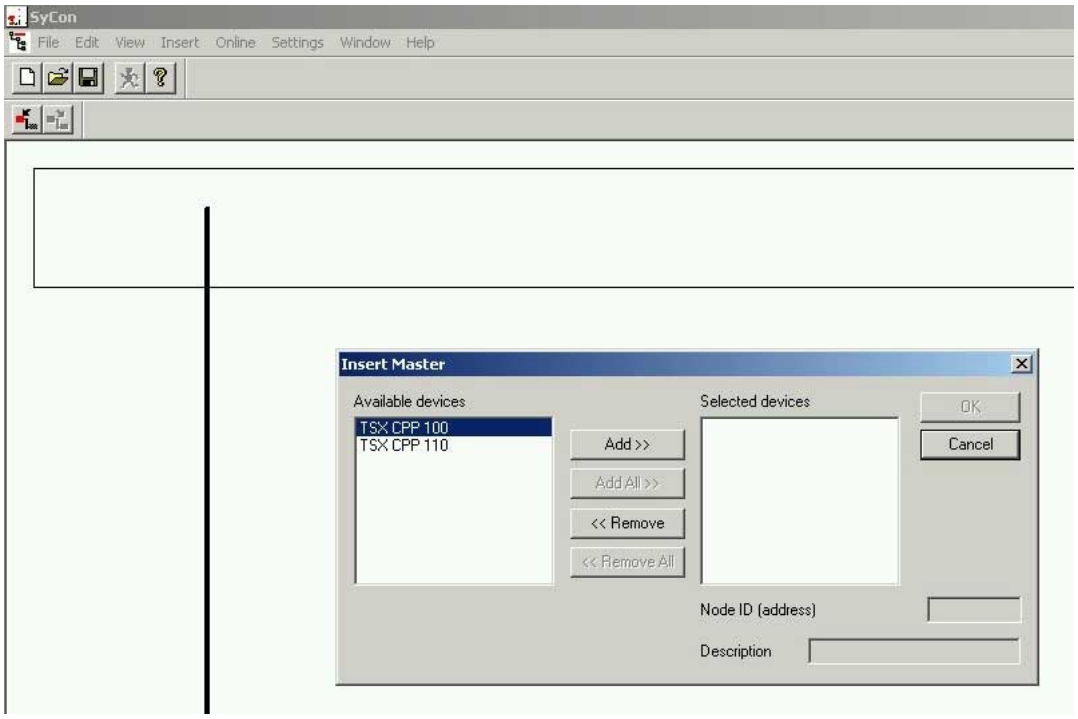


5.3.2 Selection of the CANopen master

The network master can be selected using the menu option "Insert - Master". In the example the Premium PLC TSX CPP 110 card is used.

The node address and a brief description can be entered directly.

- ▶ Confirm your selection by clicking on "OK".

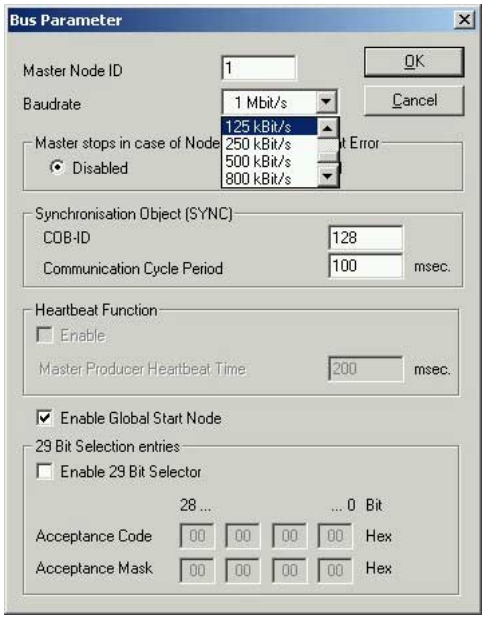


0198441113235, V1.04, 01.2006

5.3.3 Setting the bus parameters

The CANopen communication parameters are set using the menu option "Settings - Bus Parameter...". Please also consult the Operation Instructions for the SyCon configuration software.

▶ Confirm your selection by clicking on "OK".

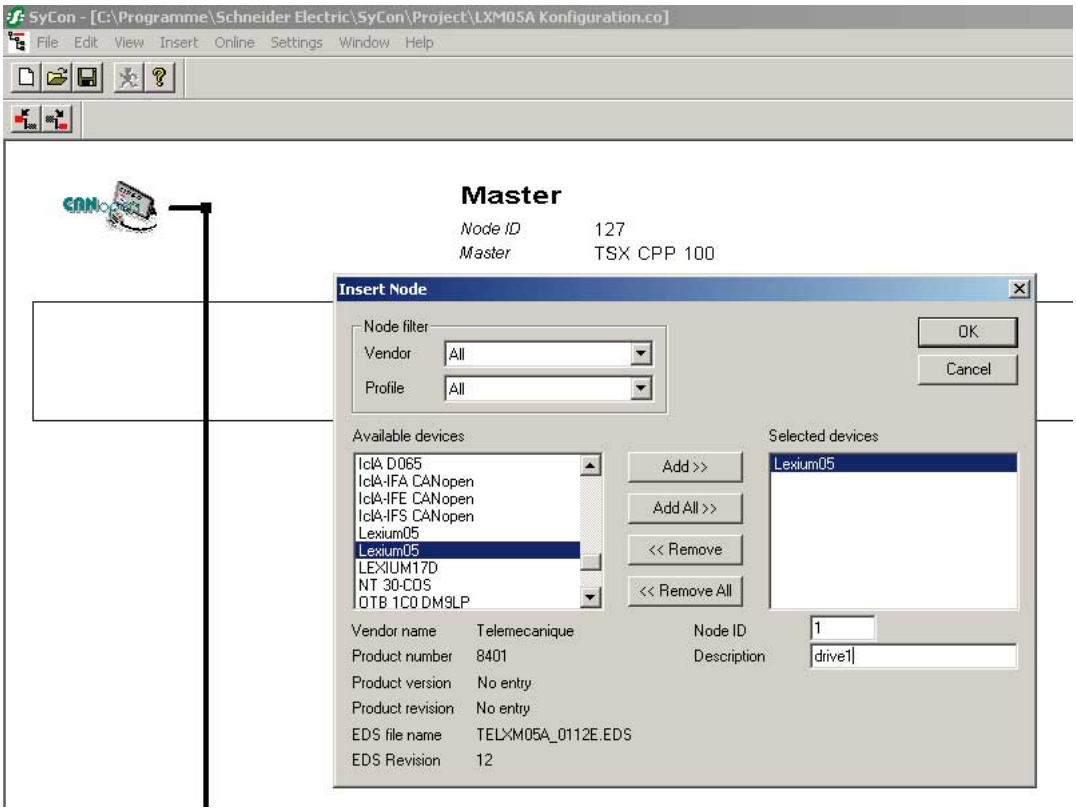


0198441113235, V1.04, 01.2006

5.3.4 Selection and insertion of nodes

Select the network nodes with "Paste - Nodes" from the menu. In the example a LXM05A is used.

▶ Confirm your selection with "OK".

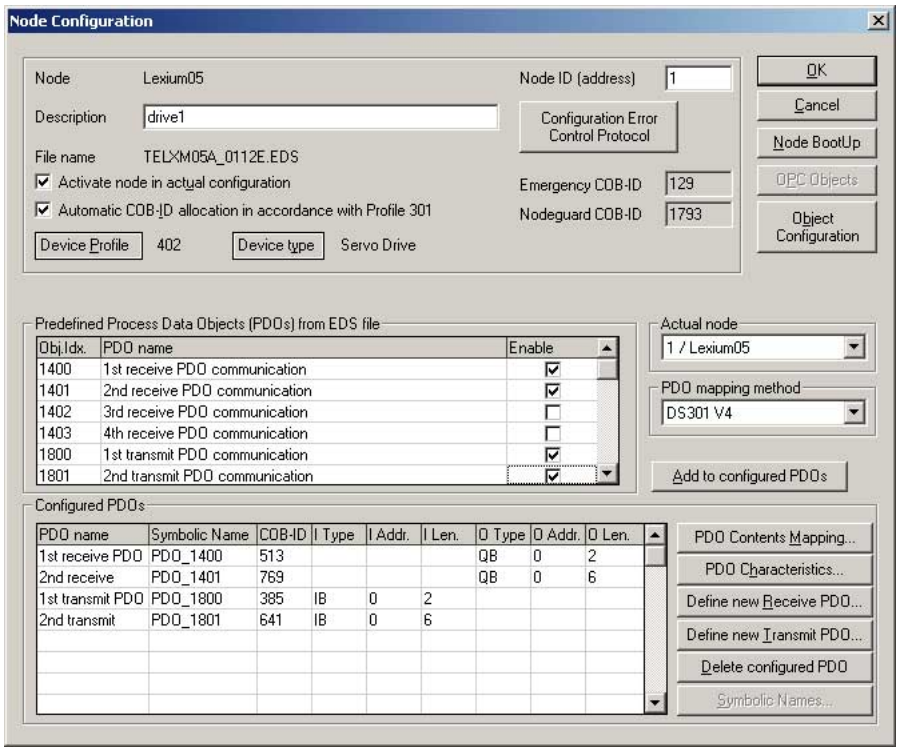


0198441113235, V1.04, 01.2006

5.3.5 Configuration of network nodes

Double-click the network node to open the node configuration. This can be used to set the communication properties of the selected node.

This is primarily the PDO characteristics of the freely configurable PDO4.



The error response of the node can be set with the "Error Control Protocol Configuration" button. The selection of whether the node is monitored by the node guarding protocol or the heartbeat protocol is made here.

▶ Confirm your selection with "OK".

Double-click on the object 1403 "4th receive PDO communication" or object 1803 "4th transmit PDO communication" to open a dialogue box in which the transmission characteristics of the PDO can be set. The default values can be imported without change.

▶ Confirm with "OK".

The mapping of the PDO4 can be set as desired with the "PDO Contents Mapping" button.

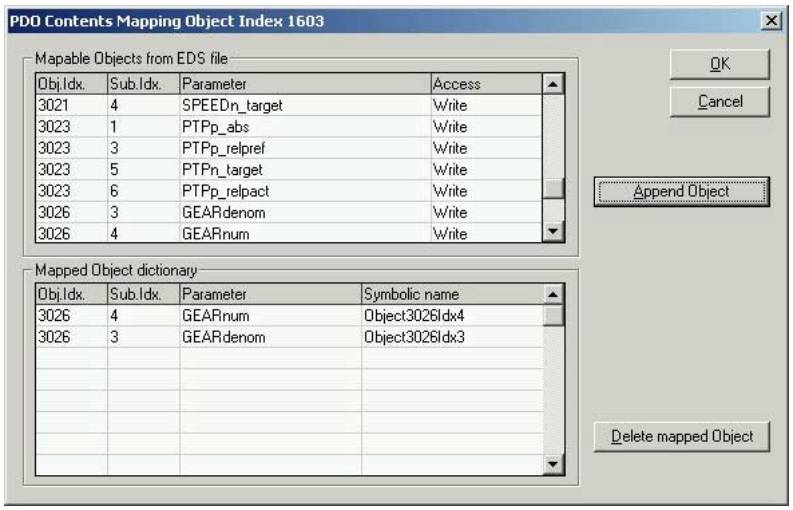
0198441113235, V1.04, 01.2006

5.3.6 Setting the mapping of the PDO4

Both in the receiving PDO4 and the transmitting PDO4 up to 4 objects can be parameterised for each using the "Append Object" button. It must be ensured that the total number of 8 bytes is not exceeded during this procedure.

In the example, the receiving PDO4 gear numerator and denominator (32bit respectively) are allocated.

▶ Confirm your selection by clicking on "OK".



Save the configuration using the menu option "File - Save as...".

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

0198441113235, V1.04, 01.2006

6 Operation

6.1 Operating modes

Local controller and fieldbus controller

With a local controller the movement is preset with analogue signals (±10V) or with RS422 signals (e.g. pulse/direction).

When the controller is connected over the fieldbus the reference values are preset via fieldbus commands.

For detailed information on setting the type of controller see the product manual under commissioning.

Standardised operating modes

The unit operates in 3 standardised operating modes. They can be started and monitored with the objects of the CANopen device profile DSP 402.

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

Manufacturer-specific operating modes

The unit operates in 4 manufacturer-specific operating modes. They use all the functions of the unit. The operating modes are started and monitored with manufacturer-specific objects.

- Current control
- Speed control
- Electronic gear
- Jog

6.2 Standardised operating modes

6.2.1 Profile position operating mode

For the profile position mode, the PDO2 must be activated. Then, motion parameters such as ramps and speeds can be set.

Example Node address 1

Work step COB-ID / data	Object value
▶ Enable R_PDO2 `601 / 23 01 14 01 01 03 00 04`	$1401:1_h$ $0400\ 0301_h$
◁ `581 / 60 01 14 01 00 00 00 00`	
▶ Enable T_PDO2 `601 / 23 01 18 01 81 02 00 04`	$1801:1_h$ $0400\ 0281_h$
◁ `581 / 60 01 18 01 00 00 00 00`	
▶ Set acceleration ramp to 2000 rpm*s `601 / 23 83 60 00 D0 07 00 00`	6083_h $0000\ 07D0_h$
◁ `581 / 60 83 60 00 00 00 00 00`	

0198441113235, V1.04, 01.2006

Work step COB-ID / data	Object value
▶ Set deceleration ramp to 4000 rpm*s 601 / 23 84 60 00 A0 0F 00 00	6084_h 0000 $0FA0_h$
◁ 581 / 60 84 60 00 00 00 00 00	
▶ Restrict setpoint speed to 6000 rpm 601 / 23 7F 60 00 70 17 00 00	$607F_h$ 0000 1770_h
◁ 581 / 60 7F 60 00 00 00 00 00	
▶ Set setpoint speed to 4000 rpm 601 / 23 81 60 00 A0 0F 00 00	6081_h 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	
▶ Activation of power circuit with PDO1 201 / 00 00 201 / 06 00 201 / 0F 00	
◁ T_PDO1 (status: operation enabled) 181 / 37 46	
▶ Start operating mode 601 / 2F 60 60 00 01 00 00 00	6060_h 01_h
◁ 581 / 60 60 60 00 00 00 00 00	
▶ Check operating status [1)] 601 / 40 61 60 00 00 00 00 00	6061_h
◁ Operating mode active 581 / 4F 61 60 00 01 00 01 00	01_h
▶ 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 status must be checked until the unit has enabled the specified operating mode.

0198441113235, V1.04, 01.2006

6.2.2 Operating mode Profile velocity

For the Profile Velocity operating mode, the PDO3 must be activated.

Example Node address 1

Work step COB-ID / data	Object value
▶ Enable 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	
▶ Enable 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 rpm*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 rpm*s 601 / 23 84 60 00 10 27 00 00	6084$_h$ 0000 2710$_h$
◁ 581 / 60 84 60 00 00 00 00 00	
▶ Restrict setpoint speed to 10000 rpm 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	
▶ Activation of power circuit with PDO1 201 / 00 00 201 / 06 00 201 / 0F 00	
◁ T_PDO1 (status: 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 status [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: send setpoint speed 1000 rpm 401 / 0F 00 E8 03 00 00	
◁ T_PDO2 with status word and position actual value 381 / 37 02 00 00 00 00	
◁ Setpoint speed reached 381 / 37 06 E8 03 00 00	

1) The operating status must be checked until the unit has enabled the specified operating mode.

0198441113235, V1.04, 01.2006

6.2.3 Operating mode Homing

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

Example Node address 1

Work step COB-ID / data	Object value
▶ Setpoint speed for movement to limit switch 100 rpm 601 / 23 99 60 01 64 00 00 00	6099:1$_h$ 0000 0064$_h$
◁ 581 / 60 99 60 01 00 00 00 00	
▶ Setpoint speed for free movement 10 rpm 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	
▶ Activation of power circuit with PDO1 201 / 00 00 201 / 06 00 201 / 0F 00	
◁ T_PDO1 (status: 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 status [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 finished 181 / 37 D6	

1) The operating status must be checked until the unit has enabled the specified operating mode.

0198441113235, V1.04, 01.2006

6.3 Manufacturer-specific operating modes

6.3.1 Current control mode.

Example Node address 1

Work step COB-ID / data	Object value
▶ Mapping R_PDO4: 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 = CUR_I_target (3020:4$_h$) 601 / 23 03 16 01 10 04 20 30	1603:1$_h$ 3020 0410$_h$
◁ 581 / 60 03 16 01 00 00 00 00	
▶ R_PDO4 Number of mapped objects = 1 601 / 2F 03 16 00 01 00 00 00	1603:0$_h$ 01$_h$
◁ 581 / 60 03 16 00 00 00 00 00	
▶ Mapping T_PDO4: number of mapped objects = 0 601 / 2F 03 1A 00 00 00 00 00	1A03:0$_h$ 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	1A03:0$_h$ 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 / 60 03 18 01 00 00 00 00	
▶ Setpoint specification with parameter 601 / 2B 1B 30 10 02 00 00 00	301B:10$_h$ 02$_h$
◁ 581 / 60 1B 30 10 00 00 00 00	
▶ NMT Start Remote Node 0 / 01 00	
◁ T_PDO1 with status word 181 / 31 66	
▶ Activation of power circuit with PDO1 201 / 00 00 201 / 06 00 201 / 0F 00	
◁ T_PDO1 (status: operation enabled) 181 / 37 46	
▶ Start operating mode 601 / 2F 60 60 00 FD 00 00 00	6060$_h$ -03$_h$
◁ 581 / 60 60 60 00 00 00 00 00	

0198441113235, V1.04, 01.2006

Work step COB-ID / data	Object value
▶ Check operating status [1)] 601 / 40 61 60 00 00 00 00 00	6061_h
◁ Operating mode active 581 / 4F 61 60 00 FD 00 01 00	-03_h
▶ PDO4 send setpoint current 1000 (10A) 501 / E8 03	
◁ T_PDO4 with current position 481 / 00 CE 09 00	

1) The operating status must be checked until the unit has enabled the specified operating mode.

6.3.2 Speed control operating mode

Example Node address 1

Work step COB-ID / data	Object value
▶ Mapping R_PDO4: 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 = SPEEDn_target ($3021:4_h$) 601 / 23 03 16 01 10 04 21 30	$1603:1_h$ $3021\ 0410_h$
◁ 581 / 60 03 16 01 00 00 00 00	
▶ R_PDO4 Number of mapped objects = 1 601 / 2F 03 16 00 01 00 00 00	$1603:0_h$ 01_h
◁ 581 / 60 03 16 00 00 00 00 00	
▶ Mapping T_PDO4: number of mapped objects = 0 601 / 2F 03 1A 00 00 00 00 00	$1A03:0_h$ 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	$1A03:0_h$ 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 / 60 03 18 01 00 00 00 00	
▶ NMT Start Remote Node 0 / 01 00	
◁ T_PDO1 with status word 181 / 31 66	

0198441113235, V1.04, 01.2006

Work step COB-ID / data	Object value
▶ Activation of power circuit with PDO1 201 / 00 00 201 / 06 00 201 / 0F 00	
◁ T_PDO1 (status: operation enabled) 181 / 37 46	
▶ Start operating mode 601 / 2F 60 60 00 FC 00 00 00	6060_h -04_h
◁ 581 / 60 60 60 00 00 00 00 00	
▶ Check operating status [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
▶ Setpoint specification with 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 send setpoint speed 1000 rpm 501 / E8 03	
◁ T_PDO4 with current position 481 / 6E 97 04 00	

1) The operating status must be checked until the unit has enabled the specified operating mode.

6.3.3 Electronic gearbox operating mode

Example Node address 1

Work step COB-ID / data	Object value
▶ Mapping R_PDO4: 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	
▶ Mapping T_PDO4: number of mapped objects = 0 601 / 2F 03 1A 00 00 00 00 00	$1A03:0_h$ 00_h
◁ 581 / 60 03 1A 00 00 00 00 00	

0198441113235, V1.04, 01.2006

Work step COB-ID / data	Object value
▶ 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	1A03:0$_h$ 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 / 30 03 18 01 00 00 00 00	1803:1$_h$ 0400 0481$_h$
▶ Signal selection position interface 601 / 2B 05 30 02 01 00 00 00 ◁ 581 / 60 05 30 02 00 00 00 00	3005:2$_h$ 01$_h$
▶ Enable operating mode with immediate synchronisation 601 / 2B 1B 30 12 01 00 00 00 ◁ 581 / 60 1B 30 12 00 00 00 00	301B:12$_h$ 01$_h$
▶ NMT Start Remote Node 0 / 01 00 ◁ T_PDO1 with status word 181 / 31 66	
▶ Activation of power circuit with PDO1 201 / 00 00 201 / 06 00 201 / 0F 00 ◁ T_PDO1 (status: operation enabled) 181 / 37 46	
▶ Start operating mode 601 / 2F 60 60 00 FE 00 00 00 ◁ 581 / 60 60 60 00 00 00 00 00	6060$_h$ -02$_h$
▶ Check operating status [1)] 601 / 40 61 60 00 00 00 00 00 ◁ Operating mode active 581 / 4F 61 60 00 FE 00 01 00	6061$_h$ -02$_h$
▶ PDO4 send gear factor 2/3 501 / 02 00 00 00 03 00 00 00 ◁ T_PDO4 with current position 481 / 53 76 01 00	

1) The operating status must be checked until the unit has enabled the specified operating mode.

0198441113235, V1.04, 01.2006

6.3.4 Jog mode

Example Node address 1

Work step COB-ID / data	Object value
▶ Speed slow movement to 100 rpm 601 / 2B 29 30 04 64 00 00 00	3029:4$_h$ 0064$_h$
◁ 581 / 60 29 30 04 00 00 00 00	
▶ Speed fast movement to 250 rpm 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	
▶ Activation of power circuit with PDO1 201 / 00 00 201 / 06 00 201 / 0F 00	
◁ T_PDO1 (status: 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 status [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$_h$
▶ Manual movement (pos. direction of 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 word181 / 37 02	
▶ Manual movement (pos. direction of 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 status must be checked until the unit has enabled the specified operating mode.

0198441113235, V1.04, 01.2006

6.4 Functions

6.4.1 ramp function

The device controls the acceleration and deceleration behaviour of the motor with ramp functions. The gradient and shape of the ramp describe the ramp function. The ramp gradient shows the motor's change of speed, and the shape of the ramp the acceleration over time.

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

Object (Index:subindex)	Description
Profile acceleration (6083_h)	Acceleration [usr]
Profile deceleration (6084_h)	Deceleration [usr]
Max profile velocity ($607F_h$)	Limitation of setpoint speed

6.4.2 Quick Stop function

Quick Stop is an emergency braking function, which stops the motor, e.g. if a fault occurs.

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

The motor can be decelerated with the Quick Stop current, object `LIM_I_maxQSTP (3011:5h)`, with the stop current, object `LIM_I_maxHalt (3011:6h)` or the deceleration ramp of the travel profile, object `Profile deceleration (6084h)`.

Object (Index:subindex)	Description
LIM_I_maxQSTP ($3011:5_h$)	Current limiting for Quick Stop [0.01A]
LIM_I_maxHalt ($3011:6_h$)	Current limiting for stop [0.01A]
Profile deceleration (6084_h)	Deceleration ramp of the travel profile

6.4.3 Motor stop

The drive can be stopped during a movement command over the field bus. If bit 8 in object `Controlword (6040h)` switches to "1", the unit brakes the motor at the deceleration ramp that was specified for the movement command. The movement and position data are retained.

The movement command is continued as soon as bit 8 is switched to "0" again.

6.4.4 Standstill window

If the motor is retained at zero speed with closed-loop control enabled, minimum speed variations prevent detection of the motor standstill. If the motor remains in the standstill window for an adjustable period, the closed-loop control reports motor standstill. Bit 10 in the status word, object `Statusword (6041h)` is set.

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

0198441113235, V1.04, 01.2006

Object (Index:subindex)	Description
Position window (6067_h)	Standstill window, permissible control deviation
Position window time (6068_h)	Time for which control deviations must apply in the standstill window for standstill to be signalled [ms]

6.4.5 Reversal of direction of rotation

The direction of rotation of the drive can be reversed with the object `POSdirOfRotat (3006:12h)`. The limit switch connections must be reversed at the same time.

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

Object (Index:subindex)	Description
POSdirOfRotat ($3006:12_h$)	Inversion of sense of rotation

6.4.6 Monitoring functions

Positioning limits

The motor can be moved to any point on the axis within the axis positioning range by specifying an absolute positioning process.

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

At a position overrun bit 15 (REF_OK) of the object `Statusword (6041h)` is set to 0.

Software limit switch

The software limit switch position is specified with the object `Max position limit (607Dh)` of the DSP 402 device profile.

The determining factor for position monitoring of the software limit switch range is the setpoint position of the position controller. Depending on the controller setting, therefore, the motor can stop before it reaches the limit switch position. Bit 2 of the object `_SigLatched (301C:8h)` reports the tripping 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 the movement the two limit switches are monitored with the input signals $\overline{\texttt{LIMN}}$ and $\overline{\texttt{LIMP}}$. When the limit switch is tripped the motor is stopped. Bit 1 of the object `_SigLatched (301C:8h)` report the tripping of the limit switch.

0198441113235, V1.04, 01.2006

Object (Index:subindex)	Description
IOsigLimN (3006:F_h)	0: inactive 1: break contact 2: make contact
IOsigLimN (3006:10_h)	0: inactive 1: break contact 2: make contact
_SigLatched (301C:8_h)	Bit 1: $\overline{\texttt{LIMP}}$ / $\overline{\texttt{LIMN}}$

Following error monitoring

Following error monitoring checks for positional discrepancies between the actual position of the motor and its setpoint. If the difference exceeds a following error threshold value, the unit reports an error. The threshold for the following error deviation can be set

Object (Index:Subindex)	Meaning
Following error window (6065_h)	Maximum permitted following error of the position controller [Inc]

Monitoring parameters

The unit and operating status can be monitored with 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 interruption

0198441113235, V1.04, 01.2006

6.5 Monitoring inputs and outputs of the device

The analogue signal and the digital signals of the device can be monitored over the fieldbus.

The analogue inputs ANA1 and ANA2 are monitored with the objects ANA1_act (3009:1_h)and ANA2_act (3009:5_h). The digital inputs are set with object _IO_act (3008:1_h). For example, a jog can be started via the interface signal by fieldbus.

Object (Index:subindex)	Description
ANA1_act (3009:1_h)	Monitoring analogue input ANA1
ANA2_act (3009:5_h)	Monitoring analogue input ANA2
_IO_act (3008:1_h)	Monitoring digital inputs and outputs [mV]

6.5.1 Backing up and restoring object data

The unit copies non-volatile saved object data to the RAM memory after the unit is switched on. The unit works with the data in RAM during operation.

The data must be transferred to the non-volatile memory with the object PAReeprSave (3004:1_h) to back up user-specific object settings in case of power failure.

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

Object (Index:Subindex)	Meaning
PAReeprSave (3004:1_h)	Backing up object settings in EEPROM
PARusrReset (3004:8_h)	Restoring object settings

0198441113235, V1.04, 01.2006

0198441113235, V1.04, 01.2006

7 Diagnostics and troubleshooting

7.1 Fieldbus communication error diagnosis

Field-mode mode must be functioning to be able to evaluate operational and error messages.

Connections to field-bus mode

If the device cannot be addressed over 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
- Field-bus cable and field-bus wiring
- Network connection to the device

The commissioning software can also be used for error diagnosis.

Baud rate and address

If a connection to a device cannot be made, check the baud rate and node address.

- The baud rate of all network devices must be set to the same value
- The node address of every device must be between 1 and 127 and must be different for device

To set the baud rate and node address see Chapter 5.2 “Address and baud rate“.

Function test on the field bus

After correct configuration of the transmission data test the field bus operation. This requires installation of a CAN configuration tool that displays CAN messages. The acknowledgement of the unit is captured by a boot-up message:

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

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

0198441113235, V1.04, 01.2006

7.2 Error diagnosis over fieldbus

7.2.1 Message objects

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

- Object `Statusword (6041h)`
  operating states, see product manual
- Object `EMCY (80h+ node-ID)`
  error message of a device with error status and error code, see Chapter 3.5 “Emergency service“
- Object `Error register (1001h)`
  error status
- Object `Error code (603Fh)`
  error code of the last occurring error
- Devices use the special SDO ABORT error message to report the failed message exchange via SDO (cancel)

7.2.2 Messages on the device status

A distinction is made between synchronous and asynchronous errors when evaluating and handling errors.

Synchronous error

The unit reports a synchronous error directly as a response to a message that cannot be evaluated. Possible causes can be faulty transmission or illegal data. For a list of synchronous errors see chapter 7.3.1 “error register“.

Asynchronous errors

Asynchronous errors are reported by the monitoring devices of the unit as soon as a unit error occurs. An asynchronous error is reported via bit 3, "Fault", of the object `statusword (6041h)`. For errors that cause a movement interruption the unit sends an EMCY message.

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

0198441113235, V1.04, 01.2006

7.3 CANopen error messages

CANopen error messages are displayed as an EMCY message. They are evaluated via the object `Error register (1001h)` and `Error code (603Fh)`. For information on the object `EMCY` see chapter 3.5 “Emergency service“.

CANopen reports errors that occur during data exchange by SDO with the special SDO error message ABORT.

7.3.1 error register

The object `Error register(1001h)` shows the error status 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	Description
0	generic error	An error has occurred
1	-	reserved
2	-	reserved
3	-	reserved
4	Communication	Error in network communications
5	Device profile-specific	Error in execution by device profile
6	-	reserved
7	Manufacturer-specific	Manufacturer-dependent error message

7.3.2 Error code table

The error code is evaluated with the object `error code (603Fh)`, an object of the DSP 402 device profile, and output as a four-character hexadecimal number. The error code shows the cause of the last interruption of movement. The meaning of the error code can be found in the product manual in the section on error diagnosis and troubleshooting.

7.3.3 SDO error message ABORT

A SDO error message is output as a response to an error in a SDO transmission. The cause of error is shown in `error code`, byte 4 to byte 7.

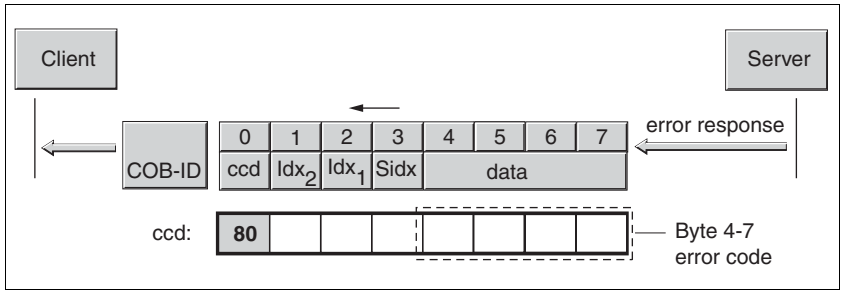


Figure 7.1 SDO error message as answer to a SDO message

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

0198441113235, V1.04, 01.2006

Error code	Description
0503 0000_h	Inverse bit not inverted
0504 0000_h	Time-out during SDO transfer
0504 0001_h	Command specifier CS incorrect or unknown
0504 0005_h	No free memory
0601 0000_h	No access to object possible
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 directory
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 not compatible
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 to great
0609 0032_h	Parameter values too small
0609 0036_h	Top value is less than bottom value
0800 0000_h	General error
0800 0020_h	Data cannot be uploaded or saved to the application.
0800 0021_h	Device control is executed locally, data cannot be uploaded or saved.
0800 0022_h	Device status prevent uploading and saving data.
0800 0023_h	Object directory either not present or cannot be generated, e.g. if data error occurs when generating 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.

0198441113235, V1.04, 01.2006

8 Service, maintenance and disposal

⚠ CAUTION

Destruction of unit components and loss of control!

Excessive currents can be created at the signal connections if the negative connection to the controller supply voltage is interrupted.

- Do not interrupt the negative connection between power supply unit and load with a fuse or switch
- Check for correct connection before switching on.
- Never connect the controller supply voltage or change its wiring while there is supply voltage present.

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

8.1 Service address

If you have any questions please contact your local dealer. Your dealer will be happy to give you the name of a customer service outlet in your area.

0198441113235, V1.04, 01.2006

0198441113235, V1.04, 01.2006

9 Object directory

9.1 Specifications for the objects

Index This index shows the position of the object in the object directory. The index value is shown in hexadecimal.

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

Object code	Description	Coding
VAR	A single 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 single data types.	9

Data type	Value range	Data length	DS 301 coding
Boolean	0 = false, 1 = true	1 byte	0001
Integer8	-128 .. +127	1 byte	0002
Integer16	-32768 .. +32767	2 byte	0003
Integer32	-2147483648 .. +2147483647	4 byte	0004
Unsigned8	0 .. 255	1 byte	0005
Unsigned16	0 .. 65535	2 byte	0006
Unsigned32	0 .. 4294967295	4 byte	0007
Visible String8	ASCII characters	8 byte	0009
Visible String16	ASCII characters	16 byte	0010

RO/RW Note on readability and writability of 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 entry: PDO mapping not possible with the object

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

Default value Factory settings.

persistent Designation of whether the value of the parameter is persistent, i.e. after switching off the unit it is retained in the memory. When changing a value via commissioning software or fieldbus, the user must explicitly store the value change in the persistent memory. When entering via HMI the unit stores the value of the parameter automatically at each change.

0198441113235, V1.04, 01.2006

9.2 Overview of object group 1000_h

Index	subindex	Name	Obj. code	Data type	Access	PDO	Description	Page
1000_h		device type	VAR	Unsigned32	ro		Device type and profile	9-7
1001_h		error register	VAR	Unsigned8	ro		error register	9-7
1003_h		predefined error field	ARR		rw		Error history, memory for error messages	9-8
1003_h	00_h	number of errors	VAR	Unsigned8	rw		Number of error entries	9-8
1003_h	01_h	error field	VAR	Unsigned32	ro		Error number	9-8
1005_h		COB-ID SYNC	VAR	Unsigned32	rw		Identifier of the synchronisation object	9-9
1008_h		manufacturer device name	VAR	Visible String8	ro		User device name	9-9
1009_h		manufacturer hardware version	VAR	Visible String8	ro		Hardware status	9-10
$100A_h$		manufacturer software version	VAR	Visible String8	ro		Software version	9-10
$100C_h$		guard time	VAR	Unsigned16	rw		Time span for node guarding [ms]	9-10
100Dh		life time factor	VAR	Unsigned8	rw		Repeat factor for the node guarding protocol	9-11
1010_h		save parameters	ARR	Unsigned32	rw		Saves parameters:	9-12
1010_h	01_h	save all parameters	VAR	Unsigned32	rw		Saves all parameters	9-12
1010_h	02_h	save communication parameters	VAR	Unsigned32	rw		Saves parameters for communications	9-12
1010_h	03_h	save application parameters	VAR	Unsigned32	rw		saves parameters for application	9-12
1011_h		restore default of parameters	ARR	Unsigned32	rw		Resets parameter values to the default setting	9-13
1011_h	01_h	restore default of all parameters	VAR	Unsigned32	rw		Resets all parameter values to the default setting	9-13
1011_h	02_h	restore default of application parameters	VAR	Unsigned32	rw		Restores parameter settings for communications to default	9-13
1011_h	03_h	restore default of communication parameters	VAR	Unsigned32	rw		sets parameter settings for the application to default	9-13
1014_h		COB-ID EMCY	VAR	Unsigned32	rw		Unsigned16	9-14
1015_h		inhibit time EMCY	VAR	Unsigned16	rw		Unsigned16	9-14
1016_h		Consumer Heartbeat Time	ARR	Unsigned32	rw		Unsigned16	9-15
1016_h	01_h	Consumer Heartbeat Time	VAR	Unsigned32	rw		Time interval and Node-Id of the "Heartbeat" receiver	9-15
1017_h		Producer Heartbeat Time	VAR	Unsigned16	rw		Time interval for producer "heartbeat"	9-15
1018_h		Identity Object	REC	Identity	ro		Identification object:	9-16
1018_h	01_h	Vendor ID	VAR	Unsigned32	ro		Vendor ID	9-16
1018_h	02_h	Product code	VAR	Unsigned32	ro		Product code	9-16
1018_h	03_h	Revision number	VAR	Unsigned32	ro		Revision number	9-16

0198441113235, V1.04, 01.2006

Index	subindex	Name	Obj. code	Data type	Access	PDO	Description	Page
1018_h	04_h	Serial number	VAR	Unsigned32	ro		Serial number	9-16
1020_h		Verify configuration	ARR	Unsigned32	rw		Retains data for configuration	9-17
1020_h	01_h	Configuration date	VAR	Unsigned32	rw		Date of configuration	9-17
1020_h	02_h	Configuration time	VAR	Unsigned32	rw		Time of configuration	9-17
1029_h		Number of elements	ARR	Unsigned8	ro		Number of values for the object	9-17
1029_h	01_h	Communication error	ARR	Unsigned8	rw		Communication error	9-17
1200_h		1st server SDO parameter	REC	SDO server param.	ro		First server SDO, settings	9-19
1200_h	01_h	COB-ID Client -> Server	VAR	Unsigned32	ro		Identifier Client -> Server	9-19
1200_h	02_h	COB-ID Server -> Client	VAR	Unsigned32	ro		Identifier Server -> Client	9-19
1201_h		2nd server SDO parameter	REC	SDO server param.	rw		Second server SDO, settings	9-20
1201_h	01_h	COB-ID Client -> Server	VAR	Unsigned32	rw		Identifier Client -> Server	9-20
1201_h	02_h	COB-ID Server -> Client	VAR	Unsigned32	rw		Identifier Server -> Client	9-20
1201_h	03_h	Node-ID SDO Client	VAR	Unsigned32	rw		Node-ID SDO Client	9-20
1400_h		1st receive PDO parameter	REC	PDO comm. param.	rw		First receive PDO (R_PDO1), settings	9-21
1400_h	01_h	COB-ID R_PDO1	VAR	Unsigned32	rw		Identifier of the R_PDO1	9-21
1400_h	02_h	transmission type R_PDO1	VAR	Unsigned8	rw		Transmission type	9-21
1401_h		2nd receive PDO parameter	REC	PDO comm. param.	rw		Second receive PDO (R_PDO2), settings	9-23
1401_h	01_h	COB-ID R_PDO2	VAR	Unsigned32	rw		Identifier of the R_PDO2	9-23
1401_h	02_h	transmission type R_PDO2	VAR	Unsigned8	rw		Transmission type	9-23
1402_h		3rd receive PDO parameter	REC	PDO comm. param.	rw		Third receive PDO (R_PDO3), settings	9-24
1402_h	01_h	COB-ID R_PDO3	VAR	Unsigned32	rw		Identifier of the R_PDO3	9-24
1402_h	02_h	transmission type R_PDO3	VAR	Unsigned8	rw		Transmission type	9-24
1403_h		4th receive PDO parameter	REC	PDO comm. param.	rw		Fourth receive PDO (R_PDO4), settings	9-25
1403_h	01_h	COB-ID R_PDO4	VAR	Unsigned32	rw		Identifier of the R_PDO4	9-25
1403_h	02_h	transmission type R_PDO4	VAR	Unsigned8	rw		Transmission type	9-25
1600_h		1st receive PDO mapping	REC	PDO mapping	ro		PDO mapping for R_PDO1, settings	9-26
1600_h	01_h	1st mapped object R_PDO1	VAR	Unsigned32	ro		First object for the mapping in R_PDO1	9-26
1601_h		2nd receive PDO mapping	REC	PDO mapping	ro		PDO mapping for R_PDO2, settings	9-27
1601_h	01_h	1st mapped object R_PDO2	VAR	Unsigned32	ro		First object for the mapping in R_PDO2	9-27

0198441113235, V1.04, 01.2006

Index	subindex	Name	Obj. code	Data type	Access	PDO	Description	Page
1601_h	02_h	2nd mapped object R_PDO2	VAR	Unsigned32	ro		Second object for the mapping in R_PDO2	9-27
1602_h		3rd receive PDO mapping	REC	PDO mapping	ro		PDO mapping for R_PDO3, settings	9-28
1602_h	01_h	1st mapped object R_PDO3	VAR	Unsigned32	ro		First object for the mapping in R_PDO3	9-28
1602_h	02_h	2nd mapped object R_PDO3	VAR	Unsigned32	ro		Second object for the mapping in R_PDO3	9-28
1603_h		4th receive PDO mapping	REC	PDO mapping	rw		PDO mapping for R_PDO3, settings	9-29
1603_h	01_h	1st mapped object R_PDO4	VAR	Unsigned32	rw		First object for the mapping in R_PDO4	9-29
1603_h	02_h	2nd mapped object R_PDO4	VAR	Unsigned32	rw		Second object for the mapping in R_PDO4	9-29
1603_h	03_h	3rd mapped object R_PDO4	VAR	Unsigned32	rw		Third object for the mapping in R_PDO4	9-29
1800_h		1st transmit PDO parameter	REC	PDO comm. param.	rw		First send PDO (T_PDO1), settings	9-29
1800_h	01_h	COB-ID T_PDO1	VAR	Unsigned32	rw		Identifier of the T_PDO1	9-29
1800_h	02_h	transmission type T_PDO1	VAR	Unsigned8	rw		Transmission type	9-29
1800_h	03_h	inhibit time T_PDO1	VAR	Unsigned16	rw		Blocking period for bus access (1=100 µs)	9-29
1800_h	04_h	reserved T_PDO1	VAR	Unsigned8	rw		Priority for CAN bus arbitration ([0-7]).	9-29
1800_h	05_h	event timer T_PDO1	VAR	Unsigned16	rw		Time span for event triggering (1=1 ms)	9-29
1801_h		2nd transmit PDO parameter	REC	PDO comm. param.	rw		Second send PDO (T_PDO2), settings	9-31
1801_h	01_h	COB-ID T_PDO2	VAR	Unsigned32	rw		Identifier of the T_PDO2	9-31
1801_h	02_h	transmission type T_PDO2	VAR	Unsigned8	rw		Transmission type	9-31
1801_h	03_h	inhibit time T_PDO2	VAR	Unsigned16	rw		Blocking period for bus access (1=100 µs)	9-31
1801_h	04_h	reserved T_PDO2	VAR	Unsigned8	rw		reserved	9-31
1801_h	05_h	event timer T_PDO2	VAR	Unsigned16	rw		Time span for event triggering (1=1 ms)	9-31
1802_h		3rd transmit PDO parameter	REC	PDO comm. param.	rw		Third send PDO (T_PDO3), settings	9-33
1802_h	01_h	COB-ID T_PDO3	VAR	Unsigned32	rw		Identifier of the T_PDO3	9-33
1802_h	02_h	transmission type T_PDO3	VAR	Unsigned8	rw		Transmission type	9-33
1802_h	03_h	inhibit time T_PDO3	VAR	Unsigned16	rw		Blocking period for bus access (1=100 µs)	9-33
1802_h	04_h	reserved T_PDO3	VAR	Unsigned8	rw		reserved	9-33
1802_h	05_h	event timer T_PDO3	VAR	Unsigned16	rw		Time span for event triggering (1=1 ms)	9-33

0198441113235, V1.04, 01.2006

Index	subindex	Name	Obj. code	Data type	Access	PDO	Description	Page
1803_h		4th transmit PDO parameter	REC	PDO comm. param.	rw		Fourth send PDO (T_PDO4), settings	9-34
1803_h	01_h	COB-ID T_PDO4	VAR	Unsigned32	rw		Identifier of the T_PDO4	9-34
1803_h	02_h	transmission type T_PDO4	VAR	Unsigned8	rw		Transmission type	9-34
1803_h	03_h	inhibit time T_PDO4	VAR	Unsigned16	rw		Blocking period for bus access (1=100 µs)	9-34
1803_h	04_h	reserved T_PDO4	VAR	Unsigned8	ro		reserved	9-34
1803_h	05_h	event timer T_PDO4	VAR	Unsigned16	rw		Time span for event triggering (1=1ms)	9-34
1A00h		1st transmit PDO mapping	REC	PDO mapping	rw		PDO mapping for T_PDO1, settings	9-36
1A00h	01_h	1st mapped object T_PDO1	VAR	Unsigned32	ro		First object for the mapping in T_PDO1	9-36
$1A01_h$		2nd transmit PDO mapping	REC	PDO mapping	rw		PDO mapping for T_PDO2, settings	9-36
$1A01_h$	01_h	1st mapped object T_PDO2	VAR	Unsigned32	ro		First object for the mapping in T_PDO2	9-36
$1A01_h$	02_h	2nd mapped object T_PDO2	VAR	Unsigned32	ro		Second object for the mapping in T_PDO2	9-36
$1A02_h$		3rd transmit PDO mapping	REC	PDO mapping	rw		PDO mapping for T_PDO3, settings	9-37
$1A02_h$	01_h	1st mapped object T_PDO3	VAR	Unsigned32	ro		First object for the mapping in T_PDO3	9-37
$1A02_h$	02_h	2nd mapped object T_PDO3	VAR	Unsigned32	ro		Second object for the mapping in T_PDO3	9-37
$1A03_h$		4th transmit PDO mapping	REC	PDO mapping	rw		PDO mapping for T_PDO4, settings	9-38
$1A03_h$	01_h	1st mapped object T_PDO4	VAR	Unsigned32	rw		First object for the mapping in T_PDO4	9-38
$1A03_h$	02_h	2nd mapped object T_PDO4	VAR	Unsigned32	rw		Second object for the mapping in T_PDO4	9-38
$1A03_h$	03_h	3rd mapped object T_PDO4	VAR	Unsigned32	rw		Third object for the mapping in T_PDO4	9-38
$1A03_h$	04_h	4th mapped object T_PDO4	VAR	Unsigned32	rw		Fourth object for the mapping in T_PDO4	9-38

0198441113235, V1.04, 01.2006

9.3 Arrangement of object group 6000_h

The product includes 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 DSP 402 definition for object group 6000_h. In this case, the data type corresponding to DSP402 must be input.
A detailed description of all parameters can be found in the product manual in the Parameters chapter.

Index	DSP 402 object name	DSP 402 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
$6060:0_h$	Modes of operation	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
$60F2:0_h$	Position Option Code	UINT16	PPoption
$60F4:0_h$	Tracking error actual value	INT32	_p_dif
$60FF:0_h$	Target velocity	INT32	PVn_target
$6502:0_h$	Supported drive modes	UINT32	SuppDriveModes

0198441113235, V1.04, 01.2006

9.4 Details of object group 1000h

9.4.1 1000_h Device type

The object shows the implemented device profile and the device type.

Object description

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

Values description

subindex	00_h, device type
Description	Device type and profile
Access	read-only
PDO mapping	–
Value range	–
Default value	0042 0192_h
can be saved	–

Bit coding, subindex 00_h

Bit	Access	Value	Description
31-24	ro	00_h	not used
23-16	ro	42_h	Bit17=1: servo drive
15-0	ro	0192_h	Device profile DS-402 (192_h)

9.4.2 1001_h Error register

The object shows the error status of the device. The detailed cause of error can be found with the object `predefined error field` (1003_h) and - for reasons of compatibility to devices with different field-bus profiles - the object `error code` ($603F_h$).

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

Object description

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

Values description

subindex	00_h, error register
Description	error register
Access	read-only
PDO mapping	–
Value range	–
Default value	–
can be saved	–

0198441113235, V1.04, 01.2006

Bit coding, subindex 00_h

Bit	Access	Value	Description
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

9.4.3 1003_h Predefined error field

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

- The entry under subindex 00_h contains the number of saved error messages.
- The current error message is stored under subindex 01_h, older messages are moved to high 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

Values description

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

subindex	01_h, error field
Description	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. Byte 16..31 additional error information, not assigned in the device.

0198441113235, V1.04, 01.2006

9.4.4 1005_h COB-ID SYNC message

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

The device can only receive SYNC messages.

For synchronisation a device in the network must send SYNC objects.

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

Object description

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

Values description

subindex	00_h, COB-ID SYNC
Description	Identifier of the synchronisation 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	Description
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 is 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

9.4.5 1008_h Manufacturer device name

The object shows the device name of the manufacturer.

Object description

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

Values description

subindex	00_h, manufacturer device name
Description	User device name
Access	read-only

0198441113235, V1.04, 01.2006

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

9.4.6 1009_h Manufacturer hardware version

The object shows the version of the device hardware.

Object description

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

Values description

subindex	00_h, manufacturer hardware version
Description	Hardware status
Access	read-only
PDO mapping	–
Value range	–
Default value	–
can be saved	–

9.4.7 $100A_h$ Manufacturer software version

The object shows the version of the device software.

Object description

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

Values description

subindex	00_h, manufacturer software version
Description	Software version
Access	read-only
PDO mapping	–
Value range	–
Default value	–
can be saved	–

9.4.8 $100C_h$ Guard time

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

0198441113235, V1.04, 01.2006

The time span for the connection monitoring of a NMT master is derived from the time span "guard time" multiplied by the "life time factor", object `Life time factor(100D`$_h$`)`.

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

Object description

Index	$100C_h$
Object name	guard time
Object code	VAR
Data type	Unsigned16

Values description

subindex	00_h, guard time
Description	Time span for node guarding [ms]
Access	read-write
PDO mapping	–
Value range	0...65535
Default value	0
can be saved	yes

9.4.9 $100D_h$ Life time factor

The object shows the factor that together with the time span "guard time" gives the time interval for the connection monitoring of a NMT master. Within this period the NMT slave device waits for a monitoring request by node guarding by the NMT master device.

life time = guard time * life time factor

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

If the connection monitoring by the NMT master remains disabled during the time interval "life time", the device reports an error and switches to error status.

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

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

Object description

Index	100Dh
Object name	life time factor
Object code	VAR
Data type	Unsigned8

Values description

subindex	00_h, life time factor
Description	Repeat factor for the node guarding protocol
Access	read-write
PDO mapping	–
Value range	0...255
Default value	0
can be saved	yes

0198441113235, V1.04, 01.2006

9.4.10 1010_h Save Parameters

The object is used to save parameters.

- subindex 01_h, all parameters
- subindex 02_h, parameters for communications
- subindex 03_h, parameter for application

Object description

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

Values description

subindex	00h, number of elements
Description	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
Description	saves all parameters
Access	read-write
PDO mapping	–
Value range	–
Default value	1
can be saved	–

subindex	02_h, save communication parameters
Description	saves parameters for communications
Access	read-write
PDO mapping	–
Value range	–
Default value	1
can be saved	–

subindex	03_h, save application parameters
Description	saves parameters for application
Access	read-write
PDO mapping	–
Value range	–
Default value	1
can be saved	–

0198441113235, V1.04, 01.2006

9.4.11 1011_h Restore Default Parameters

The object is used to restore the default parameters.

- subindex 01_h, all parameters
- subindex 02_h, parameters for communications
- subindex 03_h, parameter for application

Object description

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

Values description

subindex	00h, number of elements
Description	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
Description	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
Description	Restores parameter settings for communications to default
Access	read-write
PDO mapping	–
Value range	–
Default value	1
can be saved	–

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

0198441113235, V1.04, 01.2006

9.4.12 1014_h COB-ID emergency message

The object shows the COB-ID of the emergency object "EMCY".

Object description

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

Values description

subindex	00_h, COB-ID EMCY
Description	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	Description
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, bit 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 "Pre-Operational" status.

9.4.13 1015_h Inhibit time emergency message

The object specifies the waiting period 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

Values description

subindex	00h, inhibit time EMCY
Description	Waiting time for repeated transmission of an EMCY
Access	read-write
PDO mapping	–
Value range	0...65535
Default value	0
can be saved	yes

0198441113235, V1.04, 01.2006

9.4.14 1016_h Consumer Heartbeat Time

The object saves the settings of the "heartbeat" consumer for NMT monitoring by "heartbeat" connection message.

Object description

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

Values description

subindex	00h, number of elements
Description	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
Description	Time interval and Node-Id of the "Heartbeat" receiver
Access	read-write
PDO mapping	–
Value range	0...4294967295
Default value	0
can be saved	yes

Bit-coding subindex 01_h..03_h

Bit	Description
31..24	reserved
23..16	Node-ID
15..0	Time interval for "heartbeat" message

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

9.4.15 1017_h Producer Heartbeat Time

The object saves the time interval of the "heartbeat" producer for NMT monitoring by "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` (1016_h). Time interval zero switches monitoring off.

Object description

Index	1017_h
Object name	Producer Heartbeat Time

0198441113235, V1.04, 01.2006

Object code	VAR
Data type	Unsigned16

Values description

subindex	00_h, Producer Heartbeat Time
Description	Time interval for producer "heartbeat"
Access	read-write
PDO mapping	–
Value range	0...65535
Default value	0
can be saved	yes

9.4.16 1018_h Identity Object

The object shows information on the device.

- subindex 01_h (vendor ID) contains the identification identifier of the manufacturer,
- subindex 02_h (product ID) shows 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

Values description

subindex	00h, number of elements
Description	Number of values for the object
Access	read-only
PDO mapping	–
Value range	–
Default value	4
can be saved	–

subindex	01_h, Vendor ID
Description	Vendor ID
Access	read-only
PDO mapping	–
Value range	–
Default value	0800 0054_h
can be saved	–

subindex 02_h, Product code

0198441113235, V1.04, 01.2006

Description	Product code
Access	read-only
PDO mapping	–
Value range	–
Default value	8401
can be saved	–

subindex	03_h, Revision number
Description	Revision number
Access	read-only
PDO mapping	–
Value range	–
Default value	1
can be saved	–

subindex	04_h, Serial number
Description	Serial number
Access	read-only
PDO mapping	–
Value range	–
Default value	0
can be saved	–

9.4.17 1020_h data for 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

Values description

subindex	00_h, verify configuration
Description	Retains data for configuration
Access	read-only
PDO mapping	–
Value range	–
Default value	2
can be saved	–

subindex	01_h, configuration date
Description	Date of configuration

0198441113235, V1.04, 01.2006

Access	read-write
PDO mapping	–
Value range	–
Default value	
can be saved	yes

subindex	02_h, configuration time
Description	Time of configuration
Access	read-write
PDO mapping	–
Value range	–
Default value	
can be saved	yes

0198441113235, V1.04, 01.2006

9.4.18 1029_h Error Behaviour

The object shows the behaviour of the NMT status machine in the event of a communication error.

Object description

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

Values description

subindex	00h, number of elements
Description	Number of values for the object
Access	read-only
PDO mapping	–
Value range	–
Default value	1
can be saved	–

subindex	01_h, Communication Error
Description	Communication error
Access	read-write
PDO mapping	–
Value range	0...2
Default value	0
can be saved	yes

Settings, subindex 01_h

Value	Description
0	pre-operational (with operational status only)
1	no status change
2	stopped

9.4.19 1200_h 1st server SDO parameter

The object saves 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

Values description

subindex	00h, number of elements
Description	Number of values for the object
Access	read-only
PDO mapping	–

0198441113235, V1.04, 01.2006

Value range	–
Default value	2
can be saved	–

subindex	01_h, COB-ID Client -> Server
Description	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
Description	Identifier Server -> Client
Access	read-only
PDO mapping	–
Value range	0...4294967295
Default value	1408 + Node-ID
can be saved	yes

9.4.20 1201_h 2nd server SDO parameter

The object saves 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

Values description

subindex	00h, number of elements
Description	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
Description	Identifier Client -> Server
Access	read-write
PDO mapping	–
Value range	0...4294967295
Default value	8000 0000_h
can be saved	yes

0198441113235, V1.04, 01.2006

subindex	02_h, COB-ID Server -> Client
Description	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
Description	Node-ID SDO Client
Access	read-write
PDO mapping	–
Value range	1 ... 127
Default value	–
can be saved	yes

9.4.21 1400_h 1st receive PDO parameter

The object saves 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

Values description

subindex	00_h, number of entries
Description	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
Description	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
Description	Transmission type
Access	read-write

0198441113235, V1.04, 01.2006

PDO mapping	–
Value range	0...255
Default value	255
can be saved	yes

Bit assignment subindex 01_h

Bit	Access	Value	Description
31	rw	0_b	0: PDO is active 1: PDO is inactive
30	ro	0_b	0: RTR (see below) is possible 1: RTR not allowed
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, bit 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 controller for evaluating R_PDO data is specified via subindex 02h. The values 241..251 are reserved.

Transmission type	cyclic	acyclic	synchronous	asynchronous	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 device evaluates the received data in accordance with the SYNC object.

- With acyclic transmission (transmission type=0) the evaluation is linked to 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 shows 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 transmit data with receipt of the next SYNC

0198441113235, V1.04, 01.2006

- 253 updating transmit data with receipt of a request from a PDO consumer
- 254: Data update event-controlled, the triggering event is manufacturer-specific specified

R_PDOs with the value 255 are updated immediately with receipt of the PDOs. The triggering event is the data that are sent corresponding to the definition of the DSP 402 device profile in the PDO.

Settings R_PDO1 is processed asynchronously and event-controlled.

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

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

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

9.4.22 1401_h 2nd receive PDO_parameter

The object saves 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

Values description

subindex	00_h, Largest subindex supported
Description	Maximum supported subindex
Access	read-only
PDO mapping	–
Value range	–
Default value	2
can be saved	–

subindex	01_h, COB-ID R_PDO2
Description	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
Description	Transmission type
Access	read-write
PDO mapping	–
Value range	0...255

0198441113235, V1.04, 01.2006

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 event-controlled and must be enabled with bit 31=1 in subindex 01_h.

The byte assignment of R_PDO2 is specified via PDO mapping with the object 2nd Receive PDO mapping (1601_h). The following assignment is preset in the "profile position mode":

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

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

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

0	for an asynchronous cycle
1 to 240	assigns the receive PDO, only becomes active if a SYNC object is received
255	shows that the PDO is executed on receipt

9.4.23 1402_h 3rd receive PDO parameter

The object saves 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

Values description

subindex	00_h, Largest subindex supported
Description	Maximum supported subindex
Access	read-only
PDO mapping	–
Value range	–
Default value	2
can be saved	–

subindex	01_h, COB-ID used by PDO
Description	Identifier of the R_PDO3
Access	read-write
PDO mapping	–
Value range	0...4294967295
Default value	8000 0400_h + Node-ID

0198441113235, V1.04, 01.2006

can be saved	yes

subindex	02_h, transmission type
Description	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 (1400h)`.

Settings

R_PDO3 is processed synchronously, acyclically and event-controlled and must be enabled with bit 31=1 in subindex 01_h.

The byte assignment of the R_PDO3 is specified via PDO mapping with the object `3rd Receive PDO mapping (1602h)`. The following assignment is preset for speed mode in the "profile velocity mode":

- Bytes 0..1: control word `controlword (6040h)`
- Bytes 2..5: set speed of the travel command `Target velocity (60FFh)`

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

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

0	for an asynchronous cycle
1 to 240	assigns the receive PDO, only becomes active if a SYNC object is received
255	shows that the PDO is executed on receipt

9.4.24 1403_h 4th receive PDO parameter

The object saves 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

Values description

subindex	00_h, Largest subindex supported
Description	Maximum supported subindex
Access	read-only
PDO mapping	–
Value range	–
Default value	2
can be saved	–

0198441113235, V1.04, 01.2006

subindex	01_h, COB-ID used by PDO
Description	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
Description	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 with the object `1st receive PDO-parameters (1400h)`.

PDO settings

R_PDO4 is processed synchronously and event-controlled and must be enabled with bit 31=1 in subindex 01_h.

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

9.4.25 1600_h 1st receive PDO mapping

The object shows which objects are mapped in R_PDO1 and transmitted with the PDO. When reading the object subindex 00_h the number of mapped objects is given.

Object description

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

Values description

subindex	00_h, number of mapped objects
Description	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
Description	First object for the mapping in R_PDO1
Access	read-only

0198441113235, V1.04, 01.2006

PDO mapping	–
Value range	0...4294967295
Default value	6040 0010_h
can be saved	–

Bit coding from subindex 01_h

Every subindex entry from subindex 01_h gives the object and the bit length of the object. The object is identified via index and subindex, which refer to the object directory of the device.

Bit	Description
31..16	Index
15..8	subindex
7..0	Object length in bits

Settings

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

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

9.4.26 1601_h 2nd receive PDO mapping

The object shows which objects are mapped in R_PDO2 and transmitted with the PDO. When reading the object subindex 00_h the number of mapped objects is given.

Object description

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

Values description

subindex	00_h, number of mapped application objects in PDO
Description	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)
Description	First object for the mapping in R_PDO2
Access	read-only
PDO mapping	–
Value range	0...4294967295
Default value	6040 0010_h
can be saved	–

0198441113235, V1.04, 01.2006

subindex	02_h, PDO mapping for the second application object to be mapped (target position)
Description	Second object for the 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 (1600h)`.

Settings

The PDO assignment for R_PDO2 cannot be modified. The following assignment is preset in the "profile position mode":

- subindex 01_h: PDO mapping of the control word, object `controlword (6040h)`
- subindex 02_h: target position of the travel command, object `target position (607Ah)`.

9.4.27 1602_h 3rd receive PDO mapping

The object shows which objects are mapped in R_PDO3 and transmitted with the PDO. When reading the object subindex 00_h the number of mapped objects is given.

Object description

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

Values description

subindex	00_h, number of mapped application objects in PDO
Description	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)
Description	First object for the mapping in R_PDO3
Access	read-only
PDO mapping	–
Value range	0...4294967295
Default value	6040 0010_h
can be saved	–

0198441113235, V1.04, 01.2006

subindex	02_h, PDO mapping for the second application object to be mapped (target velocity)
Description	Second object for the 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 (1600h)`.

Settings The PDO assignment for R_PDO3 cannot be modified. The following assignment is preset for speed mode in the "profile velocity mode":

- subindex 01_h: PDO mapping of the control word, object `controlword (6040h)`
- Bytes 2..5: set speed of the travel command `Target velocity (60FFh)`

9.4.28 1603_h 4th receive PDO mapping

The object shows which objects are mapped in R_PDO4 and transmitted with the PDO. When reading the object subindex 00_h the number of mapped objects is given.

Object description

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

Values description

subindex	00h, number of elements
Description	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 at the object `1st receive PDO mapping (1600h)`.

Settings The PDO assignment for R_PDO4 can be modified.

9.4.29 1800_h 1st transmit PDO parameter

The object saves settings for the first send PDO T_PDO1.

Object description

Index	1800_h
Object name	1st transmit PDO parameter

0198441113235, V1.04, 01.2006

Object code	RECORD
Data type	PDO Communication Parameter

Values description

subindex	00_h, number of entries
Description	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
Description	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
Description	Transmission type
Access	read-write
PDO mapping	–
Value range	0...255
Default value	255
can be saved	yes

subindex	03_h, inhibit time
Description	Blocking period for bus access (1=100 µs)
Access	read-write
PDO mapping	–
Value range	0...65535
Default value	0
can be saved	yes

subindex	04_h, reserved
Description	reserved
Access	–
PDO mapping	–
Value range	0...255
Default value	–
can be saved	–

0198441113235, V1.04, 01.2006

subindex	05_h, event timer
Description	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 (1400h)`.

Settings

T_PDO1 is sent asynchronously and event-controlled at every change of the PDO data.

The byte assignment of the T_PDO1 is specified via PDO mapping with the object `1st transmit PDO mapping (1A00h)`. The following assignment is the default:

- Bytes 0..1: status word `statusword (6041h)`.

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

9.4.30 1801_h 2nd transmit PDO parameter

The object saves settings for the second send PDO T_PDO2.

Object description

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

Values description

subindex	00_h, Largest subindex supported
Description	Maximum supported subindex
Access	read-only
PDO mapping	–
Value range	–
Default value	5
can be saved	–

subindex	01_h, COB-ID used by PDO
Description	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

0198441113235, V1.04, 01.2006

Description	Transmission type
Access	read-write
PDO mapping	–
Value range	0...255
Default value	255
can be saved	yes

subindex	03_h, inhibit time
Description	Blocking period for bus access (1=100 µs)
Access	read-write
PDO mapping	–
Value range	0...65535
Default value	0
can be saved	yes

subindex	04_h, reserved
Description	reserved
Access	–
PDO mapping	–
Value range	0...255
Default value	–
can be saved	–

subindex	05_h, event timer
Description	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 (1400h)`.

Settings T_PDO2 is sent 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 assignment is preset in the "profile position 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 "Pre-Operational" status.

0198441113235, V1.04, 01.2006

9.4.31 1802_h 3rd transmit PDO parameter

The object saves settings for the third send PDO T_PDO3.

Object description

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

Values description

subindex	00_h, Largest subindex supported
Description	Maximum supported subindex
Access	read-only
PDO mapping	–
Value range	–
Default value	5
can be saved	–

subindex	01_h, COB-ID used by PDO
Description	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
Description	Transmission type
Access	read-write
PDO mapping	–
Value range	0...255
Default value	255
can be saved	yes

subindex	03_h, inhibit time
Description	Blocking period for bus access (1=100 µs)
Access	read-write
PDO mapping	–
Value range	0...65535
Default value	0
can be saved	yes

subindex	04_h, reserved
Description	reserved
Access	–

0198441113235, V1.04, 01.2006

PDO mapping	–
Value range	0...255
Default value	–
can be saved	–

subindex	05_h, event timer
Description	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 (1400h)`.

Settings

T_PDO3 is sent synchronously and acyclically.

The byte assignment of the T_PDO3 is specified via PDO mapping with the object `3rd transmit PDO mapping (1A02h)`. The following assignment is preset for speed mode in the "profile velocity mode":

- Bytes 0..1: Status word`statusword (6041h)`
- Bytes 2..5: current speed `velocity actual value (606Ch)`.

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

9.4.32 1803_h 4th transmit PDO parameter

The object saves settings for the fourth send PDO T_PDO4.

Object description

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

Values description

subindex	00_h, Largest subindex supported
Description	Maximum supported subindex
Access	read-only
PDO mapping	–
Value range	–
Default value	5
can be saved	–

subindex	01_h, COB-ID used by PDO
Description	Identifier of the T_PDO4
Access	read-write
PDO mapping	–

0198441113235, V1.04, 01.2006

Value range	0...4294967295
Default value	C000 0480_h + Node-ID
can be saved	yes

subindex	02_h, transmission type
Description	Transmission type
Access	read-only
PDO mapping	–
Value range	0...255
Default value	254
can be saved	yes

subindex	03_h, inhibit time
Description	Blocking period for bus access (1=100 µs)
Access	read-write
PDO mapping	–
Value range	0...65535
Default value	0
can be saved	yes

subindex	04_h, reserved
Description	reserved
Access	–
PDO mapping	–
Value range	0...255
Default value	–
can be saved	–

subindex	05_h, event timer
Description	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 sent asynchronously and event-driven.

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

0198441113235, V1.04, 01.2006

9.4.33 $1A00_h$ 1st transmit PDO mapping

The object shows which objects are mapped in T_PDO1 and transmitted with the PDO. When reading the object subindex 00_h the number of mapped objects is given.

Object description

Index	1A00h
Object name	1st transmit PDO mapping
Object code	RECORD
Data type	PDO mapping

Values description

subindex	00_h, number of mapped objects
Description	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
Description	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 (1600h)`.

Settings

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

- subindex 1: PDO mapping of the status word, object `statusword (6041h)`

9.4.34 $1A01_h$ 2nd transmit PDO mapping

The object shows which objects are mapped in T_PDO2 and transmitted with the PDO. When reading the object subindex 00_h the number of mapped objects is given.

Object description

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

Values description

subindex	00_h, number of mapped application objects in PDO
Description	Number of values for the object

0198441113235, V1.04, 01.2006

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)
Description	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)
Description	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 (1600h)`.

Settings

The PDO assignment for T_PDO2 cannot be modified. The following assignment is preset in the "profile position mode":

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

9.4.35 $1A02_h$ 3rd transmit PDO mapping

The object shows which objects are mapped in T_PDO3 and transmitted with the PDO. When reading the object subindex 00_h the number of mapped objects is given.

Object description

Index	$1A02_h$
Object name	3rd transmit PDO mapping
Object code	RECORD
Data type	PDO mapping

Values description

subindex	00_h, number of mapped application objects in PDO
Description	Number of values for the object

0198441113235, V1.04, 01.2006

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)
Description	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)
Description	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 (1600h)`.

Settings

The PDO assignment for T_PDO3 cannot be modified. The following assignment is preset for speed mode in the "profile velocity mode":

- Bytes 0..1: Status word`statusword (6041h)`
- Bytes 2..5: current speed `velocity actual value (606Ch)`.

9.4.36 $1A03_h$ 4th transmit PDO mapping

The object shows which objects are mapped in T_PDO4 and transmitted with the PDO. When reading the object subindex 00_h the number of mapped objects is given.

Object description

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

Values description

subindex	00h, number of elements
Description	Number of values for the object
Access	read-write
PDO mapping	–

0198441113235, V1.04, 01.2006

Value range	0 ... 4
Default value	0
can be saved	yes

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

Settings The PDO assignment for T_PDO4 can be modified.

0198441113235, V1.04, 01.2006

0198441113235, V1.04, 01.2006

10 Glossary

10.1 Terms and Abbreviations

AC	Alternating Current
CAN	(**C**ontroller **A**rea **N**etwork), standardized open Fieldbus over which the drives and other devices from different manufacturers communicate with one another.
CANopen	Device and manufacturer-independent description language for communication in the CAN bus
CiA	**C**AN **i**n **A**utomation, CAN interest group, sets standards for CAN and CANopen.
COB	(**C**ommunication **OB**ject) communication object, transport unit in a CAN network.
COB-ID	(**C**ommunication **OB**ject-**Id**entifier) uniquely identifies every communications object in a CAN network
DC	Direct current
Default value	Factory settings.
DriveCom	specification of the DSP 402 status machine was created in accordance with the DriveCom specification.
DS 301	standardises the CANopen communications profile
DSP 402	standardises the CANopen device profile for drives and positioning controls
E	Encoder
EDS	(**E**lectronic **D**ata **S**heet) electronic data sheet
Electronic gear	An input speed is recalculated by the drive system using the values of an adjustable gear factor to derive a new output speed for the motor movement.
EMC	Electromagnetic compatibility
EMCY object	Emergency Object
Encoder	Sensor for recording the angular position of a rotating element. The encoder is mounted on the motor and signals the angular position of the rotor.
Error class	Classification of operational faults into groups corresponding to the error responses
Heartbeat	used for unconfirmed connection message from network devices.
HMI	Human Machine Interface, handheld operating unit.
I/O	Inputs/Outputs
Input device	A device that can be connected to the RS232 interface for commissioning, either the HMI handheld operating unit or a PC with the commissioning software..
Life-Guarding	(monitoring for signs of life) for monitoring the connection of a NMT master

0198441113235, V1.04, 01.2006

Limit switch	Switch that signals an overrun of the permissible travel range.
Mapping	assignment of object directory entries to PDOs
node ID	Node address assigned to a device on the network.
NMT	network management (NMT), component of the CANopen communications profile, tasks: initialising network and devices, starting, stopping, monitoring devices
Node Guarding	Monitoring function with slave at an interface for cyclic communication.
Object directory	List of all parameters, values and functions available in the unit. Every entry is uniquely references via index (16 bit) and subindex (8 bit).
Parameter	Device functions and values that can be set and called by the user.
PDO	Process Data Object
persistent	Designation of whether the value of the parameter is persistent, i.e. after switching off the unit it is retained in the memory. When changing a value via commissioning software or fieldbus, the user must explicitly store the value change in the persistent memory. When entering via HMI the unit stores the value of the parameter automatically at each change.
Power circuit	see power amplifier
Power amplifier	A device that generates current for controlling the motor in accordance with the positioning signals from the controller.
Quick Stop	Quick stop, function used to provide quick braking of the motor via a command or in the event of a fault.
R_PDO	Receive PDO
SDO	Service Data Object
SYNC object	Synchronisation object
T_PDO	Transmit PDO

0198441113235, V1.04, 01.2006

11 Index

0198441113235, V1.04, 01.2006

D

0198441113235, V1.04, 01.2006

E

F

0198441113235, V1.04, 01.2006

G

H

I

J

L

M

0198441113235, V1.04, 01.2006

N

O

0198441113235, V1.04, 01.2006

P

Q

0198441113235, V1.04, 01.2006

R

S

0198441113235, V1.04, 01.2006

T

0198441113235, V1.04, 01.2006