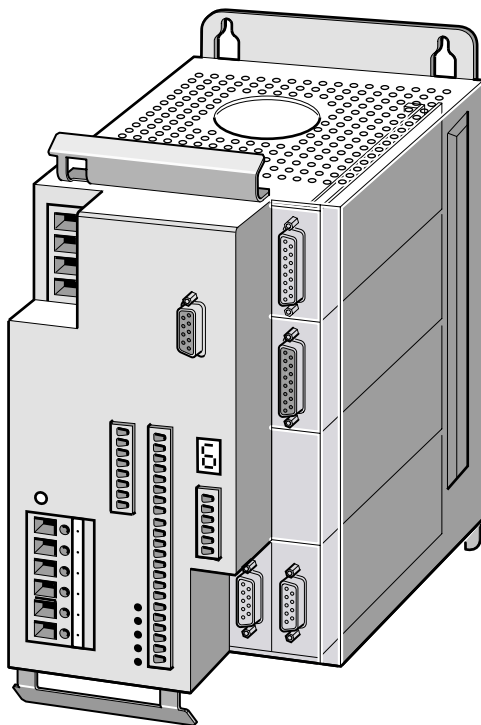


## Technical Documentation

---



Positioning controller in Field bus

### **Twin Line, CANopen**

Operating system: 01.xxx

Order No.: 9844 1113 141

Edition: -000, 09.00

#### **A product from:**

SIG Positec BERGERLAHR  
GmbH & Co. KG

Breslauer Str. 7  
D-77933 Lahr

Tel. (07821) 946 01  
Fax (07821) 946 313

<http://www.sig-berger.de>

# **Twin Line**

#### **Advice, Service and Sales:**

SIG Positec Automation GmbH

Breslauer Str. 7  
D-77933 Lahr

Tel. (07821) 946 02  
Fax (07821) 946 220

<http://www.sig-positec.de>

---



## Table of content

<b>Indexes</b>	<b>V-5</b>
Abbreviations	V-5
Product names	V-5
Technical terms	V-6
Writing conventions and warning symbols	V-7
<b>1 CAN-Bus and CANopen</b>	<b>1-1</b>
1.1 CAN-Bus	1-1
1.2 CANopen	1-2
1.2.1 CANopen descriptive language	1-2
1.2.2 Communication layers	1-2
1.2.4 CANopen profiles	1-4
1.3 Documentation, literature references, addresses	1-5
<b>2 Field bus operation</b>	<b>2-1</b>
2.1 Field bus devices in the CAN-Bus	2-1
2.2 Requirements for field bus operation	2-1
2.3 Operating modes and functions in field bus operation	2-2
<b>3 CANopen Communication</b>	<b>3-1</b>
3.1 Communication profile	3-1
3.1.1 Object dictionary	3-1
3.1.2 Communication objects	3-2
3.2 Service data communication	3-7
3.2.1 Overview	3-7
3.2.2 SDO data exchange	3-7
3.3 Process data communication	3-11
3.3.1 Overview	3-11
3.3.2 PDO data exchange	3-11
3.4 Synchronization	3-20
3.5 Emergency service	3-22
3.5.1 Error evaluation and error handling	3-22
3.6 Network management services	3-23
<b>4 Settings for operation in CAN bus</b>	<b>4-1</b>
4.1 EMC	4-1
4.2 Device installation	4-1
4.2.1 Setting address and baud rate	4-1
4.2.2 Profile selection	4-2
4.2.3 Connection of the Twin Line unit	4-3

<b>5</b>	<b>Operating modes</b>	<b>5-1</b>
5.1	Overview	5-1
5.2	Operating states and transitions	5-3
5.3	Setting and monitoring operating modes	5-9
5.3.1	Setting the operating mode	5-9
5.3.3	Monitoring the operating mode	5-10
5.4	Point-to-point operation as „profile position mode“ in accordance with DSP 402	5-12
5.4.1	Function	5-12
5.4.2	initiating positioning	5-12
5.5	Speed operation as „profile velocity mode“ in accordance with DSP 402	5-18
5.5.1	Function	5-18
5.5.2	Initiating speed mode	5-18
5.5.3	Objects and settings.	5-19
5.6	Homing as „homing mode“ in accordance with DSP 402	5-22
5.6.1	Function	5-22
5.6.2	Initiating homing	5-22
5.7	Manufacturer-specific operating modes	5-24
5.7.1	Overview	5-24
5.7.2	Objects for device monitoring and change of status.	5-24
5.7.4	Objects for monitoring status during movement.	5-25
5.7.6	Speed mode.	5-28
5.7.7	Point-to-point mode	5-29
<b>6</b>	<b>Operating functions</b>	<b>6-1</b>
6.1	List control and list processing	6-1
6.2	Teach In processing	6-3
6.3	Normalization	6-4
6.4	Ramp function	6-5
6.5	Quick Stop function	6-6
6.6	Standstill window	6-6
6.7	Inversion of rotation.	6-6
6.8	Fast position capture	6-7
6.9	Monitoring functions	6-8
6.9.1	Monitoring of axis signals	6-8
6.9.2	Monitoring internal signals.	6-9
6.10	Adding external bleed resistor	6-9
6.11	Motor stop.	6-9
6.12	Monitoring and changing signal interface inputs and outputs	6-10

6.13	Saving and restoring object data. . . . .	6-10
<b>7</b>	<b>Diagnostics and Error Correction . . . . .</b>	<b>7-1</b>
7.1	Error diagnosis field bus communication. . . . .	7-1
7.2	Error diagnosis via field bus . . . . .	7-3
7.2.1	Reporting objects . . . . .	7-3
7.2.2	Signals on the device status . . . . .	7-3
7.3	CANopen error messages . . . . .	7-4
7.3.1	Error register . . . . .	7-4
7.3.2	Error code table. . . . .	7-4
7.3.3	SDO error message ABORT . . . . .	7-4
<b>8</b>	<b>Service. . . . .</b>	<b>8-1</b>
8.1	Service address . . . . .	8-1
<b>9</b>	<b>Object Directory. . . . .</b>	<b>9-1</b>
9.1	Overview . . . . .	9-1
9.1.1	Specifications for the objects . . . . .	9-1
9.1.2	Objects sorted by object name . . . . .	9-2
9.2	Position control objects . . . . .	9-14
	<b>Index . . . . .</b>	<b>A-1</b>



## Indexes

### Abbreviations

Abbrevia- tion	Explanation
ACM	AC synchronous actuator
CAN	Controller Area Network, Standardized Bus System
ccd	Command code: Command code, part of an SDO message
COB	Communication Object, base object for transporting data in a CAN network 2048 COB's are permitted in a CAN network and can be identified via a unique COB ID.
COB ID	Communication object identifier, part of the CANopen message for identifying objects and for defining bus access priorities
DS, DSP	Draft standard Draft standard proposal
EMCY	Emergency object, object for fast transmission of error messages in the network
Inc	Increments
ISO	International Standards Organization
LSB	Lowest significant bit, bit with the lowest value $2^0$
MSB	Most significant bit, bit with the highest value, e.g. for a byte that is bit $2^7$
NMT	Network Management, services provided by the CAN Application Layer
node ID	node identifier: node address
OSI	Open Systems Interconnection, reference model for data communication, representation as layer model with distributed tasks for each layer
PDO	Process Data Object for fast transmission of data in the CAN network, a difference is made between T_PDO (Transmit PDO) in order to transmit data and R_PDO (Receive PDO) to receive data
SDO	Service Data Object for transmitting system data, a difference is made between T_SDO (Transmit SDO) in order to transmit data and R_SDO (Receive SDO) to receive data
SM	Stepping motor
SYNC	Synchronization object, object for synchronizing devices on the network

### Product names

Abbrevia- tion	Product designation	Term used
TLC5xx	Positioning controller	Positioning controller

## Technical terms

<i>Broadcast</i>	Type of data transmission in the network, one device sends a message to all devices on the network
<i>Bus arbitration</i>	System in the field bus for avoiding data collision when several devices on the bus transmit simultaneously. A device with higher priority which is ready to transmit is given first access. . Priority is defined through the COB ID.
<i>CAN terminal</i>	Communications interface of the positioning controller for connecting to the CAN bus
<i>Client</i>	First the sender, then the receiver of CAN messages in the client-server relationship, starts transmission by transmitting to the server, the point of reference is the server object directory
<i>Consumer</i>	Receiver of CAN messages in the producer-receiver relationship of network devices
<i>Default values</i>	Factory settings, pre-set values when leaving the factory
<i>Event timer</i>	When an event-driven period of time finishes, transmission of a PDO message is triggered. Event timers can be used alongside other event-triggering functions such as the changing of a signal at the input of a monitored device.
<i>Error class</i>	Summary of malfunctions in groups corresponding to error responses
<i>Heartbeat</i>	Heartbeat is a monitoring function with which one network device can check whether another network device is ready to send and receive data.
<i>Index</i>	The index is a 16-bit value through which any object in a device's object directory can be uniquely addressed.
<i>Inhibit time</i>	A PDO can be assigned a minimum waiting time for repeat transmissions in order to relieve the data transfer volume on the field bus. After the first transmission, the PDO is not re-sent until the delay has expired.
<i>Master</i>	First the sender, then the receiver of CAN messages in a master-slave relationship among network devices, the master controls the communication of the slaves.
<i>Node guarding</i>	Node guarding ist eine Überwachungsfunktion, mit der einer oder mehrere Netzwerkteilnehmer regelmäßig auf Sende- und Empfangsbereitschaft geprüft wird.
<i>Node address</i>	Address of a network device, every device on the network has a unique node address.
<i>Parameters</i>	Device data and values which can be set by the user
<i>Producer</i>	Producer of CAN messages in the producer-consumer relationship between network devices
<i>RS422 level</i>	The signal status is determined by means of the differential voltage between a positive signal and an inverted negative signal. Two signal wires must therefore be connected to produce a signal.
<i>Server</i>	First sender, then receiver of CAN messages in the client-server relationship, responds to requests from a client, point of reference is the server object directory.
<i>Slave</i>	First receiver, then sender of CAN messages in a master-slave relationship between network devices, the slave responds to enquiries from a master.



<i>Sub-index</i>	Every object is addressed by an index. The data belonging to an addressed object are stored in its sub-indexes. The object value is determined by addressing the index and sub-index.
<i>State machine</i>	<p>A state machine defines operating states and transitions with which under CANopen the network or the behavior of network devices can be controlled and changed.</p> <p>The state machine of the NMT services describes the initialization phase and start-up phase for operating the network devices. The state machine for drive units defines control points and setting options for operating a positioning drive or a positioning controller.</p>

## Writing conventions and warning symbols

<i>Action symbol „►“</i>	This symbol identifies step-by-step instructions which can be carried out in the sequence in which they appear. If the unit shows a recognizable response to a particular step in the instructions, this is shown after the description of the action. In this way you will receive direct confirmation of whether the particular step has been correctly carried out.
<i>Enumeration symbol „•“</i>	The enumeration symbol is used to list the individual points of an information set which is being described. If a sequence of steps or processes is being described, the first step to be carried out is listed first.



*This symbol identifies general notes which supply additional information about the unit.*



*For passages which are preceded by this symbol, it may be necessary to discuss more detailed information with SIG Positec Automation's Service Department. You will find contact addresses for SIG Positec Automation in the manual under „Service address“.*



## 1 CAN-Bus and CANopen

### 1.1 CAN-Bus

The CAN-Bus (CAN: Controller Area Network) was originally developed for fast, cost-effective data transmission in automotive engineering. In the meantime the CAN-Bus is also used in industrial automation, and has been further developed for communication at the field bus level.

#### *Characteristics of the CAN-Bus*

The CAN-Bus is a standardized, open bus through which units, sensors and actuators from different manufacturers communicate with each other. Characteristics of the CAN-Bus are

- Multi-master capability  
Every device on the field bus can send and receive data on its own without being dependent on a „controlling“ master functionality.
- Message-oriented communication  
Devices can be integrated into a running network without the whole system having to be re-configured. It is not necessary to announce the address of a new device on the network.
- Prioritization of messages  
For time-critical applications, high-priority messages are transmitted first.
- Residual error probability  
Various safety procedures in the network reduce the probability of a faulty data transmission going undetected to under  $10^{-11}$ . In practical terms, transmission can be assumed to be 100% secure.

#### *Transmission technology*

Several network devices are connected in the CAN-Bus 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. PLC
- PCs
- Input/output modules
- Drive controllers
- Analytical devices
- Sensors and actuators.

## 1.2 CANopen

### 1.2.1 CANopen descriptive language

CANopen is a device and manufacturer-independent descriptive language for communicating in a CAN-Bus. Although CANopen was originally used in industrial applications to control process sequences, it is now being used in many areas of network communication, e. g. in medical engineering, building automation and vehicle control.

### 1.2.2 Communication layers

CANopen uses CAN-Bus technology for data communication.

CANopen builds on the basic network services of data communication, following the ISO-OSI layer model. Three layers secure data communication in the CAN-Bus

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

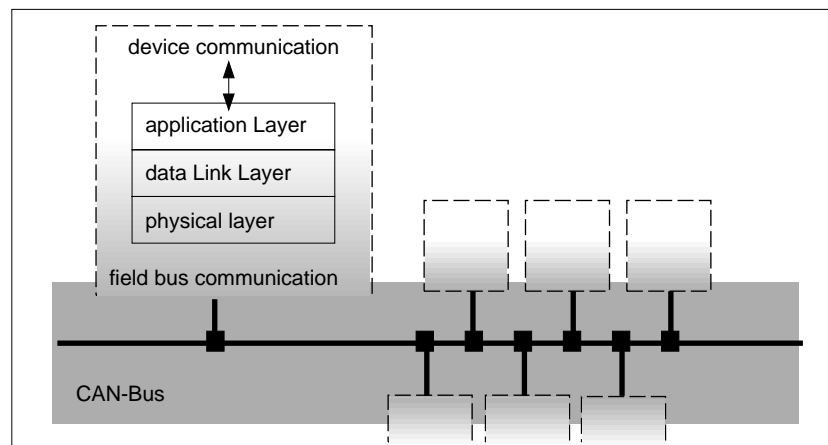


Fig 1.1 CANopen layer model

<i>CAN Physical Layer</i>	The physical layer defines the electrical properties of the CAN-Bus such as the plug connectors, length and type of cables, as well as bit coding and bit timing.
<i>CAN Data Link Layer</i>	The data security layer looks after the connection between the devices on a network. It assigns priorities to individual data packages, and carries out error monitoring and error corrections.
<i>CANopen Application Layer</i>	The application layer uses communication objects (COB) for exchanging data between individual network devices. Communication objects are the basic building blocks for setting up a CANopen application.

### 1.2.3 Objects

All processes under CANopen are carried out via objects. Objects carry out various tasks, serving as communication objects for data transmission to the field bus, controlling the connection procedure or monitoring network devices. Device-specific objects are in direct contact with the device. Device functions can be used and changed through them.

**Object dictionary** The central point of connection for all objects is the object dictionary of every network device. Other devices can find here a list of all the objects through which they can make contact with that device.

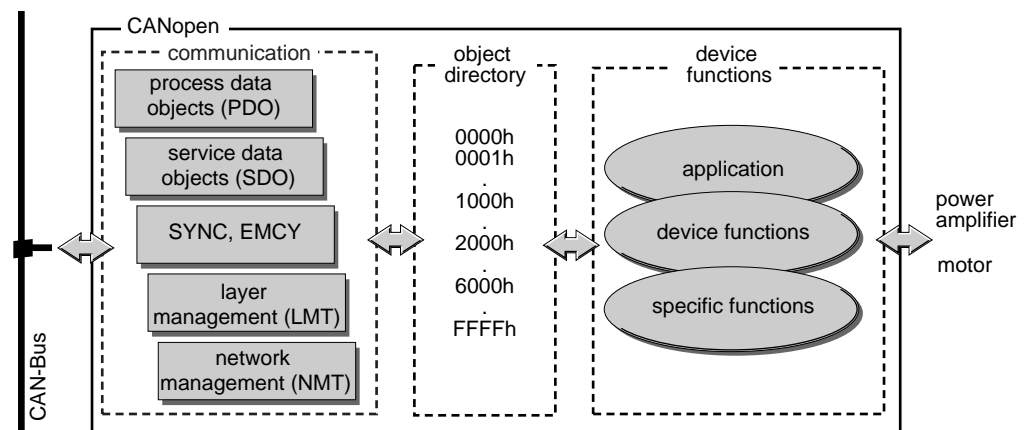


Fig 1.2 Device model with object dictionary

Included in the list are objects for describing the data types and for carrying out communication tasks and device functions under CANopen.

**Object index** Every object is addressed via a 16 bit index, which is represented as a four-digit hexadecimal number. The objects are grouped in the object dictionary.

Index (hex)	Object groups
0000	reserved
0001-009F	Static and complex data types
00A0-0FFF	reserved
1000-1FFF	Communication profile, standardized in the DS 301
2000-5FFF	Manufacturer-specific device profiles
6000-9FFF	Standardized device profiles, e g. in the DSP 402
A000-FFFF	reserved

You will find a list of all the objects which can be used for the device under CANopen in Chapter „Object Directory“ from page 9-1.

**Object group: data types**

With the data types, messages passing through the network as a stream of bits are assigned the same significance for sender and receiver. They are specified via the data type objects.

**Profile object groups**

CANopen objects take on various tasks in field bus operation. Profiles put the objects together in accordance with their assigned tasks.

## 1.2.4 CANopen profiles

### *Standardized profiles*

Standardized profiles describe objects which can be used on different devices without any additional adaptation. The CAN in Automation Association (CiA) has standardized on various profiles. These include:

- the communications profile DS-301,
- the device profile, „Drives and motion profile“ DSP-402

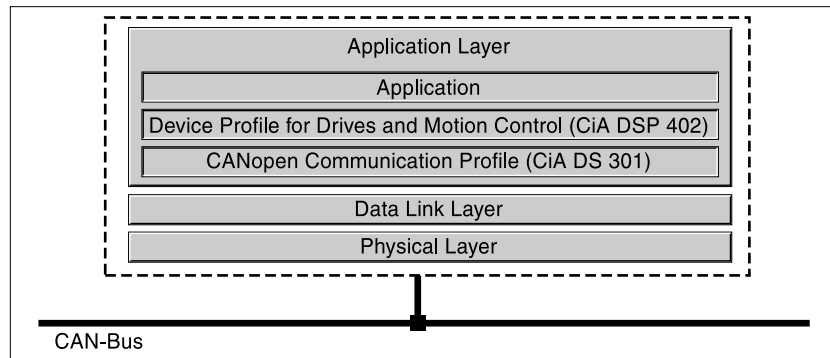


Fig 1.3 CANopen reference model

### *Communications profile DS-301*

The CANopen communications profile forms the interface between device profiles and the CAN-Bus. It was specified in 1995 under the name DS-301, and defines uniform standards for the exchange of data between different device types under CANopen.

The objects of the communication profile in the device take on the task of exchanging data and parameters with other network devices, and initialize, control and monitor the device in the network. Objects of the communication profile are:

- Process data objects PDO
- Service data objects SDO
- Objects with special functions for synchronization SYNC, and for error messages and response EMCY
- Network management objects NMT for initializing the device and monitoring it for errors and status.

You will find details on the communication profile objects in Chapter „CANopen Communication“ from page 3-1.

### *Device profile DSP-402*

The device profile, „Drives and motion profile“ DSP-402, describes standardized objects for the positioning, monitoring and setting of drives. Object tasks are

- monitoring devices and supervising status (Device Control)
- standardized parameter setting
- changing, monitoring and executing operating modes (Modes of Operation)

### *Manufacturer-specific profiles*

The basic functions of a device can be used with the objects of standardized device profiles. The entire functional scope only becomes available with manufacturer-specific device profiles. They define the objects with which the special functions of a device can be used under CANopen.

### 1.3 Documentation, literature references, addresses

*CANopen Standards* CANopen documentation from the Association CAN in Automation (CiA).

- DS 201 to DS 207 CAN Application Layer (CAL)  
Version 1.1, Feb. 1996, The CAN in Automation Association
- CiA Draft Standard 301  
CANopen Application Layer and Communication Profile  
Version 4.0, June 1999, The CAN in Automation Association
- CiA Draft Standard Proposal DSP-402  
CANopen Device Profile for Drives and Motion Control  
Version 1.0, October 1997, The CAN in Automation Association

*TLC5xx-Dokumentation* Technical documentation, positioning controller for synchronous AC actuators TLC53x,  
SIG Positec Berger Lahr, see following table for order numbers

Technical documentation, positioning controller for stepping motors TLC51x,  
SIG Positec Berger Lahr, see following table for order numbers

*CANopen documentation* Technical documentation, field bus control with CANopen  
SIG Positec Berger Lahr, see following table for order numbers

*CAN field bus technology* Technical documentation, online command processing for Twin Line devices on CAN field bus,  
SIG Positec Berger Lahr, see following table for order numbers

Designation	Order Numbers	
Documentation	TLC51x	TLC53x
German (D):	9844 1113 118	9844 1113 110
English (GB):	9844 1113 117	9844 1113 111
French (F):	9844 1113 119	9844 1113 112
Italian (I):	9844 1113 120	9844 1113 114
Documentation	CAN field bus	CANopen
German (D):	9844 1113 122	9844 1113 140
English (GB):	9844 1113 121	9844 1113 141
French (F):	9844 1113 123	
Italian (I):	9844 1113 124	

Controller Aerea Network  
Etschberger, Konrad, Carl Hanser Verlag, Munich Vienna  
ISBN 3-446-19431-2

*CAN Association* CAN in Automation (CiA)  
Am Weichselgarten 26  
D-91058 Erlangen  
www.can-cia.de





## 2 Field bus operation

### 2.1 Field bus devices in the CAN-Bus

Different field bus devices from SIG Positec Automation can be operated in the same field bus segment. CANopen offers a uniform basis for exchanging commands and data between Twin Line units and other CAN bus devices.

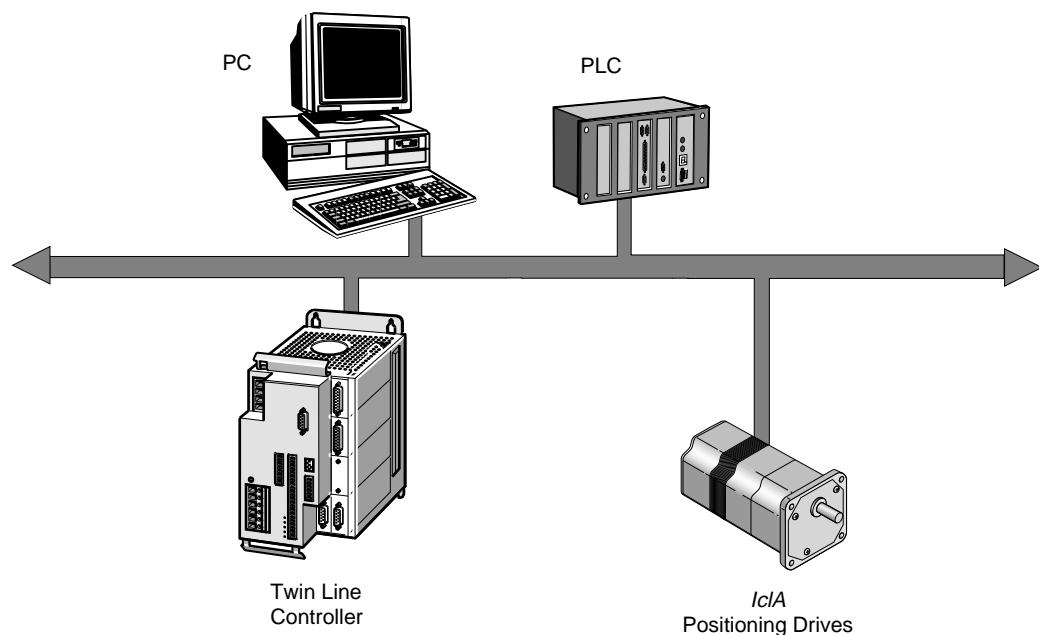


Fig 2.1 Field bus devices in a network

### 2.2 Requirements for field bus operation

*Hardware* In order to operate the positioning controller through CANopen, the unit must be fitted with the CAN-C field bus module in slot M4 and connected to a CAN field bus.

*Field bus access* Data exchange and control of Twin Line units can be performed through various access channels:

- remotely controlled through the field bus
- locally through the RS232 interface with the HMI hand-held operating unit, the TL CT operating software or through the signal interface.

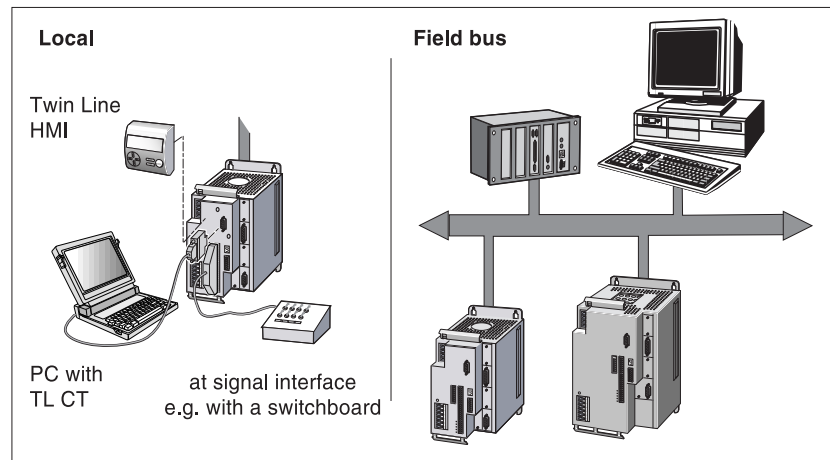


Fig 2.2 Local and remote access to Twin Line units

In order to work with the positioning controller through the field bus, the „field bus“ access channel must be enabled. The positioning controller automatically enables this access if no local operating unit connected to the positioning controller is controlling an operation or setting up a control unit.

You will find further information on access control in the positioning controller manual in the chapter on operating modes.

## 2.3 Operating modes and functions in field bus operation

Twin Line units work with the following operating modes and functions in field bus operation, depending on the equipment fitted and the model:

- Relative and absolute positioning
- Speed mode
- Electronic gear (depending on the type)
- Positioning mode with relative and absolute positioning
- Reference movement and dimension setting
- Manual movement

The operating functions include

- List control
- Teach-In
- Normalization
- Ramp functions
- Brake function
- Monitoring functions.

The parameter settings of the positioning controller can be called up and changed via the field bus, signal interface inputs can be monitored and diagnostic and error monitoring functions set to work.

### 3 CANopen Communication

#### 3.1 Communication profile

CANopen carries out the communication between network devices via object directories and objects. A network device can use Process Data Objects (PDO) and Service Data Objects (SDO) in order to read the object data from the object dictionary of another device and - if permitted - write them back with different values.

Accessing the objects of the network devices enables parameter values to be swapped, movement functions of individual CAN-Bus devices to be initiated or e. g. status information to be requested.

##### 3.1.1 Object dictionary

Every CANopen device administers an object dictionary in which all the objects required for CANopen communication with the device are listed.

*Index, Sub-index*

The objects are addressed in the object dictionary via a 16 bit long index. One or more 8 bit long sub-index entries for each object specify individual data fields in the object. Index and sub-index are shown in hexadecimal notation, identifiable by the attached „h“.

The following example shows index and sub-index entries for the *software position limit* object (607Dh) for identifying software limit switch positions.

Index	Sub-index	Name	Explanation
607Dh	00h	-	Number of data fields
607Dh	01h	min. position limit	Lower value limit switch
607Dh	02h	max. position limit	Upper value limit switch

*Dictionary set-up*

The objects are fitted into the object dictionary structure, sorted by index values. The following table shows an overview of the object dictionary in accordance with the CANopen specification.

Index range (hex)	Object groups
0000h	reserved
0001h-001Fh	static data types
0020h-003Fh	complex data types
0040h-005Fh	manufacturer-specific data types
0060h-007Fh	static data types for device profiles
0080h-009Fh	complex data types for device profiles
00A0h-0FFFh	reserved
1000h-1FFFh	communication profile
2000h-5FFFh	manufacturer-specific profiles
6000h-9FFFh	standardized device profiles
A000h-FFFFh	reserved

*Object descriptions in the manual* For CANopen programming with the positioning controller, the objects in the following object groups are described separately:

- 1xxxh objects: Communication objects in this Chapter
- 6xxxh objects: standardized device profile objects in the Chapters entitled „Operating modes“ and „Operating functions“.
- 2xxxh objects: manufacturer-specific objects in so far as they are needed for controlling the device, in the Chapters entitled „Operating modes“ and „Operating functions“.

*Standardized objects* Standardized objects form the basis for using the same application programs for different network devices of one device type. The only condition is that the devices list the objects in their directories. Standardized objects are defined in the communications profile and in various device profiles, and for the positioning controller in device profile DPS 402.

### 3.1.2 Communication objects

*Overview* Communication objects are standardized in the CANopen communication profile, DS-301. The objects can be divided into four groups corresponding to their tasks:

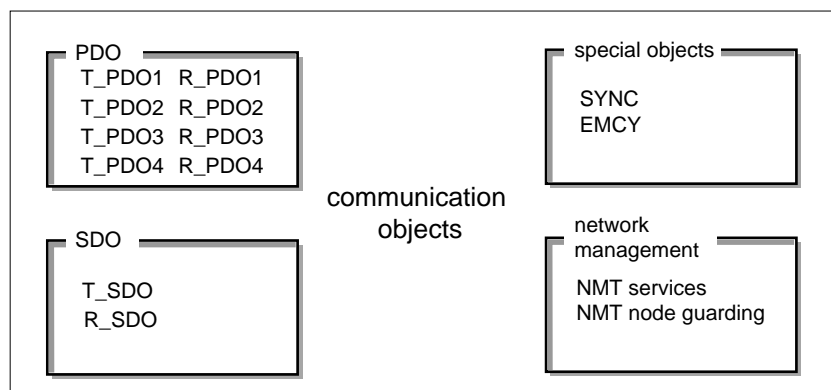


Fig 3.1 Communication objects, seen from the perspective of the device:  
T\_...: „Transmit“, R\_...: „Receive“

- Process Data Objects (PDO) for real-time transmission of process data
- Service Data Objects (SDO) for read/write access to the object dictionary
- Objects for controlling CAN messages:
  - SYNC object (synchronization object) for synchronizing network devices
  - EMCY object (emergency object) for displaying errors in a device or in its periphery.
- Network Management Services:
  - NMT services for initialization and network control (NMT: network management)
  - NMT guarding objects for monitoring network devices

*CAN Message* Data are exchanged on the CAN-Bus in the form of CAN messages. A CAN message transmits the communication object and a variety of administration and control information which ensures loss-free and error-free transmission of the data.

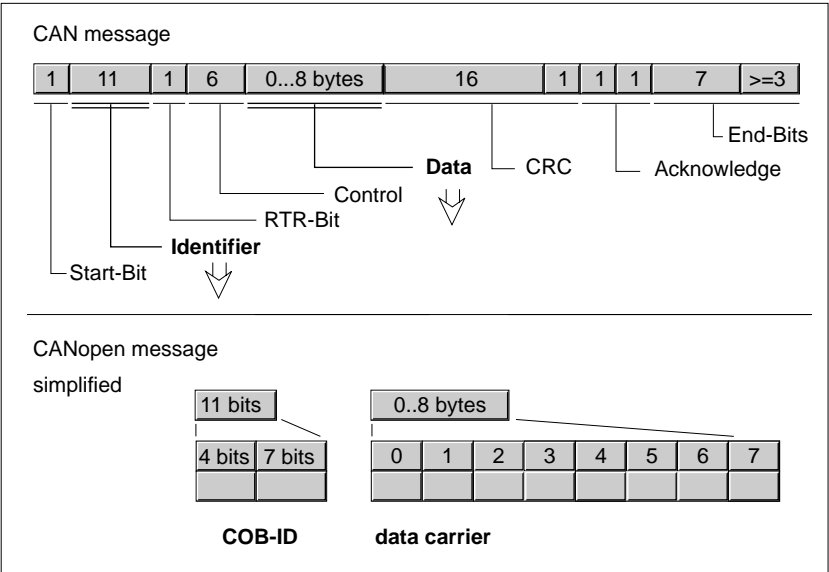


Fig 3.2 CAN message and CANopen message in simplified form

CANopen message

For work with CANopen objects and data exchange, the CAN message can be shown in a simplified form, as most of the bits are used to ensure that data transmission is free of error. These bits are automatically removed from the received message by the data security layer, the data link layer of the layer model, and inserted before a message is sent.

The two bit fields, „Identifier“ and „Data“ make up the simplified CANopen message. The „Identifier“ corresponds to the „COB-ID“ and the „Data“ field to the data carrier of a CANopen message which can be max. eight bytes in size.

COB-ID

The COB-ID (Communication object identifier) fulfils two tasks in controlling communication objects:

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

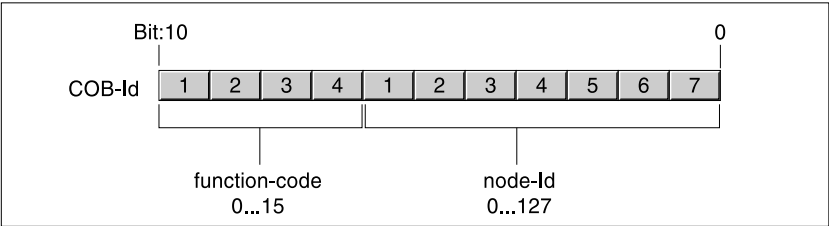


Fig 3.3 COB-ID with function code and node address (node-Id)

An 11 bit COB Identifier is defined for the positioning controller in accordance with the CAN 2.0A specification, consisting of two parts:

- Function code, 4 bits in size
- Node ID, 7 bits in size.

Function code

The function code classifies the communication objects. As the function code bits in the COB-ID are more significant, the function code also controls transmission priorities at the same time: objects with a small function code are assigned high priority in transmission, e. g. an object

with the function code „1“ will be transmitted before an object with the function code „3“, if both access the bus at the same time.

#### Node address

Every network device is configured before network operation. In the process it is assigned a unique, seven bit long node address (node-ID) between 1 and 127 (7Fh). The device address „0“ is reserved for „broadcast“ transmissions, in which messages are sent simultaneously to all devices on the network.

#### COB-IDs of Communication Objects

The following table shows the COB-IDs of all communication objects in accordance with their factory settings. The „Index of object parameters“ column indicates the index of special objects with which the communication objects settings can be read or changed by SDO.

Communication object	Function code	Node address (node-ID) [1...127]	COB-ID decimal (hexadecimal)	Index of object parameters
NMT Start/Stop Service	0 0 0 0	0 0 0 0 0 0 0	0	–
SYNC-Objekt	0 0 0 1	0 0 0 0 0 0 0	128 (80h)	1005h....1007h
EMCY-Objekt	0 0 0 1	x x x x x x x	128 (80h) + node-Id	1014h, 1015h
T_PDO1	0 0 1 1	x x x x x x x	384 (180h) + node-Id	1800h
R_PDO1	0 1 0 0	x x x x x x x	512 (200h) + node-Id	1400h
T_PDO2	0 1 0 1	x x x x x x x	640 (280h) + node-Id	1801h
R_PDO2	0 1 1 0	x x x x x x x	768 (300h) + node-Id	1401h
T_PDO3	0 1 1 1	x x x x x x x	896 (380h) + node-Id	1802h
R_PDO3	1 0 0 0	x x x x x x x	1024 (400h) + node-Id	1402h
T_PDO4	1 0 0 1	x x x x x x x	1152 (480h) + node-Id	1803h
R_PDO4	1 0 1 0	x x x x x x x	1280 (500h) + node-Id	1403h
T_SDO	1 0 1 1	x x x x x x x	1408 (580h) + node-Id	–
R_SDO	1 1 0 0	x x x x x x x	1536 (600h) + node-Id	–
NMT error control	1 1 1 0	x x x x x x x	1792 (700h) + node-Id	100Eh
LMT Services <sup>1)</sup>	1 1 1 1	1 1 0 0 1 0 x	2020 (7E4h), 2021 (7E5h)	
NMT Identify Service <sup>1)</sup>	1 1 1 1	1 1 0 0 1 1 0	2022 (7E6h)	
DBT Services <sup>1)</sup>	1 1 1 1	1 1 0 0 x x x	2023 (7E7h), 2024 (7F8h)	
NMT Services <sup>1)</sup>	1 1 1 1	1 1 0 1 0 0 x	2025 (7E9h), 2026 (7EAh)	

1) not supported by the positioning controller

#### Example of COB-ID selection

For a device with the node address 5, the COB-ID of the communication object T\_PDO1 is:

$$384 + \text{node-ID} = 384 (180h) + 5 = 389 (185h).$$

#### Data carrier

The data carrier of the CANopen message can take up to eight bytes of data. Besides the data carrier for SDOs and PDOs, special carrier types are also defined in the CANopen profile:

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

Data carriers are described by the relevant communication objects.

### 3.1.3 Communication Relationships

CANopen uses three relationships for communication between network devices:

- The Master-Slave relationship
- The Client-Server relationship
- The Producer-Consumer relationship

#### *The Master-Slave relationship*

A „Master“ in the network controls the traffic of messages. A „Slave“ only responds to requests from the master.

The Master-Slave relationship is used with network management objects in order to ensure controlled network start-up and to monitor network device connections.

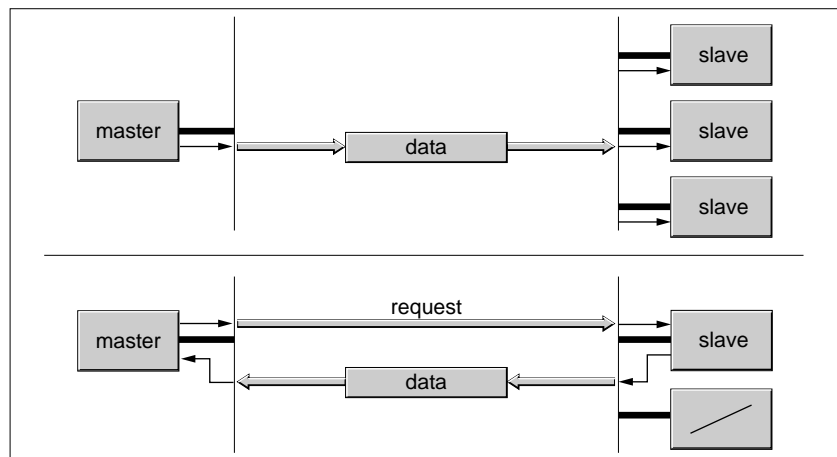


Fig 3.4 The Master-Slave relationship

The exchange of messages can be carried out with or without confirmation. If the master sends an unconfirmed CAN message, it may be received by one, several or no slaves.

In order to confirm the message, the master requests a message from a particular slave which then responds with the desired data.

#### *The Client-Server relationship*

A Client-Server relationship is always established between two network devices. The server is the device whose object list is used during the exchange of data. The client addresses and initiates the exchange of messages, and expects confirmation from the server.

A Client-Server relationship is used with SDOs in order to transmit configuration data and long messages.

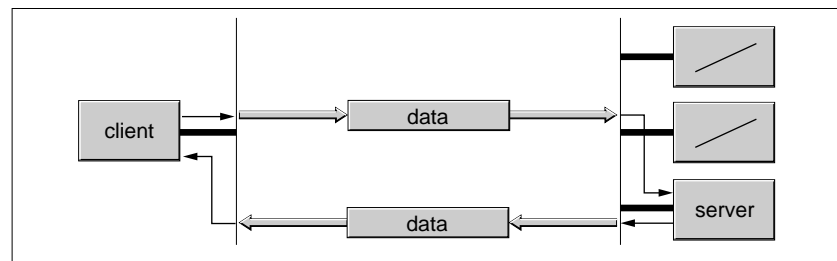


Fig 3.5 The Client-Server relationship

*The Producer-Consumer relationship*

The client addresses and transmits a CAN message to a server. The server evaluates the message and sends reply data by way of confirmation.

The Producer-Consumer relationship is used for the exchange of messages containing process data, as the relationship allows fast exchange of data without any overhead data.

A producer sends data, a consumer receives them.

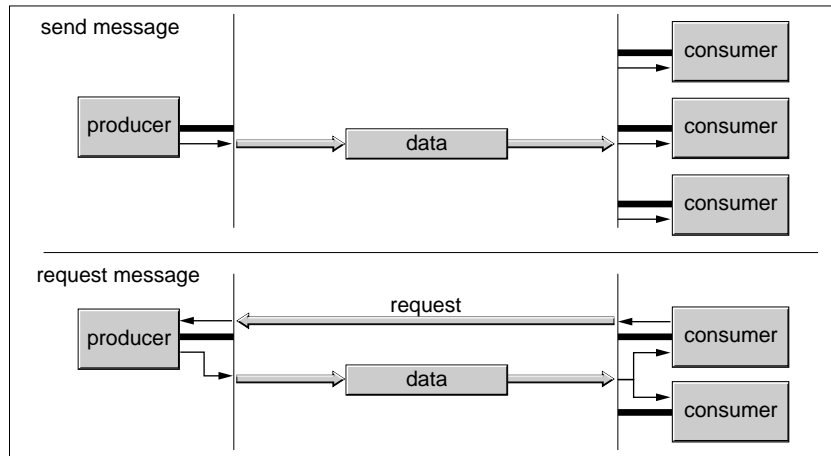


Fig 3.6 The Producer-Consumer relationship

The producer sends a message which may be received by one or several or no network devices. It receives no confirmation of receipt. Message transmission can be triggered by

- an internal event, e. g. by a „target position reached“ signal
- through the synchronization object SYNC
- by a request from a consumer.

You will find details on the function of the Producer-Consumer relationship and on requesting messages in the section „Process data communication“ from page 3-11.



## 3.2 Service data communication

### 3.2.1 Overview

Service Data Objects (*SDO: service data object*) can be used to access the entries in an object dictionary through its index and sub-index. The values of the objects can be read and - if permitted - also changed.

Every network device has at least one server-SDO in order to be able to respond to read or write requests from another device. A client-SDO is only needed in order to request SDO messages from the object dictionary of other devices, or to change them there.

Data are sent by means of T\_SDO in the client or server-SDO, and received by means of R\_SDO. An SDO is always 8 bytes in length.

SDOs have a higher COB-ID than PDOs and are therefore transmitted on the CAN-Bus with a lower priority status.

### 3.2.2 SDO data exchange

A Service Data Object SDO transmits parameter data between two network devices. The exchange of data follows the client-server relationship. The server is the device to which an SDO message refers.

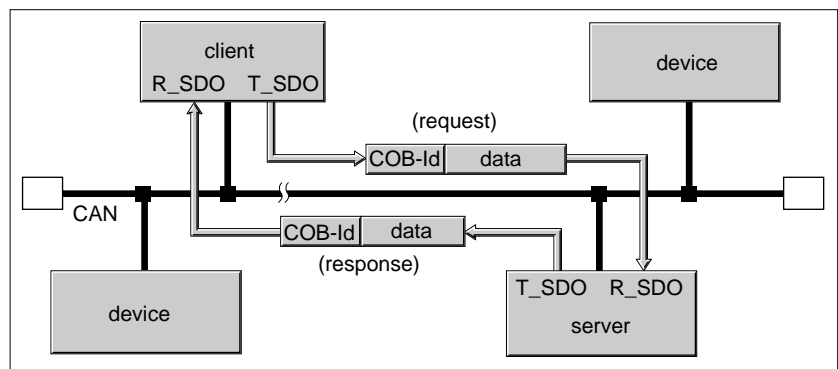


Fig 3.7 SDO message exchange with request and response

#### Message types

The client-server communication is initiated by the client in order to transfer parameter values to the server or to fetch them from the server. In both cases, the client initiates communication with a request and receives a response from the server.

3.2.3 SDO message

An SDO message consists in simplified form of the COB-ID and the SDO data carrier in which up to 4 bytes of data can be transmitted. Longer sequences of data are spread over several SDO messages, using a special protocol.

The positioning controller transmits SDOs of up to 4 bytes in length. Larger data volumes such as 8 byte values belonging to the "Visible String 8" data type, can be spread over several SDOs and transmitted in successive 4 byte blocks.

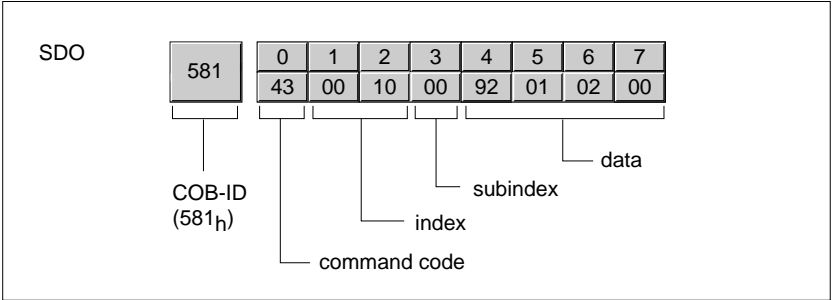


Fig 3.8 Example of an SDO message

COB ID and data framework

R\_SDOs and T\_SDOs have different COB IDs, see table on page 3-4. The data framework of an SDO message consists of

- A command code (ccd: command code), containing the SDO message type and the length of the transmitted value in encrypted form.
- The index and sub-index, which point to the object whose data are being transmitted in the SDO message. In the event of an error, the faulty SDO is itself specified in the index and sub-index.
- Data, which can comprise up to four bytes.

Evaluation of numerical values

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

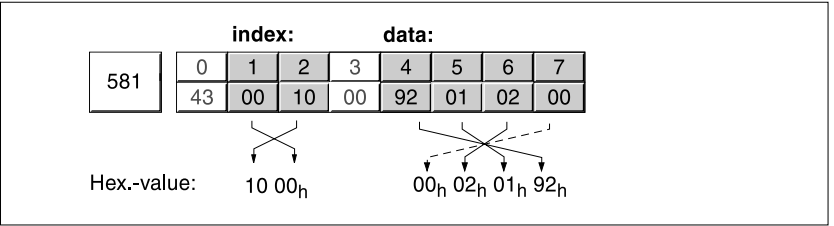


Fig 3.9 Conversion of numerical values larger than 1 byte

### 3.2.4 Reading and writing data

**Writing data** The client initiates a write request by communicating index, sub-index, data length and value.

The server sends a response confirming whether the data have been correctly processed. The response contains the same index and sub-index, but not data.

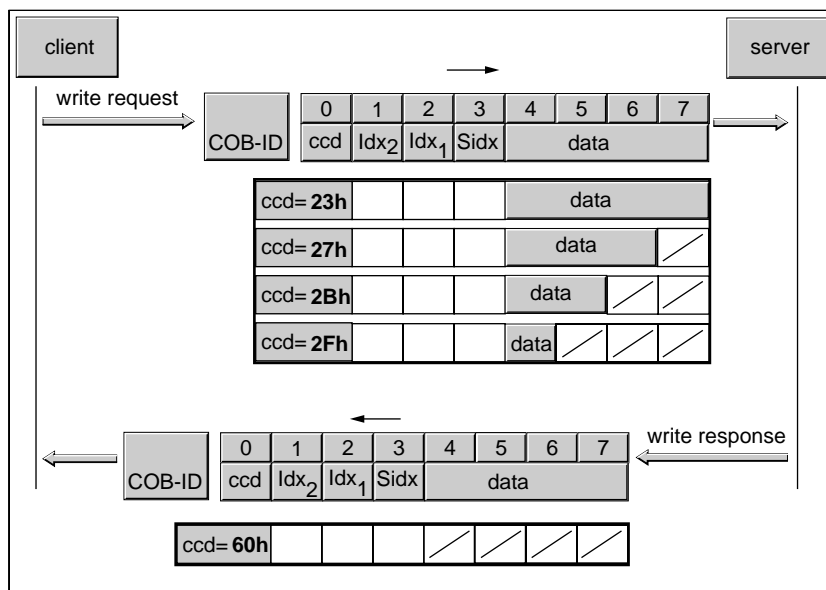


Fig 3.10 Writing a parameter value

Unused bytes in the data field are shown in the graphic as an oblique stroke. Their contents are not defined.

**ccd coding** The following table shows the command code used for writing parameter values. It varies with the type of message and length of data transmitted.

Messages	Length of data used				Explanation
	4 bytes	3 bytes	2 bytes	1 byte	
write request	23h	27h	2Bh	2Fh	transmit parameters
write response	60h	60h	60h	60h	response
error response	80h	80h	80h	80h	error

**Error response** If a message can not be evaluated without errors, the server sends an error signal.

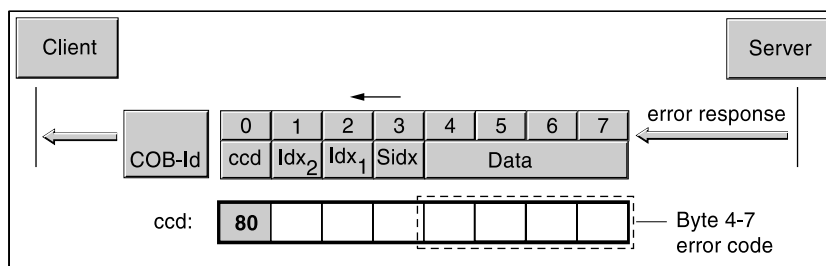


Fig 3.11 Error response, cause of error encoded in bytes 4 ..7 (error code)

You will find details on evaluating error responses in the section „SDO error message“ on page 7-4.

*Reading data* The client initiates a read request by communicating the index and sub-index which point to the object or object value whose value it wishes to read.

The server confirms the request by sending the required data. The SDO response contains the same index and sub-index. The length of the response data is given in the command code „ccd“.

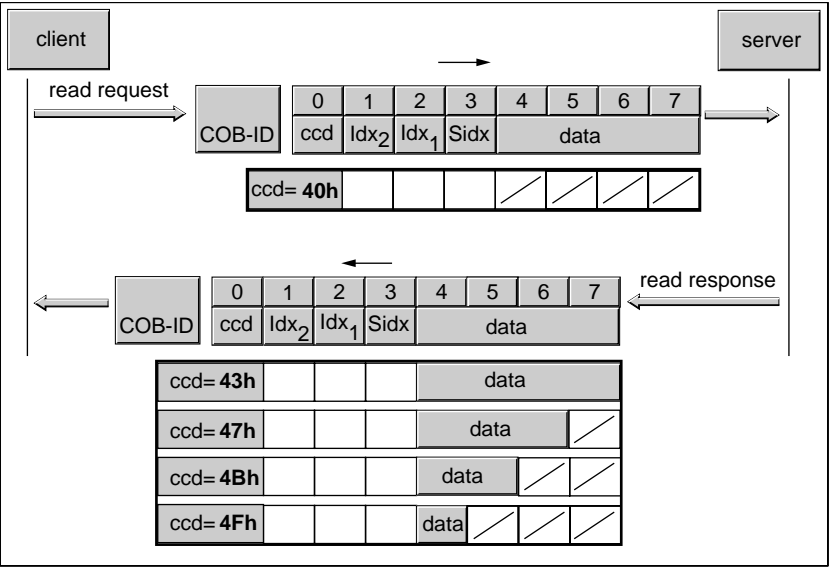


Fig 3.12 Reading a parameter value

Unused bytes in the data field are shown in the graphic as an oblique stroke. Their contents are not defined.

*ccd coding* The following table shows the command code used for transmitting a read value. It varies with the type of message and length of data transmitted.

Messages	Length of data used				Explanation
	4 bytes	3 bytes	2 bytes	1 byte	
read request	40h	40h	40h	40h	request read value
read response	43h	47h	4Bh	4Fh	return read value
error response	80h	80h	80h	80h	error

*Error response* If a message can not be evaluated without errors, the server sends an error signal. You will find details on evaluating error responses in the section „SDO error message“ on page 7-6.

### 3.3 Process data communication

#### 3.3.1 Overview

Process Data Objects (PDO: process data object) are used for real-time data exchange of process data such as the actual or set positions or operating status of the device. Transmission can be carried out very quickly because no additional administration data are sent and because no response is required from the receiver.

The flexible data length of a PDO message also increases the data throughput. A PDO message can transmit up to eight bytes of data. If only two bytes are used, only two bytes are transmitted.

The length of a PDO message and assignment of the data fields is defined via the PDO mapping procedure. You will find information on PDO mapping on page 3-17.

PDO messages can be swapped between network devices which produce or work with process data.

#### 3.3.2 PDO data exchange

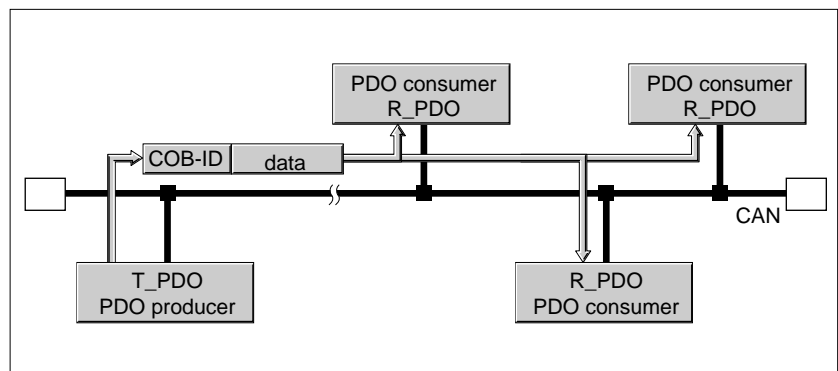


Fig 3.13 PDO message exchange

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

- synchronized
- event-driven, asynchronously
- at the request of a consumer, asynchronously

The task of controlling how the synchronized data are processed, is performed by the SYNC object. Synchronous PDO messages are transmitted immediately like all other PDO messages, but they are not evaluated until the next SYNC. Synchronized data exchange allows several drives to be started at the same time, for example.

PDO messages which are called up on request or as a result of an event, are evaluated by the network device immediately. For example, an emergency stop message should be sent asynchronously to enable the device to carry out an immediate shutdown.

The transmission type can be set separately for every PDO via sub-index 02h (transmission type). The objects are listed in the table on page 3-15.

*Requesting messages*      One or more network devices with consumer functions can request PDO messages from a producer. The producer is identified via the COB-ID in the request, and it responds with the required PDO.

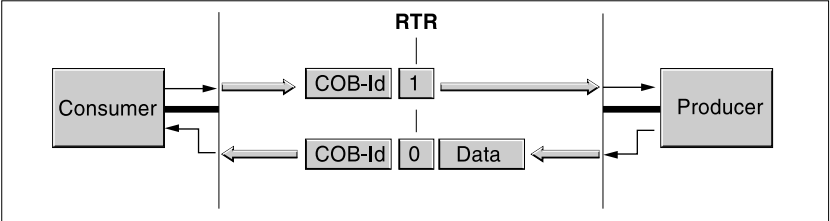


Fig 3.14    Requesting a message with RTR=1

The RTR bit in a CAN message is used to detect a request (RTR: remote transmission request). The COB-ID remains the same for both messages:

RTR=0: transmission of data

RTR=1: request for data.

*Setting the RTR request*      Each PDO can be set separately to define whether it should respond to RTR requests or not. The code is switched on or off via sub-index 01h, bit 30 of every PDO. The objects for this are listed in the table on page 3-15. Sub-index 02h of the object defines the transmission type. The PDO only responds to a request by RTR bit if RTR transmission has been switched on for that PDO. The sub-index values for using the RTR bit are:

Objects 140xh,180xh (x: 0..4) Sub-index 02h, „transmission type“	Explanation
252	RTR activated, synchronous
253	RTR activated, asynchronous

You will find an overview of all values relating to sub-index 02h in the object dictionary for the relevant object.

The positioning controller cannot request PDOs, but it can respond to PDO requests.

### 3.3.3 PDO Messages

*T\_PDO, R\_PDO* There is one PDO each available for sending and receiving PDO messages.

- The T\_PDO to transmit PDO messages (T: transmit),
- The R\_PDO to receive PDO messages (R: receive).

The number of T\_PDOs and R\_PDOs used by the device, can be determined via the *number of PDOs supported* object (1004h).



*The PDO settings given below correspond to the standard presets for the device unless otherwise specified, and can be read and set via communication profile objects.*

The positioning controller uses eight PDOs, four receive PDOs and four transmit PDOs. In the standard setting, all PDOs are evaluated or transmitted on an event-driven basis.

*Receive PDOs* The objects shown in PDOs R\_PDO1, R\_PDO2 and R\_PDO3 can be changed by means of PDO mapping. The objects for R\_PDO4 are permanently set.

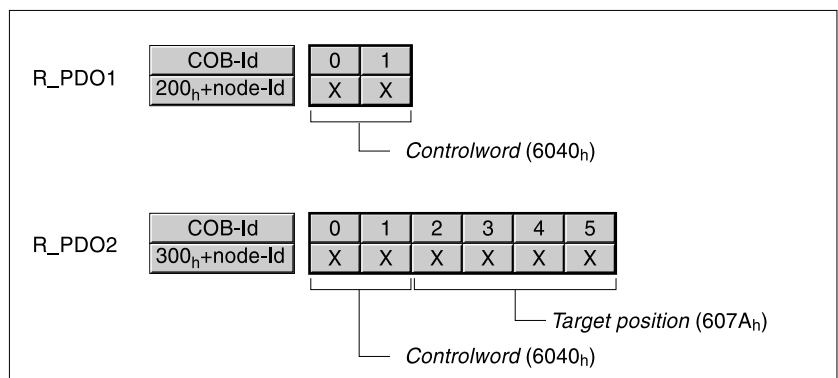


Fig 3.15 receive PDOs R\_PDO1 and R\_PDO2

*R\_PDO1* The first receive PDO contains the status machine's control word, object *Controlword* (6040h), with which the operating status of the drive can be set.

R\_PDO1 is evaluated asynchronously and can be configured by means of PDO mapping.

You will find information on the control word for the status machine in the „Operating modes“ chapter on page 5-3.

*R\_PDO2* The second receive PDO receives the control word and target position of a movement command, *Target position* object (607Ah), for a point-to-point positioning process in „profile position mode“. R\_PDO2 can be configured by means of PDO mapping.

The positioning controller processes R\_PDO2 synchronously with reception of the next SYNC object. R\_PDO2 can be used for the synchronized start-up of several drives.

You will find details on the SYNC object in the section „Synchronization“ from page 3-20.

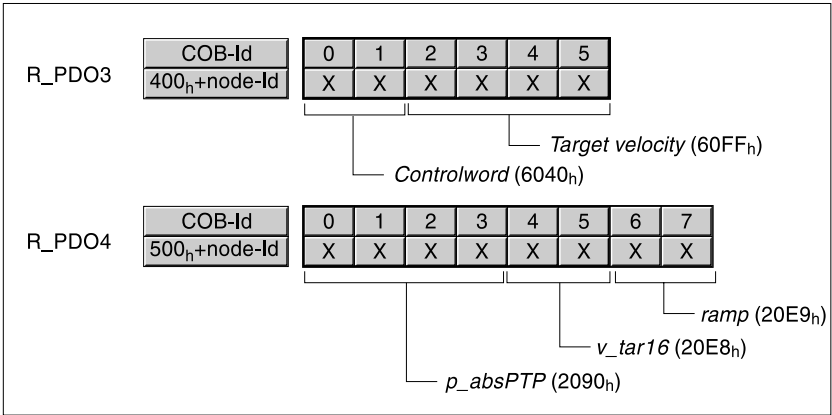


Fig 3.16 Receive PDOs, R\_PDO3 and R\_PDO4

- R\_PDO3** The third receive PDO contains the control word and set speed, *Target velocity* object (60FFh), for speed operation in „profile velocity mode“.
- R\_PDO3 is evaluated asynchronously. R\_PDO3 can also be used to carry other objects by means of PDO mapping.
- R\_PDO4** The fourth receive PDO is used to transmit manufacturer-specific object values for starting an absolute point-to-point positioning process:
- the target position, *p\_absPTP* object (2090h),
  - the set speed, *v\_tar16* object (20E8h)
  - the ramp values for acceleration and deceleration, *ramp* object (20E9h).

Receipt of R\_PDO4 also triggers an absolute point-to-point positioning operation. No other objects can be transmitted with R\_PDO4.

**Transmit PDOs** The objects shown in PDOs T\_PDO1, T\_PDO2 and T\_PDO3 can be changed by means of PDO mapping. The objects for T\_PDO4 are permanently set.

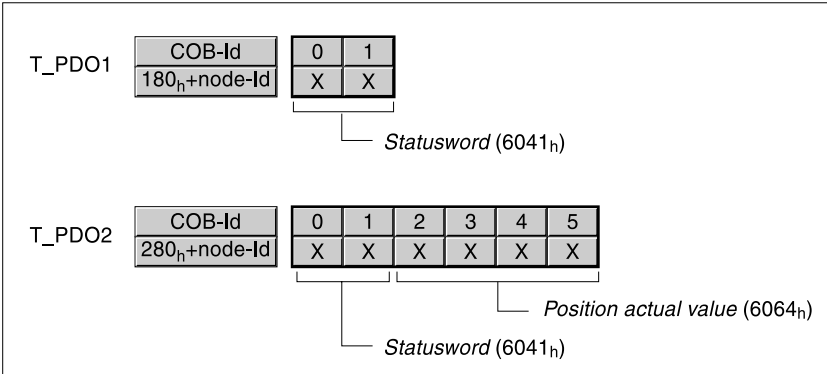


Fig 3.17 Transmit PDO T\_PDO1 and T\_PDO2

- T\_PDO1** The first transmit PDO contains the status word, the *Statusword* object (6041h), of the status machine.
- T\_PDO1 is event-driven and transmitted asynchronously with every change of the status word. T\_PDO1 can also be used to carry other objects by means of PDO mapping.
- You will find information about the status word of the status machine in the chapter „CANopen-status machine“ on page 5-4.



**T\_PDO2** The second transmit PDO carries the status word and the current position of the motor, the *Position actual value* object (6064h), for monitoring a point-to-point positioning operation in „profile position mode“.

T\_PDO2 is transmitted after a SYNC object has been received and on an event-driven basis. T\_PDO2 can also be used to carry other objects by means of PDO mapping.

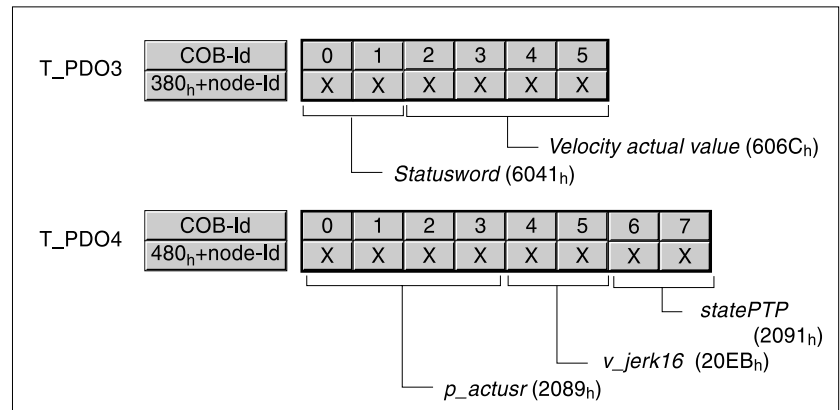


Fig 3.18 Transmit PDOs, T\_PDO3 and T\_PDO4

**T\_PDO3** The third transmit PDO shows the status word and current speed, the *Velocity actual value* object (606Ch), for monitoring speed operations in „profile velocity mode“.

T\_PDO3 is event-driven and transmitted asynchronously with every change of the status word. T\_PDO3 can also be used to carry other objects by means of PDO mapping.

**T\_PDO4** The fourth transmit PDO is used to transmit manufacturer-specific object values for monitoring an absolute point-to-point positioning process:

- the current position, the *p\_actusr* object (2089h),
- the current speed, the *v\_jerk16* object (20EBh)
- status information on the positioning mode, the *statePTP* object (2091h).

T\_PDO4 is event-driven and transmitted asynchronously with every change of the status information. T\_PDO4 cannot be used to carry other objects.

**PDO Settings** PDO settings can be read and changed by means of four communication objects:

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

Object	Explanation
4th transmit PDO parameter (1803h)	Settings for T_PDO4

*Activating PDOs* In the standard PDO setting, R\_PDO1 and T\_PDO1 are activated. The other PDOs must be first activated before the positioning controller can use them.

PDOs are activated via bit 31 (valid-Bit) in sub-index 01h of the particular communication object:

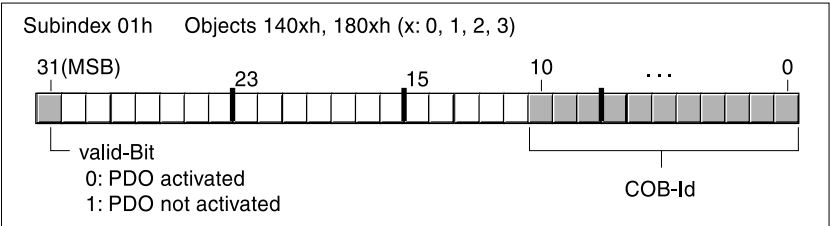


Fig 3.19    Activating PDOs via sub-index 01h, bit 31

Example setting for R\_PDO3 in object 1402h:

- sub-index 01h = 8000 04xxh: R\_PDO3 not activated
- sub-index 01h = 0000 04xxh: R\_PDO3 activated.

Values for „x“ in the example are dependent on the setting of the COB ID.

*PDO time intervals* The time intervals, „inhibit time“ and „event timer“ can be set for every transmit PDO.

The time interval, „inhibit time“, can be used to reduce the load on the CAN bus which for example can arise through the continuous transmission of T\_PDOs. If an interval time is set to a value other than zero, a PDO which has been transmitted is only re-sent after the interval has elapsed. The time is set via sub-index 04h.

The time interval „event timer“ triggers an event message on a cyclical basis. After the time interval has elapsed, the positioning controller transmits the event-driven T\_PDO. The time is set via sub-index 05h.

### 3.3.4 PDO mapping

Up to 8 bytes of data from different parts of the object dictionary can be transmitted in a PDO message. Showing the data in a PDO message is termed PDO mapping.

The following graphic shows the data exchange between PDOs and object directory for the objects in T\_PDO1, T\_PDO2, R\_PDO1 and R\_PDO2 for standard PDO settings.

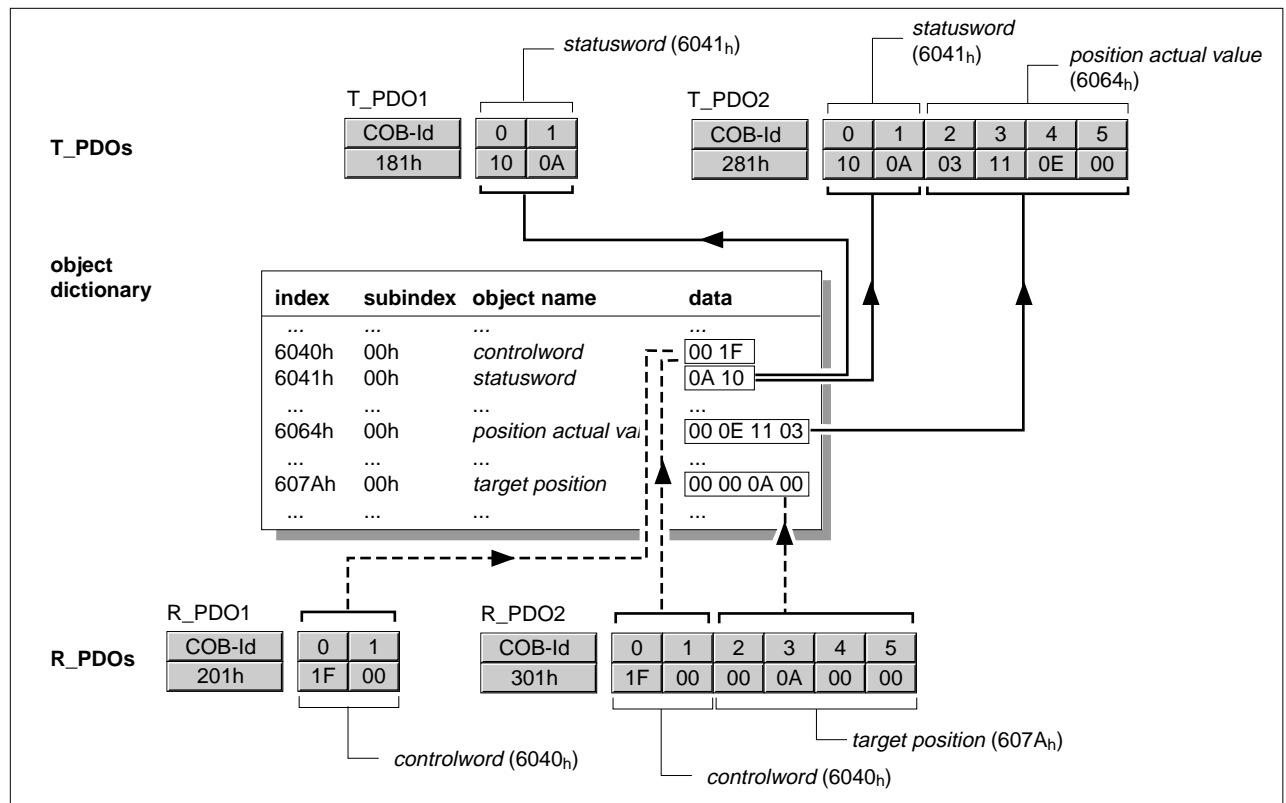


Fig 3.20 PDO mapping, here for a network device with node address = 1

#### dynamic and static PDO mapping

The settings for PDO mapping are defined for every PDO in a specially assigned communication object. If the PDO mapping settings for a PDO can be changed, we speak of dynamic PDO mapping for the PDO. Dynamic PDO mapping allows flexibility in the arrangement of various process data during operation.

With static PDO mapping, all objects are shown in accordance with a defined, unchangeable setting in the relevant PDO.

The positioning controller uses dynamic and static PDO mapping which can be read and – if dynamic mapping is possible – changed via the following communication objects.

Object	Explanation
1st receive PDO mapping (1600h)	PDO mapping for R_PDO1, dynamic
2nd receive PDO mapping (1601h)	PDO mapping for R_PDO2, dynamic
3rd receive PDO mapping (1602h)	PDO mapping for R_PDO3, dynamic
4th receive PDO mapping (1603h)	PDO mapping for R_PDO4, static
1st transmit PDO mapping (1A00h)	PDO mapping for T_PDO1, dynamic

Object	Explanation
2nd transmit PDO mapping (1A01h)	PDO mapping for T_PDO2, dynamic
3rd transmit PDO mapping (1A02h)	PDO mapping for T_PDO3, dynamic
4th transmit PDO mapping (1A03h)	PDO mapping for T_PDO4, static

*objects for PDO mapping* Not all objects can be transmitted by PDO. This applies to most communication objects and some device profile objects. Details are given for every object in the object directory from page 9-1 as to whether it can be operated via PDO mapping.

*PDO mapping memory* Up to 64 1 bit values can be shown for 64 different objects in one PDO. Every communication object for setting PDO mapping also offers 64 sub-index entries. A sub-index entry contains three pieces of information about the object: the index, the sub-index and the number of bits which the object occupies in the PDO.

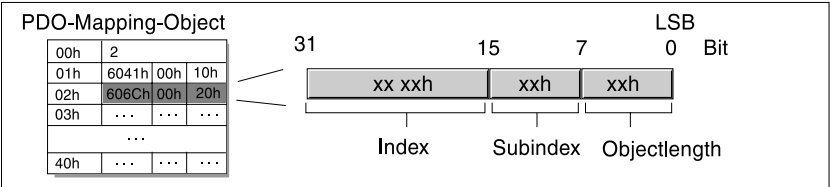


Fig 3.21 Structure of entries for PDO mapping

Sub-index 00h in the communication object holds the number of valid sub-index entries.

*setting PDO mapping* In order to change PDO mapping entries, sub-index 00h must be set to „0“ to deactivate the current PDO mapping. The new objects can then be entered in the communication object for PDO mapping by SDO transfer. After every transmission, the CANopen application checks whether the object exists in the object directory, and sends back an SDO error message in the event of an error.

If all the objects exist, the number of PDO objects mapped must be entered in sub-index 00h of the communication object. Here too the CANopen application checks the entries and sends back an SDO error message in the event of an error.

Example: PDO mapping for  
R\_PDO2 and T\_PDO2

The following graphic shows the entries in communication objects 2nd receive PDO mapping (1601h) and 2nd transmit PDO mapping (1A01h) for PDO mapping of R\_PDO2 and T\_PDO2 with two objects each.

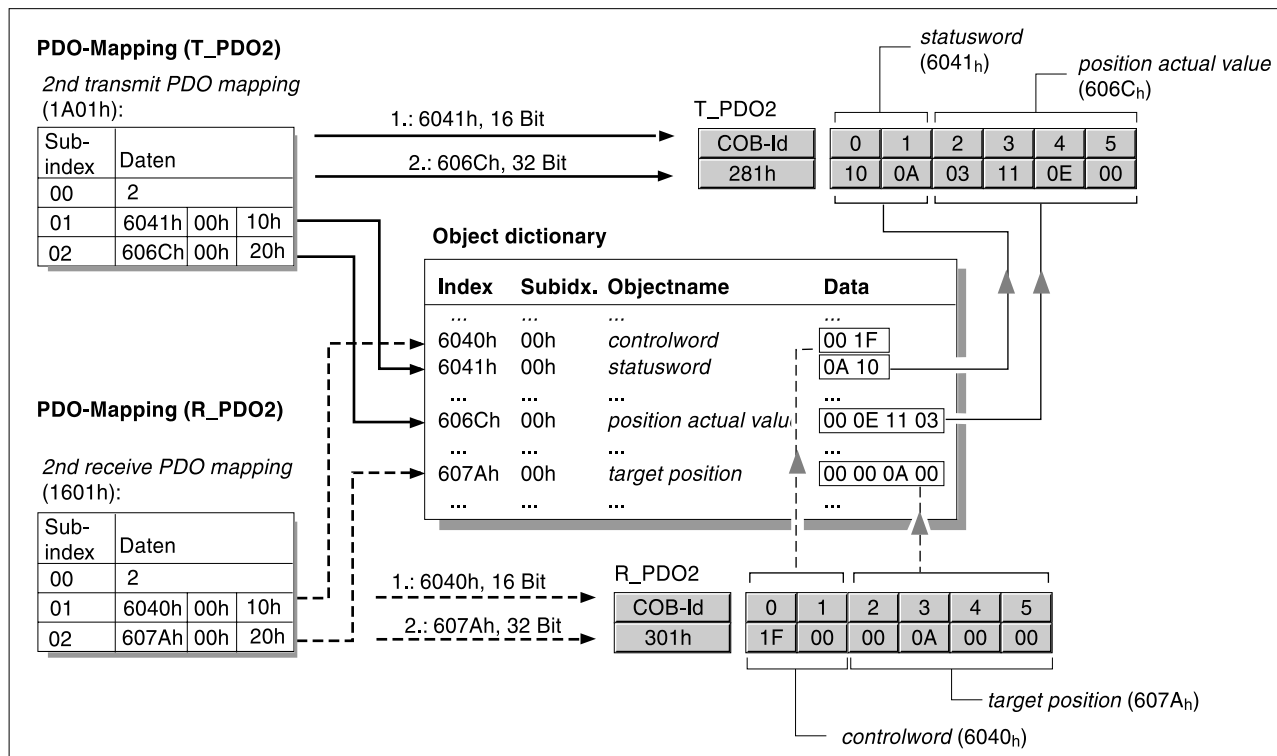


Fig 3.22 PDO mapping entries for T\_PDO2 and R\_PDO2

### 3.4 Synchronization

The SYNC synchronization object controls the synchronous exchange of messages between network devices, e. g. to enable several drives to be started up simultaneously.

The exchange of data follows the producer-consumer relationship. The SYNC object is sent to all devices by a network device, and can be evaluated by all devices which support synchronous PDOs.

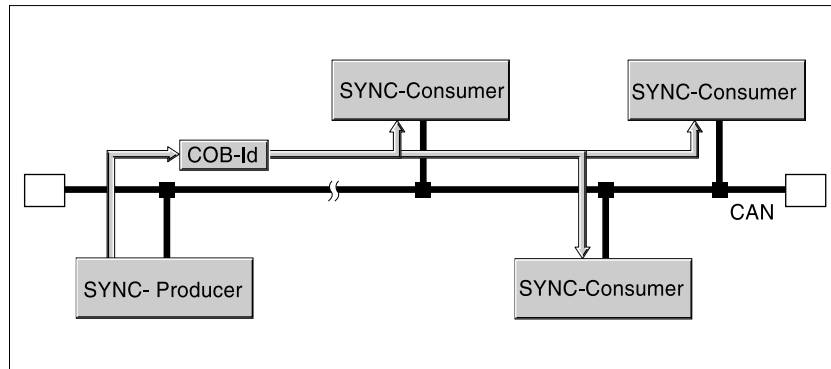


Fig 3.23 SYNC message

*Synchronization times* Two time values define the behavior of synchronous data transmission, the cycle time and the synchronous window length.

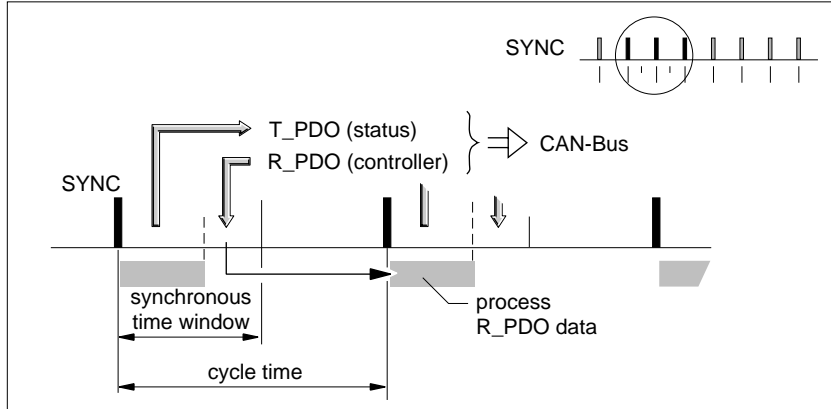


Fig 3.24 Synchronization times

The cycle time defines the interval between two SYNC messages. It is set by means of the *Communication cycle period* object (1006h).

The synchronous window length defines the period of time during which synchronous PDO messages must be received and sent. The window length is defined via the *Synchronous window length* object (1007h).

*Synchronous data transmission* Seen from the perspective of the SYNC receiver, status data are first sent in a T\_PDO, and then the new control data are received via an R\_PDO. The control data, however, are not processed until the next SYNC message arrives. The SYNC object itself does not transmit data.

*Cyclical and acyclical transmission*

The synchronous exchange of messages can be carried out cyclically or acyclically.

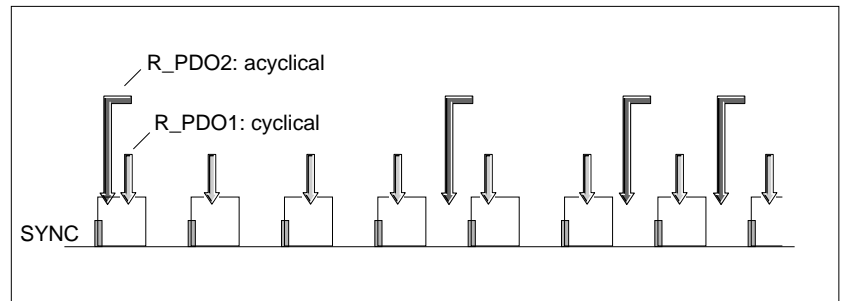


Fig 3.25 Cyclical and acyclical transmission

With cyclical transmission PDO messages are exchanged continuously in a defined cycle, e. g. with every SYNC message.

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

Whether a PDO operates cyclically or acyclically is stored in the *transmission type* sub-index (02h) of the relevant PDO parameter, e. g. for R\_PDO1 in the *1st receive PDO parameter* object (1400h), sub-index 02h.

*COB-ID, SYNC object*

In order to ensure fast transmission of the SYNC object, it is sent without confirmation and with high priority.

The COB-ID of the SYNC object is set to the standard value of 128 (80h). This value can be changed after the network has been initialized, by means of the *COB-ID SYNC Message* object (1005h).

*„start“ PDO*

For standard PDO settings, R\_PDO2/T\_PDO2 and R\_PDO3/T\_PDO3 are received and transmitted synchronously. Both PDOs are used for starting and monitoring operating modes. The synchronization allows an operating mode to be started simultaneously on several devices, and so, for example, to synchronize the advance of a multi-motor portal drive.

3.5 Emergency service

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

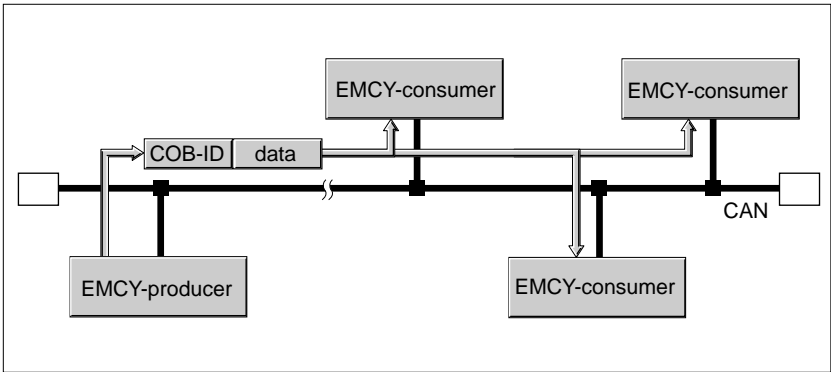


Fig 3.26 Error messages via EMCY objects

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 sending the message is ready for operation in the CAN network.

A boot-up message consists of the COB-ID of the EMCY object, and is transmitted without any data. The standard setting of the COB-ID is 128 (80h).

3.5.1 Error evaluation and error handling

EMCY message

If an internal device fault occurs, the device switches into error status in accordance with the CANopen status machine, see the Chapter entitled „CANopen state machine“ on page 5-4. At the same time it sends an EMCY message with the error register and error code.

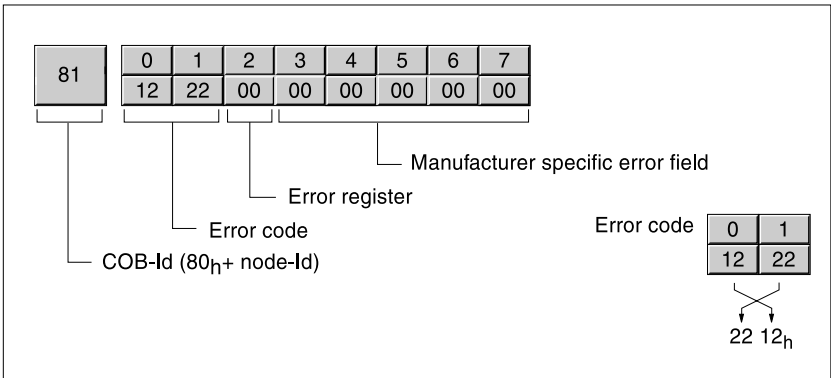


Fig 3.27 EMCY message

Byte 0, 1 - Error code: value also stored in the *Error code* object (603Fh).

Byte 2 - Error register: value also stored in the *Error register* (1001h) object.

Bytes 3-7, manufacturer-specific error field: manufacturer-specific errors, see „Error code table“ on page 7-3.



**COB-ID** The COB-ID is calculated from the node address for every device in the network which supports an EMCY object.

$$\text{COB-ID} = \text{Function code EMCY object (80h)} + \text{node-ID}$$

The function code of the COB-ID can be changed by means of the *COB-ID emergency* object (1014h).

**Error register and error code** The error register reports the error status of the device in bit-coded form. Bit 0 remains set as long as the error remains. The other bits designate the error type. The exact cause of the error can be determined through the error code. The error code is transmitted as a 2-byte value in Intel format, and must be converted byte by byte for evaluation.

You will find a list of all error messages as well as device responses and remedial measures in the Chapter entitled „Diagnostics and Error Correction“ from page 7-1 .

**Error memory** The positioning controller stores the error register in the *Error register* object (1001h), and the last error to have occurred in the *Error code* object (603Fh). At the same time, the last 20 error messages are stored in the order in which they have occurred in the *error field* object (2400h).

### 3.6 Network management services

Network Management (NMT) is part of the CANopen communication profile, and is used to initialize the network and network devices as well as to start, stop and monitor the devices during network operation.

NMT services are carried out in a master-slave relationship. The NMT master addresses individual NMT slaves via their node address. A message with the node address „0“ goes to all NMT slaves simultaneously.

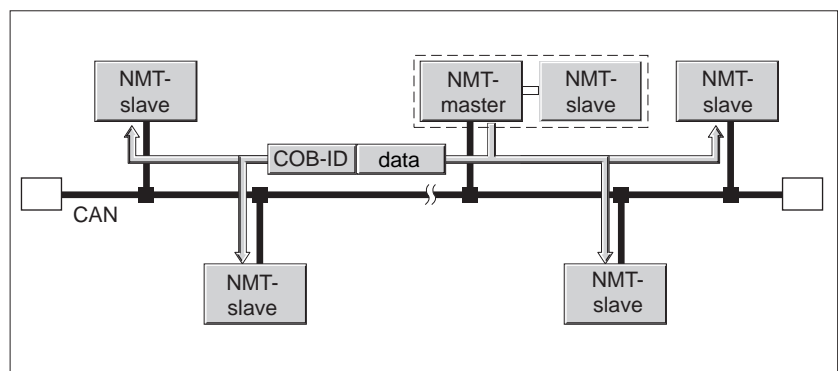


Fig 3.28 NMT services via the master-slave relationship

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

**NMT Network Services** NMT Services can be divided into two groups:

- Services for monitoring devices, for initializing devices for CANopen communication and for controlling the behavior of devices during network operation,
- Services for monitoring connections in order to ensure error-free network operation

3.6.1 NMT Services for monitoring devices

*NMT Status Machine* The NMT status machine describes the initialization and operating states of an NMT slave in network operation.

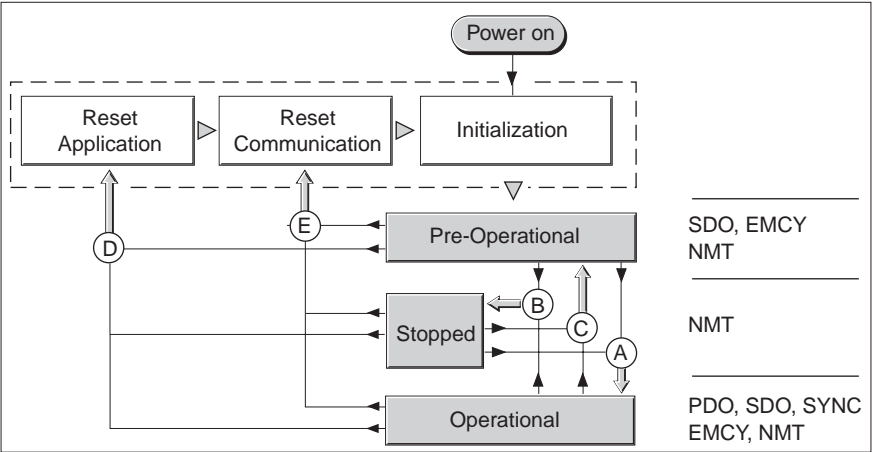


Fig 3.29 NMT Status Machine and available communication objects

All the communication objects which can be used in the relevant network status, are shown on the right-hand side of the graphic.

*Initialization* When the power supply is switched on (power on), an NMT slave automatically goes through an initialization phase which prepares it for CAN-Bus operation. After initialization is finished, the slave switches to „Pre Operational“ status and sends a boot-up message by EMCY object, containing no data. From now on, an NMT master can control the operational behavior of the NMT slave in the network by using five NMT services, shown as the letters A to E in the above graphic.

NMT Service	Tran- sition	Explanation
Start remote node	A	Switch to „Operational“ status Commence normal network operation with all devices
Stop remote node	B	Switch to „Stopped“ status Stop device communication in the network. If connection monitoring is activated, it remains on
Enter Pre-Operational	C	Enter „Pre-Operational“ status All communication objects can be used apart from PDOs.  The „Pre-Operational“ status can be used for configuration by SDOs: - PDO mapping - Start of synchronization - Start of connection monitoring
Reset node	D	Switch to „Reset application“ status Load stored device profiles data and switch automatically to „Reset communication“ status after the „Pre-Operational“ status
Reset communication	E	Switch to „Reset communication“ status Load stored communication profiles data and switch automatically to „Pre-Operational“ status

**Remanent data** If the power supply is switched on (power on), the unit loads the remanent object data from the EEPROM into RAM.

**NMT message** The NMT services for monitoring devices are transmitted as unconfirmed messages with the COB-ID = 0. They therefore receive as standard the highest transmission priority on the CAN-Bus.

The NMT device service data carrier consists of two bytes.

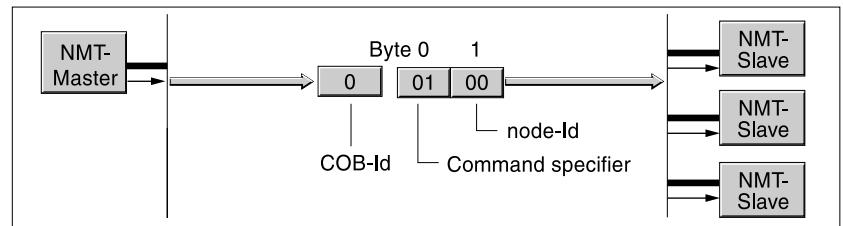


Fig 3.30 NMT message

The first byte, the „Command specifier“, specifies the NMT service used.

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

The second byte addresses the receiver of the NMT message with a node address between 1 and 127 (7Fh). A message with the node address „0“ is addressed to all NMT-Slaves.

### 3.6.2 NMT services for connection monitoring

The connection monitoring function monitors the communication status of network devices to enable the system to react to the failure of a device or to an interruption in the network.

There are three connection monitoring NMT services available with the positioning controller under CANopen:

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

#### Node guarding

In carrying out its „node guarding“ and „life guarding“ monitoring functions, the NMT master requests an NMT status signal at regular intervals and expects to receive a response by the end of the interval. The interval is set via the parameter „guard time“.

The following graphic shows an error message generated due to not having received a response from the NMT slave by the end of the third cycle.

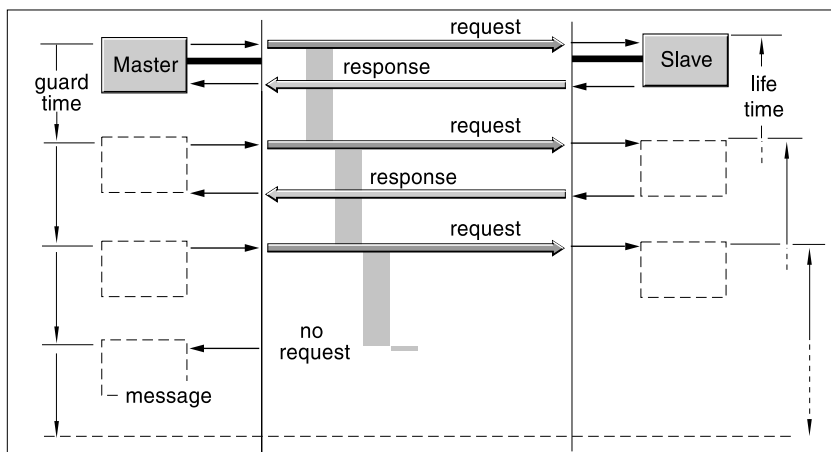


Fig 3.31 „Node guarding“ and „Life guarding“ with intervals

The NMT master reports a connection fault to the upstream master program if

- the slave does not respond by the end of the „guard time“ interval
- if the slave's NMT status has changed without any prompting by the NMT master.

#### Life guarding

If „Life guarding“ is activated, the NMT master must send its enquiry to the slave within the adjustable „life time“ period. If the period is exceeded, the NMT slave assumes that the NMT master has failed, and signals a connection fault. The „life time“ counter is restarted with every enquiry.

#### Intervals

The „guard time“ and „life time“ intervals are set via two communication objects:

- the „guard time“ interval via the *Guard time* object (100Ch)
- the „life time“ interval as a factor of the „guard time“ interval via the *Life time factor* object (100Dh)

$$\text{„life time“} = \text{guard time (100Ch)} \times \text{life time factor (100Dh)}$$

**COB-ID** Connection monitoring is carried out with the aid of the communication object *NMT error control* (700h+node-ID). The COB-ID is worked out from the node address for every NMT slave:

$$\text{COB-ID} = \text{Function code } \textit{NMT error control} (700\text{h}) + \text{node-ID}.$$

The function code of the COB-ID can be changed in the „Pre-Operational“ NMT status using the *COB-ID guarding protocol* object (100Eh).

**NMT message** On receiving a request from the NMT master, the NMT slave responds by sending a data byte.

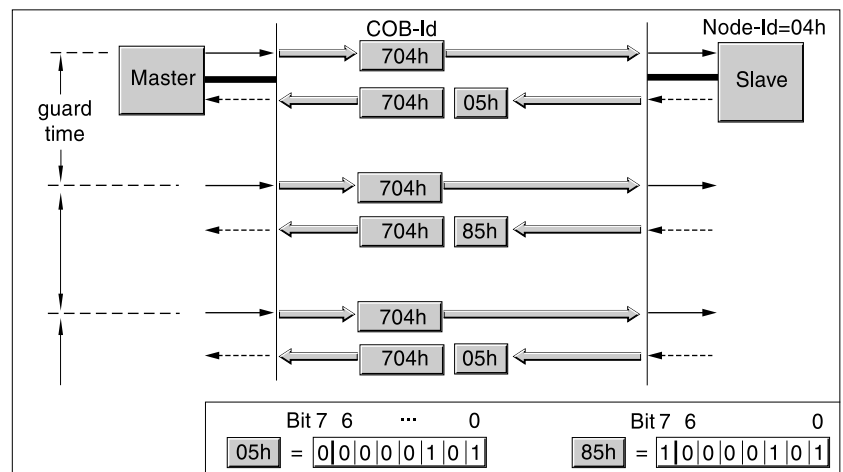


Fig 3.32 Response of the NMT slave

Bits 0 to 6 designate the NMT status of the slave:

4 (04h): „Stopped“

5 (05h): „Operational“

127 (7Fh) : „Pre-Operational“

Bit 7 changes its status between „0“ and „1“ after every „guard time“ interval, to enable the NMT master to recognize and ignore any second response in the same interval period. The first request at the start of the connection monitoring function, begins with bit 7 = 0.

Connection monitoring may not be activated during the initialization phase of a device. The status of bit 7 is reset as soon as the device passes the NMT status „Reset communication“.

Connection monitoring continues to run in the NMT status „Stopped“.

**Heartbeat** „Heartbeat“ is used for continuous connection reports from individual devices. „Heartbeat“ and „node guarding“ may not be used at the same time.

If a device supports monitoring by means of „Heartbeat“ and „Node/Life guarding“, „Heartbeat“ monitoring should be preferred. A supervising NMT master is not required. The task of sending connection messages and of carrying out the monitoring function can be performed by every „Heartbeat“ device.

In accordance with the producer-consumer relationship, the „Heartbeat“ producer transmits a data byte at „Heartbeat Producer Time“ intervals. „Heartbeat“ consumers monitor the „Heartbeat“ messages. The consumer expects to receive the message before the end of the „Heartbeat Consumer Time“ interval. If it receives the message later, it generates an error message.

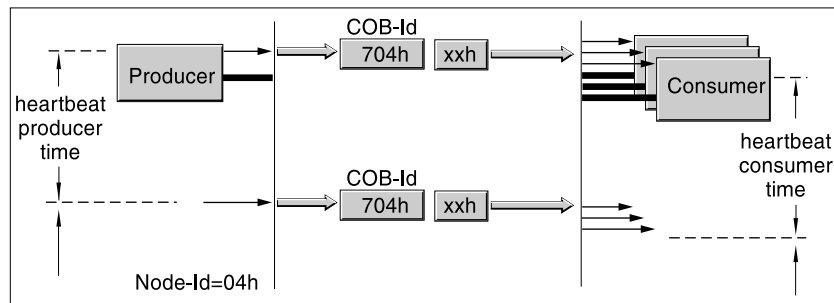


Fig 3.33 „Heartbeat“ monitoring

Data byte for NMT status evaluation of the „Heartbeat“ producer:

0 (00h): „Boot-Up“

4 (04h): „Stopped“

5 (05h): „Operational“

127 (7Fh) : „Pre-Operational“

**Intervals** The „Heartbeat producer time“ interval is set via the *Producer heartbeat time* object (1017h). The „Heartbeat consumer time“ intervals for one or more consumers are specified via the *Producer heartbeat time* object (1016h). The value zero deactivates the monitoring of a consumer.

The intervals are given in 1 ms steps and may not be set lower for the consumer than for the producer. The producer's interval re-starts after every „Heartbeat“ message is received.

**Start of monitoring function** The „Heartbeat“ monitoring function begins as soon as the producer's interval is larger than zero. If „Heartbeat“ monitoring is already active during the NMT status change to „Pre-Operational“, „Heartbeat“ monitoring will start as soon as the „Boot-Up“ message is sent. You will find information on the „Boot-Up“ message on page 3-22.

Twin Line units can monitor each other by means of „Heartbeat“ messages. In doing so, they perform the consumer and producer functions simultaneously.

## 4 Settings for operation in CAN bus

### 4.1 EMC

Where wiring is used in an electromagnetically severe environment, EMC requirements must be observed when laying cables and making connections.

#### *EMC measures*

The following measures are necessary with a view to trouble-free field bus operation. They complement the device-specific EMC measures in the device manual.

EMC measures	Effect
Use cables with braided screens and foil screens	discharge interference voltage
Field bus cables can be laid in one duct together with signal cables and analogue cables. Do not lay them with DC and AC cables over 60 V.	avoid mutual interference
Use potential equalization lines for systems with - wide-area installations - different power supplies - cross-building networks	discharge interference currents
Use fine-wire potential equalization lines	discharge also of high-frequency interference currents

To protect against interference, screens for digital cables are connected on both sides. Differences in potential can lead to unacceptable currents in the screen, and must be prevented by the use of potential equalization lines. For cables up to 200 m in length, a diameter of 16 mm<sup>2</sup> is sufficient, but for longer lengths a diameter of 20 mm<sup>2</sup> must be used.

### 4.2 Device installation

Correct mechanical and electrical installation of the Twin Line unit is a prerequisite of network installation, as is that the units are successfully set up.

Carry out the set-up in accordance with the manual. The Twin Line unit is now ready to be used in the network.

#### 4.2.1 Setting address and baud rate

Up to 32 controllers can be addressed in a CAN-Bus network branch, and up to 128 controllers in an extended network. Every device is identified by means of a unique address. The network address for a Twin Line unit is preset to 127.

The baud rate is preset to 125 kbaud.

#### *Setting address and baud rate*

There are two ways of setting the address and the baud rate. The „Settings.IO\_mode“ parameter (29:31) is used to set the unit up for one of the two options:

- „IO\_mode“ = „0“: setting via the signal interface
- „IO\_mode“ = „1“ or „2“: setting via parameters.

When making the setting via the signal interface, the controller interprets the switching states of inputs ADR\_1 to ADR\_64 as the network address, and that of inputs BAUD\_1 and BAUD\_4 as the baud rate setting. You will find assignment and setting options for the interface described in the manual in the chapter on connecting the signal interface.

When using parameters for setting address and baud rate, the address is entered locally by means of an operating unit in parameter „M4.addr-Can“ (24:24), and the baud rate in parameter „M4.baudCan“ (24:25).

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

#### 4.2.2 Profile selection

The positioning controller can work with different field bus profiles. When running under CANopen, the field bus profile can be set via inputs MODE1 and MODE2 of the signal interface or via parameter M4.profilCan (24:23).

The parameter can be changed by means of the TL CT operating software or with the hand-held operating unit, TL HMI.

Profile	Signal-interface	Parameter
SIG Positec-profile	MODE1=0 MODE2=0	M4.profilCan =0
CANopen-profile	MODE1=1 MODE2=0	M4.profilCan =1

Each branch of the network can only be run with one network profile. All devices must be set for the same network profile.



### 4.2.3 Connection of the Twin Line unit

For connecting to a CAN-Bus network, Twin Line units are fitted with the CAN-C module in slot M4.

**Module interface** The CAN-C module is fitted with a 9-pin, UNC threaded sub-D plug and sub-D socket. The pin assignment is identical for both interface connections.

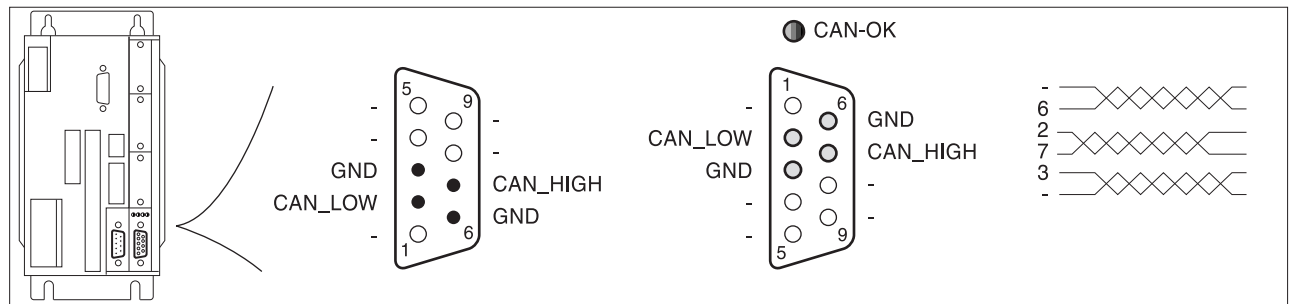


Fig 4.1 Interface connections for the field bus module with plug and socket

Pin	Signal	Colour <sup>1)</sup>	Pair	Explanation	I/O
1	-	-	1	not assigned	-
6	GND	green	1	ground	-
2	CAN_LOW	white	2	data line, inverted	I, O
7	CAN_HIGH	brown	2	data line	I, O
3	GND	grey	3	ground	-
8	-	pink	3	not assigned	-
4	-	-	-	not assigned	-
9	-	-	-	not assigned	-
5	-	-	-	not assigned	-

1) Colour details relate to the cable available as an accessory.

For units with a hood, the cable must be led downwards from the connection point.

#### Cable specifications

- screened cable
- minimum diameter of signal wires: 0.14 mm<sup>2</sup>
- twisted pairs
- screen earthed at both ends
- maximum length dependent on number of network devices, baud rate and signal transmission times. The higher the baud rates, the shorter the bus cable must be.  
Guidelines: 40 m at 1 Mbit/s, 500 m at 100 kbit/s

#### Display

The LED on the CAN-C module lights for c. 2 seconds when field bus data have been correctly received.

#### 4.2.4 Termination

Field bus devices are connected in segments in a linear arrangement, consisting of 32 master and slave devices. Each segment must be terminated at both ends with an active bus terminator or a 120  $\Omega$  terminating resistor. Termination can be activated via a switch in the connector of the outer bus devices using the network accessory cable supplied with the Twin Line unit.

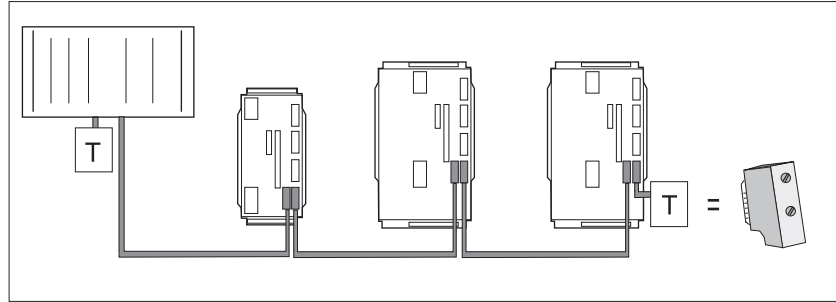


Fig 4.2 Termination of a bus segment via connector or terminator (T)

When there are more than 32 devices, individual segments are fitted with amplifiers - repeaters. Each segment must then be terminated separately.

## 5 Operating modes

### 5.1 Overview

*Operating states* The positioning controller switches between various operating states after it has been switched on and also in order to execute an operating mode. Some of these changes of status are carried out by the drive automatically. Others must be explicitly set in order to guarantee safe operation.

*Operating modes* The positioning controller functions in four automatic modes:

- speed mode, manufacturer-specific or in accordance with CANopen profile DSP 402 as „profile velocity mode“
- point-to-point mode, manufacturer-specific or in accordance with CANopen profile DSP 402 as „profile position mode“
- homing mode, manufacturer-specific or in accordance with CANopen profile DSP 402 as „homing mode“
- electronic gear mode, when a module is fitted to slot M1.

The electronic gear mode can only be operated with manufacturer-specific objects.

Access to manufacturer-specific operating modes is granted via the "SIG mode" in the DSP 402 device profile.

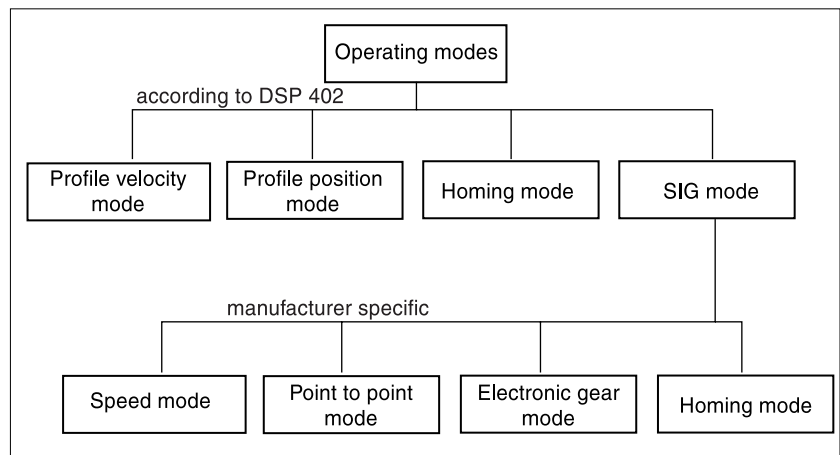


Fig 5.1 operating modes with the positioning controller

*Standardized operating modes* The standardized operating modes can be started and monitored with the objects of the CANopen device profile DSP 402. These include

- point-to-point mode with the standardized objects in the „profile position mode“ object group
- speed mode with the standardized objects in the „profile velocity mode“ object group
- homing mode with the standardized objects in the „homing mode“ object group.

*Manufacturer-specific operating modes* The manufacturer-specific operating modes exploit the full functional scope of the positioning controller. The operating modes are started and monitored using manufacturer-specific objects.

You will find details on the manufacturer-specific objects described in the manual for the positioning controller. The names are listed in the parameter description:

<groupname>.<parametername>

Example: „Status.n\_ref“ designates the „n\_ref“ parameter in the „Status“ group.

## 5.2 Operating states and transitions

In order to initiate and monitor an operating mode, the positioning controller runs through a series of operating states after being switched on which are either activated automatically or via CANopen objects.

*State machine* The relationships between operating states and transitions in the positioning controller are shown under CANopen in the state machine of the DSP 402.

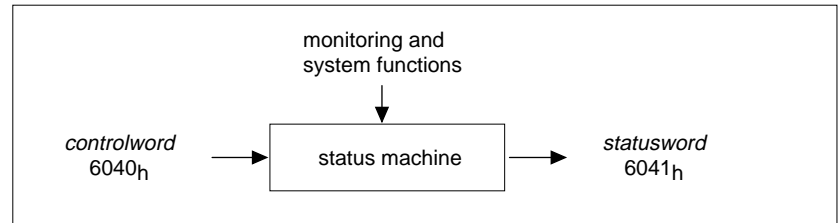


Fig 5.2 Changing and monitoring operating status

Operating states are set by the user by means of the CANopen object *controlword* (6040h), and monitored with the *statusword* object (6041h). Monitoring functions and system functions in the unit such as temperature and current monitoring, supervise and influence the operating states.

The state machine of the DSP 402 refers to drive units. In contrast, the NMT state machine for the DS 301 describes the operating states in network operation. You will find details on the NMT state machine in the Chapter entitled „CANopen Communication“ on page 3-23.

5.2.1 CANopen state machine

The state machine of the DSP 402 can be shown graphically as a flow chart.

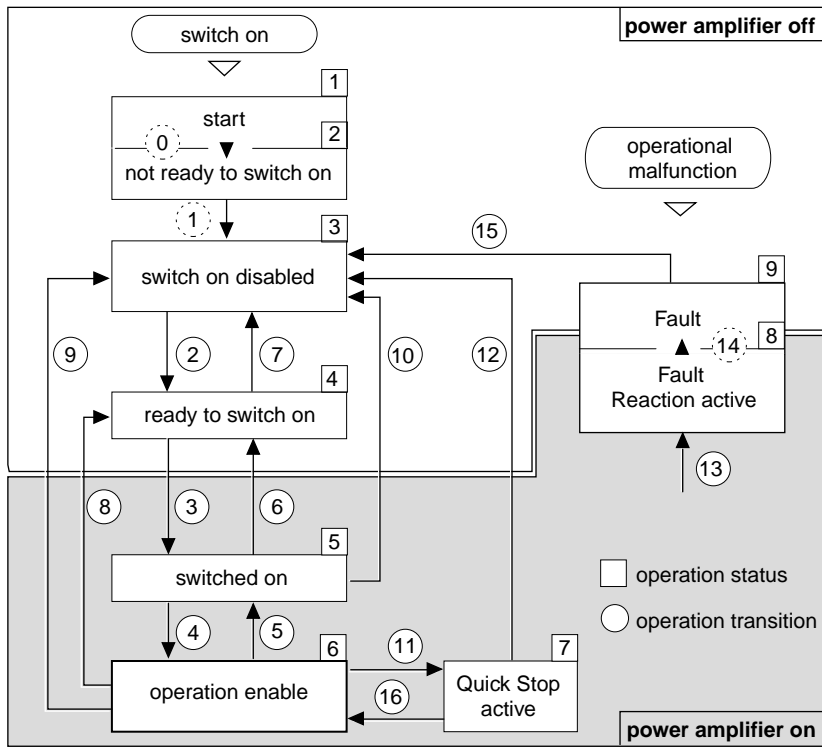


Fig 5.3 Diagram of state machine

Operating states are shown in the diagram as squares from 1 to 9, and the transitions as circles from 0 to 16.

Operating states

Status	Operating status	Explanation
1	Start	24-V switched on, Initialization of internal electronics
	Not ready to switch on	power amplifier is not ready to switch on
3	Switch on disabled	Switching on power amplifier is disabled
4	Ready to switch on	power amplifier ready to switch on
5	Switched on	power amplifier switched on, no active mode
6	Operation enable	Unit operating in the set mode
7	Quick Stop active	Quick-Stop will be executed
8	Fault reaction active	Fault detected, fault reaction will be activated, if possible
9	Fault	

The Positioning controller displays the operating states via the 7 segment display. The operating displays are explained in the technical documentation on the device in the chapter on diagnostics and error correction.

**Transitions** Transitions are triggered by a command or in response to a monitoring signal. A command is transmitted by means of the *controlword* object (6040h).

The following table shows transitions which are triggered by a command. Transitions 0, 1, 14 run automatically in the unit and are not activated by commands.

Transition	Status from..to	Command	Response
2	3→4	Shutdown	no response
3	4→5	Switch On	switch on power amplifier
4	5→6	Enable Operation	execute prescribed movement command
5	6→5	Disable Operation	abort movement command
6	5→4	Shutdown	switch off power amplifier
7	4→3	Quick Stop	no response
8	6→4	Shutdown	switch off power amplifier immediately, no „Quick Stop“
9	6→3	Disable Voltage	switch off power amplifier immediately, no „Quick Stop“
10	5→3	Disable Voltage	switch off power amplifier immediately, no „Quick Stop“
11	6→7	Quick Stop	execute „Quick Stop“
12	7→3	Disable voltage	switch off power amplifier immediately, even if „Quick Stop“ is still active
15	9→3	Fault Reset	acknowledge error in order to leave „Fault“ status
16	7→6	Enable Operation	from „Quick Stop“ status, continue set movement command

**Fault response** Transition 13 triggers a fault response as soon as a monitoring signal reports a malfunction to which the unit must respond.

A malfunction can for example be reported by a temperature sensor. The unit interrupts the current movement command, executes a fault response, e. g. an emergency stop by means of Quick Stop or switching off the power amplifier, and then switches into the operating status „Fault“. Every malfunction is reported on the network by means of an EMCY message.

In order to leave the operating status „Fault“, the cause of the fault must be corrected and the fault status bit in the *controlword* object reset.

## 5.2.2 Changing operating states

*with standardized objects*

When standardized operating modes are being used, operating states are set by means of the control word, the *controlword* object (6040h). For a status change, bits 0 to 3 and bit 7 are relevant.

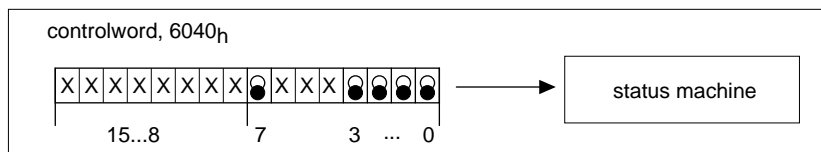


Fig 5.4 Monitoring operating states with the status word

Obj. <i>controlword</i> (6040h)		Bit 7, Reset Fault	Bit 3, Enable operation	Bit 2, Quick Stop	Bit 1, Disable Voltage	Bit 0, Switch On
Command	Change of status to					
2, 6, 8: Shutdown	4: Ready to switch on	X	X	1	1	0
3: Switch on	5: Switched on	X	X	1	1	1
7, 9, 10, 12: Disable Voltage	3: Switch on disabled	X	X	X	0	X
7, 10: Quick Stop 11: Quick Stop	3: Switch on disabled 7: Quick Stop active	X	X	0	1	X
5: Disable operation	5: Switched on	X	0	1	1	1
4, 16: Enable operation	6: Operation enable	X	1	1	1	1
15: Fault reset	3: Switch on disabled	0 -> 1	X	X	X	X

Bit states in the fields marked with an „X“ have no significance for the particular change of status. Bits 4 to 6 are used for settings specific to the operating mode. You will find details in the description of the individual operating modes further on in this chapter.

In order to achieve the speediest possible access to a status change, the control word data are mapped in the first two bytes of PDOs, R\_PDO1, R\_PDO2 and R\_PDO3.

*with manufacturer-specific objects*

The operating states of manufacturer-specific operating modes must be set using manufacturer-specific objects. The positioning controller makes the *driveCtrl* object (2040h) available for this purpose. The operating status can be changed by activating bits 0 to 3.

The bit values are always 0, with the result that writing to a bit automatically causes it to change from 0 -> 1.

Bits 3..0	Control word	Explanation
0 0 0 1	Disable	switch off power amplifier
0 0 1 0	Enable	switch on power amplifier
0 1 0 0	Quick Stop	effect stop by means of Quick Stop
1 0 0 0	Fault Reset	acknowledge error message



### 5.2.3 Monitoring operating states

*with standardized objects* When standardized operating modes are being used, operating states are monitored via bits 0 to 3, 5 and 6 of the status word which can be read through the *statusword* object (6041h).

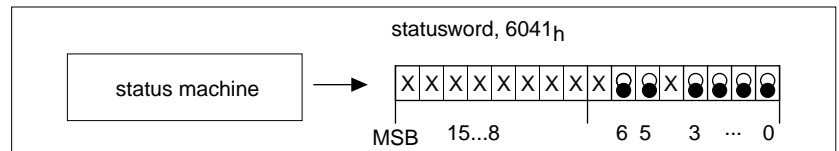


Fig 5.5 Monitoring operating state with the status word

Obj. <i>statusword</i> (6041h)	Bit 6, Switch on disable	Bit 5, Quick Stop	Bit 3, Fault	Bit 2, Operation enable	Bit 1, Switch on	Bit 0, Ready to switch on
<b>status</b>						
2: Not ready to switch on	0	X	0	0	0	0
3: Switch on disabled	1	X	0	0	0	0
4: Ready to switch on	0	1	0	0	0	1
5: Switched on	0	1	0	0	1	1
6: Operation enable	0	1	0	1	1	1
7: Quick Stop active	0	0	0	1	1	1
9: Fault	0	X	1	1	1	1

*Bit 4, voltage disable* Bit 4 zeigt die Reaktion auf den Zustandswechsel "Disable Voltage". Wird das Bit zurückgesetzt, ist die Endstufe ausgeschaltet.

*Bit 7, warning* Bit 7=1 indicates a warning message without interrupting the operation.

*Bits 8, 14 and 15* These bits are assigned manufacturer-specific functions.

*Bit 9, remote* If bit 9 is set, the unit will carry out commands over the CAN bus by PDO. If bit 9 is reset, the unit will work in local mode. SDO messages can still be passed over the CAN bus.

*Bits 10..13* Bits 10 to 13 are used to monitor the current operating mode. You will find details on bit states in the section on „Monitoring the operating mode“ on page 5-10.

In order to achieve the speediest possible access to the operating status, the control word data are mapped in the first two bytes of PDOs, T\_PDO1, T\_PDO2 and T\_PDO3.

*CANopen example „switching operating status“* The CANopen program example shows the switching of operating states for a network device with the node address 1.

COB ID	Data	Explanation
0	01 00	For all devices: NMT status change to "Operational" and start
181	xx 04	T_PDO1: report on current status (any)
201	00 00	R_PDO1: request for status change to Switch On disabled
181	40 00	T_PDO1: defined operating status: Switch On disabled
201	06 00	R_PDO1: request for status change to Ready to Switch On
181	21 04	T_PDO1: change of data in status word; Ready to Switch On
201	07 00	R_PDO1: request for status change Ready to Switch On -> Switched On
181	23 04	T_PDO1: change of data in status word: Switched On
201	0F 00	R_PDO1: request for status change Switched On -> Operation Enable
181	27 04	T_PDO1: change of data in status word: Operation Enable

*with manufacturer-specific objects*

The operating states of manufacturer-specific operating modes are monitored by means of the manufacturer-specific object, *driveStat* (2041h). Bits 0 to 3 give information on the operating status:

Bits 3..0 Operating status		Explanation
-	-	24V switched on
...0001	1 - Start	initialisation of device electronic systems
...0010	2 - Not ready to switch on	amplifier is not ready to switch on.
...0011	3 - Switch on disabled	amplifier disabled.
...0100	4 - Ready to switch on	amplifier is ready to switch on.
...0101	5 - Switched on	amplifier switched on
...0110	6 - Operation enable	device working in the set mode.
...0111	7 - Quick Stop active	Quick Stop to be performed.
...1000	8 - Fault reaction active	fault reaction activated
...1001	9 - Fault	fault display

Operating states 0..3, 5, 8 and 9 are transitional states which the controller does not remain in if working properly.



*When a manufacturer-specific operating mode is being carried out, changes of status and status monitoring must be conducted with manufacturer-specific objects, changes of status and initiation of standardized operating modes with the objects, controlword (6040) and statusword (6041h).*

*Control words and status words belonging to each operating mode group, use different starting and monitoring mechanisms for passing on and monitoring data.*

You will find a description of the function of the two objects, *driveCtrl* (2040h) and *driveStat* (2041h) along with examples in the field bus manual „CAN-Bus“ for the positioning controller, in the chapter on running the unit in a field bus. The objects can be found under the following parameters in the manual:

- *driveCtrl* (2040h): *Commands.driveCtrl* parameter (28:1)
- *driveStat* (2041h): *Status.driveStat* parameter (28:2).

## 5.3 Setting and monitoring operating modes

### 5.3.1 Setting the operating mode

**Requirements** One operating mode cannot be carried out in parallel to a second. If one operating mode is active, a switch can only be made to a different mode once the current operation has been completed or aborted.

An operating mode has been completed when the drive is at a standstill, e.g. when the target position in a positioning operation has been reached. If an error occurs during an operation which leads to the current mode being aborted, the operation can be continued or a switch made to a different mode once the error has been corrected.

**changing the operating mode** When changing the operating mode under CANopen, a distinction is made between standardized and manufacturer-specific objects.

There are two objects available for displaying and changing standardized operating modes.

- *modes of operation display* (6061h): display current operating mode
- *modes of operation* (6060h): change operating mode.

Both objects use the same values:

Operating mode	Value	Description
point-to-point operation in „profile position mode“	1	from page 5-12
speed operation in „profile velocity mode“	3	from page 5-17
homing in „homing mode“	6	from page 5-23
SIG mode, manufacturer-specific mode	-1	from page 5-25

**Manufacturer-specific operating modes** In order to change to a manufacturer-specific mode, *modes of operation* (6060h) must be set to „-1“ to switch to SIG mode. Operating modes can then be initialized and monitored using manufacturer-specific objects.

If *modes of operation display* (6061h) is showing SIG mode with „-1“, the *xMode act* object (2042h), bits 0..4 will reveal the set manufacturer-specific mode. The bit values are specific to each device.

### 5.3.2 initiating operating mode

For standardized operating modes the unit receives the request to start the set mode via bit 4 in the *controlword* object (6040h). Bits 4 to 6 are assigned specifically for each operating mode.

Operating mode	controlword (6040h)	Explanation
point-to-point operation in „profile position mode“	Bit 4: <i>New setpoint</i>	0-> 1: Start positioning operation or prepare follow-on positioning
	Bit 5: <i>Change set immediately</i>	Only applies to <i>New setpoint</i> 0->1: 0: Activate new setpoints when target position reached 1: Activate new setpoints immediately
	Bit 6: <i>Absolute / relative</i>	0: Relative positioning 1: Absolute positioning
homing in „homing mode“	Bit 4: <i>Homing operation start</i>	1: Start homing
	Bits 5, 6: -	not assigned

For speed operations in „profile velocity mode“, bits 4 to 6 are not assigned. Speed mode begins when a setpoint is communicated to the device.

#### Manufacturer-specific operating modes

Operating modes are activated when the manufacturer-specific starting-object is received. A new setpoint or value for the operating mode is conveyed with the starting object.

Operating mode	Starting object	Twin Line parameters
point-to-point mode		
- absolute positioning	<i>p_absPTP</i> (2090h)	PTP.p_absPTP
- relative positioning	<i>p_relPTP</i> (2092h)	PTP.p_relPTP
Speed mode	<i>velocity</i> (2095h)	VEL.velocity
Electronic gear	<i>startGear</i> (209Dh)	Gear.startGear
Manual movement	<i>startMan</i> (20A9h)	Manual.startMan
homing	<i>startHome</i> (20F0h)	Home.startHome

You will find details of the values and settings of the starting objects in the manual for the positioning controller.

### 5.3.3 Monitoring the operating mode

#### Standardized operating modes

The drive reports status information about the set operating mode via bits 10 to 13 in the *statusword* object (6041h).

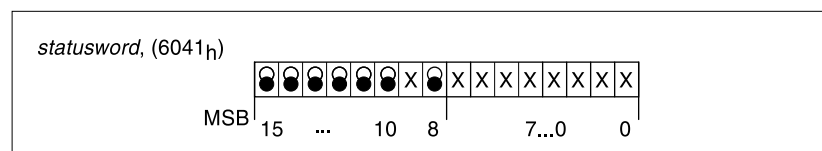


Fig 5.6 Status messages on operating mode

*Target position and limit switches*

<b>statusword (6041h)</b>	<b>Explanation</b>
Bit 10: <i>target reached</i>	1->0: New target position conveyed 0->1: Target position reached (setpoint = actual value) or motor standstill after stop request
Bit 11: <i>internal limit active</i>	0->1: limit switch position passed

*Status of operating mode*

Bits 12 and 13 assigned specifically for each operating mode.

<b>Operating mode</b>	<b>statusword (6041h)</b>	<b>Explanation</b>
Point-to-point operation in „profile position mode“	Bit 12: <i>setpoint acknowledge</i>	0: Acceptance of new position possible 1: New target position received
	Bit 13: <i>following error</i>	1: contouring error
homing in „homing mode“	Bit 12: <i>Homing attained</i>	1: homing performed
	Bit 13: <i>Homing error</i>	0: homing error-free
speed operation in „profile velocity mode“	Bit 12: Speed=0	0: Motor moving 1: Motor at standstill
	Bit 13: <i>max_slippage error</i>	0: Actual speed within tolerance

*Manufacturer-specific operating modes*

In manufacturer-specific operating modes, status objects give information about the processing status of the current operating mode; bits 13..15 report the status of the operating mode.

Monitoring can be carried out jointly for all operating modes with the status object *driveStat* (2041h) or with the status objects of each operating mode. The *driveStat* object (2041h) always displays the values for the active operating mode.

<b>Operating mode</b>	<b>Status object</b>	<b>Bit 13, x_add_info</b>	<b>Bit 14, x_end</b>	<b>Bit 15, x_err</b>	<b>Twin Line param.</b>
Any activated mode	<i>driveStat</i> (2041h)	x_add_info	x_end	x_err	Status.driveStat (28:2)
point-to-point mode	<i>StatePTP</i> (2091h)	Target position reached	motion_end	motion_err	PTP.StatePTP (35:2)
Speed mode	<i>StateVEL</i> (2096h)	Set speed reached	vel_end	vel_err	VEL.StateVEL (36:2)
Electronic gear	<i>StateGear</i> (209Eh)	0	gear_end	gear_err	Gear.StateGear (38:2)
Manual movement	<i>StateMan</i> (20F5h)	0	manu_end	manu_err	Manual.StateMan (41:2)
homing	<i>StateHome</i> (20F0h)	0	ref_end	ref_err	Home.StateHome (40:2)

The „x“ stands for the operating mode code, e.g. „x“=„manu“ for manual movement.

As soon as the operation has been started, bit 14 „x\_end“ changes to „0“. When the operation is finished, bit 14 is re-activated, thereby indicating that further processing steps are enabled. The change of signal is suppressed if one processing operation is directly followed by another in a different operating mode.

## 5.4 Point-to-point operation as „profile position mode“ in accordance with DSP 402

### 5.4.1 Function

In point-to-point mode, the drive is moved from a starting position to a target position by means of an adjustable movement profile. The value for the target position can be given as a relative or absolute position.

The movement profile can include settings for acceleration and deceleration ramps as well as terminal velocity.

#### *Relative and absolute positioning*

The new target position is conveyed with the *target position* object (607Ah). With relative positioning, the target position is measured from the current starting position of the drive, with absolute positioning from the zero point. For an absolute positioning process, a reference point must be defined.

Relative or absolute positioning is set via bit 6 in the control word *controlword* (6040h).

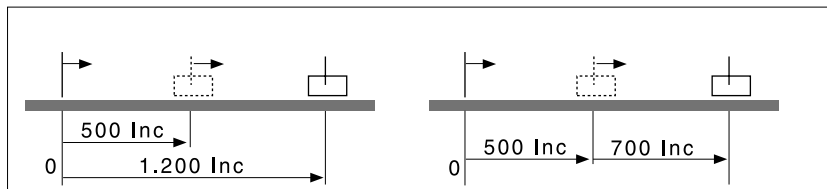


Fig 5.7 Absolute positioning (left) and relative positioning (right)

#### *Target position*

The target position is conveyed in user-defined units (usr) with the *target position* object (607Ah), and can be transmitted for standard settings together with the control word *controlword* (6040h) in R\_PDO2.

#### *Current position*

The current position can be determined via two objects:

- *position actual value* (6063h) supplies an absolute value in increments (Inc).
- *position actual value* (6064h) supplies an absolute value in user-defined units (usr).

The current position in user-defined units is transmitted for standard settings as a 4 byte value together with the *statusword* object (6041h) in T\_PDO2.

### 5.4.2 initiating positioning

#### *Requirements*

The point-to-point operating mode must be set via the *modes of operation* object (6060h), see the section on „Setting the operating mode“ on page 5-9.

The power amplifier must be switched on. The control word *controlword* (6040h) must be used to change to the „Operation enabled“ operating status, see „Operating states and transitions“ from page 5-5.

#### *Settings*

Object values for target position, movement profile, area of travel and normalization factors must contain valid values for the positioning operation. For object settings, see section on „Objects and settings“ from page 5-14.

*Control signals and monitoring signals*

If the operating mode, operating status and positioning values have been set, the operating mode can be activated and monitored via the control and status words.

Object / signal	Explanation
<i>controlword</i> (6040h)	
Bit 4: <i>New setpoint</i>	0-> 1: Start positioning operation or prepare follow-on positioning
Bit 5: <i>Change set immediatly</i>	Only applies to <i>New setpoint</i> = "1": 0: Activate new setpoints when target position reached 1: Activate new setpoints immediately
<i>statusword</i> (6041h)	
Bit 10: <i>target reached</i>	0: New target position stored 1: Stored target position reached
Bit 12: <i>setpoint acknowledge</i>	0: Acceptance of new position possible 1: New target position received
Bit 13: <i>following error</i>	0->1: contouring error has occurred

*trigger positioning*

The positioning operation is initiated with bit 4 „New setpoint“. The positioning operation can be triggered in two ways depending on bit 6, „Change set immediatly“.

- „Change set immediatly“ = 0:

On the rising edge of bit 4 „New setpoint“, the controller stores new positioning values but does not yet activate them. A new positioning operation is only initiated when the target position has been reached and bit 10 „Target reached“ changes to „1“.

- „Change set immediatly“ = 1:

On the rising edge of bit 4 „New setpoint“, new positioning values are immediately activated and a follow-on positioning operation triggered. The drive moves to the new target position without stopping on the way.

If a relative positioning operation is initiated during a running positioning process, the new positioning value is added to the setpoint for the running operation. The drive moves directly to the new setpoint.

*Positioning completed*

The target position has been reached as soon as bit 10 in the status word, *target reached* changes to „1“. The positioning controller automatically reports changes in status to the network client via the status word of event-driven T\_PDOs. On standard settings, this applies to T\_PDO1, T\_PDO2 and T\_PDO3.

*Contouring errors*

If the setpoint and actual position differ significantly, the positioning controller interrupts the operation with a contouring error message. The contouring error is displayed by means of the status word via bit 13, *following error*. The permissible deviation between setpoint and actual position can be set via the manufacturer-specific object, *following error window* (2010h). Details on monitoring operations are described in the chapter „Operating functions“.

5.4.3 Objects and settings

Positioning mode can be set and executed via the standardized objects of the CANopen „profile position mode“ in accordance with DSP 402. Additional functions can be achieved with manufacturer-specific objects.

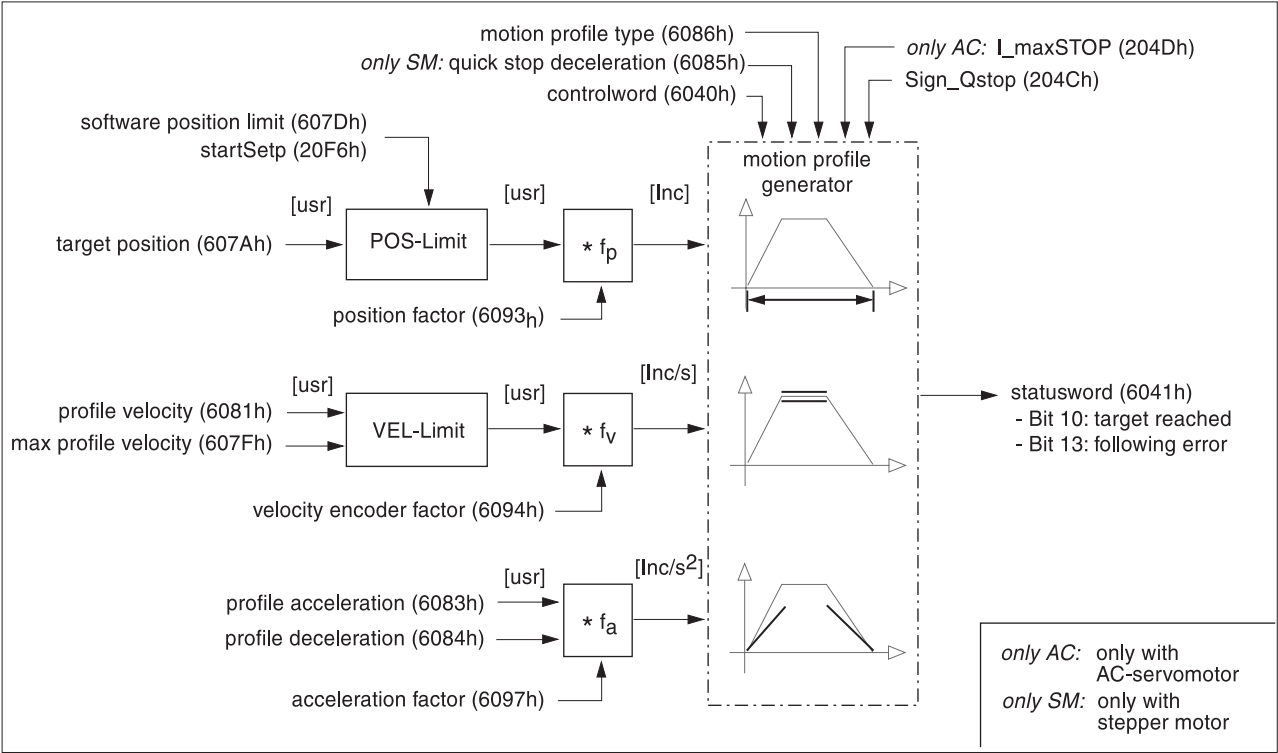


Fig 5.8 Objects for point-to-point operation as „profile position mode“

**Target position** A new position value is conveyed with the *target position* object (607Ah). The position value is given in user-defined units (usr). It is based on the current position of the drive for a relative positioning process and on the reference point for an absolute positioning operation.

The reference point is set in the homing operating mode.

**Movement profiles** In a positioning process, the drive acts in accordance with a movement profile, defined via acceleration and deceleration ramps as well as terminal velocity. The following objects are available for defining a movement profile:

Object	Explanation
max profile velocity (607Fh)	Maximum terminal velocity
profile velocity (6081h)	Terminal velocity
profile acceleration (6083h)	Acceleration ramp
profile deceleration (6084h)	Deceleration ramp
quick stop deceleration (6085h)	Quick Stop deceleration (specific to device)
motion profile type (6086h)	Ramp shape (specific to device)



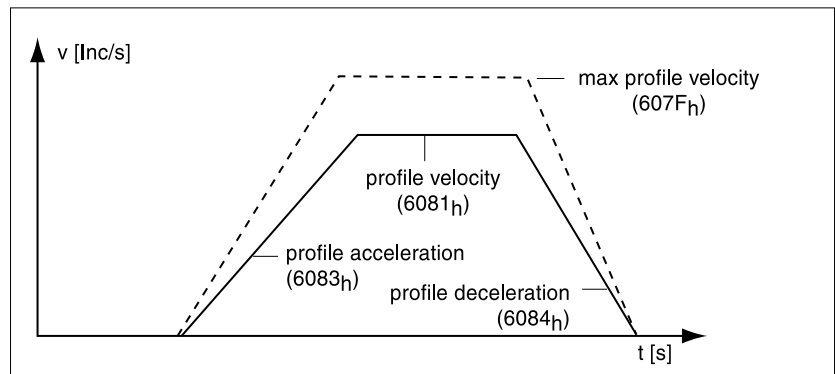


Fig 5.9 Ramp settings

The profile settings apply for both directions of motion of the drive. The maximum value for the profile velocity, *max profile velocity* (607Fh), limits the speed value for a movement profile.

The profile settings can be transmitted without triggering a movement or influencing a running operation. The objects *profile velocity* (6081h), *profile acceleration* (6083h) and *profile deceleration* (6084h) are only adopted when the operating mode is initiated by the control word.

#### Current limitation on AC actuators

In a positioning controller for AC actuators, the maximum values for acceleration and deceleration ramps are defined by the current limitation which can be set via sub-indexes 01h and 02h of the *position control parameter set* object (60FBh). For the Quick Stop operating status, a separate deceleration ramp can be set via the manufacturer-specific object, *I\_maxSTOP* (204Dh).

#### Ramp profile and Quick Stop ramp for stepping motors

With a positioning controller for driving stepping motors, the *motion profile type* object (6086h) can be used to choose between a linear and a sin-squared ramp shape optimized for the motor.

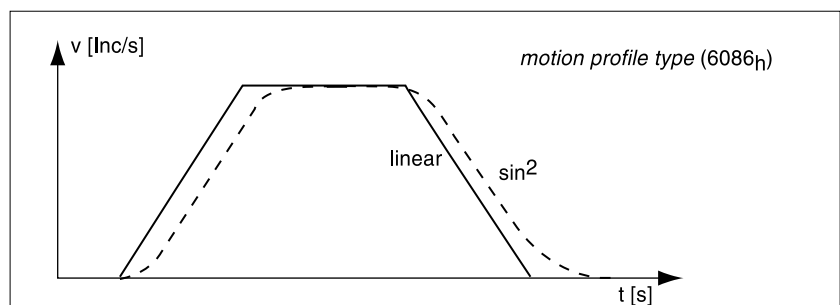


Fig 5.10 Linear and sin-squared ramp

In the Quick Stop operating status, a stepping motor is decelerated using the ramp settings of the *quick stop deceleration* object (6085h).

#### Area of travel

The area of travel can be limited by software limit switches whose positions are given in relation to the reference point. Before executing a movement command, the positioning controller checks the area limits: if the target position lies beyond the software limit switches, the positioning controller reports an error and does not execute the movement command.

Upper and lower limit switch positions, SW\_LimN and SW\_LimP are stored in the standardized object *software position limit* (607Dh), but can also be set via the manufacturer-specific objects *SW\_LimN* (2053h) and *SW\_LimP* (2052h).

If the reference point is moved as a result of the dimension setting homing method, the software limit switch values must be adapted accordingly.

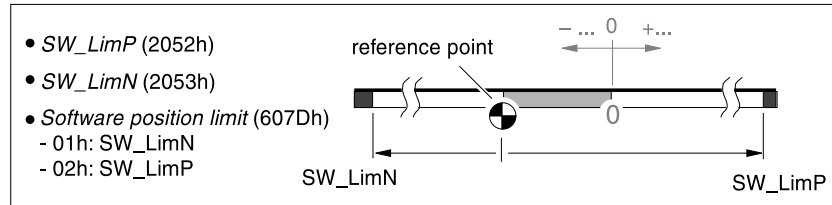


Fig 5.11 Area of travel and area limits

The graphic shows the reference point before a dimension setting process and the position „0“, which becomes the new reference point after dimension setting. The software limit switches must be adapted to fit the new reference point.

#### Normalization factors

Object values for position, speed and acceleration can be given in user-defined units such as m, mm/s or mm/s<sup>2</sup>. To generate the movement profile values, the positioning controller converts the user-defined units into internal increments (Inc) by means of normalization factors. There are separate normalization factors available for position, speed and acceleration values:

Object	Explanation
<i>position factor</i> (6093h)	normalization factor, position values
<i>velocity encoder factor</i> (6094h)	normalization factor, speed values
<i>acceleration factor</i> (6097h)	normalization factor, acceleration values

The setting options for normalization factors are explained in the chapter on „Operating functions“ on page 6-4.

#### CANopen example „Positioning with SDO, PDO and SYNC“

The example shows a positioning operation being set up by SDO, change of operating status by PDO1 and initiation via PDO2 with SYNC reception. The node address is 1.

COB ID	Data	Explanation
601	23 01 14 01 01 03 00 00	R_SDO: activate R_PDO2, sub-index 01h=0000 0301h
581	60 01 14 00 00 00 00 00	T_SDO: OK
601	23 01 18 01 81 02 00 00	R_SDO: activate T_PDO2, sub-index 01h=0000 0281h
581	60 01 18 00 00 00 00 00	T_SDO: OK
601	23 83 60 00 D0 07 00 00	T_SDO: set acceleration ramp: 2000 Inc/s/s
581	60 83 60 00 00 00 00 00	R_SDO: OK
601	23 81 60 00 F4 01 00 00	T_SDO: set speed: 500 Inc/s
581	60 81 60 00 00 00 00 00	R_SDO: OK
601	23 84 60 00 E8 03 00 00	T_SDO: set deceleration ramp: 1000 Inc/s/s
581	60 84 60 00 00 00 00 00	R_SDO: OK
0	01 00	NMT Protocol; Start Remote Node
181	40 00	T_PDO1: change of data in status word
201	06 00	R_PDO1: request status change: SwitchOnDisabled->ReadytoSwitchOn
181	21 00	T_PDO1: change of data in status word
201	0F 00	R_PDO1: request status change: ReadytoSwitchOn->OperationEnable
181	27 00	T_PDO1: change of data in status word
601	2F 60 60 00 01	T_SDO: switch to profile position mode
581	60 60 60 00 00	R_SDO: OK
301	1F 00 DC 05 00 00	R_PDO2: bytes 0,1: controlword, bytes 2..5: target pos. 5DCh = 1500 usr

COB ID	Data	Explanation
80		SYNC: SYNC object for initiating positioning
181	27 10	T_PDO1: setpoint acknowledge
201	0F 00	R_PDO1: bit 4: reset new setpoint
181	27 00	T_PDO1: setpoint acknowledge = 0
181	27 14	T_PDO1: bit 10, target reached = 1_

## 5.5 Speed operation as „profile velocity mode“ in accordance with DSP 402

### 5.5.1 Function

In speed mode, the drive accelerates up to an adjustable set speed. The movement profile can include settings for acceleration and deceleration ramps.

*set speed* The set speed is entered in user-defined units via the *target velocity* object (60FFh). New speed values are accepted immediately by the positioning controller while a movement task is running.

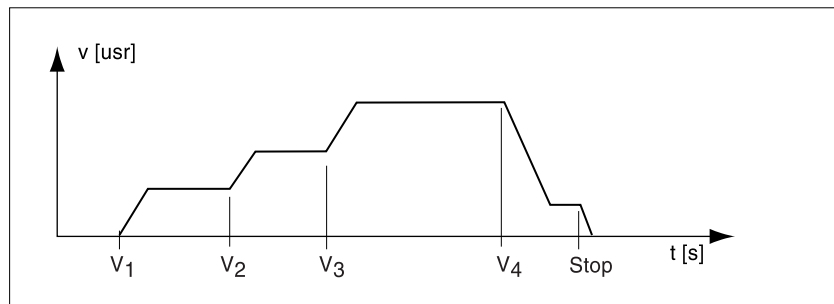


Fig 5.12 Movement tasks with speeds  $v_1$  to  $v_4$

For standard settings of R\_PDO3, the set speed *target velocity* (60FFh) is transmitted with the control word *controlword* (6040h) in R\_PDO3.

*current speed* The current speed can be determined via two objects:

- *velocity sensor actual value* (6069h) returns the speed in increments (Inc).
- *velocity actual value* (606Ch) returns the speed in user-defined units (usr), e.g. in „m/s“.

For standard settings of T\_PDO3, the speed, *velocity actual value* (606Ch), is transmitted as a 4 byte value together with the *statusword* object (6041h) in T\_PDO3.

### 5.5.2 Initiating speed mode

*Requirements* The speed operating mode must be set with the *modes of operation* object (6060h), see the section on „Setting the operating mode“ on page 5-9.

The power amplifier must be switched on. The control word *controlword* (6040h) must be used to change to the „Operation enabled“ operating status, see „Setting the operating mode“ from page 5-5.

*Settings* Object values for movement profile and normalization must contain valid values for the positioning process. For object settings, see section on „Objects and settings“ from page 5-19.

*Triggering speed mode* If operating mode, operating status and object values have been set, the operating mode can be initiated by communicating a set speed in the *target velocity* object (60FFh) or via PDO2.

*Speed mode monitoring* The movement operation can be monitored via the status word:

Object / signal	Explanation
<i>statusword (6041h)</i>	
Bit 10: <i>target reached</i>	0: new set speed stored 1: set speed reached
Bit 12: <i>speed=0</i>	0: Motor moving 1: Motor at standstill

### 5.5.3 Objects and settings

Speed mode can be set and executed via the standardized objects of the CANopen „profile velocity mode“ in accordance with DSP 402. Additional functions can be achieved with manufacturer-specific objects.

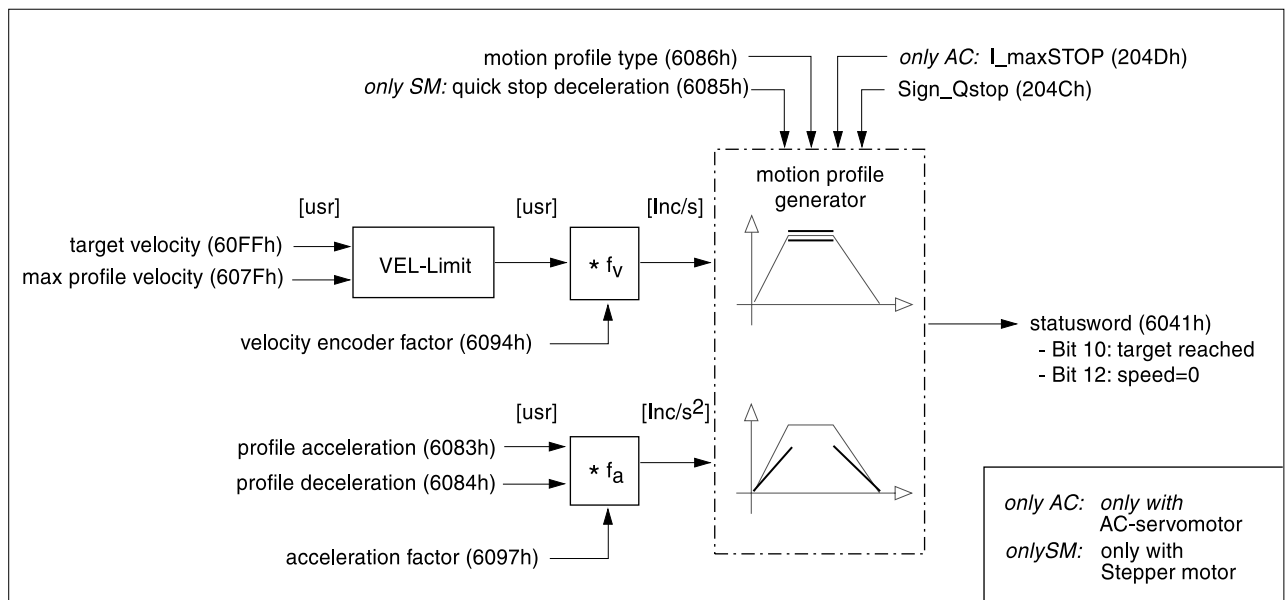


Fig 5.13 Speed operation: „profile velocity mode“

- Set speed* The set speed is conveyed with the *target velocity* object (60FFh) in user-defined units (usr).
- Movement profiles* Acceleration and deceleration phases are defined via the movement profile. The following objects are available for this purpose from the „profile position mode“:

Object	Explanation
<i>max profile velocity</i> (607Fh)	Maximum terminal velocity
<i>profile acceleration</i> (6083h)	Acceleration ramp
<i>profile deceleration</i> (6084h)	Deceleration ramp
<i>quick stop deceleration</i> (6085h)	Quick Stop deceleration (specific to device)
<i>motion profile type</i> (6086h)	Ramp shape (specific to device)

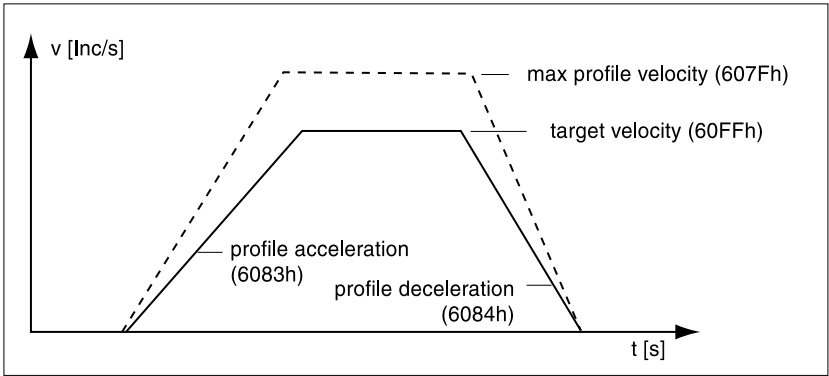


Fig 5.14 Ramp settings

The profile settings apply for both directions of motion of the drive. The maximum value for the profile velocity, *max profile velocity* (607Fh), limits the speed value for a movement profile.

Current limitation on AC actuators

In a positioning controller for AC actuators, the maximum values for acceleration and deceleration ramps are defined by the current limitation which can be set via sub-indexes 01h and 02h of the *position control parameter set* object (60FBh). A separate deceleration ramp can be set for the Quick Stop operating status via the manufacturer-specific object, *L\_maxSTOP* (204D).

Ramp profile and Quick Stop ramp for stepping motors

With a positioning controller for driving stepping motors, the *motion profile type* object (6086h) can be used to choose between a linear and a sin-squared ramp shape optimized for the motor.

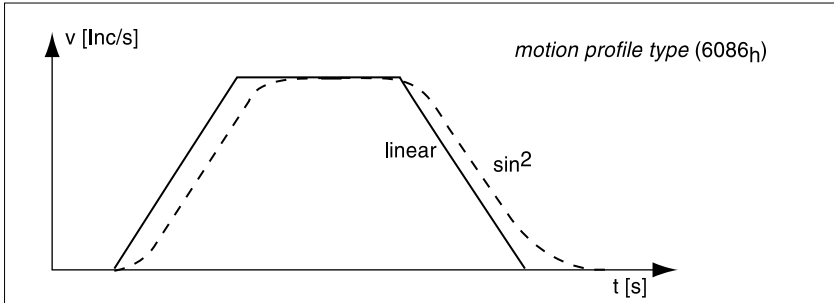


Fig 5.15 Linear and sin-squared ramp

In the Quick Stop operating status, a stepping motor is decelerated using the ramp settings of the *quick stop deceleration* object (6085h).

Normalization factors

Object values for position, speed and acceleration can be given in user-defined units such as m, mm/s or mm/s<sup>2</sup>. To generate the movement profile values, the positioning controller converts the user-defined units into internal increments (Inc) by means of normalization factors. There are separate normalization factors available for speed and acceleration values:

Object	Explanation
<i>velocity encoder factor</i> (6094h)	normalization factor, speed values
<i>acceleration factor</i> (6097h)	normalization factor, acceleration values

The setting options for normalization factors are explained in the chapter on „Operating functions“ on page 6-4.

*CANopen example*  
*„Speed mode with SDO, PDO and SYNC“*

The example shows speed mode being set by SDO, change of operating status by PDO1 and initiation via PDO3 with SYNC reception. The node address is 1.

COB ID	Data	Explanation
601	23 02 14 01 01 04 00 00	R_SDO: activate R_PDO3, sub-index 01h=0000 0401h
581	60 02 14 00 00 00 00 00	T_SDO: OK
601	23 02 18 01 81 03 00 00	R_SDO: activate T_PDO3, sub-index 01h=0000 0381h
581	60 02 18 00 00 00 00 00	T_SDO: OK
601	23 83 60 00 D0 07 00 00	T_SDO: set acceleration ramp: 2000 Inc/s/s
581	60 83 60 00 00 00 00 00	R_SDO: OK
601	23 84 60 00 E8 03 00 00	T_SDO: set deceleration ramp: 1000 Inc/s/s
581	60 84 60 00 00 00 00 00	R_SDO: OK
0	01 00	NMT Protokoll; Start Remote Node
181	40 00	T_PDO1: change of data in status word
201	06 00	R_PDO1: request status change: SwitchOnDisabled->ReadytoSwitchOn
181	21 00	T_PDO1: change of data in status word
201	0F 00	R_PDO1: request status change: ReadytoSwitchOn->OperationEnable
181	27 00	T_PDO1: change of data in status word
601	2F 60 60 00 03	T_SDO: switch to profile velocity mode
581	60 60 60 00 00	R_SDO: OK
401	1F 00 85 00 00 00	R_PDO3: bytes 0,1: controlword, bytes 2..5: target vel. 85h = 133 r.p.m.
80		SYNC: SYNC object for initiating positioning

## 5.6 Homing as „homing mode“ in accordance with DSP 402

### 5.6.1 Function

The homing operating mode is used to create an absolute dimensional reference between the position of the motor and a defined axis position. Homing can be effected by

- one of eight reference movements
- dimension setting.

The reference movement defines a reference point based on defined position, e.g. a limit switch or reference switch position. Dimension setting defines a reference point relative to the current position of the drive.

### 5.6.2 Initiating homing

**Requirements** The homing operating mode must be set with the *modes of operation* object (6060h), see the section on „Setting the operating mode“ on page 5-9.

The control word *controlword* (6040h) must be used to change to the „Operation enabled“ operating status, see „Setting the operating mode“ from page 5-5.

**Settings** Object values for the homing method and homing speed must contain valid values for the positioning operation. For object settings, see section on „Objects and settings“ from page 5-23.

**Control signals and monitoring signals** If the operating mode, operating status and object values have been set, the operating mode can be activated and monitored via the control and status words.

Object / signal	Explanation
<i>controlword</i> (6040h)	
Bit 4: <i>homing operation start</i>	0: no homing active 0-> 1: initiation of homing
<i>statusword</i> (6041h)	
Bit 13: homing error	0: no error in homing 1: error in homing
Bit 12: homing attained	0: Bit 13=0: homing not yet concluded Bit 13=1: homing contains error 1: Bit 13=0: homing carried out Bit 13=1: -

**trigger positioning** The positioning operation is initiated by setting bit 4, „homing operation start“, in the control word.



### 5.6.3 Objects and settings

*homing method* The homing method is selected via the *homing method* object (6098h). The following methods are supported:

Referencing to...	
Method 1	Limit switch $\overline{\text{LIMN}}$ with index pulse
Method 2	Limit switch $\overline{\text{LIMP}}$ with index pulse
Method 7	Reference switch $\overline{\text{REF}}$ with index pulse, first movement in clockwise direction
Method 11	Reference switch $\overline{\text{REF}}$ with index pulse, first movement in anti-clockwise direction
Method 17	Limit switch $\overline{\text{LIMN}}$ with no index pulse
Method 18	Limit switch $\overline{\text{LIMP}}$ with no index pulse
Method 23	Reference switch $\overline{\text{REF}}$ with no index pulse, first movement in clockwise direction
Method 27	Reference switch $\overline{\text{REF}}$ with no index pulse, first movement in anti-clockwise direction
Method 35	Dimension setting to current position

*Homing speed* During a homing movement, the drive moves at a reduced speed. It approaches the switching window of a limit switch or reference switch at search speed. Once it has found the switch, it moves to the edge of the switch's switching area at exit speed. Both speeds are stored in sub-indexes 01h and 02h of the *homing speeds* object (6099h) in user-defined units.

*Movement of reference point* Carrying out dimension setting with the *startSetp* object (20F6h) can cause the zero point for absolute positioning to move to the current position of the drive. The zero point thereby becomes the new reference point.

Values for software limit switches must then be corrected by the differential between the old reference point and the new setpoint.

## 5.7 Manufacturer-specific operating modes

### 5.7.1 Overview

The manufacturer-specific operating modes are described in the manuals for the Twin Line unit. For working in CANopen, the objects for each operating mode are compared below with the Twin Line parameters in the manuals. You will find information on the CANopen objects

- through the object directory from page 9-1
- through the parameter directory at the back of the manuals.

*Object directory* The object directory collects the objects into groups and sorts each group by ascending index. You will find the manufacturer-specific objects in the group called „Manufacturer-specific objects“ from sub-index 2000h.

*Parameter directory* In the manuals, the objects can be identified via their parameter names. The parameters are combined into groups in the parameter overviews, and sorted by ascending Twin Line index.

### 5.7.2 Objects for device monitoring and change of status

#### *Device monitoring*

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>OnlAuto</i> (2060h)	Commands.OnlAuto (29:30)	Access to operating mode setting
<i>IO_mode</i> (2061h)	Settings.IO_mode (29:31)	Significance of I/O signal assignment
<i>SetICtrl</i> (2043h)	Commands.SetCtrl (28:4)	switch controller parameter set
<i>ActCtrl</i> (2070h)	Status.ActCtrl (31:4)	Active controller parameter set

*Status change* Operating status and status change can be executed in the manufacturer-specific operating modes via objects *driveCtrl* (2040h) and *driveStat* (2041h).

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>driveCtrl</i> (2040h)	Commands.driveCtrl (28:1)	Control word for change of status
<i>driveStat</i> (2041h)	Status.driveStat (28:2)	Status word for the operating status

### 5.7.3 Objects for displaying and initiating operating modes

In order to be able to work with manufacturer-specific objects, the *modes of operation* object (6060h) must be switched to the value „-1“ (SigMode).

Object	Twin Line parameter (index:sub-index).	Explanation
<i>xMode</i> (2051h)	Status.xMode (28:3)	Axis operating mode to be set after power on
<i>xMode_act</i> (2042h)	Status.xMode_act (28:3)	Current axis operating mode with additional information

*Initiation of an operating mode* A manufacturer-specific mode is initiated by means of the starting object for each mode, e.g. for manual mode via the *startMan* object (20A9h).

### 5.7.4 Objects for monitoring status during movement

During movement, the positioning controller can be monitored with the following objects. Object values are read only.

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>p_ref</i> (2071h)	Status.p_ref (31:5)	Set position of rotor [Inc]
<i>p_act</i> (2072h)	Status.p_act (31:6)	Motor position in int. units [Inc]
<i>p_dif</i> (2073h)	Status.p_dif (31:7)	Contouring error [Inc]
<i>n_ref</i> (2074h)	Status.n_ref (31:8)	Set speed [r.p.m.]
<i>n_act</i> (2075h)	Status.n_act (31:9)	Actual speed [r.p.m.]
<i>I_ref</i> (2076h)	Status.I_ref (31:10)	Set current, Q component (torque-forming) (100=1A)
<i>I_act</i> (2077h)	Status.I_act (31:12)	Current motor current, Q component (torque-forming) (100=1A)
<i>p_abs</i> (2078h)	Status.p_abs (31:16)	Absolute position per motor revolution (modulo value) [Inc]
<i>I2tM_act</i> (2079h)	Status.I2tM_act (31:17)	I <sup>2</sup> t motor sum [%]
<i>I2tPA_act</i> (207Ah)	Status.I2tPA_act (31:18)	I <sup>2</sup> t amplifier sum [%]
<i>I2tB_act</i> (207Bh)	Status.I2tB_act (31:19)	I <sup>2</sup> t ballast sum [%]
<i>UDC_act</i> (207Ch)	Status.UDC_act (31:20)	DC line voltage (10=1V)
<i>TM_act</i> (207Fh)	Status.TM_act (31:24)	Motor temperature [°C]
<i>TPA_act</i> (2080h)	Status.TPA_act (31:25)	Amplifier temperature [°C]
<i>p_refGear</i> (2081h)	Status.p_refGear (31:26)	Electronic gear set position [Inc]
<i>v_refGear</i> (2082h)	Status.v_refGear (31:27)	Electronic gear set speed [Inc/s]
<i>v_ref</i> (2083h)	Status.v_ref (31:28)	Rotor position set speed p_ref [Inc/s]
<i>acc_ref</i> (2084h)	Status.acc_ref (31:29)	Acceleration of position controller setpoint p_ref [r.p.m.*s]

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>p_target</i> (2085h)	Status.p_target (31:30)	Target position of movement profile generator [usr]
<i>p_jerkusr</i> (2086h)	Status.p_jerkusr (31:31)	Actual position of movement profile generator [usr]
<i>p_tarOffs</i> (2087h)	Status.p_tarOffs (31:32)	Target position of offset positioning in electronic gear [Inc]
<i>p_refOffs</i> (2088h)	Status.p_refOffs (31:33)	Actual position of offset positioning in electronic gear [Inc]
<i>p_actusr</i> (2089h)	Status.p_actusr (31:34)	Actual position of motor in user-defined units [usr]
<i>v_jerkusr</i> (208Ah)	Status.v_jerkusr (31:35)	Actual speed of movement profile generator [usr]
<i>n_refOffs</i> (208Bh)	Status.n_refOffs (31:36)	Actual speed of offset positioning in electronic gear [r.p.m.]
<i>p_remaind</i> (208Ch)	Status.p_remaind (31:37)	Residual value of positional normalization of set position p_ref [Inc]
<i>v_target</i> (208Dh)	Status.v_target (31:38)	Target speed of movement profile generator

## 5.7.5 Manual movement

Manual movement is executed either as „classical manual movement“ or as „united inching“. In both processing modes, the motor is moved a prescribed distance via starting signals. In the „classical manual movement“ the motor changes to continuous movement if the starting signal is applied for a longer time.

### Start and monitoring

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>startMan</i> (20A9h)	Manual.startMan (41:1)	Start of manual movement with transfer of control bit
<i>stateMan</i> (20F5h)	Manual.StateMan (41:2)	Acknowledgment: manual movement

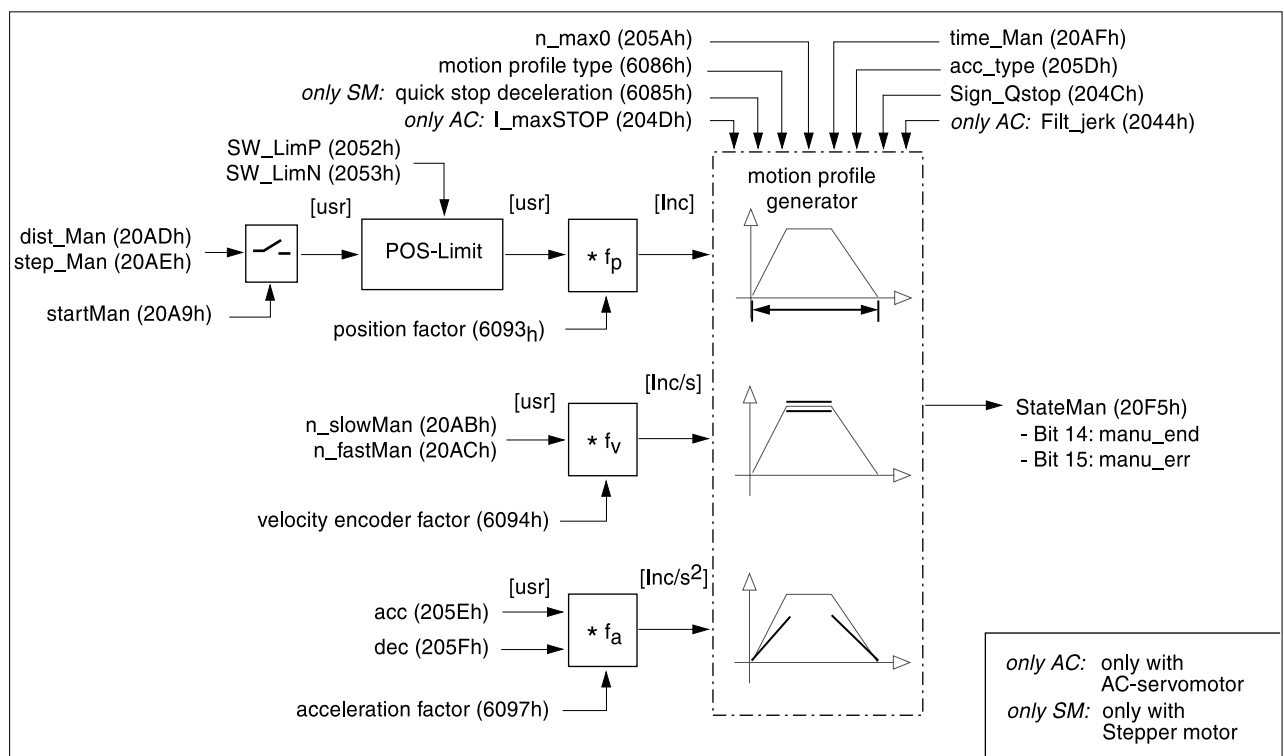


Fig 5.16 Manual mode

### Settings

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>typeMan</i> (20AAh)	Manual.typeMan (41:3)	Type of manual movement
<i>n_slowMan</i> (20ABh)	Manual.n_slowMan (41:4)	Speed for slow manual movement
<i>n_fastMan</i> (20ACh)	Manual.n_fastMan (41:5)	Speed for fast manual movement
<i>I_maxMan</i> (2050h)	Manual.I_maxMan (28:25)	max. current in manual movement

Classical manual movement

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>step_Man</i> (20AEh)	Manual.step_Man (41:7)	Inching distance, defined path at the start of manual movement
<i>time_Man</i> (20AFh)	Manual.time_Man (41:8)	Delay, classical

United inching

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>dist_Man</i> (20ADh)	Manual.dist_Man (41:6)	Inching distance, defined path per inch cycle in united inching

You will find settings for ramp values, jerk filter, Quick Stop, normalization factors and monitoring functions described in the chapter on „Operation functions“ from page 6-1.

5.7.6 Speed mode

In speed mode the motor is given a set speed and movement is initiated with no target position. The motor moves at this speed until another set speed is transmitted or the operating mode is terminated.

Start and monitoring

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>velocity</i> (2095h)	VEL.velocity (36:1)	Start of a speed change with transfer of set speed [usr]
<i>stateVEL</i> (2096h)	VEL.StateVEL (36:2)	Acknowledgment: Speed profile mode

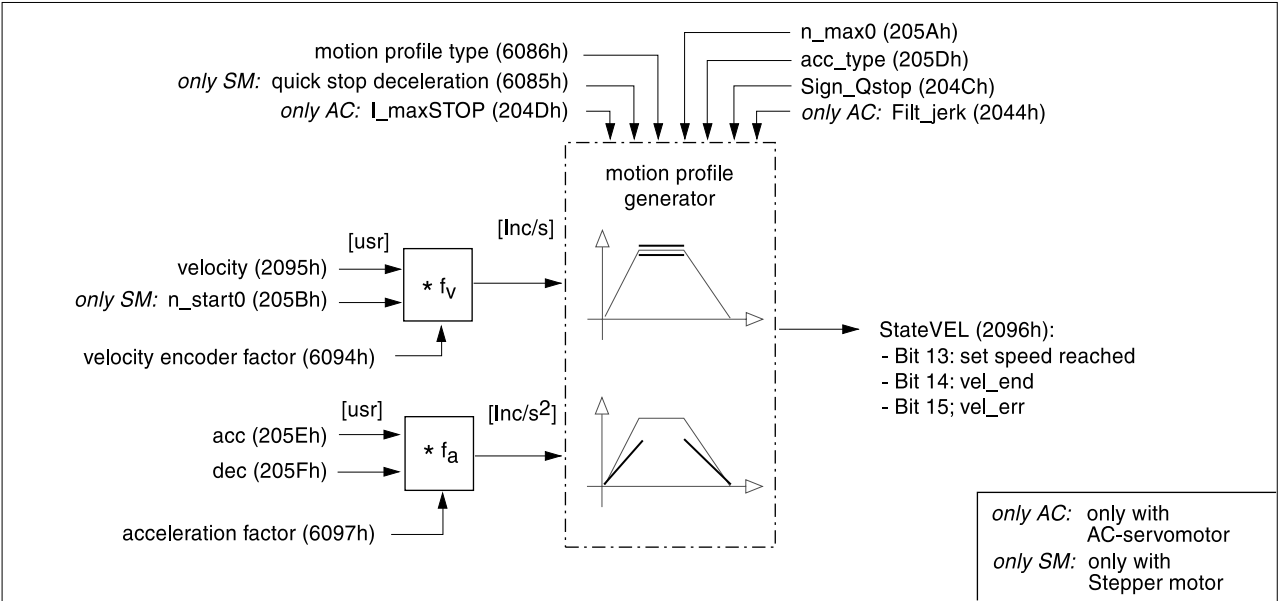


Fig 5.17 Speed mode

You will find settings for ramp values, jerk filter, Quick Stop, normalization factors and monitoring functions described in the chapter on „Operating functions“ from page 6-1.

### 5.7.7 Point-to-point mode

In point-to-point mode (also PtP mode, PtP: Point to Point), the motor is positioned from point A to point B by means of a positioning command. The positioning path is given in the form of absolute values based on the reference point or relative values based on the current position of the axis.

#### Start and monitoring

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>p_absPTP</i> (2090h)	PTP.p_absPTP (35:1)	Initiation of absolute positioning with transfer of absolute target position value [usr]
<i>statePTP</i> (2091h)	PTP.StatePTP (35:2)	Acknowledgment: PTP positioning
<i>p_relPTP</i> (2092h)	PTP.p_relPTP (35:3)	Initiation of relative positioning with transfer of value for the distance [usr]
<i>continue</i> (2093h)	PTP.continue (35:4)	Continuation of an interrupted positioning operation with transfer of any value
<i>v_tarPTP</i> (2094h)	PTP.v_target (35:5)	Set speed of PTP positioning [usr]

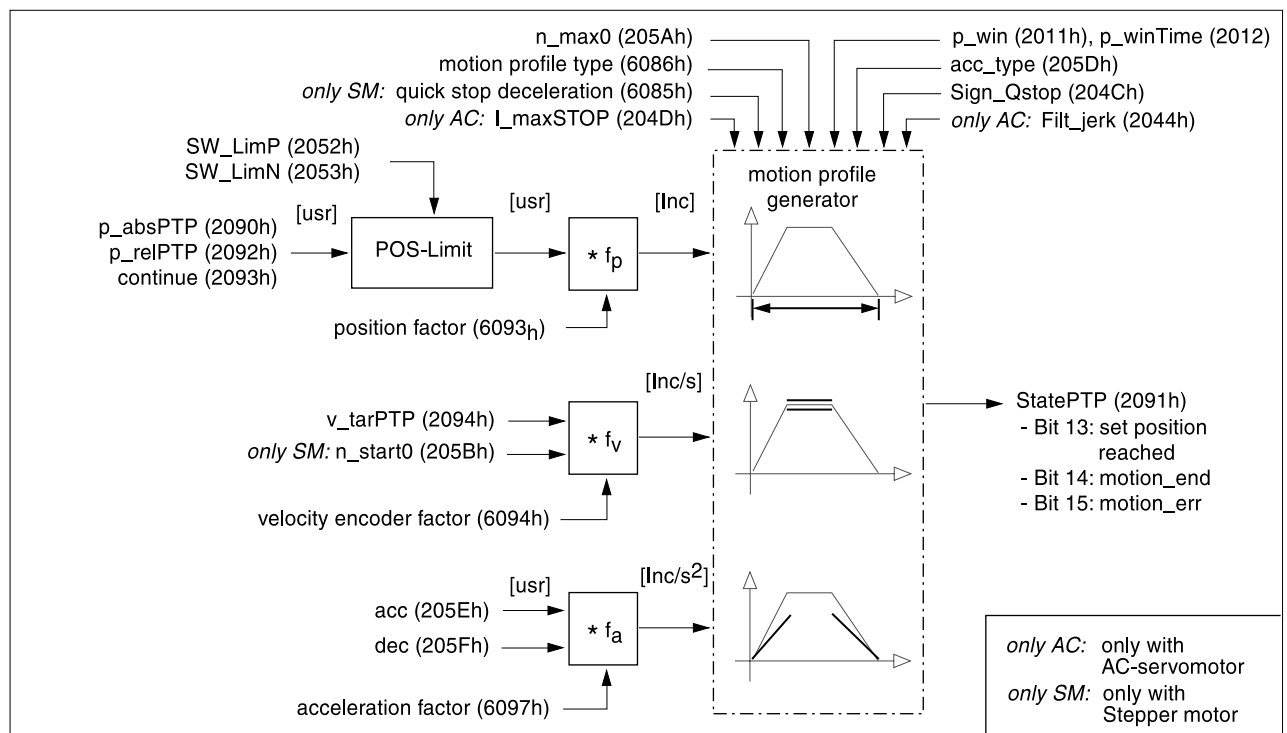


Fig 5.18 point-to-point mode

You will find settings for ramp values, jerk filter, Quick Stop, normalization factors and monitoring functions described in the chapter on „Operating functions“ from page 6-1.



### 5.7.8 Electronic gear

In the electronic gear operating mode, the positioning controller calculates a new position setpoint for the movement of the motor from a predefined position and an adjustable gear factor. The operating mode is used when one or more motors are to follow the position control reference signal of an NC system or an encoder.

#### Start and monitoring

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>startGear</i> (209Dh)	Gear.startGear (38:1)	Initiation of electronic gear processing with selection of processing mode
<i>stateGear</i> (209Eh)	PTP.StatePTP (35:2)	Acknowledgment: PTP positioning

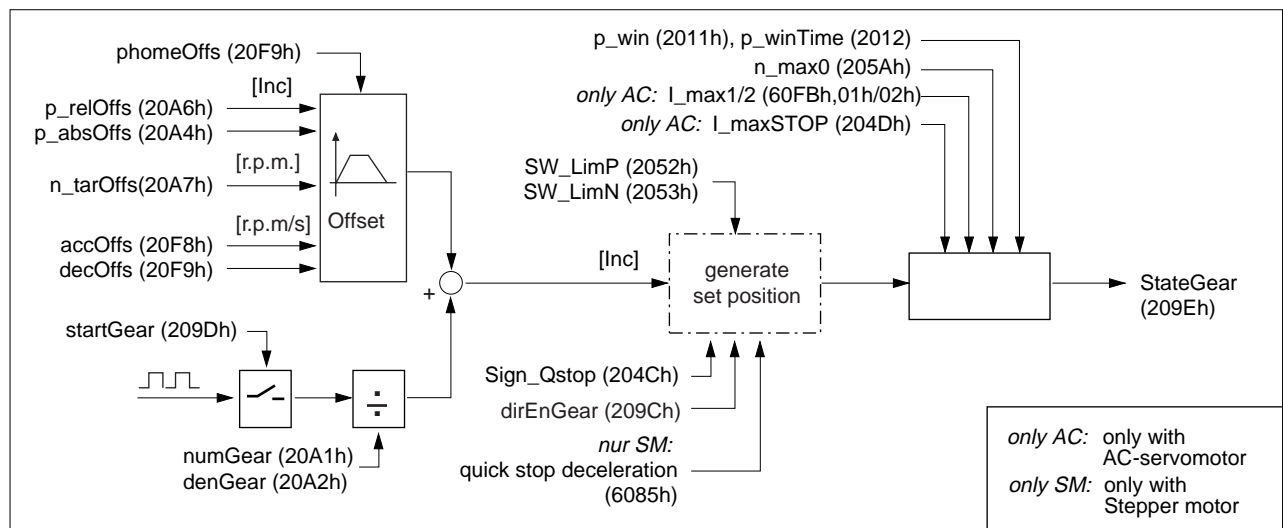


Fig 5.19 Electronic gear

#### Settings

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>n_maxGear</i> (209Fh)	Gear.n_maxGear (38:5)	Maximum speed [r.p.m.]
<i>numGear</i> (20A1h)	Gear.numGear (38:7)	Gear factor counter
<i>denGear</i> (20A2h)	Gear.denGear (38:8)	Gear factor denominator
<i>dirEnGear</i> (209Ch)	Gear.dirEnGear (38:13)	Direction of movement enable

#### Offset positioning

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>p_absOffs</i> (20A4h)	Gear.p_absOffs (39:1)	Initiation of absolute offset positioning with transfer of position value
<i>stateOffs</i> (20A5h)	Gear.stateOffs (39:2)	Acknowledgment: Offset positioning

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>p_relOfs</i> (20A6h)	Gear.p_relOfs (39:3)	Initiation of relative offset positioning with transfer of value for the distance [Inc]
<i>n_tarOfs</i> (20A7h)	Gear.n_tarOfs (39:5)	Offset positioning set speed [Inc/s]
<i>phomeOfs</i> (20A8h)	Gear.phomeOfs (39:6)	Dimension setting in offset positioning [Inc]
<i>accOfs</i> (20F8h)	Gear.accOfs (39:7)	Offset positioning acceleration ramp [rev/(min*s)]
<i>decOfs</i> (20F9h)	Gear.decOfs (39:8)	Offset positioning deceleration ramp [rev/(min*s)]

You will find settings for ramp values, jerk filter, Quick Stop, and monitoring functions described in the chapter on „Operating functions“ from page 6-1

### 5.7.9 Homing

The homing operating mode is used to create an absolute dimensional reference between the position of the motor and a defined axis position. Homing can be effected by means of a reference movement or dimension setting.

#### Start and monitoring

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>startHome</i> (20F0h)	Home.startHome (40:1)	Initiation of homing operating mode
<i>stateHome</i> (20F1h)	Home.StateHome(40:2)	Acknowledgment: homing
<i>startSetp</i> (20F6h)	Home.startSetp (40:3)	Dimension setting on set position (absolute position) [usr]

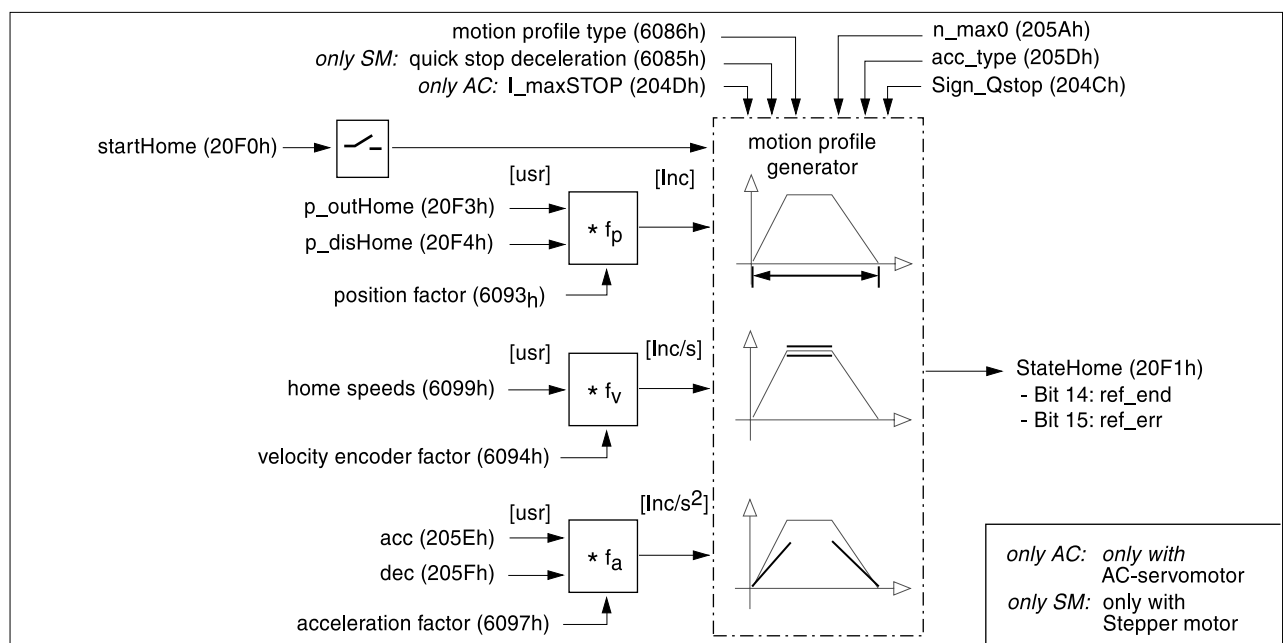


Fig 5.20 homing

#### Settings

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>homing speeds</i> (6099h, 01h)	Home.v_Home (40:4)	Speed for searching reference switch [usr]
<i>homing speeds</i> (6099h, 02h)	Home.v_outHome (40:5)	Speed for free movement to position 'p_homeOut' [usr]
<i>p_outHome</i> (20F3h)	Home.p_outHome (40:6)	Exit path, automatically selected after reference has been found [usr]
<i>p_disHome</i> (20F4h)	Home.p_disHome (40:7)	Safety gap from the switching edge to the reference point [usr]

You will find settings for ramp values, jerk filter, Quick Stop, normalization factors and monitoring functions described in the chapter on „Operating functions“ from page 6-1.



## 6 Operating functions

### 6.1 List control and list processing

**Overview** List controlled operation runs in the background during the execution of a movement command. If the motor overruns an axis position which is stored in the list, an interface signal is changed or a new speed value activated.

The positioning controller stores two separate lists with 64 fields each for position entries. Before values from a list are entered, a list type must be assigned:

- Position / speed list  
In this list a speed value is filed for every position entry.
- Position / signal lists  
A signal level is stored for every position entry and the interface output TRIGGER set to it.

You will find details on list control and list processing in the positioning controller manual in the chapter on functions.

#### *Start and monitoring*

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>startList</i> (20B4h)	List.startList (44:1)	Activate new list processing operation, running list processing is deactivated beforehand.
<i>stateList</i> (20B5h)	List.stateList (44:2)	Acknowledgement and status: list processing

#### *Settings*

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>cntList1</i> (20B7h)	List.cntList1 (44:4)	List 1: number of available list entries
<i>bgnList1</i> (20B8h)	List.bgnList1 (44:6)	List 1: starting number, first entry for list processing starting number < finishing number
<i>endList1</i> (20B9h)	List.endList1 (44:7)	List 1: finishing number, last entry for list processing finishing number > starting number
<i>actList1</i> (20BAh)	List.actList1 (44:8)	List 1: active processing number starting number =< active proc. No. =< fin. No.
<i>cntList2</i> (20BDh)	List.cntList2 (44:12)	List 2: number of available list entries
<i>bgnList2</i> (20BEh)	List.bgnList2 (44:14)	List 2: starting number, first entry for list processing starting number < finishing number
<i>endList2</i> (20BFh)	List.endList2 (44:15)	List 2: finishing number, last entry for list processing finishing number > starting number
<i>actList2</i> (20C0h)	List.actList2 (44:16)	List 2: active processing number starting number =< active proc. No. =< fin. No.

*Processing lists*

List data for both lists are administered under CANopen via the eight manufacturer-specific objects : *List1\_...* (2200h..2203h) and *List2\_...* (2300h..2303h) with 64 sub-index values each. The table shows the allocation for sub-index 01h of the objects. Sub-indexes 02h to 40h have the same significance.

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>List1_Type</i> (2200h,01h)	L1Data0.typeList1 (1100:1)	List 1: list type for ALL following list entries (1101:x..1163:x)
<i>List1_Position</i> (2201h,01h)	L1Data0.posList1 (1100:2)	List 1: position
<i>List1_Signal</i> (2202h,01h)	L1Data0.signList1 (1100:3)	List 1: signal status
<i>List1_Velocity</i> (2203h,01h)	L1Data0.velList1 (1100:4)	List 1: set speed
<i>List2_Type</i> (2300h,01h)	L2Data0.typeList2 (1200:1)	List 2: list type for ALL following list entries (1201:x..1263:x)
<i>List2_Position</i> (2301h, 01h)	L2Data0.posList2 (1200:2)	List 2: position
<i>List2_Signal</i> (2302h, 01h)	L2Data0.signList2 (1200:3)	List 2: signal status
<i>List2_Velocity</i> (2303h, 01h)	L2Data0.velList2 (1200:4)	List 2: set speed

*Comparison with Twin Line parameters*

Unlike under CANopen, the list data with Twin Line parameters are not administered via 64 sub-index entries but 64 parameter groups with 4 sub-index entries each. For list 1, the data records are stored in parameter groups L1Data0...L1Data63 and for list 2 under L2Data0...L2Data63.

The following table shows the memory allocation for the first two objects, *List1\_Type* (2200h) and *List1\_Position* (2201h) compared with the Twin Line parameters.

Object (index, sub-index)	Twin Line parameter (index: sub-index).
<i>List1_Type</i> (2200h)	-
typeList1 (2200h,01h)	L1Data1.typeList1 (1100:1)
typeList1 (2200h,02h)	L1Data1.typeList2 (1101:1)
...	...
typeList1 (2200h,64h)	L1Data1.typeList64 (1163:1)
<i>List1_Position</i> (2201h)	
posList1 (2201h, 01h)	L1Data0.posList1 (1100:2)
posList1 (2201h, 02h)	L1Data0.posList2 (1101:2)
...	...
posList1 (2201h, 64h)	L1Data0.posList64 (1163:2)

## 6.2 Teach In processing

**Overview** Teach-In processing offers the facility for capturing current position values by moving the motor, and transferring them to a previously determined memory area.

The data are stored in a position/signal list or a position/speed list.

The positioning controller accepts position values as absolute values in user-defined units.

You will find details on teach-in processing in the positioning controller manual in the chapter on functions.

### Start and monitoring

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>storeTeac</i> (20B0h)	Teach.storeTeac (43:1)	Teach-In processing, select memory location list number for storing a position value (0...63)
<i>stateTeac</i> (20B1h)	Teach.stateTeac (43:2)	Acknowledgment: teach-in processing

### Settings

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>memNrTeac</i> (20B2h)	Teach.memNrTeac (43:3)	List for teach-In processing
<i>p_actTeac</i> (20B3h)	Teach.p_actTeac (43:4)	current motor position for teach-in processing [usr]
<i>List1_Type</i> (2200h,01h)	L1Data0.typeList1 (1100:1)	List 1: list type for ALL following list entries (1101:x..1163:x)
<i>List2_Type</i> (2300h,01h)	L2Data0.typeList2 (1200:1)	List 1: list type for ALL following list entries (1101:x..1163:x)

### 6.3 Normalization

**Overview** Normalization converts user-defined units into internal units of the positioning controller and vice versa. The positioning controller stores position, speed and acceleration values in user-defined units. The positioning controller applies a separate normalization factor to each of the values.

As a result, neither position nor speed values need to be recalculated or re-entered after a change to a motor with a different resolution.

You will find details on normalization in the positioning controller manual in the chapter on functions.

#### Normalization factors

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>position factor (6093h)</i>		
<i>pNormNum</i> (6093h, 01h)	Motion.pNormNum (29:7)	position normalization numerator
<i>pNormDen</i> (6093h, 02h)	Motion.pNormDen (29:8)	position normalization denominator
<i>velocity factor (6094h)</i>		
<i>vNormNum</i> (6094h,01h)	Motion.vNormNum (29:9)	speed normalization numerator
<i>vNormDen</i> (6094h,02h)	Motion.vNormDen (29:10)	speed normalization denominator
<i>acceleration factor (6097h)</i>		
<i>aNormNum</i> (6097h,01h)	Motion.aNormNum (29:11)	acceleration normalization numerator
<i>aNormDen</i> (6097h,02h)	Motion.aNormDen (29:12)	acceleration normalization denominator

The normalization factors apply for manufacturer-specific and standardized operating modes in the DSP 402 profile.

#### Residual value for user normalization

Apart from electronic gear mode, movement data are given in user-defined units in all operating modes. Internally the positioning controller works with the resolution of the motor, for AC actuators with a Sincoder, e.g. with 16384 inc., and moves to the nearest internal position corresponding to the user-defined position.

An interruption to the movement or a change from operating with internal resolution to operating with user-defined resolution can lead to deviations between the actual position of the motor and the nearest possible user-defined position. The differential can be viewed via the *p\_remaind* object (208Ch).

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>p_remaind</i> (208Ch)	Status.p_remaind (31:37)	Residual value of positional normalization of set position <i>p_ref</i> [Inc]



## 6.4 Ramp function

The positioning controller controls the acceleration and deceleration behaviour of the motor by means of ramp functions. The gradient and shape of the ramp describe the ramp function. The ramp gradient shows the speed change of the motor, and the ramp shape the acceleration over time.

You will find details on the ramp function in the positioning controller manual in the chapter on functions.

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>acc_type</i> (205Dh)	Motion.acc_type (29:25)	Shape of acceleration curve
<i>acc</i> (205Eh)	Motion.acc (29:26)	acceleration [usr]
<i>dec</i> (205Fh)	Motion.dec (29:27)	deceleration [usr]
<i>v_target0</i> (205Ch)	Motion.v_target0 (29:23)	set speed [usr]

*with stepping motors*

Additional parameters for setting the movement profile are available for positioning controllers for stepping motors.

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>n_max0</i> (205Ah)	Motion.n_max0 (29:21)	speed limitation for movement profile [r.p.m.]
<i>n_start0</i> (205Bh)	Motion.n_start0 (29:22)	starting speed [usr]

*with AC actuators*

A positioning controller for AC actuators can carry out a soft, jerk-free change of speed by adding the jerk filter.

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>Filt_jerk</i> (2044h)	Motion.Filt_jerk (28:5)	jerk filter

## 6.5 Quick Stop function

Quick Stop is an emergency brake function which stops the motor, e.g. because of a fault.

You will find details on the Quick Stop function in the positioning controller manual in the chapter on functions.

*with stepping motors*

positioning controllers for stepping motors control the Quick Stop deceleration via the Quick Stop ramp, the *quick stop deceleration* object (6085h) or via the deceleration ramp of the movement profile, the *dec* object (205Fh).

*with AC actuators*

AC actuators can be decelerated via the Quick Stop current, *I\_maxSTOP* (204Dh), or via the movement profile deceleration ramp, *dec* (205Fh).

The deceleration function for Quick Stop is set by means of the *SignQ-stop* object (204Ch).

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>SignQstop</i> (204Ch)	Settings.SignQstop (28:20)	monitoring signals which trigger Quick Stop
<i>I_maxSTOP</i> (204Dh)	Settings.I_maxSTOP (28:22)	current limitation for Quick Stop [A]
<i>quick_stop_deceleration</i> (6085h)	Settings.dec_Stop	Quick Stop deceleration

## 6.6 Standstill window

*with AC actuators*

If the motor is held at zero speed when under active control, minimal fluctuations in speed prevent standstill from being detected. If the motor remains in the standstill window for an adjustable length of time, the controller signals standstill. Bit 10 in the status word, *statusword* (6041h), is set.

You will find details on the standstill window function in the positioning controller manual in the chapter on functions.

*Settings*

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>p_win</i> (2011h)	Settings.p_win (12:13)	standstill window, permissible control deviation [Inc]
<i>p_winTime</i> (2012h)	Settings.p_winTime (12:14)	time for which any control deviations must be within the standstill window for standstill to be reported [ms]

## 6.7 Inversion of rotation

The direction of rotation of the drive can be inverted by means of the *invertDir* object (2045h). The limit switch connections must be swapped at the same time.

You will find details on the inversion of the direction of rotation in the positioning controller manual in the chapter on functions.

*Settings*

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>invertDir</i> (2045h)	Motion.invertDir (28:6)	Reversal of sense of rotation

**6.8 Fast position capture**

Position values can be recorded via two adjustable channels with a timing accuracy of 5  $\mu$ s.

The „Capture.TrigSign“ parameter defines the signal source of a position value capture: the inputs CAPTURE1 and CAPTURE2 of the signal interface or the index pulse of a position sensor.

You will find details on fast position capture in the positioning controller manual in the chapter on functions.

*Start and monitoring*

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>TrigStart</i> (2019h)	Capture.TrigStart (20:16)	initiate triggering (bits 0..1): 0: no change 1: reset triggering and re-start abort triggering (bit14=1) repeat triggering (bit15) 0: one-off triggering 1: continuous triggering
<i>TrigStat</i> (202Ah)	Capture.TrigStat (20:17)	status triggering carried out

*Settings*

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>TrigSign</i> (2016h)	Capture.TrigSign (20:13)	selection of trigger signals for position storage bits 3..2: signal - channel 2 (K2) bits 1..0: signal - channel 1 (K1)
<i>TrigType</i> (2017h)	Capture.TrigType (20:14)	position source for position storage
<i>TrigLevl</i> (2018h)	Capture.TrigLevl (20:15)	signal level for trigger channels bit status: 0: triggering at 1->0 change 1: triggering at 0->1 change
<i>TrigPact1</i> (202Bh)	Capture.TrigPact1 (20:18)	actual position of motor for triggering on channel 1 [Inc]
<i>TrigPact2</i> (202Ch)	Capture.TrigPact2 (20:19)	actual position of motor for triggering on channel 2 [Inc]
<i>TrigPref1</i> (202Dh)	Capture.TrigPref1 (20:20)	setpoint electronic gear for triggering on channel 1 [Inc]
<i>TrigPref2</i> (202Eh)	Capture.TrigPref2 (20:21)	setpoint electronic gear for triggering on channel 2 [Inc]

## 6.9 Monitoring functions

### 6.9.1 Monitoring of axis signals

**Positioning limits** Within the positioning range of the axis, the motor can be moved to any axis point by specifying an absolute positioning process.

The area of travel of the axis is given in internal units within the range of  $-2^{31}$  to  $+2^{31}$  increments. For internal units, the resolution of the motor sensor is given in increments.

In a position overrun, bit 2 of the *IntSigSR* object (2062h) is set.

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>IntSigSr</i> (2062h)	Status.IntSigSR (29:34)	monitoring signals bit 2: position overrun

**Software limit switches** The software limit switch position is set by means of the *SW\_LimP* (2052h) and *SW\_LimN* objects (2053h) and activated through *SW\_Enabl* (2054h). The limit switch values are also mapped in the *software position limit* object (607Dh) of the device profile DSP 402.

The position controller setpoint determines position monitoring of the software limit switch range. Depending on the controller settings, the motor can therefore stop before it reaches the limit switch position. Bits 5 and 6 of the *IntSigSR* object (2062h) signal if the limit switch position is passed.

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>SW_LimP</i> (2052h) <i>software position limit</i> (607Dh, 02h)	Motion.SW_LimP (29:4)	software limit switches for pos. position limit LIMP condition: $SW\_LimP > SW\_LimN$ [usr]
<i>SW_LimN</i> (2053h) <i>software position limit</i> (607Dh, 01h)	Motion.SW_LimN (29:5)	software limit switches for pos. position limit LIMN condition: $SW\_LimN < SW\_LimP$ [usr]
<i>SW_Enabl</i> (2054h)	Motion.SW_Enabl (29:6)	set monitoring of software limit switches
<i>IntSigSr</i> (2062h)	Status.IntSigSR (29:34)	monitoring signals Bit 5/6: $SW\_LimP/SW\_LimN$

**Limit switch and STOP signal** During movements both limit switches are monitored via the input signals LIMN and LIMP. If the drive meets a limit switch, the positioning controller stops the motor. Limit switch overrun is signalled to the input device.

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>SignEnabl</i> (2046h)	Settings.SignEnabl (28:13)	signal enable for monitoring inputs 0: disabled 1: enabled
<i>SignLevel</i> (2047h)	Settings.SignLevel (28:14)	signal level for monitoring inputs 0: response if level = 0 1: response if level = 1

## 6.9.2 Monitoring internal signals

*Monitoring contouring errors with AC actuators*

Contouring error monitoring checks for positional deviation between the actual position of the motor and its setpoint. If the difference exceeds a contouring error threshold, the positioning controller signals an error. The threshold for the contouring error is adjustable.

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>p_maxDiff</i> (2010h)	Settings.p_maxDiff (12:11)	maximum permissible contouring error of position controller [Inc]

*Monitoring parameters*

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

- *FltSig* (2049h), *FltSig\_SR* (204Ah) and *IntSigSr* (2062h) for monitoring internal signals from the device
- *action\_st* (2048h) for monitoring the operating status
- *error code* (603Fh) with which the cause of the last interruption can be determined.

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>FltSig</i> (2049h)	Status.FltSig (28:17)	monitoring signals
<i>FltSig_SR</i> (204Ah)	Status.FltSig_SR (28:18)	stored monitoring signals
<i>IntSigSr</i> (2062h)	Status.IntSigSR (29:34)	internal monitoring signals
<i>action_st</i> (204Bh)	Status.action_st (28:19)	action word, stored error class bits
<i>error code</i> (603Fh)	Status.StopFault (32:7)	cause of last interruption

## 6.10 Adding external bleed resistor

*with AC actuators*

If a positioning controller for AC actuators uses the bleed resistor control *TL\_BRC* with external bleed resistor, the *TL\_BRC* object (203Fh) must be set to „1“.

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>TL_BRC</i> (203Fh)	Settings.TL_BRC (28:26)	external bleed resistor control TL BRC

## 6.11 Motor stop

The drive can be stopped during a movement command over the field bus. If bit 8 in the *controlword* object (6040h) changes to „1“, the positioning controller brakes the drive using the deceleration ramp which has been set for the movement command. Movement data and position data are retained.

The drive continues an interrupted movement command as soon as bit 8 changes back to „0“.

## 6.12 Monitoring and changing signal interface inputs and outputs

The analogue signal and the digital signals of the signal interface can be monitored and switched over the field bus.

The analogue input ANALOG\_IN is monitored via the *AnalogIn* object (2028h), and the digital inputs via the *digital inputs* object (60FDh). This allows a manual movement to be monitored on the field bus through the interface signals, for example.

The interface outputs can be selected and activated by switching the relevant bits by means of the *digital outputs* object (60FEh).

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>digital inputs</i> (60FDh)	–	monitor digital inputs
<i>digital outputs</i> (60FEh)	–	switch digital outputs
<i>AnalogIn</i> (2028h)	Status.AnalogIn (20:8)	monitor analogue input on ANALOG_IN input [mV]

## 6.13 Saving and restoring object data

The positioning controller copies permanently saved object data to the device's RAM memory after it has been switched on. During operation, the positioning controller works with the data in RAM. In order to save object settings, list data or data records so that they are protected from power failures, they must be transferred to the permanent memory using the *eeprSave* object (2008h).

The *default* object (2009h) resets the controller's settings or the device settings to their factory settings.

Object (Index)	Twin Line parameter (index:sub-index).	Explanation
<i>eeprSave</i> (2008h)	Commands.eeprSave (11:6)	save object data to EEPROM
<i>default</i> (2009h)	Commands.default (11:8)	initialise object data with default values factory settings

## 7 Diagnostics and Error Correction

### 7.1 Error diagnosis field bus communication

#### *connections for field bus operation*

In order to be able to evaluate operating messages and error messages, it is essential that field bus operation is working.

If the drive cannot be contacted over the field bus, first check the connections. You will find technical data on the unit as well as information on network and device installation in the manual on the positioning controller. Check:

- 24 V power supply
- power supply connections to the unit
- field bus cables and wiring
- network connections to the unit

You can also use the TL CT operating software for error diagnostics. Connect a PC to the positioning controller's RS232 interface, and start TL CT. You will find details on the TL CT operating software in the manual on software.

#### *function test on the field bus*

If the positioning controller has been correctly connected, check the settings for baud rate, profile selection and node address. The IO\_mode object (2061h) is preset to „0“ with the result that all settings are defined through the signal interface inputs:

- baud rate via inputs BAUD\_1, BAUD\_2 and BAUD4
- node address via inputs ADR\_1 to ADR\_64
- profile selection for CANopen via inputs MODE1=1 and MODE2=0.

If IO\_mode (2061h) is set to „1“ or „2“, the settings can be defined via the TL CT or the TL HMI.

Once the transmission data have been correctly set, check the operation of the field bus. To do so, a CAN configuration tool must be installed which displays CAN messages. The response is captured by the drive through a boot-up message:

- Switch the power supply to the drive off and back on again.
- Watch the network messages shortly after the drive is switched on. The positioning controller transmits a 1 byte long boot-up message after bus initialization: 128 (80h)+node-ID.

With the node address at its factory setting of 127 (7Fh), boot-up message 255 (FFh) is transmitted over the bus. The drive can then be started up via LMT and NMT services.



*If network operation cannot be established, the network capability of the device must be checked by SIG Positec. Contact a SIG Positec service representative.*

#### *baud rate and address*

If no connection can be established with a network device, check the baud rate and node address.

- The baud rate must be set at the same rate for all network devices.
- The node address for every device must lie between 1 and 127, and must be set differently for every device.

You can set baud rate and node address for the positioning controller through parameters with the operating software or the hand-held operating software, see „Setting address and baud rate“ on page 4-1.



## 7.2 Error diagnosis via field bus

### 7.2.1 Reporting objects

There are several objects for supplying information on the operating status and error status of the positioning controller:

- *Statusword* object (6041h),  
Operating states which are reported via the status word, are described in the Chapter entitled „Operating modes“ on page 5-7.
- *EMCY* object (80h+ node-ID),  
Error message from a network device with error status and error code, see Chapter entitled „Emergency service“ from page 3-22
- *Error register* object (1001h) error status
- *Error code* object (603Fh) error code of the last error to occur
- Network devices use a special SDO error message to report faulty message exchange by SDO.

### 7.2.2 Signals on the device status

A distinction is made between synchronous and asynchronous errors in evaluating and dealing with errors.

<i>Synchronous errors</i>	The positioning controller signals a synchronous error as a direct response to a message which cannot be evaluated. The causes may be faulty transmission or invalid data. You will find a list of synchronous errors on page 7-5.
<i>Asynchronous errors</i>	Asynchronous errors are reported by the monitoring systems on the positioning controller as soon as a device error occurs. An asynchronous error is reported via bit 3, „Fault“ of the <i>statusword</i> object (6041h) and via bits 5..7 of the <i>driveStat</i> object (2041h). For faults which lead to the movement being interrupted, the positioning controller sends a SYNC message.
<i>Fault evaluation</i>	The <i>driveStat</i> object (2041h) distinguishes between internal monitoring (bit 5=1), external monitoring (bit 6 =1) and warnings (bit 7 =1). Internal monitoring signals report too high a temperature in the power amplifier or positional overrun, for example. External signals are triggered by limit switches or stop switches, and reported by external monitoring devices.  If bits 5 or 6 of the <i>driveStat</i> object (2041h) are set, the cause of the fault can be determined via the <i>IntSigSr</i> (2062h) and <i>FltSig</i> objects (2049h).

### 7.3 CANopen error messages

CANopen error messages are displayed via an EMCY message. They are evaluated by means of the *Error register* (1001h) and *Error code* (603Fh) objects. You will find information on the EMCY object in the chapter entitled „Emergency service“ from page 3-22.

Errors occurring in data exchange by SDO, are reported by CANopen by means of a special SDO error message.

#### 7.3.1 Error register

The *Error register* object (1001h) displays the error status of a network device in bit-encoded form. The exact cause of the error must be determined from the error code table. Bit 0 is set as soon as an error occurs.

Bit	Report	Explanation
0	generic error	An error has occurred.
1	-	
2	-	
3	-	
4	communication	error in network communication
5	Device profile-specific	error in carrying out device profile
6	reserved	reserved
7	manufacturer-specific	manufacturer-specific error message

#### 7.3.2 Error code table

The error code is evaluated via the *error code* object (603Fh), an object in the device profile DSP 402, and displayed as a four-digit hexadecimal number. The error code specifies the cause of the last interruption to movement. The significance of the error code is given in the positioning controller's manual in the chapter on error diagnosis and error correction.

#### 7.3.3 SDO error message ABORT

An SDO error message is issued in response to a faulty SDO transmission. The cause of the error is given in the „error code“, byte 4 to byte 7.

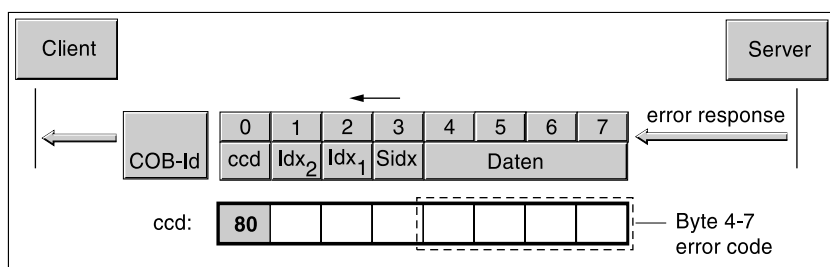


Fig 7.1 SDO error report as response to an SOD message

The following table shows all error messages which can occur in the exchange of data with the positioning controller. The byte sequence must be converted to Intel format before being evaluated.

Example:

Error code 0607 0013h is transmitted as 1300 0706h  
byte 7: 06h, byte 6: 07h, byte 5: 00h, byte 4: 13h

Error code	Explanation
0503 0000h	inversion bit not switched
0504 0000h	time-out on SDO transfer
0504 0001h	command specifier CS incorrect or unknown
0504 0002h	-
0504 0003 h	-
0504 0004h	-
0504 0005h	no free memory
0601 0000h	no access to object possible
0601 0001h	no read access, as object write-only (wo)
0601 0002h	no write access, as object read-only (ro)
0602 0000h	object not in object directory
0604 0041h	object does not support PDO mapping
0604 0042h	PDO mapping: number or length of objects exceed byte length of the PDO
0604 0043h	parameters not compatible
0604 0047h	device has detected internal incompatibility
0606 0000h	hardware fault, access denied
0607 0010h	data type and parameter length mismatch
0607 0012h	data type mismatch, parameter too long
0607 0013h	data type mismatch, parameter too short
0609 0011h	sub-index not supported
0609 0030h	value range of parameter too large (only relevant for write access)
0609 0031h	parameter values too large
0609 0032h	parameter values too small
0609 0036h	upper value smaller than lower value
0800 0000h	general fault
0800 0020h	Data cannot be transmitted or stored for the application.
0800 0021h	Device monitoring carried out locally, data cannot be transmitted or stored.
0800 0022h	Device status prevents transmission and storage of data.
0800 0023h	Object directory either not present or cannot be generated, e.g. when data error occurs in generating from file.
0800 xxxxh	Manufacturer-specific error, xxxx corresponds to the error number of the device. It is given in the error code table in the device manual.



## 8 Service

### 8.1 Service address

If you have queries or problems, please refer them to your SIG Positec contact person or directly to SIG Positec Automation. SIG Positec Automation will be happy to give you the name of their customer service in your area.

SIG Positec Automation GmbH  
Breslauer Str. 7  
D-77933 Lahr

Tel.: (07821) 946 - 02  
Fax: (07821) 946 - 220

<http://www.sig-positec.de>

#### Hardware hotline

questions on devices, service, set-up on site

Telephone: +49 (0) 7821 - 946 - 257  
Fax: +49 (0) 7821 - 946 - 430

Lotus Notes: Hotline, Hardware  
Internet e-mail: [hw.hotline@sig-positec.de](mailto:hw.hotline@sig-positec.de)

#### Software hotline

questions on software, field bus

Telephone: +49 (0) 7821 - 946 - 360  
Fax: +49 (0) 7821 - 946 - 430

Lotus Notes: Hotline, Software  
Internet e-mail: [sw.hotline@sig-positec.de](mailto:sw.hotline@sig-positec.de)

#### RSS office

Repairs and Spare Parts Service

Telephone: +49 (0) 7821 - 946 - 606  
Fax: +49 (0) 7821 - 946 - 202

Lotus Notes: RED, Buero  
Internet, e-mail: [red@sig-positec.de](mailto:red@sig-positec.de)



## 9 Object Directory

### 9.1 Overview

#### 9.1.1 Specifications for the objects

*Index* The index identifies the position of the object in the object directory. The index value is given in hexadecimal.

*Object code* The object code identifies the data structure of the object.

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

Data type	Value range	Data length
Boolean	0=false, 1=true	1 byte
Integer8	-128 .. +127	1 byte
Integer16	-32768 .. +32767	2 bytes
Integer32	-2147483648 .. +2147483647	4 bytes
Unsigned8	0 .. 255	1 byte
Unsigned16	0 .. 65535	2 bytes
Unsigned32	0 .. 4294967295	4 bytes
Visible String8	ASCII characters	8 bytes
Visible String16	ASCII characters	16 bytes

*access* **ro**: read-only, value can only be read

**rw**: read-write: value can be read and written to

**wo**: write-only: value can only be written to

*PDO* **R\_PDO**: mapping for R\_PDO possible

**T\_PDO**: mapping for T\_PDO possible

no details: PDP mapping not possible with the object

*value range* Specifies the permissible range in which the object value is defined and valid.

*default value* Default values can be adjusted by loading and initializing with the stored factory settings.

*storable* ✓: values can be saved in the positioning drive's memory, and are available when the unit is switched on again.

–: values are lost when the positioning drive is switched off.

## 9.1.2 Objects sorted by object name

Name	Index	Sub-index	Obj. Code	Data type	Access	PDO	Description	Page
1st mapped object R_PDO1	1600h	01h	VAR	Unsigned32	rw		first object for mapping in R_PDO1	9-30
1st mapped object R_PDO2	1601h	01h	VAR	Unsigned32	rw		first object for mapping in R_PDO2	9-31
1st mapped object R_PDO3	1602h	01h	VAR	Unsigned32	rw		first object for mapping in R_PDO3	9-31
1st mapped object R_PDO4	1603h	01h	VAR	Unsigned32	ro		first object for mapping in R_PDO4	9-32
1st mapped object T_PDO1	1A00h	01h	VAR	Unsigned32	rw		first object for mapping in T_PDO1	9-39
1st mapped object T_PDO2	1A01h	01h	VAR	Unsigned32	rw		first object for mapping in T_PDO2	9-39
1st mapped object T_PDO3	1A02h	01h	VAR	Unsigned32	rw		first object for mapping in T_PDO3	9-40
1st mapped object T_PDO4	1A03h	01h	VAR	Unsigned32	ro		first object for mapping in T_PDO4	9-41
1st receive PDO mapping	1600h		REC	PDO Mapping	rw		PDO mapping for R_PDO1, settings	9-30
1st receive PDO parameter	1400h		REC	PDO comm. param.	rw		first receive PDO (R_PDO1), settings	9-23
1st transmit PDO mapping	1A00h		REC	PDO Mapping	rw		PDO mapping for T_PDO1, settings	9-39
1st transmit PDO parameter	1800h		REC	PDO comm. param.	rw		first transmit PDO (T_PDO1), settings	9-33
2nd mapped object R_PDO2	1601h	02h	VAR	Unsigned32	rw		second object for mapping in R_PDO2	9-31
2nd mapped object R_PDO3	1602h	02h	VAR	Unsigned32	rw		second object for mapping in R_PDO3	9-31
2nd mapped object R_PDO4	1603h	02h	VAR	Unsigned32	ro		second object for mapping in R_PDO4	9-32
2nd mapped object T_PDO2	1A01h	02h	VAR	Unsigned32	rw		second object for mapping in T_PDO2	9-39
2nd mapped object T_PDO3	1A02h	02h	VAR	Unsigned32	rw		second object for mapping in T_PDO3	9-40
2nd mapped object T_PDO4	1A03h	02h	VAR	Unsigned32	ro		second object for mapping in T_PDO4	9-41
2nd receive PDO mapping	1601h		REC	PDO Mapping	rw		PDO mapping for R_PDO2	9-31
2nd receive PDO parameter	1401h		REC	PDO comm. param.	rw		second receive PDO (R_PDO2), settings	9-26
2nd transmit PDO mapping	1A01h		REC	PDO Mapping	rw		PDO mapping for T_PDO2, settings	9-39
2nd transmit PDO parameter	1801h		REC	PDO comm. param.	rw		second transmit PDO (T_PDO2), settings	9-35
3rd mapped object R_PDO4	1603h	03h	VAR	Unsigned32	ro		third object for mapping in R_PDO4	9-32



Name	Index	Sub-index	Obj. Code	Data type	Access	PDO	Description	Page
3rd mapped object T_PDO4	1A03h	03h	VAR	Unsigned32	ro		third object for mapping in T_PDO4	9-41
3rd receive PDO mapping	1602h		REC	PDO Mapping	rw		PDO mapping for R_PDO3, settings	9-31
3rd receive PDO parameter	1402h		REC	PDO comm. param.	rw		third receive PDO (R_PDO3), settings	9-27
3rd transmit PDO mapping	1A02h		REC	PDO Mapping	rw		PDO mapping for T_PDO3, settings	9-40
3rd transmit PDO parameter	1802h		REC	PDO comm. param.	rw		third transmit PDO (T_PDO3), settings	9-36
4th receive PDO mapping	1603h		REC	PDO Mapping	ro		PDO mapping for R_PDO3, settings	9-32
4th receive PDO parameter	1403h		REC	PDO comm. param.	ro		fourth receive PDO (R_PDO4), settings	9-28
4th transmit PDO mapping	1A03h		REC	PDO Mapping	rw		PDO mapping for T_PDO4, settings	9-41
4th transmit PDO parameter	1803h		REC	PDO comm. param.	ro		fourth transmit PDO (T_PDO4), settings	9-37
acc	205Eh		VAR	Unsigned32	rw		Acceleration [usr]	9-62
acc_ref	2084h		VAR	Unsigned16	ro		Acceleration of the position controller setpoint p_ref [rpm*s]	9-73
acc_type	205Dh		VAR	Unsigned16	rw		Shape of acceleration curve	9-61
acceleration factor	6097h		ARR	Integer32	rw		Acceleration calibration factor	9-120
accOffs	20F8h		VAR	Integer32	rw		Acceleration ramp for offset positioning [r.p.m/s]	9-100
ActCtrl	2070h		VAR	Unsigned16	ro		Active controller parameter set	9-67
action_st	204Bh		VAR	Unsigned16	ro		Action word, saved error class bits	9-57
actList1	20BAh		VAR	Integer16	ro		List 1: current processing number	9-91
actList2	20C0h		VAR	Integer16	ro		List 2: current processing number	9-93
AnalogIn	2028h		VAR	Unsigned16	ro		Analogue input at input ANALOG_IN [mV]	9-48
aNormDen	6097h	02h	VAR	Integer32	rw		Acceleration calibration denominator	9-120
aNormNum	6097h	01h	VAR	Integer32	rw		Acceleration calibration numerator	9-120
bgnList1	20B8h		VAR	Unsigned16	rw		List 1: starting number, first entry for list processing	9-90
bgnList2	20BEh		VAR	Unsigned16	rw		list 2 starting number, first entry for list processing	9-92
bitmask	60FEh	02h	VAR	Unsigned32	rw		enabling and disabling device outputs	9-128
cntList1	20B7h		VAR	Unsigned16	rw		List 1: number of available list entries	9-90
cntList2	20BDh		VAR	Unsigned16	rw		List 2: number of available list entries	9-91
COB-ID EMCY	1014h		VAR	Unsigned32	rw		identifier for Emergency object	9-20
COB-ID R_PDO1	1400h	01h	VAR	Unsigned32	rw		identifier for R_PDO1	9-23
COB-ID R_PDO2	1401h	01h	VAR	Unsigned32	rw		identifier for R_PDO2	9-26

Name	Index	Sub-index	Obj. Code	Data type	Access	PDOs	Description	Page
COB-ID R_PDO3	1402h	01h	VAR	Unsigned32	rw		identifier for R_PDO3	9-27
COB-ID R_PDO4	1403h	01h	VAR	Unsigned32	rw		identifier for R_PDO4	9-28
COB-ID SYNC	1005h		VAR	Unsigned32	rw		identifier for synchronization object	9-16
COB-ID T_PDO1	1800h	01h	VAR	Unsigned32	rw		identifier for T_PDO1	9-33
COB-ID T_PDO2	1801h	01h	VAR	Unsigned32	rw		identifier for T_PDO2	9-35
COB-ID T_PDO3	1802h	01h	VAR	Unsigned32	rw		identifier for T_PDO3	9-36
COB-ID T_PDO4	1803h	01h	VAR	Unsigned32	rw		identifier for T_PDO4	9-37
communication cycle period	1006h		VAR	Unsigned32	rw		interval between two successive SYNC messages [ $\mu$ sec]	9-17
compatibility entry R_PDO1	1400h	04h	VAR	Unsigned8	rw		priority for CAN bus arbitration	9-23
compatibility entry R_PDO2	1401h	04h	VAR	Unsigned8	rw		priority for CAN bus arbitration	9-26
compatibility entry R_PDO3	1402h	04h	VAR	Unsigned8	rw		priority for CAN bus arbitration	9-27
compatibility entry R_PDO4	1403h	04h	VAR	Unsigned8	ro		priority for CAN bus arbitration	9-28
Consumer Heartbeat Time	1016h		ARR	Unsigned32	rw		consumer "heartbeat"	9-21
Consumer Heartbeat Time	1016h	01h	ARR	Unsigned32	rw		interval and node ID of "heartbeat" receiver	9-21
continue	2093h		VAR	Unsigned16	rw		Continuation of interrupted positioning with transfer of any value	9-78
controlword	6040h		VAR	Unsigned16	rw	R_PDO	control word for changing operating status	9-109
dec	205Fh		VAR	Unsigned32	rw		Deceleration [usr]	9-62
decOffs	20F9h		VAR	Integer32	rw		Deceleration ramp for offset positioning [r.p.m/s]	9-101
default	2009h		VAR	Unsigned16	wo		Initialize parameters with default values Factory setting	9-43
denGear	20A2h		VAR	Integer32	rw		Gearbox factor denominator	9-82
device type	1000h		VAR	Unsigned32	ro		Device type and profile	9-14
digital inputs	60FDh		VAR	Unsigned32	ro	T_PDO	signal status of digital inputs	9-127
digital outputs	60FEh		REC	Unsigned32	rw		signal status of digital outputs	9-128
dirEnGear	209Ch		VAR	Unsigned16	rw		Release of movement direction, Reversing the sense of rotation inverts the movement direction	9-80
dist_Man	20ADh		VAR	Unsigned16	rw		Inch travel, defined travel per jog cycle on travel-limited inching [usr]	9-86
DnRamp1	20D3h		VAR	Unsigned32	rw		deceleration ramp selection 1 [usr]	9-94
DnRamp2	20D5h		VAR	Unsigned32	rw		deceleration ramp selection 2 [usr]	9-95
DnRamp3	20D7h		VAR	Unsigned32	rw		deceleration ramp selection 3 [usr]	9-96
drive data	6510h		REC	User Defined	ro		electronic data sheet for electronic systems	9-131
driveCtrl	2040h		VAR	Unsigned16	rw	R_PDO	Control word for change of state	9-52

Name	Index	Sub-index	Obj. Code	Data type	Access	PDO	Description	Page
driveStat	2041h		VAR	Unsigned16	ro	T_PDO	Status word for the operational state of the device	9-52
eeprSave	2008h		VAR	Unsigned16	wo		Save parameter values in EEPROM memory	9-42
endList1	20B9h		VAR	Unsigned16	rw		List 1: finishing number, last entry for list processing	9-91
endList2	20BFh		VAR	Unsigned16	rw		List 2: finishing number, last entry for list processing	9-92
error code	603Fh		VAR	Unsigned16	ro		last error	9-109
error field	1003h	01h	ARR	Unsigned32	ro		error number	9-15
error number	2400h	14h	ARR	Unsigned16	rw		Coded error number	9-108
Error register	1001h		VAR	Unsigned8	ro		Error register	9-14
Error-Field	2400h		ARR	Unsigned16	rw		error memory	9-108
event timer R_PDO1	1400h	05h	VAR	Unsigned16	rw		time span for event triggering	9-23
event timer R_PDO2	1401h	05h	VAR	Unsigned16	rw		time span for event triggering	9-26
event timer R_PDO3	1402h	05h	VAR	Unsigned16	rw		time span for event triggering	9-27
event timer R_PDO4	1403h	05h	VAR	Unsigned16	rw		time span for event triggering	9-28
event timer T_PDO1	1800h	05h	VAR	Unsigned16	rw		time span for event triggering	9-33
event timer T_PDO2	1801h	05h	VAR	Unsigned16	rw		time span for event triggering	9-35
event timer T_PDO3	1802h	05h	VAR	Unsigned16	rw		time span for event triggering	9-36
event timer T_PDO4	1803h	05h	VAR	Unsigned16	rw		time span for event triggering	9-37
f_Chop	2014h		VAR	Unsigned16	rw		Switching frequency of the current module	9-45
Filt_jerk	2044h		VAR	Unsigned16	rw		Jerk filter	9-54
Filt_nRef1	60FBh	17h	VAR	Unsigned16	rw		Parameter set 1: Filter time constant reference variable filter of the set-point speed (100=1ms)	9-122
Filt_nRef2	60FBh	18h	VAR	Unsigned16	rw		Parameter set 2: Filter time constant reference variable filter of the set-point speed (100=1ms)	9-122
Flt_pDiff	204Fh		VAR	Unsigned16	rw		Error response to contour error	9-58
FltSig	2049h		VAR	Unsigned32	ro		Monitoring signals	9-56
FltSig_SR	204Ah		VAR	Unsigned32	ro		Saved monitoring signals	9-56
following error actual value	60F4h		VAR	Integer32	rw		current contouring error	9-122
guard time	100Ch		VAR	Unsigned16	rw		time span for node guarding [ms]	9-19
homing method	6098h		VAR	Integer8	rw		homing method	9-120
homing speeds	6099h		ARR	Integer32	rw		speeds for homing travel	9-121
I_0	2065h		VAR	Unsigned16	rw		Phase current, standstill (100=1Arms)	9-64
I_0M	6410h	29h	VAR	Unsigned16	rw		Motor continuous current at standstill (100=1A)	9-130
I_acc	2066h		VAR	Unsigned16	rw		Phase current, acceleration / deceleration (100=1Arms)	9-64
I_act	2077h		VAR	Integer16	ro	T_PDO	Current motor current [100=1A]	9-69

Name	Index	Sub-index	Obj. Code	Data type	Access	PDOs	Description	Page
I_const	2067h		VAR	Unsigned16	rw		Phase current, constant movement (100=1Arms)	9-65
I_max1	60FBh	01h	VAR	Unsigned16	rw		Parameter set 1: Current limit in all operating modes apart from manual and quick stop [100=1A]	9-122
I_max2	60FBh	02h	VAR	Unsigned16	rw		Parameter set 2: Current limit in all operating modes apart from manual and quick stop (100=1A)	9-122
I_maxM	6410h	06h	VAR	Unsigned16	rw		Max. motor current [100=1A]	9-130
I_maxMan	2050h		VAR	Unsigned16	rw		Max. current manual operation (100=1A)	9-59
I_maxPA	6510h	01h	VAR	Unsigned16	ro		Peak current of the unit [100=1A]	9-131
I_maxSTOP	204Dh		VAR	Unsigned16	rw		Current limit for quick stop [100=1A]	9-58
I_nomM	6410h	07h	VAR	Unsigned16	rw		Nominal motor current [100=1A]	9-130
I_nomPA	6510h	02h	VAR	Unsigned16	ro		Nominal current of the unit [100=1A]	9-131
I_ref	2076h		VAR	Integer16	ro		Setpoint current [100=1A]	9-69
I2tB_act	207Bh		VAR	Integer16	ro		I <sup>2</sup> t total ballast [%]	9-71
I2tM_act	2079h		VAR	Integer16	ro		I <sup>2</sup> t total motor [%]	9-70
I2tPA_act	207Ah		VAR	Integer16	ro		I <sup>2</sup> t total power amplifier [%]	9-70
Identity Object	1018h		REC	Identity	ro		identification object	9-22
inhibit time EMCY	1015h		VAR	Unsigned16	rw		delay before repeat transmission of an EMCY	9-21
inhibit time R_PDO1	1400h	03h	VAR	Unsigned16	rw		inhibit time for bus access (1=100 µsec)	9-23
inhibit time R_PDO2	1401h	03h	VAR	Unsigned16	rw		inhibit time for bus access (1=100 µsec)	9-26
inhibit time R_PDO3	1402h	03h	VAR	Unsigned16	rw		inhibit time for bus access (1=100 µsec)	9-27
inhibit time R_PDO4	1403h	03h	VAR	Unsigned16	ro		inhibit time for bus access (1=100 µsec)	9-28
inhibit time T_PDO1	1800h	03h	VAR	Unsigned16	rw		inhibit time for bus access (in [100 µsec])	9-33
inhibit time T_PDO2	1801h	03h	VAR	Unsigned16	rw		inhibit time for bus access (in [100 µsec])	9-35
inhibit time T_PDO3	1802h	03h	VAR	Unsigned16	rw		inhibit time for bus access (in [100 µsec])	9-36
inhibit time T_PDO4	1803h	03h	VAR	Unsigned16	ro		inhibit time for bus access (in [100 µsec])	9-37
IntSigSr	2062h		VAR	Unsigned32	ro		Monitoring signals	9-63
invertDir	2045h		VAR	Unsigned16	rw		Inversion of sense of rotation	9-54
IO_mode	2061h		VAR	Unsigned16	rw		Significance of I/O signal assignment	9-63
KFAp1	60FBh	15h	VAR	Unsigned16	rw		Parameter set 1: Speed controller feed forward control acceleration (1000=1As*min/rev)	9-122

Name	Index	Sub-index	Obj. Code	Data type	Access	PDO	Description	Page
KFAp2	60FBh	16h	VAR	Unsigned16	rw		Parameter set 2: Speed controller feed forward control acceleration (10 000=1mAs*min/rev)	9-122
KFDn1	60FBh	0Dh	VAR	Unsigned16	rw		Parameter set 1: Speed controller feed forward control D-factor [10 000=1mAs*min/rev]	9-122
KFDn2	60FBh	0Eh	VAR	Unsigned16	rw		Parameter set 2: Speed controller feed forward control D-factor (10 000=1mAs*min/rev)	9-122
KFPn1	60FBh	0Bh	VAR	Unsigned16	rw		Parameter set 1: Speed controller feed forward control P-factor [100=1A*min/rev]	9-122
KFPn2	60FBh	0Ch	VAR	Unsigned16	rw		Parameter set 2: Speed controller feed forward control P-factor (100=1A*min/rev)	9-122
KFPp1	60FBh	13h	VAR	Unsigned16	rw		Parameter set 1: Position controller feed forward control speed [-]	9-122
KFPp2	60FBh	14h	VAR	Unsigned16	rw		Parameter set 2: Position controller feed forward control speed [-]	9-122
KPn1	60FBh	05h	VAR	Unsigned16	rw		Parameter set 1: Speed controller P-factor [1000=A*min/rev]	9-122
KPn2	60FBh	06h	VAR	Unsigned16	rw		Parameter set 2: Speed controller P-factor (1000=1Amin/rev)	9-122
KPp1	60FBh	0Fh	VAR	Unsigned16	rw		Parameter set 1: Position controller P- factor [10=1/s]	9-122
KPp2	60FBh	10h	VAR	Unsigned16	rw		Parameter set 2: Position controller P- factor [1/s]	9-122
life time faktor	100Dh		VAR	Unsigned8	rw		repeat factor for node guarding protocol	9-19
List1_Position	2201h		ARR	Integer32	rw		position values for list 1	9-105
List1_Signal	2202h		ARR	Unsigned16	rw		signal states for list 1	9-105
List1_Type	2200h		ARR	Unsigned16	rw		list type for list 1	9-104
List1_Velocity	2203h		ARR	Integer32	rw		speed values for list 1	9-106
List2_Position	2301h		ARR	Integer32	rw		position values for list 2	9-107
List2_Signal	2302h		ARR	Unsigned16	rw		signal states for list 2	9-107
List2_Type	2300h		ARR	Unsigned16	rw		list type for list 2	9-106
List2_Velocity	2303h		ARR	Integer32	rw		speed values for list 2	9-108
M_nomM	6410h	08h	VAR	Unsigned16	rw		Nominal torque [Ncm]	9-130
manufacturer device name	1008h		VAR	Visible String8	ro		User device name	9-18
manufacturer hardware version	1009h		VAR	Visible String8	ro		hardware version	9-18
manufacturer software version	100Ah		VAR	Visible String8	ro		software version	9-19
manufacturer status register	1002h		VAR	Unsigned32	ro		status of motor and configuration	9-14
MassRecord	2101h	28h	ARR	Unsigned16	rw		mass system for PTP record processing	9-102

Name	Index	Sub-index	Obj. Code	Data type	Access	PDOs	Description	Page
max position limit	607Dh	02h	ARR	Integer32	rw		upper software limit switch SW_LimP	9-115
max profile velocity	607Fh		VAR	Unsigned32	rw		maximum permissible profile speed [usr]	9-116
memNrTeac	20B2h		VAR	Unsigned16	rw		List for teach-in processing	9-88
min position limit	607Dh	01h	ARR	Integer32	rw		lower software limit switch SW_LimN	9-115
modes of operation	6060h		VAR	Integer8	wo	R_PDO	setting the operating mode	9-112
modes of operation display	6061h		VAR	Integer8	rw	T_PDO	displaying current operating mode	9-112
monitorM	206Bh		VAR	Unsigned16	rw		Motor monitoring	9-66
motion profile type	6086h		VAR	Integer16	rw		selection for movement profile	9-118
motor data	6410h		REC	User Defined	ro		electronic data sheet for the motor	9-130
motor manufacturer	6404h		VAR	Visible String16	ro		motor manufacturer	9-129
n_50%	2069h		VAR	Unsigned32	rw		Motor speed with 50% of the standstill momentum [r.p.m]	9-65
n_90%	2068h		VAR	Unsigned32	rw		Motor speed with 90% of the standstill momentum [r.p.m]	9-65
n_act	2075h		VAR	Integer16	ro	T_PDO	Actual speed [rpm]	9-68
n_fastMan	20AC h		VAR	Unsigned32	rw		Speed for fast manual travel [usr]	9-86
n_max0	205Ah		VAR	Integer32	rw		Speed limit for travel profile [r.p.m.]	9-60
n_max1	60FBh	03h	VAR	Unsigned16	rw		Parameter set 1: Max. speed [rpm]	9-122
n_max2	60FBh	04h	VAR	Unsigned16	rw		Parameter set 2: Max. speed [rpm]	9-122
n_maxGear	209Fh		VAR	Integer32	rw		Max. speed [rpm]	9-82
n_nomM	6410h	05h	VAR	Unsigned16	rw		Nominal motor speed [rpm]	9-130
n_ref	2074h		VAR	Integer16	ro		Setpoint speed [rpm]	9-68
n_refOffs	208Bh		VAR	Integer32	ro		Actual speed of offset positioning in electronic gearbox [r.p.m.]	9-75
n_slowMan	20ABh		VAR	Unsigned32	rw		Speed for slow manual travel [usr]	9-85
n_start0	205Bh		VAR	Integer32	rw		Start-stop speed [usr]	9-61
n_tarOffs	20A7h		VAR	Integer32	rw		Setpoint speed of offset positioning [inc/s]	9-84
Name	2020h		VAR	Visible String8	rw		User device name	9-47
nameM	6410h	28h	VAR	Visible String16	rw		Motor name	9-130
number of errors	1003h	00h	VAR	Unsigned8	rw		number of error entries	9-15
numGear	20A1h		VAR	Integer32	rw		Gearbox factor numerator	9-82
OnIAuto	2060h		VAR	Unsigned16	rw		Access to the mode setting	9-62
p_abs	2078h		VAR	Unsigned16	ro	T_PDO	Absolute position per motor revolution (modulo value) [inc]	9-69
p_absOffs	20A4h		VAR	Integer32	rw		Start of absolute offset positioning with transfer of position	9-83

Name	Index	Sub-index	Obj. Code	Data type	Access	PDO	Description	Page
p_absPTP	2090h		VAR	Integer32	rw	R_PDO	Start of absolute positioning with transfer of absolute target position value [usr]	9-77
p_act	2072h		VAR	Integer32	ro	T_PDO	Motor position / rev. [inc]	9-67
p_actTeac	20B3h		VAR	Integer32	ro		current motor position in teach-in processing [usr]	9-89
p_actusr	2089h		VAR	Integer32	ro	T_PDO	Actual position of motor in operator units [usr]	9-75
p_dif	2073h		VAR	Integer32	ro	T_PDO	Contouring error [inc]	9-68
p_DifPeak	2013h		VAR	Unsigned32	rw		Max. contouring error reached [Inc] write access resets value	9-44
p_disHome	20F4h		VAR	Unsigned32	rw		Safety distance of switching edge to reference point	9-99
p_jerkusr	2086h		VAR	Integer32	ro	T_PDO	Actual position of travel profile generator [usr]	9-74
p_maxDiff	2010h		VAR	Unsigned32	rw		Maximum permitted contour error of the position controller [inc]	9-43
p_outHome	20F3h		VAR	Unsigned32	rw		Run-out distance, is automatically approached when reference is found [usr]	9-99
p_ref	2071h		VAR	Integer32	ro		Setpoint position of rotor [inc]	9-67
p_refGear	2081h		VAR	Integer32	ro	T_PDO	Setpoint position of electronic gearbox [inc]	9-72
p_refOffs	2088h		VAR	Integer32	ro	T_PDO	Actual position of offset positioning in electronic gearbox [inc]	9-74
p_relOffs	20A6h		VAR	Integer32	rw		Start of relative offset positioning with transfer of travel value [inc]	9-83
p_relPTP	2092h		VAR	Integer32	rw	R_PDO	Start of relative positioning with value transfer for travel [usr]	9-78
p_remaind	208Ch		VAR	Integer32	ro		Residual value of position calibration of position setpoint p_ref [inc]	9-76
p_target	2085h		VAR	Integer32	ro		Target position of travel profile generator [usr]	9-73
p_tarOffs	2087h		VAR	Integer32	ro		Target position of offset positioning in electronic gearbox [inc]	9-74
p_win	2011h		VAR	Unsigned16	rw		Standstill window, permissible control deviation (inc)	9-44
p_winTime	2012h		VAR	Unsigned16	rw		Time, for which control deviations must apply in the standstill window for standstill to be signalled [ms]	9-44
phomeOffs	20A8h		VAR	Integer32	rw		Sizing in offset positioning [inc]	9-84
physical outputs	60FEh	01h	VAR	Unsigned32	rw	R_PDO	signal interface outputs	9-128
pNormDen	6093h	02h	VAR	Integer32	rw		Position calibration denominator	9-118
pNormNum	6093h	01h	VAR	Integer32	rw		Position calibration numerator	9-118
position actual value	6064h		VAR	Integer32	ro	T_PDO	current position of drive [usr]	9-113
position actual value*	6063h		VAR	Integer32	ro		current position of drive in increments [inc]	9-113

Name	Index	Sub-index	Obj. Code	Data type	Access	PDOs	Description	Page
position control parameter set	60FBh		REC	Unsigned16	rw		Control parameters set	9-122
position factor	6093h		ARR	Integer32	rw		position normalization factor	9-118
position window time	6068h		VAR	Unsigned16	rw		Time, for which control deviations must apply in the standstill window for standstill to be signalled [ms]	9-114
posList1	2201h	40h	ARR	Integer32	rw		List 1: position	9-105
posList2	2301h	40h	ARR	Integer32	rw		List 2: position	9-107
PosRecord	2102h	28h	ARR	Integer32	rw		setpoint for PTP record processing [usr]	9-102
predefined error field	1003h		ARR	Unsigned32	rw		error history, memory for error messages	9-15
Producer Heartbeat Time	1017h		VAR	Unsigned16	rw		interval for producer "heartbeat"	9-22
Product code	1018h	02h	VAR	Unsigned32	ro		product code	9-22
profile acceleration	6083h		VAR	Unsigned32	rw	R_PDO	acceleration default for positioning profile [usr]	9-117
profile deceleration	6084h		VAR	Unsigned32	rw	R_PDO	deceleration default for positioning profile [usr]	9-117
profile velocity	6081h		VAR	Unsigned32	rw	R_PDO	speed default for positioning profile [usr]	9-116
PULSE-C	2032h		VAR	Unsigned16	rw		Setting position encoder PULSE-C	9-50
quick stop deceleration	6085h		VAR	Unsigned32	rw		Deceleration ramp for quick stop [rev/(min*s)]	9-118
ramp	20E9h		VAR	Unsigned16	rw	R_PDO	ramp values for acceleration and deceleration [usr]	9-97
RampRecord	2104h	28h	ARR	Unsigned16	rw		ramp selection for record	9-103
Record_Mass	2101h		ARR	Unsigned16	rw		record data: mass system	9-102
Record_Position	2102h		ARR	Integer32	rw		record data: position	9-102
Record_Ramp	2104h		ARR	Unsigned16	rw		record data: ramp	9-103
Record_Type	2100h		ARR	Unsigned16	rw		record data: record type	9-101
Record_Velocity	2103h		ARR	Integer32	rw		record data: speed	9-103
reserved T_PDO1	1800h	04h	VAR	Unsigned8	rw		priority for CAN bus arbitration ([0- 7])	9-33
reserved T_PDO2	1801h	04h	VAR	Unsigned8	rw		reserved	9-35
reserved T_PDO3	1802h	04h	VAR	Unsigned8	rw		reserved	9-36
reserved T_PDO4	1803h	04h	VAR	Unsigned8	ro		reserved	9-37
RESO-C	2035h		VAR	Unsigned16	rw		Setting position encoder RESO -C	9-51
Revision number	1018h	03h	VAR	Unsigned32	ro		revision number	9-22
Serial number	1018h	04h	VAR	Unsigned32	ro		serial number	9-22
serial_number	2007h		VAR	Unsigned32	ro		Device serial number, max. 9 digits	9-42
SetCtrl	2043h		VAR	Unsigned16	rw		switch control parameters set	9-53
Sign_SR	2048h		VAR	Unsigned16	ro		Saved signal states of external monitoring signals	9-56
SignEnabl	2046h		VAR	Unsigned16	rw		Signal enable for monitoring inputs	9-55



Name	Index	Sub-index	Obj. Code	Data type	Access	PDO	Description	Page
SignLevel	2047h		VAR	Unsigned16	rw		Signal level for monitoring inputs	9-55
signList1	2202h	40h	ARR	Unsigned16	rw		List 1: signal state	9-105
signList2	2302h	40h	ARR	Unsigned16	rw		List 2: signal state	9-107
SignQstop	204Ch		VAR	Unsigned16	rw		Check signals which initiate quick stop	9-57
SM_toggle	206Ah		VAR	Unsigned16	rw		Short minimal motor movement when switching on the amplifier	9-66
software position limit	607Dh		ARR	Integer32	rw		software limit switch	9-115
speed during search for switch	6099h	01h	ARR	Integer32	rw		Speed for search of reference switch [usr]	9-121
speed during search for zero	6099h	02h	ARR	Integer32	rw		Freewheel speed to 'p_outHome' position [usr]	9-121
startGear	209Dh		VAR	Unsigned16	rw	R_PDO	Start of electronic gearbox processing with selection of processing mode	9-80
startHome	20F0h		VAR	Unsigned16	rw	R_PDO	Start of operating mode referencing	9-97
startList	20B4h		VAR	Unsigned16	rw		activate new list processing, running list processing is deactivated beforehand	9-89
startMan	20A9h		VAR	Unsigned16	rw	R_PDO	Start of manual travel with transfer of control bits	9-84
startRec	20D0h		VAR	Unsigned16	rw	R_PDO	start of record processing with transfer of record number	9-93
startSetp	20F6h		VAR	Integer32	rw		Sizing on sizing position (set absolute position) [usr]	9-100
stateGear	209Eh		VAR	Unsigned16	ro		Acknowledgement: gearbox processing	9-81
stateHome	20F1h		VAR	Unsigned16	ro		Acknowledgement: referencing	9-98
stateList	20B5h		VAR	Unsigned16	ro		acknowledgement and status: list processing	9-89
stateMan	20F5h		VAR	Unsigned16	ro		Acknowledgement: manual travel	9-99
stateOffs	20A5h		VAR	Unsigned16	ro		Acknowledgement: offset positioning	9-83
StatePTP	2091h		VAR	Unsigned16	ro	T_PDO	Acknowledgement: PTP positioning	9-77
stateRec	20D1h		VAR	Unsigned16	ro		acknowledgement: record processing	9-93
stateTeac	20B1h		VAR	Unsigned16	ro		Acknowledgement: teach-in processing	9-88
StateVEL	2096h		VAR	Unsigned16	ro		Acknowledgement: speed profile mode	9-79
statusword	6041h		VAR	Unsigned16	ro	T_PDO	status word for evaluating the operating status	9-110
step_Man	20AEh		VAR	Unsigned16	rw		Inch travel, defined travel on manual travel start [usr]	9-86
storeTeac	20B0h		VAR	Unsigned16	rw		Teach-In processing, select memory address	9-87
SW_Enabl	2054h		VAR	Unsigned16	rw		Set monitoring of software limit switches	9-60

Name	Index	Sub-index	Obj. Code	Data type	Access	PDOs	Description	Page
SW_LimN	2053h		VAR	Integer32	rw		Software limit switch for negativ Position limit LIMN [usr]	9-59
SW_LimP	2052h		VAR	Integer32	rw		Software limit switch for positiv Position limit LIMP [usr]	9-59
synchronous window length	1007h		VAR	Unsigned32	rw		time window for PDO processing after SYNC [μsec]	9-17
target position	607Ah		VAR	Integer32	rw	R_PDO	target position (setpoint) [usr]	9-115
target velocity	60FFh		VAR	Integer32	rw	R_PDO	set speed	9-129
time_Man	20AFh		VAR	Unsigned16	rw		Classical waiting time [ms]	9-87
TL_BRC	203Fh		VAR	Unsigned16	rw		external ballast resistance control TL BRC	9-51
TM_act	207Fh		VAR	Integer16	ro		Temperature of motor [°C]	9-71
TNn1	60FBh	07h	VAR	Unsigned16	rw		Parameter set 1: Speed controller integral time I-factor [100=1ms]	9-122
TNn2	60FBh	08h	VAR	Unsigned16	rw		Parameter set 2: Speed controller integral time I-factor (100=1ms)	9-122
TPA_act	2080h		VAR	Integer16	ro		Temperature of power amplifier [°C]	9-72
transmission type R_PDO1	1400h	02h	VAR	Unsigned8	rw		transmission type	9-23
transmission type R_PDO2	1401h	02h	VAR	Unsigned8	rw		transmission type	9-26
transmission type R_PDO3	1402h	02h	VAR	Unsigned8	rw		transmission type	9-27
transmission type R_PDO4	1403h	02h	VAR	Unsigned8	ro		transmission type	9-28
transmission type T_PDO1	1800h	02h	VAR	Unsigned8	rw		transmission type	9-33
transmission type T_PDO2	1801h	02h	VAR	Unsigned8	rw		transmission type	9-35
transmission type T_PDO3	1802h	02h	VAR	Unsigned8	rw		transmission type	9-36
transmission type T_PDO4	1803h	02h	VAR	Unsigned8	ro		transmission type	9-37
TrigLevl	2018h		VAR	Unsigned16	rw		Signal level for trigger channels	9-46
TrigPact1	202Bh		VAR	Integer32	ro		Actual position of motor on triggering on channel 1 (inc)	9-49
TrigPact2	202Ch		VAR	Integer32	ro		Actual position of motor on triggering on channel 2 (inc)	9-49
TrigPref1	202Dh		VAR	Integer32	ro		Setpoint of electrical gearbox on triggering on channel 1 (inc)	9-49
TrigPref2	202Eh		VAR	Integer32	ro		Setpoint of electrical gearbox on triggering on channel 2 (inc)	9-50
TrigSign	2016h		VAR	Unsigned16	rw		Selection of trigger signals for position storage	9-45
TrigStart	2019h		VAR	Unsigned16	rw		start triggering	9-47
TrigStat	202Ah		VAR	Unsigned16	ro		Status of trigger channels	9-48
TrigType	2017h		VAR	Unsigned16	rw		Position source for position storage	9-46

Name	Index	Sub-index	Obj. Code	Data type	Access	PDO	Description	Page
TVn1	60FBh	09h	VAR	Unsigned16	rw		Parameter set 1: Speed controller derivative time D-factor [100=1ms]	9-122
TVn2	60FBh	0Ah	VAR	Unsigned16	rw		Parameter set 2: Speed controller derivative time D-factor (100=1ms)	9-122
TVp1	60FBh	11h	VAR	Unsigned16	rw		Parameter set 1: Position controller derivative time D-factor (100=1ms)	9-122
TVp2	60FBh	12h	VAR	Unsigned16	rw		Parameter set 2: Position controller derivative time D-factor (100=1ms)	9-122
typeList1	2200h	40h	ARR	Unsigned16	rw		List 1: list type for ALL following list entries	9-104
typeList2	2300h	40h	ARR	Unsigned16	rw		List 2: list type for ALL following list entries	9-106
TypeM	6410h	01h	VAR	Integer32	rw		Motor type, consecutive numbers	9-130
typeMan	20AAh		VAR	Unsigned16	rw		Type of manual travel	9-85
TypeRecord	2100h	28h	ARR	Unsigned16	rw		record data type for ALL the following record data entries	9-101
UDC_act	207Ch		VAR	Integer16	ro		DC-line voltage [10=1V]	9-71
UpRamp1	20D2h		VAR	Unsigned32	rw		acceleration ramp selection 1 [usr]	9-94
UpRamp2	20D4h		VAR	Unsigned32	rw		acceleration ramp selection 2 [usr]	9-95
UpRamp3	20D6h		VAR	Unsigned32	rw		acceleration ramp selection 3	9-95
v_jerk16	20EBh		VAR	Unsigned16	ro	T_PDO	actual speed for manufacturer-specific PTP positioning	9-97
v_jerkusr	208Ah		VAR	Integer32	ro	T_PDO	Actual speed of travel profile generator [usr]	9-75
v_ref	2083h		VAR	Integer32	ro		Speed of the rotor position setpoint value p_ref [inc/s]	9-73
v_refGear	2082h		VAR	Integer32	ro		Setpoint speed of electronic gearbox [inc/s]	9-72
v_tar16	20E8h		VAR	Unsigned16	rw	R_PDO	set speed for manufacturer-specific PTP positioning [usr]	9-96
v_target	208Dh		VAR	Integer32	ro		Target speed of travel profile generator	9-76
v_target0	205Ch		VAR	Integer32	rw		Setpoint speed [usr]	9-61
v_tarPTP	2094h		VAR	Integer32	rw	R_PDO	Setpoint speed of PTP positioning [usr]	9-78
velList1	2203h	40h	ARR	Integer32	rw		List 1: setpoint speed	9-106
velList2	2303h	40h	ARR	Integer32	rw		List 2: setpoint speed	9-108
velocity	2095h		VAR	Integer32	rw	R_PDO	Start of speed change with transfer of setpoint speed [usr]	9-79
velocity actual value	606Ch		VAR	Integer32	ro	T_PDO	current speed of drive [usr]	9-114
velocity encoder factor	6094h		ARR	Integer32	rw		speed normalization factor	9-119
velocity sensor actual value	6069h		VAR	Integer32	ro	T_PDO	current speed of drive in [in/sec]	9-114
VelRecord	2103h	28h	ARR	Integer32	rw		set speed [usr]	9-103
Vendor ID	1018h	01h	VAR	Unsigned32	ro		vendor ID	9-22
vNormDen	6094h	02h	VAR	Integer32	rw		Speed calibration denominator	9-119

Name	Index	Sub-index	Obj. Code	Data type	Access	PDO	Description	Page
vNormNum	6094h	01h	VAR	Integer32	rw		Speed calibration numerator	9-119
xMode_act	2042h		VAR	Unsigned16	ro		Current axis operating mode with additional information	9-53

## 9.2 Position control objects

### 1000h *Device type*

The object gives details of the device profile and device type used.

#### *Object description*

Index	1000h
Object name	device type
PDO mapping	VAR
Object code	Unsigned32

#### *Value description*

Sub-index	00h, device type
Explanation	Device type and profile
Access	read-only
PDO mapping	–
Value range	–
Default value	0x00020192
Storable	✓

#### *Bit coding, sub-index 00h*

Bit	Access	Value	Explanation
31-24	ro	00 <sub>h</sub>	not used
23-16	ro	02 <sub>h</sub>	Bit17=1: actuator
15-0	ro	0192 <sub>h</sub>	Device profile DSP-402 (402=192h)

### 1001h *Error register*

The object displays the error status of the device. The exact cause of the error can be determined via the *error code* object (603Fh). The *pre-defined error field* object (1003h) displays the error history of the drive.

Errors are signalled when they occur by means of an EMCY message.

#### *Object description*

Index	1001h
Object name	error register
PDO mapping	VAR
Object code	Unsigned8

#### *Value description*

Sub-index	00h, error register
Explanation	Error register
Access	read-only
PDO mapping	–
Value range	–
Default value	–
Storable	–

*Bit coding, sub-index 00h*

Bit	Access	Value	Explanation, if bit = 1
0	ro	–	Error (generic error)
1	ro	–	Current
2	ro	–	Voltage
3	ro	–	Temperature
4	ro	–	Communication error
5	ro	–	Device profile error
6	ro	–	reserved
7	ro	–	manufacturer-specific)

If a bit is set, it displays an error message. If sub-index 00h=0, there are no errors.

## 1002h **Manufacturer Status Register**

The object displays the operating status and error status of the positioning controller in bitcoded form. The value is always zero for the positioning controller. Information on operating states can be requested via the *statusword* object (6060h) of the status machine. Error states are shown by the *error code* object (603Fh).

*Object description*

Index	1002h
Object name	manufacturer status register
PDO Mapping	VAR
Object code	Unsigned32

*Value description*

Sub-index	00h, manufacturer status register
Explanation	status of motor and configuration
Access	read-only
PDO Mapping	–
Value range	–
Default value	0
Storable	–

## 1003h **Predefined error field**

The object stores the last error messages to be displayed as EMCY messages.

- The entry in sub-index 00h contains the number of error messages stored.
- The current error message is filed in sub-index 01h and older messages are moved to higher sub-index entries.
- Writing a '0' to sub-index 00h resets the error list.

*Object description*

Index	1003h
Object name	predefined error field
PDO Mapping	ARRAY
Object code	Unsigned32

*Value description*

Sub-index	00h, number of errors
Explanation	number of error entries
Access	
PDO Mapping	–
Value range	0...1
Default value	1
Storable	–
Sub-index	01h, error field
Explanation	error number
Access	read-only
PDO Mapping	–
Value range	–
Default value	0
Storable	–

*Bit coding, sub-index 00h..05h*

byte 0..15: error code.byte 16..31 additional error information, not assigned for the positioning controller.

**1005h COB-Id SYNC message**

The object gives the COB ID of the SYNC object, and defines whether a device sends or receives SYNC messages.

The positioning controller can only receive SYNC messages.

*Object description*

Index	1005h
Object name	COB-ID SYNC
PDO Mapping	VAR
Object code	Unsigned32

*Value description*

Sub-index	00h, COB-ID SYNC
Explanation	identifier for synchronization object
Access	
PDO Mapping	–
Value range	0...4294967295
Default value	0x80000080
Storable	–

*Bit coding, sub-index 00h*

Bit	Access	Value	Explanation
31	ro	0 <sub>b</sub>	1: device can receive SYNC messages (SYNC consumer)
30	ro	1 <sub>b</sub>	1: device can transmit SYNC messages (SYNC producer)
29	ro	0 <sub>b</sub>	0: 11-bit identifier (CAN 2.0A) 1: 29-bit identifier (CAN 2.0B)
28-11	ro	0000 <sub>h</sub>	only relevant if bit 29=1, not used by the positioning controller.
10-7	rw	0001 <sub>b</sub>	function code, bits 10..7 of the COB ID

Bit	Access	Value	Explanation
6-0	ro	7F <sub>h</sub>	Node address, bits 6..0 of the COB ID

A network device must transmit SYNC objects for the purpose of synchronization.

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

## 1006h *Communication cycle period*

The object specifies the cycle time for synchronisation. The cycle time is the time span between two successive SYNC messages sent by a SYNC producer.

### *Object description*

Index	1006h
Object name	communication cycle period
PDO Mapping	VAR
Object code	Unsigned32

### *Value description*

Sub-index	00h, communication cycle period
Explanation	interval between two successive SYNC messages [μsec]
Access	
PDO Mapping	–
Value range	0...4294967295
Default value	0
Storable	–

### *Values, sub-index 00h*

Value	Explanation
0	No synchronisation
> 0	Send SYNC messages with set cycle time.

## 1007h *Synchronous windows length*

The object specifies the time window within a SYNC cycle period. The time window starts when a SYNC message is received. Within the time window, a SYNC consumer must have received its synchronous PDOs to enable them to be processed with the following SYNC message. If a synchronous PDO arrives later, it cannot be evaluated until the next but one SYNC.

### *Object description*

Index	1007h
Object name	synchronous window length
PDO Mapping	VAR
Object code	Unsigned32

*Value description*

Sub-index	00h, synchronous window length
Explanation	time window for PDO processing after SYNC [ $\mu$ sec]
Access	
PDO Mapping	–
Value range	0...4294967295
Default value	0
Storable	–

The value 0" deactivates the object.

The COB ID can be changed by a NMT master in the pre-operational NMT status.

**1008h    *Manufacturer device name***

This object identifies the manufacturer's device name.

*Object description*

Index	1008h
Object name	manufacturer device name
PDO Mapping	VAR
Object code	Visible String8

*Value description*

Sub-index	00h, manufacturer device name
Explanation	User device name
Access	read-only
PDO Mapping	–
Value range	–
Default value	–
Storable	–

The following objects contain further information about the device:  
 - objects 6404h, 6410h: motor data

**1009h    *Manufacturer hardware version***

This object gives the version of device hardware.

*Object description*

Index	1009h
Object name	manufacturer hardware version
PDO Mapping	VAR
Object code	Visible String8

*Value description*

Sub-index	00h, manufacturer hardware version
Explanation	hardware version
Access	read-only
PDO Mapping	–
Value range	–
Default value	–
Storable	–



**100Ah Manufacturer software version**

This object gives the version of device software.

*Object description*

Index	100Ah
Object name	manufacturer software version
PDO Mapping	VAR
Object code	Visible String8

*Value description*

Sub-index	00h, manufacturer software version
Explanation	software version
Access	read-only
PDO Mapping	–
Value range	–
Default value	–
Storable	–

**100Ch Guard time**

This object gives the time span for monitoring the connections (node guarding) of an NMT slave.

*Object description*

Index	100Ch
Object name	guard time
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, guard time
Explanation	time span for node guarding [ms]
Access	
PDO Mapping	–
Value range	0...65535
Default value	0
Storable	–

The time span for monitoring the connections of an NMT master is derived from the guard time multiplied by the life time factor, the *Life time factor* object (100Dh).

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

**100Dh Life time factor**

This object gives the factor which together with the guard time makes up the interval for monitoring the connections of an NMT master. Within this time span the NMT slave expects to receive a node guarding enquiry from the NMT master.

life time = guard time \* life time factor

The value 0 deactivates monitoring of the NMT master.

*Object description*

Index	100Dh
Object name	life time faktor
PDO Mapping	VAR
Object code	Unsigned8

*Value description*

Sub-index	00h, life time faktor
Explanation	repeat factor for node guarding protocol
Access	
PDO Mapping	–
Value range	0...255
Default value	0
Storable	–

If no connection monitoring signals are received from the NMT master during the life time interval, the positioning controller reports an error and changes to error status.

The time factor can be changed in the pre-operational NMT status.

The guard time interval is set by means of the *Guard time* object (100Ch).

**1014h COB-ID-Emergency message**

This object gives the COB ID of the Emergency objectEMCY.

*Object description*

Index	1014h
Object name	COB-ID EMCY
PDO Mapping	VAR
Object code	Unsigned32

*Value description*

Sub-index	00h, COB-ID EMCY
Explanation	identifier for Emergency object
Access	
PDO Mapping	–
Value range	0...4294967295
Default value	0x40000080+NodeID
Storable	–

*Bit coding, sub-index 00h*

Bit	Access	Value	Explanation
31, 30	ro	0 <sub>b</sub>	reserved
29	ro	0 <sub>b</sub>	0: 11-bit identifier (CAN 2.0A) 1: 29-bit identifier (CAN 2.0B)
28-11	ro	0000 <sub>h</sub>	only relevant if bit 29=1, not used by the positioning controller.
10-7	rw	0001 <sub>b</sub>	Function code, bits 10...-7 of the COB ID
6-0	ro	-	Node address, bits 6-0 of the COB ID

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

**1015h    *Inhibit time emergency message***

This object defines the delay for repeat transmissions of EMCY messages in multiples of 100 ms.

*Object description*

Index	1015h
Object name	inhibit time EMCY
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, inhibit time EMCY
Explanation	delay before repeat transmission of an EMCY
Access	
PDO Mapping	–
Value range	0...65535
Default value	0
Storable	–

**1016h    *Consumer Heartbeat Time***

This object stores the settings of the Heartbeat consumer for NMT monitoring by means of Heartbeat connection reports.

*Object description*

Index	1016h
Object name	Consumer Heartbeat Time
PDO Mapping	ARRAY
Object code	Unsigned32

*Value description*

Sub-index	00h, number of elements
Explanation	number of values for the object
Access	read-only
PDO Mapping	–
Value range	–
Default value	1
Storable	–
Sub-index	01h, Consumer Heartbeat Time
Explanation	interval and node ID of "heartbeat" receiver
Access	
PDO Mapping	–
Value range	0...4294967295
Default value	0
Storable	–

*Bit coding sub-index 01h..2Fh*

Bit	Explanation
31..24	reserved
23..16	Node ID
15..0	Interval between heartbeat reports

The interval is given in multiples of 1 ms, and must be larger than the producer Heartbeat time, contained in the *Producer Heartbeat Time* object (1017h). If the interval is zero, the device specified via node ID is not being monitored.

**1017h    *Producer Heartbeat Time***

This object stores the interval of the Heartbeat producer for NMT monitoring by Heartbeat connection reports in multiples of 1 ms.

*Object description*

Index	1017h
Object name	Producer Heartbeat Time
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, Producer Heartbeat Time
Explanation	interval for producer "heartbeat"
Access	
PDO Mapping	–
Value range	0...65535
Default value	0
Storable	–

The producerHeartbeat time must be shorter than the interval of the Heartbeat consumer, contained in the *Consumer Heartbeat Time* object (1016h). An interval of zero switches off the monitoring function.

**1018h    *Identity Object***

This object gives information about the device. Sub-index 01h (vendor Id) contains the identification code for the manufacturer, sub-index 02h (product Id) gives the manufacturer-specific product code and sub-index 03h (revision number) identifies special CANopen properties for the device. Sub-index 04h (serial number) contains the serial number of the device.

*Object description*

Index	1018h
Object name	Identity Object
PDO Mapping	RECORD
Object code	Identity

*Value description*

Sub-index	00h, number of elements
Explanation	number of values for the object
Access	read-only
PDO Mapping	–
Value range	–
Default value	4
Storable	–
Sub-index	01h, Vendor ID
Explanation	vendor ID
Access	read-only
PDO Mapping	–
Value range	–
Default value	0x0100002E
Storable	–

Sub-index	02h, Product code
Explanation	product code
Access	read-only
PDO Mapping	—
Value range	—
Default value	0
Storable	—

Sub-index	03h, Revision number
Explanation	revision number
Access	read-only
PDO Mapping	—
Value range	—
Default value	0
Storable	—

Sub-index	04h, Serial number
Explanation	serial number
Access	read-only
PDO Mapping	—
Value range	—
Default value	0
Storable	—

### 1400h **1st receive PDO parameter**

The object stores the settings for the first receive PDO, R\_PDO1.

#### *Object description*

Index	1400h
Object name	1st receive PDO parameter
PDO Mapping	RECORD
Object code	PDO Communication Parameter

#### *Value description*

Sub-index	00h, number of elements
Explanation	number of values for the object
Access	read-only
PDO Mapping	—
Value range	—
Default value	5
Storable	—

Sub-index	01h, COB-ID R_PDO1
Explanation	identifier for R_PDO1
Access	—
PDO Mapping	—
Value range	0...4294967295
Default value	0x00000200+nodeID
Storable	—

Sub-index	02h, transmission type R_PDO1
Explanation	transmission type
Access	
PDO Mapping	–
Value range	0...255
Default value	255
Storable	–
Sub-index	03h, inhibit time R_PDO1
Explanation	inhibit time for bus access (1=100 µsec)
Access	
PDO Mapping	–
Value range	0...65535
Default value	0
Storable	–
Sub-index	04h, compatibility entry R_PDO1
Explanation	priority for CAN bus arbitration
Access	
PDO Mapping	–
Value range	0...255
Default value	0
Storable	–
Sub-index	05h, event timer R_PDO1
Explanation	time span for event triggering
Access	
PDO Mapping	–
Value range	0...65535
Default value	0
Storable	–

*Bit assignment, sub-index 01h*

Bit	Access	Value	Explanation
31	rw	0 <sub>b</sub>	0: PDO is active 1: PDO is inactive
30	ro	0 <sub>b</sub>	0: RTR (see below) is possible 1: RTR not permitted
29	ro	0 <sub>b</sub>	0: 11-bit identifier (CAN 2.0A) 1: 29-bit identifier (CAN 2.0B)
28-11	ro	0000 <sub>h</sub>	only relevant if bit 29=1, not used by the positioning controller.
10-7	rw	0100 <sub>b</sub>	Function code, bits 10..-7 of the COB ID
6-0	ro	-	Node address, bits 6-0 of the COB ID

**Bit 31** An R\_PDO can only be used if bit 31=0.

**Bit 30: RTR-Bit** If a network device supports R\_PDOs with RTR (remote transmission-request), it can request a PDO from a PDO producer by means of RTR = 0 in accordance with the producer-consumer relationship.

*Bit coding, sub-index 02h*

The positioning controller cannot request PDOs, but it can respond to a request for a PDO, see RTR bit for T\_PDO1 settings (1800h).

The control for evaluating R\_PDO data is defined via sub-index 02h. The values 241..251 are reserved.

Type of transmission	cyclical	acyclical	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 data received according to the SYNC object.

- When transmission is acyclical (transmission type=0), evaluation is linked to the SYNC object, while transmission of the PDO is not. When a PDO message is received, it is evaluated with the following SYNC.

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

The values 252 to 254 are relevant for updating - but not for transmitting - T\_PDOs.

- 252: Updating of transmission data with reception of the next SYNC
- 253: Updating of transmission data with the reception of a request from a PDO consumer.
- 254: Updating of data on event-driven basis, the trigger event defined by the manufacturer

R\_PDOs with the value 255 are updated immediately upon receipt of the PDO. The data which are transmitted in the PDO in accordance with the definition of the device profile DSP 402, constitute the trigger event.

*Sub-index 03h:* The Inhibit time interval is only relevant for T\_PDOs.

A T\_PDO cannot be re-sent before the Inhibit time period has elapsed. The value is given in multiples of 100 ms.

*Sub-index 04h:* This value is reserved and is not used. Read or write access triggers an SDO error message.

*Sub-index 05h:* The event timer interval is only relevant for T\_PDOs.

A T\_PDO is transmitted when the event timer period has elapsed. At the same time, the period is re-started. The transmission type must be set via sub-index 02h to 254 or 255.

*Settings* R\_PDO1 is processed asynchronously and on an event-driven basis.

The byte assignment of the R\_PDO1 is defined via PDO mapping by means of the *1st Receive PDO mapping* object (1600h). The following assignment is preset for R\_PDO1:

- bytes 0..1: *controlword* (6040h).

The COB ID of the object can be changed in the pre-operational NMT status.

### 1401h **2nd receive PDO-parameter**

The object stores settings for the second receive PDO, R\_PDO2.

#### *Object description*

Index	1401h
Object name	2nd receive PDO parameter
PDO Mapping	RECORD
Object code	PDO Communication Parameter

#### *Value description*

Sub-index	00h, number of elements
Explanation	number of values for the object
Access	read-only
PDO Mapping	—
Value range	—
Default value	5
Storable	—

Sub-index	01h, COB-ID R_PDO2
Explanation	identifier for R_PDO2
Access	—
PDO Mapping	—
Value range	0...4294967295
Default value	0x80000300+nodeID
Storable	—

Sub-index	02h, transmission type R_PDO2
Explanation	transmission type
Access	—
PDO Mapping	—
Value range	0...255
Default value	0
Storable	—

Sub-index	03h, inhibit time R_PDO2
Explanation	inhibit time for bus access (1=100 µsec)
Access	—
PDO Mapping	—
Value range	0...65535
Default value	0
Storable	—

Sub-index	04h, compatibility entry R_PDO2
Explanation	priority for CAN bus arbitration
Access	—
PDO Mapping	—
Value range	0...255
Default value	0
Storable	—



Sub-index	05h, event timer R_PDO2
Explanation	time span for event triggering
Access	
PDO Mapping	—
Value range	0...65535
Default value	0
Storable	—

The significance of the bit states and sub-index values is described by means of the *1st receive PDO-parameters* object (1400h).

*Settings* R\_PDO2 is processed synchronously, acyclically and on an event-driven basis, and must be activated before being used via bit 31=1 in sub-index 01h.

The byte assignment of the R\_PDO2 is defined via PDO mapping with the *2nd Receive PDO mapping* object (1601h). The following assignment is preset for point-to-point operation in the profile position mode:

- bytes 0..1: *controlword* (6040h).
- bytes 2..5: target position of the movement command, *Target position* (607Ah)

The COB ID of the object can be changed in the pre-operational NMT status.

## 1402h *3rd receive PDO-parameter*

This object stores settings for the third receive PDO, R\_PDO3.

### *Object description*

Index	1402h
Object name	3rd receive PDO parameter
PDO Mapping	RECORD
Object code	PDO Communication Parameter

### *Value description*

Sub-index	00h, number of elements
Explanation	number of values for the object
Access	read-only
PDO Mapping	—
Value range	—
Default value	5
Storable	—

Sub-index	01h, COB-ID R_PDO3
Explanation	identifier for R_PDO3
Access	
PDO Mapping	—
Value range	0...4294967295
Default value	0x80000400+nodeID
Storable	—

Sub-index	02h, transmission type R_PDO3
Explanation	transmission type
Access	
PDO Mapping	–
Value range	0...255
Default value	0
Storable	–
Sub-index	03h, inhibit time R_PDO3
Explanation	inhibit time for bus access (1=100 µsec)
Access	
PDO Mapping	–
Value range	0...65535
Default value	0
Storable	–
Sub-index	04h, compatibility entry R_PDO3
Explanation	priority for CAN bus arbitration
Access	
PDO Mapping	–
Value range	0...255
Default value	0
Storable	–
Sub-index	05h, event timer R_PDO3
Explanation	time span for event triggering
Access	
PDO Mapping	–
Value range	0...65535
Default value	0
Storable	–

The significance of the bit states and sub-index values is described by means of the *1st receive PDO-parameters* object (1400h).

#### Settings

R\_PDO3 is processed synchronously, acyclically and on an event-driven basis, and must be activated before being used via bit 31=1 insub-index 01h.

The byte assignment of the R\_PDO3 is defined via PDO mapping by means of the *3rd Receive PDO mapping* object (1602h). The following assignment is preset for speed operations in profile velocity mode:

- bytes 0..1: *controlword* (6040h).
- bytes 2..5: set speed of movement command, *Target velocity* (60FFh)

The COB ID of the object can be changed in the pre-operational NMT status.

### 1403h *4th receive PDO-parameter*

The object stores settings for the fourth receive PDO, R\_PDO4.

*Object description*

Index	1403h
Object name	4th receive PDO parameter
PDO Mapping	RECORD
Object code	PDO Communication Parameter

*Value description*

Sub-index	00h, number of elements
Explanation	number of values for the object
Access	read-only
PDO Mapping	–
Value range	–
Default value	5
Storable	–

Sub-index	01h, COB-ID R_PDO4
Explanation	identifier for R_PDO4
Access	–
PDO Mapping	–
Value range	0...4294967295
Default value	0x80000500+nodeID
Storable	–

Sub-index	02h, transmission type R_PDO4
Explanation	transmission type
Access	read-only
PDO Mapping	–
Value range	–
Default value	254
Storable	–

Sub-index	03h, inhibit time R_PDO4
Explanation	inhibit time for bus access (1=100 µsec)
Access	read-only
PDO Mapping	–
Value range	–
Default value	0
Storable	–

Sub-index	04h, compatibility entry R_PDO4
Explanation	priority for CAN bus arbitration
Access	read-only
PDO Mapping	–
Value range	–
Default value	0
Storable	–

Sub-index	05h, event timer R_PDO4
Explanation	time span for event triggering
Access	–
PDO Mapping	–
Value range	0...65535
Default value	0
Storable	–

The significance of the bit states and sub-index values is described by means of the *1st receive PDO-parameters* object (1400h).

*PDO settings* R\_PDO4 is processed asynchronously, and on an event-driven basis, and must be activated before being used via bit 31=1 in sub-index 01h. The byte assignment of the R\_PDO4 is defined by PDO mapping by means of the *4th Receive PDO mapping* object (1603h), and cannot be changed. The following assignment is permanently set for the manufacturer-specific point-to-point mode:

- bytes 0..3: target position of the movement command *p\_absPTP* (2090h)
- bytes 4..5: set speed of movement command, *v\_targetr* (20E8h)
- bytes 6..7: ramp values for the movement command, *ramp* (20E9h).

The COB ID of the object can be changed in the pre-operational NMT status.

### 1600h *1st receive PDO mapping*

This object states which objects are shown in R\_PDO1 and transmitted with the PDO. When sub-index 00h of the object is read, the number of objects shown is supplied.

#### *Object description*

Index	1600h
Object name	1st receive PDO mapping
PDO Mapping	RECORD
Object code	PDO Mapping

#### *Value description*

Sub-index	00h, number of elements
Explanation	number of values for the object
Access	
PDO Mapping	—
Value range	1...8
Default value	1
Storable	—

Sub-index	01h, 1st mapped object R_PDO1
Explanation	first object for mapping in R_PDO1
Access	
PDO Mapping	—
Value range	0...4294967295
Default value	0x60400010
Storable	—

#### *Bit coding from sub-index 01h*

Every sub-index entry from sub-index 01h specifies the object and the its byte length. The object is identified via the index and sub-index which refer to the device's object directory.

Bit	Explanation
31..16	Index
15..8	Sub-index
7..0	length of object in bytes

*Settings* The PDO assignment for R\_PDO1 can be changed. The following assignment is preset:

- Sub-index01h: PDO mapping of the control word object, *control-word* (6040h).

### 1601h **2nd receive PDO mapping**

This object states which objects are shown in the R\_PDO2 and transmitted with the PDO. When sub-index 00h of the object is read, the number of objects shown is supplied.

#### Object description

Index	1601h
Object name	2nd receive PDO mapping
PDO Mapping	RECORD
Object code	PDO Mapping

#### Value description

Sub-index	00h, number of elements
Explanation	number of values for the object
Access	
PDO Mapping	–
Value range	1...8
Default value	2
Storable	–

Sub-index	01h, 1st mapped object R_PDO2
Explanation	first object for mapping in R_PDO2
Access	
PDO Mapping	–
Value range	0...4294967295
Default value	0x60400010
Storable	–

Sub-index	02h, 2nd mapped object R_PDO2
Explanation	second object for mapping in R_PDO2
Access	
PDO Mapping	–
Value range	0...4294967295
Default value	0x607A0020
Storable	–

The significance of the bit states is described by means of the *1st receive PDO-mapping* object (1600h).

#### Settings

The PDO assignment for R\_PDO2 can be changed. The following assignment is preset for point-to-point operation in the profile position mode:

- Sub-index01h: PDO mapping of the control word object, *control-word* (6040h).
- Sub-index 02h: Target position of the movement command, *Target position* object (607Ah).

### 1602h **3rd receive PDO mapping**

This object states which objects are mapped in R\_PDO3 and transmitted with the PDO. When sub-index 00h of the object is read, the number of objects shown is supplied.

*Object description*

Index	1602h
Object name	3rd receive PDO mapping
PDO Mapping	RECORD
Object code	PDO Mapping

*Value description*

Sub-index	00h, number of elements
Explanation	number of values for the object
Access	
PDO Mapping	–
Value range	1...8
Default value	2
Storable	–

Sub-index	01h, 1st mapped object R_PDO3
Explanation	first object for mapping in R_PDO3
Access	
PDO Mapping	–
Value range	0...4294967295
Default value	0x60400010
Storable	–

Sub-index	02h, 2nd mapped object R_PDO3
Explanation	second object for mapping in R_PDO3
Access	
PDO Mapping	–
Value range	0...4294967295
Default value	0x60FF0020
Storable	–

The significance of the bit states is described by means of the *1st receive PDO-mapping* object (1600h).

*Settings*

The PDO assignment for R\_PDO3 can be changed. The following assignments are preset for speed operations in profile velocity mode:

- Sub-index 01h: PDO mapping of the control word object, *control word* (6040h).
- bytes 2..5: set speed of movement command, *Target velocity* (60FFh)

**1603h 4th receive PDO mapping**

This object states which objects are shown in R\_PDO4 and transmitted with the PDO. When sub-index 00h of the object is read, the number of objects shown is supplied.

*Object description*

Index	1603h
Object name	4th receive PDO mapping
PDO Mapping	RECORD
Object code	PDO Mapping

*Value description*

Sub-index	00h, number of elements
Explanation	number of values for the object
Access	read-only
PDO Mapping	–
Value range	–
Default value	3
Storable	–
Sub-index	01h, 1st mapped object R_PDO4
Explanation	first object for mapping in R_PDO4
Access	read-only
PDO Mapping	–
Value range	–
Default value	0x20900020
Storable	–
Sub-index	02h, 2nd mapped object R_PDO4
Explanation	second object for mapping in R_PDO4
Access	read-only
PDO Mapping	–
Value range	–
Default value	0x20E80010
Storable	–
Sub-index	03h, 3rd mapped object R_PDO4
Explanation	third object for mapping in R_PDO4
Access	read-only
PDO Mapping	–
Value range	–
Default value	0x20E90010
Storable	–

The significance of the bit states is described by means of the *1st receive PDO-mapping* object (1600h).

*Settings*

The PDO assignment for R\_PDO4 cannot be changed. The following assignment is set for manufacturer-specific operating modes:

- bytes 0..3: target position of the movement command *p\_absPTP* (2090h)
- bytes 4..5: set speed of movement command, *v\_targetr* (20E8h)
- bytes 6..7: ramp values for the movement command, *ramp* (20E9h).

**1800h    1st transmit PDO parameter**

This object stores settings for the first transmit PDO, T\_PDO1.

*Object description*

Index	1800h
Object name	1st transmit PDO parameter
PDO Mapping	RECORD
Object code	PDO Communication Parameter

*Value description*

Sub-index	00h, number of elements
Explanation	number of values for the object
Access	read-only
PDO Mapping	–
Value range	–
Default value	5
Storable	–
Sub-index	01h, COB-ID T_PDO1
Explanation	identifier for T_PDO1
Access	–
PDO Mapping	–
Value range	0...4294967295
Default value	0x00000180+nodeID
Storable	–
Sub-index	02h, transmission type T_PDO1
Explanation	transmission type
Access	–
PDO Mapping	–
Value range	0...255
Default value	255
Storable	–
Sub-index	03h, inhibit time T_PDO1
Explanation	inhibit time for bus access (in [100 µsec])
Access	–
PDO Mapping	–
Value range	0...65535
Default value	0
Storable	–
Sub-index	04h, reserved T_PDO1
Explanation	priority for CAN bus arbitration ([0-7])
Access	–
PDO Mapping	–
Value range	0...255
Default value	0
Storable	–
Sub-index	05h, event timer T_PDO1
Explanation	time span for event triggering
Access	–
PDO Mapping	–
Value range	0...65535
Default value	0
Storable	–

The significance of the bit states and sub-index values is described by means of the *1st receive PDO-parameters* object (1400h).

*Settings*

T\_PDO1 is transmitted asynchronously and on an event-driven basis after every change to the PDO data.

The byte assignment of the T\_PDO1 is defined via PDO mapping with the *1st transmit PDO mapping* object (1A00h). The following assignment is preset:



- bytes 0..1: *statusword* (6041h)

The COB ID of the object can be changed in the pre-operational NMT status.

## 1801h **2nd transmit PDO parameter**

This object stores settings for the second transmit PDO, T\_PDO2.

### *Object description*

Index	1801h
Object name	2nd transmit PDO parameter
PDO Mapping	RECORD
Object code	PDO Communication Parameter

### *Value description*

Sub-index	00h, number of elements
Explanation	number of values for the object
Access	read-only
PDO Mapping	–
Value range	–
Default value	5
Storable	–

Sub-index	01h, COB-ID T_PDO2
Explanation	identifier for T_PDO2
Access	–
PDO Mapping	–
Value range	0...4294967295
Default value	0x80000280+nodeID
Storable	–

Sub-index	02h, transmission type T_PDO2
Explanation	transmission type
Access	–
PDO Mapping	–
Value range	0...255
Default value	0
Storable	–

Sub-index	03h, inhibit time T_PDO2
Explanation	inhibit time for bus access (in [100 µsec])
Access	–
PDO Mapping	–
Value range	0...65535
Default value	0
Storable	–

Sub-index	04h, reserved T_PDO2
Explanation	reserved
Access	–
PDO Mapping	–
Value range	0...255
Default value	0
Storable	–

Sub-index	05h, event timer T_PDO2
Explanation	time span for event triggering
Access	
PDO Mapping	–
Value range	0...65535
Default value	0
Storable	–

The significance of the bit states and sub-index values is described by means of the *1st receive PDO-parameters* object (1400h).

*Settings* T\_PDO2 is transmitted synchronously and acyclically.

The byte assignment of the T\_PDO2 is defined via PDO mapping with the *2nd transmit PDO mapping* object (1A01h). The following assignment is preset for point-to-point operation in the profile position mode:

- bytes 0..1: *statusword* (6041h)
- Bytes 2..5: current position, *position actual value* (6064h).

The COB ID of the object can be changed in the pre-operational NMT status.

## 1802h *3rd transmit PDO parameter*

This object stores settings for the third transmit PDO, T\_PDO3.

### *Object description*

Index	1802h
Object name	3rd transmit PDO parameter
PDO Mapping	RECORD
Object code	PDO Communication Parameter

### *Value description*

Sub-index	00h, number of elements
Explanation	number of values for the object
Access	read-only
PDO Mapping	–
Value range	–
Default value	5
Storable	–

Sub-index	01h, COB-ID T_PDO3
Explanation	identifier for T_PDO3
Access	
PDO Mapping	–
Value range	0...4294967295
Default value	0x80000380+nodeID
Storable	–

Sub-index	02h, transmission type T_PDO3
Explanation	transmission type
Access	
PDO Mapping	–
Value range	0...255
Default value	0
Storable	–

Sub-index	03h, inhibit time T_PDO3
Explanation	inhibit time for bus access (in [100 µsec])
Access	
PDO Mapping	—
Value range	0...65535
Default value	0
Storable	—
Sub-index	04h, reserved T_PDO3
Explanation	reserved
Access	
PDO Mapping	—
Value range	0...255
Default value	0
Storable	—
Sub-index	05h, event timer T_PDO3
Explanation	time span for event triggering
Access	
PDO Mapping	—
Value range	0...65535
Default value	0
Storable	—

The significance of the bit states and sub-index values is described by means of the *1st receive PDO-parameters* object (1400h).

*Settings* T\_PDO3 is transmitted synchronously and acyclically.

The byte assignment of the T\_PDO3 is defined via PDO mapping with the *3rd transmit PDO mapping* object (1A02h). The following assignment is preset for speed operations in profile velocity mode:

- bytes 0..1: *statusword* (6041h)
- bytes 2..5: current speed, *velocity actual value* (606Ch).

The COB ID of the object can be changed in the pre-operational NMT status.

## 1803h *4th transmit PDO parameter*

This object stores settings for the fourth transmit PDO, T\_PDO4.

### *Object description*

Index	1803h
Object name	4th transmit PDO parameter
PDO Mapping	RECORD
Object code	PDO Communication Parameter

### *Value description*

Sub-index	00h, number of elements
Explanation	number of values for the object
Access	read-only
PDO Mapping	—
Value range	—
Default value	5
Storable	—

Sub-index	01h, COB-ID T_PDO4
Explanation	identifier for T_PDO4
Access	
PDO Mapping	–
Value range	0...4294967295
Default value	0x80000480+nodeID
Storable	–
Sub-index	02h, transmission type T_PDO4
Explanation	transmission type
Access	read-only
PDO Mapping	–
Value range	–
Default value	254
Storable	–
Sub-index	03h, inhibit time T_PDO4
Explanation	inhibit time for bus access (in [100 µsec])
Access	read-only
PDO Mapping	–
Value range	–
Default value	0
Storable	–
Sub-index	04h, reserved T_PDO4
Explanation	reserved
Access	read-only
PDO Mapping	–
Value range	–
Default value	0
Storable	–
Sub-index	05h, event timer T_PDO4
Explanation	time span for event triggering
Access	
PDO Mapping	–
Value range	0...65535
Default value	1000
Storable	–

The significance of the bit states and sub-index values is described by means of the *1st receive PDO-parameters* object (1400h).

#### Settings

T\_PDO4 is transmitted asynchronously and on an event-driven basis.

The byte assignment of the R\_PDO4 is defined by PDO mapping by means of the *4th transmit PDO mapping* object (1A03h), and cannot be changed. The following assignment is permanently set for the manufacturer-specific point-to-point mode:

- bytes 0..3: current position, *p\_actusr* (2089h)
- bytes 4..5: current speed, *v\_jerkusr* (208Ah)
- bytes 6..7: statusPtP mode, *StatePTP* (2091h).

The COB ID of the object can be changed in the pre-operational NMT status.

**1A00h 1st transmit PDO mapping**

This object states which objects are shown in the T\_PDO1 and transmitted with the PDO. When sub-index 00h of the object is read, the number of objects shown is supplied.

*Object description*

Index	1A00h
Object name	1st transmit PDO mapping
PDO Mapping	RECORD
Object code	PDO Mapping

*Value description*

Sub-index	00h, number of elements
Explanation	number of values for the object
Access	
PDO Mapping	—
Value range	1...8
Default value	1
Storable	—

Sub-index	01h, 1st mapped object T_PDO1
Explanation	first object for mapping in T_PDO1
Access	
PDO Mapping	—
Value range	0...4294967295
Default value	0x60410010
Storable	—

The significance of the bit states is described by means of the *1st receive PDO-mapping* object (1600h).

*Settings* The PDO assignment for T\_PDO1 can be changed. The following assignment is preset:

- Sub-index1: PDO mapping of the status word *statusword* object (6041h)

**1A01h 2nd transmit PDO mapping**

This object states which objects are shown in the T\_PDO2 and transmitted with the PDO. When sub-index 00h of the object is read, the number of objects shown is supplied.

*Object description*

Index	1A01h
Object name	2nd transmit PDO mapping
PDO Mapping	RECORD
Object code	PDO Mapping

*Value description*

Sub-index	00h, number of elements
Explanation	number of values for the object
Access	
PDO Mapping	—
Value range	1...8
Default value	2
Storable	—

Sub-index	01h, 1st mapped object T_PDO2
Explanation	first object for mapping in T_PDO2
Access	
PDO Mapping	–
Value range	0...4294967295
Default value	0x60410010
Storable	–

Sub-index	02h, 2nd mapped object T_PDO2
Explanation	second object for mapping in T_PDO2
Access	
PDO Mapping	–
Value range	0...4294967295
Default value	0x60640020
Storable	–

The significance of the bit states is described by means of the *1st receive PDO-mapping* object (1600h).

**Settings** The PDO assignment for T\_PDO2 can be changed. The following assignment is preset for point-to-point operation in the profile position mode:

- Sub-index1: PDO mapping of the status word *statusword* object (6041h)
- Sub-index2: PDO mapping of the current position, *position actual value* object (6064h).

### 1A02h **3rd transmit PDO mapping**

This object states which objects are shown in the T\_PDO3 and transmitted with the PDO. When sub-index 00h of the object is read, the number of objects shown is supplied.

#### Object description

Index	1A02h
Object name	3rd transmit PDO mapping
PDO Mapping	RECORD
Object code	PDO Mapping

#### Value description

Sub-index	00h, number of elements
Explanation	number of values for the object
Access	
PDO Mapping	–
Value range	1...8
Default value	2
Storable	–

Sub-index	01h, 1st mapped object T_PDO3
Explanation	first object for mapping in T_PDO3
Access	
PDO Mapping	–
Value range	0...4294967295
Default value	0x60410010
Storable	–

Sub-index	02h, 2nd mapped object T_PDO3
Explanation	second object for mapping in T_PDO3
Access	
PDO Mapping	—
Value range	0...4294967295
Default value	0x606C0020
Storable	—

The significance of the bit states is described by means of the *1st receive PDO-mappingobject* (1600h).

*Settings* The PDO assignment for T\_PDO3 can be changed. The following assignment is preset for speed operations in profile velocity mode:

- bytes 0..1: *statusword* (6041h)
- bytes 2..5: current speed, *velocity actual value* (606Ch).

### 1A03h **4th transmit PDO mapping**

This object states which objects are mapped in the T\_PDO4 and transmitted with the PDO. When sub-index 00h of the object is read, the number of objects shown is supplied.

#### *Object description*

Index	1A03h
Object name	4th transmit PDO mapping
PDO Mapping	RECORD
Object code	PDO Mapping

#### *Value description*

Sub-index	00h, number of elements
Explanation	number of values for the object
Access	read-only
PDO Mapping	—
Value range	—
Default value	3
Storable	—

Sub-index	01h, 1st mapped object T_PDO4
Explanation	first object for mapping in T_PDO4
Access	read-only
PDO Mapping	—
Value range	—
Default value	0x20890020
Storable	—

Sub-index	02h, 2nd mapped object T_PDO4
Explanation	second object for mapping in T_PDO4
Access	read-only
PDO Mapping	—
Value range	—
Default value	0x208A0010
Storable	—

Sub-index	03h, 3rd mapped object T_PDO4
Explanation	third object for mapping in T_PDO4
Access	read-only
PDO Mapping	–
Value range	–
Default value	0x20910010
Storable	–

The significance of the bit states is described by means of the *1st receive PDO-mapping* object (1600h).

*Settings* The PDO assignment for T\_PDO4 cannot be changed. The following assignment is permanently set for the manufacturer-specific point-to-pointmode:

- bytes 0..3: current position, *p\_actusr* (2089h)
- bytes 4..5: current speed, *v\_jerkusr* (208Ah)
- bytes 6..7: statusPtP mode, *StatePTP* (2091h).

## 2007h *serial\_number*

The object gives the max. nine-digit serial number of the device.

### Object description

Index	2007h
Object name	serial_number
PDO Mapping	VAR
Object code	Unsigned32

### Value description

Sub-index	00h, serial_number
Explanation	Device serial number, max. 9 digits
Access	read-only
PDO Mapping	–
Value range	–
Default value	–
Storable	–

## 2008h *eeprSave*

With this object, values can be saved from the device's temporary memory to its permanent EEPROM.

### Object description

Index	2008h
Object name	eeprSave
PDO Mapping	VAR
Object code	Unsigned16

### Value description

Sub-index	00h, eeprSave
Explanation	Save parameter values in EEPROM memory
Access	write-only
PDO Mapping	–
Value range	0...65535
Default value	0
Storable	–



Bit coding, sub-index 00h

Bit	Significance for bit value 1
0	save device parameters
1	save record data, only for TLC4xx
2	save list 1 list data
3	save list 2 list data
4	save user-defined data

The respective object bits are set to 1 in order to write the data.

## 2009h *default*

The object initialises the device settings and resets controller parameters and or device parameters.

Object description

Index	2009h
Object name	default
PDO Mapping	VAR
Object code	Unsigned16

Value description

Sub-index	00h, default
Explanation	Initialize parameters with default values Factory setting
Access	write-only
PDO Mapping	–
Value range	0...2
Default value	0
Storable	–

Bit coding, sub-index 00h

Bit	Significance for bit value 1
0	-
1	reset controller parameters
2	reset device parameters to factory settings

In order to write data, the respective object bits must be set to 1 and the object transmitted to the device.

## 2010h *p\_maxDiff*

This object gives the maximum permissible contouring error for the position controller.

Object description

Index	2010h
Object name	p_maxDiff
PDO Mapping	VAR
Object code	Unsigned32

*Value description*

Sub-index	00h, p_maxDiff
Explanation	Maximum permitted contour error of the position controller [inc]
Access	
PDO Mapping	–
Value range	0...131072
Default value	16384
Storable	✓

**2011h**    ***p\_win***

This object defines the standstill window. The window gives the permissible clockwise control deviation up to which the device still reports motor at standstill. The anti-clockwise value corresponds to the clockwise value. Both give the size of the standstill window in increments.

*Object description*

Index	2011h
Object name	p_win
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, p_win
Explanation	Standstill window, permissible control deviation (inc)
Access	
PDO Mapping	–
Value range	0...65535
Default value	10
Storable	✓

**2012h**    ***p\_winTime***

The object determines the time for which any control deviations must be within the standstill window for standstill to be reported.

*Object description*

Index	2012h
Object name	p_winTime
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, p_winTime
Explanation	Time, for which control deviations must apply in the standstill window for standstill to be signalled [ms]
Access	
PDO Mapping	–
Value range	0...32767
Default value	16
Storable	✓

**2013h**    ***p\_DifPeak***

The object gives the largest contouring error reached by the position controller. Write access resets the value.

*Object description*

Index	2013h
Object name	p_DifPeak
PDO Mapping	VAR
Object code	Unsigned32

*Value description*

Sub-index	00h, p_DifPeak
Explanation	Max. contouring error reached [Inc] write access resets value
Access	
PDO Mapping	–
Value range	0
Default value	0
Storable	–

**2014h    *f\_Chop***

This object can be used to change the switching frequency of the power amplifier.

*Object description*

Index	2014h
Object name	f_Chop
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, f_Chop
Explanation	Switching frequency of the current module
Access	
PDO Mapping	–
Value range	0...2
Default value	1
Storable	✓

*Sub-index 00h*

Value	Explanation
0	4kHz
1	8kHz
2	16kHz

Default value is 1 - 8 kHz, exception TLxx38: default value 0- 4 kHz

**2016h    *TrigSign***

With this object, the trigger signals for storing position values are defined.

*Object description*

Index	2016h
Object name	TrigSign
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, TrigSign
Explanation	Selection of trigger signals for position storage
Access	
PDO Mapping	–
Value range	0...15
Default value	4
Storable	–

*Sub-index 00h*

Value	Explanation
0	CAPTURE1
1	CAPTURE2
2	Index pulse (setpoint sensor)
3	Index pulse (actual position sensor)

**2017h TrigType**

The object defines the sensor for recording position values. The choice depends on the device type and on the module configuration.

*Object description*

Index	2017h
Object name	TrigType
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, TrigType
Explanation	Position source for position storage
Access	
PDO Mapping	–
Value range	0...1
Default value	1
Storable	–

*Sub-index 00h*

Value	Explanation
0	Actual position sensor, for stepping motors only with module in M2
1	Setpoint sensor with a module in M1.

**2018h TrigLevl**

This object can be used to change the signal level of trigger channels.

*Object description*

Index	2018h
Object name	TrigLevl
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, TrigLevl
Explanation	Signal level for trigger channels
Access	
PDO Mapping	–
Value range	0...3
Default value	1
Storable	–

*Sub-index 00h*

Bit	Explanation
0	Trigger level on channel 1 0: triggering at 1->0 change 1: triggering at 0->1 change
1	Trigger level on channel 2 0: triggering at 1->0 change 1: triggering at 0->1 change

**2019h TrigStart**

The object starts, stops and controls fast position capture via CAPTURE inputs.

*Object description*

Index	2019h
Object name	TrigStart
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, TrigStart
Explanation	start triggering
Access	
PDO Mapping	–
Value range	0...65535
Default value	0
Storable	–

*Sub-index 00h*

Bit	Significance for bit value 1
0	Triggering on channel 1
1	Triggering on channel 2
14	Abort triggering
15	Repeat triggering

**2020h Name**

With this object, the device can be given its own designation.

*Object description*

Index	2020h
Object name	Name
PDO Mapping	VAR
Object code	Visible String8

*Value description*

Sub-index	00h, Name
Explanation	User device name
Access	
PDO Mapping	–
Value range	–
Default value	–
Storable	✓

**2028h    *AnalogIn***

The object can be used to enter an analogue value on the analogue input ANALOG\_IN.

*Object description*

Index	2028h
Object name	AnalogIn
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, AnalogIn
Explanation	Analogue input at input ANALOG_IN [mV]
Access	read-only
PDO Mapping	–
Value range	–
Default value	0
Storable	–

**202Ah    *TrigStat***

This object gives information on the status of the trigger channels.

*Object description*

Index	202Ah
Object name	TrigStat
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, TrigStat
Explanation	Status of trigger channels
Access	read-only
PDO Mapping	–
Value range	–
Default value	0
Storable	–

*Sub-index 00h*

Bit	Significance for bit value 1
0	Triggering present on channel 1
1	Triggering present on channel 2

**202Bh TrigPact1**

The object stores the actual position of the motor on triggering on channel 1.

*Object description*

Index	202Bh
Object name	TrigPact1
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, TrigPact1
Explanation	Actual position of motor on triggering on channel 1 (inc)
Access	read-only
PDO Mapping	—
Value range	—
Default value	—
Storable	—

**202Ch TrigPact2**

The object stores the actual position of the motor on triggering on channel 2.

*Object description*

Index	202Ch
Object name	TrigPact2
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, TrigPact2
Explanation	Actual position of motor on triggering on channel 2 (inc)
Access	read-only
PDO Mapping	—
Value range	—
Default value	—
Storable	—

**202Dh TrigPref1**

The object stores the setpoint on triggering on channel 1 in electronic gear mode.

*Object description*

Index	202Dh
Object name	TrigPref1
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, TrigPref1
Explanation	Setpoint of electrical gearbox on triggering on channel 1 (inc)
Access	read-only
PDO Mapping	–
Value range	–
Default value	–
Storable	–

**202Eh TrigPref2**

The object stores the setpoint on triggering on channel 2 in electronic gear mode.

*Object description*

Index	202Eh
Object name	TrigPref2
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, TrigPref2
Explanation	Setpoint of electrical gearbox on triggering on channel 2 (inc)
Access	read-only
PDO Mapping	–
Value range	–
Default value	–
Storable	–

**2032h PULSE-C**

This object can be used to set the position sensor module, PULSE-C, in slot M1.

*Object description*

Index	2032h
Object name	PULSE-C
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, PULSE-C
Explanation	Setting position encoder PULSE-C
Access	
PDO Mapping	–
Value range	0...65535
Default value	0
Storable	✓

*Sub-index 00h*

Bit	Explanation
2	maximum frequency 0: 200KHz 1: 25KHz



Bit	Explanation
3	Signal shape 0: Puls-Dir 1: Pulse <sub>forwards</sub> /Pulse <sub>backwards</sub>

**2035h RESO-C**

This object defines the settings for the position sensormodule, RESO-C, in slot M2.

*Object description*

Index	2035h
Object name	RESO-C
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, RESO-C
Explanation	Setting position encoder RESO -C
Access	
PDO Mapping	–
Value range	0...3
Default value	2
Storable	✓

*Sub-index 00h*

Bit 0	Bit 1	Exciter frequency
0	0	3.5 kHz
0	1	5 kHz
1	0	6.5 kHz
1	1	10 kHz

**203Fh TL\_BRC**

This object is used to register the external load resistor control TL BRC.

*Object description*

Index	203Fh
Object name	TL_BRC
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, TL_BRC
Explanation	external ballast resistance control TL BRC
Access	
PDO Mapping	–
Value range	0...1
Default value	0
Storable	✓

*Sub-index 00h*

Value	Explanation
0	No TL BRC connected
1	TL BRC connected

**2040h** *driveCtrl*

This object is used with manufacturer-specific operating modes for controlling changes of status.

For standardized operating modes in accordance with DSP 402, *controlword* (6040h) is used.

*Object description*

Index	2040h
Object name	driveCtrl
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, driveCtrl
Explanation	Control word for change of state
Access	
PDO Mapping	R_PDO
Value range	0...15
Default value	0
Storable	–

*Sub-index 00h*

After the object has been write accessed, the sub-index value is automatically reset to zero.

Bit	Significance for bit value 1
0, disable	switch off power amplifier
1, enable	switch on power amplifier
2, Stop	Stop drive by means of Quick Stop
3, FaultReset	acknowledge error message
Bits 4..15	not assigned

**2041h** *driveStat*

This object is used with manufacturer-specific operating modes for monitoring operating states.

For standardized operating modes in accordance with DSP 402, *status-word* (6041h) is used.

*Object description*

Index	2041h
Object name	driveStat
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, driveStat
Explanation	Status word for the operational state of the device
Access	read-only
PDO Mapping	T_PDO
Value range	–
Default value	0
Storable	–

*Sub-index 00h*

Bit	Significance for bit value 1
Bits 0..3	Current operating status of the device The bit coding is specific to the device and corresponds to the values of the TL parameter, Status. driveStat (28:2) in the manual.
Bit 5	Signal from internal monitoring function, for signals see objects <i>IntSigSr</i> (2062h) and <i>FltSig</i> (2049h).
Bit 6	Signal from external monitoring function, for signals see object <i>Sign_SR</i> (2048h)
Bit 7	Warning signal, for signals see object <i>FltSig</i> (2049h)

**2042h    *xMode\_act***

This object shows the current manufacturer-specific operating mode. The operating modes supported vary according to the device type. You will find a description of the bit coding of the object under the TL parameter Status.xMode (28:3) in the manual.

For standardized operating modes in accordance with DSP 402, *status-word* (6041h) is used.

*Object description*

Index	2042h
Object name	xMode_act
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, xMode_act
Explanation	Current axis operating mode with additional information
Access	read-only
PDO Mapping	–
Value range	–
Default value	0
Storable	–

**2043h    *SetCtrl***

This object can be used to switch between both control parameter sets for TLCx3x devices.

*Object description*

Index	2043h
Object name	SetCtrl
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, SetCtrl
Explanation	switch control parameters set
Access	–
PDO Mapping	–
Value range	0...2
Default value	1
Storable	–

*Sub-index 00h*

Value	Explanation
1	Parameter set 1
2	Parameter set 2

**2044h** *Filt\_jerk*

The object can be used to switch off the jerk filter in TLCx3x devices.

*Object description*

Index	2044h
Object name	Filt_jerk
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, Filt_jerk
Explanation	Jerk filter
Access	
PDO Mapping	–
Value range	0...65535
Default value	0
Storable	✓

*Sub-index 00h*

Value	Explanation
0	off
>0	Filter setting

**2045h** *invertDir*

This object reverses the motor's sense of rotation.

*Object description*

Index	2045h
Object name	invertDir
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, invertDir
Explanation	Inversion of sense of rotation
Access	
PDO Mapping	–
Value range	0...1
Default value	0
Storable	✓

*Sub-index 00h*

Value	Explanation
0	no reversal of direction
1	Reversal of sense of rotation

**2046h SignEnabl**

Signal interface monitoring signals can be switched on and off with this object.

*Object description*

Index	2046h
Object name	SignEnabl
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, SignEnabl
Explanation	Signal enable for monitoring inputs
Access	
PDO Mapping	–
Value range	0...15
Default value	15
Storable	✓

*Sub-index 00h*

Bit	Significance for bit value 1
0	$\overline{\text{LIMP}}$ activate
1	$\overline{\text{LIMN}}$ activate
2	$\overline{\text{STOP}}$ activate
3	$\overline{\text{REF}}$ activate

**2047h SignLevel**

This object can be used to change the signal level of the monitoring inputs at the signal interface.

*Object description*

Index	2047h
Object name	SignLevel
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, SignLevel
Explanation	Signal level for monitoring inputs
Access	
PDO Mapping	–
Value range	0...15
Default value	0
Storable	✓

*Sub-index 00h*

Bit value 0: response on Low level  
bit value 1: response onHigh level.

Bit	Explanation
0	$\overline{\text{LIMP}}$
1	$\overline{\text{LIMN}}$
2	$\overline{\text{STOP}}$
3	$\overline{\text{REF}}$

**2048h Sign\_SR**

The object gives information on the signal level of the monitoring inputs.

*Object description*

Index	2048h
Object name	Sign_SR
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, Sign_SR
Explanation	Saved signal states of external monitoring signals
Access	read-only
PDO Mapping	–
Value range	–
Default value	0
Storable	–

*Sub-index 00h*

Bit value 0: response on Low level  
bit value 1: response on High level.

Bit	Explanation
0	$\overline{\text{LIMP}}$
1	$\overline{\text{LIMN}}$
2	$\overline{\text{STOP}}$
3	$\overline{\text{REF}}$

**2049h FltSig**

This object shows the status of the internal monitoring signals. The bit assignment is specific to each device and can be determined from the manual via TL parameter, Status.FltSig (28:17).

*Object description*

Index	2049h
Object name	FltSig
PDO Mapping	VAR
Object code	Unsigned32

*Value description*

Sub-index	00h, FltSig
Explanation	Monitoring signals
Access	read-only
PDO Mapping	–
Value range	–
Default value	0
Storable	–

**204Ah FltSig\_SR**

This object shows the status of the stored internal monitoring signals. The bit assignment is specific to the device and can be determined from the manual via TL parameter Status.FltSig\_SR(28:18) or Status.FltSig (28:17).

*Object description*

Index	204Ah
Object name	FltSig_SR
PDO Mapping	VAR
Object code	Unsigned32

*Value description*

Sub-index	00h, FltSig_SR
Explanation	Saved monitoring signals
Access	read-only
PDO Mapping	–
Value range	–
Default value	0
Storable	–

**204Bh    *action\_st***

This object gives information on the stored error response(error class) and on the movement status of the drive. The bit assignment is specific to each device and can be determined from the manual via TL parameter, Status.action\_st (28:19).

*Object description*

Index	204Bh
Object name	action_st
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, action_st
Explanation	Action word, saved error class bits
Access	read-only
PDO Mapping	–
Value range	–
Default value	0
Storable	–

**204Ch    *SignQstop***

This object defines the response to monitoring signal which trigger Quick Stop. Settings are device-specific.

*Object description*

Index	204Ch
Object name	SignQstop
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, SignQstop
Explanation	Check signals which initiate quick stop
Access	–
PDO Mapping	–
Value range	0...65535
Default value	0
Storable	✓

*Sub-index 00h*

Bit	Explanation
0	$\overline{\text{LIMP}}$
1	$\overline{\text{LIMN}}$
2	$\overline{\text{STOP}}$
3	$\overline{\text{REF}}$
4..6	not assigned
7	SW_STOP

Bit values	for stepping motors	for AC actuators
0	Stop with deceleration ramp	Stop with deceleration ramp
1	Stop with Quick Stop ramp	Stop with Quick Stop current

**204Dh** *I\_maxSTOP*

This object sets the current limitation for Quick Stop when operating TLCx3x devices.

*Object description*

Index	204Dh
Object name	I_maxSTOP
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, I_maxSTOP
Explanation	Current limit for quick stop [100=1A]
Access	
PDO Mapping	–
Value range	0...29999
Default value	1000
Storable	✓

**204Fh** *Flt\_pDiff*

The object defines the response to a contouring error.

*Object description*

Index	204Fh
Object name	Flt_pDiff
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, Flt_pDiff
Explanation	Error response to contour error
Access	
PDO Mapping	–
Value range	0...3
Default value	3
Storable	✓



*Sub-index 00h*

Value	Explanation
1	Error class 1
2	Error class 2
3	Error class 3

**2050h** *I\_maxMan*

The maximum current for manual mode is set by means of this object.

*Object description*

Index	2050h
Object name	I_maxMan
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, I_maxMan
Explanation	Max. current manual operation (100=1A)
Access	
PDO Mapping	–
Value range	0...29999
Default value	1000
Storable	✓

**2052h** *SW\_LimP*

The object stores the position of the software limit switch SW\_LimP relative to the reference point. The value must be larger than the value for SW\_LimN (2053h).

*Object description*

Index	2052h
Object name	SW_LimP
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, SW_LimP
Explanation	Software limit switch for positiv Position limit LIMP [usr]
Access	
PDO Mapping	–
Value range	-2147483648...2147483647
Default value	2147483647
Storable	✓

**2053h** *SW\_LimN*

The object stores the position of the software limit switch SW\_LimN relative to the reference point. The value must be smaller than the value for SW\_LimP (2052h).

*Object description*

Index	2053h
Object name	SW_LimN
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, SW_LimN
Explanation	Software limit switch for negativ Position limit LIMN [usr]
Access	
PDO Mapping	–
Value range	-2147483648...2147483647
Default value	-2147483648
Storable	✓

**2054h SW\_Enabl**

This object can be used to switch monitoring via software limit switches SW\_LimP and SW\_LimN on and off.

*Object description*

Index	2054h
Object name	SW_Enabl
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, SW_Enabl
Explanation	Set monitoring of software limit switches
Access	
PDO Mapping	–
Value range	0...65535
Default value	0
Storable	✓

*Sub-index 00h*

Bit	Significance for bit value 1
5	SW_LIMP on
6	SW_LIMN on

**205Ah n\_max0**

This object limits the speed for the current movement profile.

*Object description*

Index	205Ah
Object name	n_max0
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, n_max0
Explanation	Speed limit for travel profile [r.p.m.]
Access	
PDO Mapping	–
Value range	1...12000
Default value	3000
Storable	✓

**205Bh    *n\_start0***

The starting speed for stepping motors can be changed with this object. The value is only relevant for TLCx1x devices.

*Object description*

Index	205Bh
Object name	n_start0
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, n_start0
Explanation	Start-stop speed [usr]
Access	
PDO Mapping	–
Value range	0...6000
Default value	12
Storable	✓

**205Ch    *v\_target0***

The object stores the set speed of the movement profile.

*Object description*

Index	205Ch
Object name	v_target0
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, v_target0
Explanation	Setpoint speed [usr]
Access	
PDO Mapping	–
Value range	1...2147483647
Default value	60
Storable	✓

**205Dh    *acc\_type***

This object sets the shape of the acceleration curve. The setting can only be changed for TLCx1x devices.

*Object description*

Index	205Dh
Object name	acc_type
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, acc_type
Explanation	Shape of acceleration curve
Access	
PDO Mapping	–
Value range	1...2
Default value	1
Storable	✓

*Sub-index 00h*

Value	Explanation
1	Linear ramp shape
2	exponential ramp shape, only for stepping motors

**205Eh** *acc*

The object gives the values for the acceleration ramp for the set movement profile.

*Object description*

Index	205Eh
Object name	acc
PDO Mapping	VAR
Object code	Unsigned32

*Value description*

Sub-index	00h, acc
Explanation	Acceleration [usr]
Access	
PDO Mapping	–
Value range	60...2000000
Default value	600
Storable	✓

**205Fh** *dec*

The object gives the values for the deceleration ramp for the set movement profile.

*Object description*

Index	205Fh
Object name	dec
PDO Mapping	VAR
Object code	Unsigned32

*Value description*

Sub-index	00h, dec
Explanation	Deceleration [usr]
Access	
PDO Mapping	–
Value range	60...2000000
Default value	600
Storable	✓

**2060h** *OnlAuto*

This object can be used to limit access to operating mode settings.

*Object description*

Index	2060h
Object name	OnlAuto
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, OnlAuto
Explanation	Access to the mode setting
Access	
PDO Mapping	–
Value range	0...1
Default value	0
Storable	–

*Sub-index 00h*

Value	Explanation
0	All channels have access, e.g. TL HMI and TL CT via the RS232 interface
1	Operating modes can only be set and controlled over the fieldbus.

**2061h IO\_mode**

The object determines the function of interface signals for operating the device.

*Object description*

Index	2061h
Object name	IO_mode
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, IO_mode
Explanation	Significance of I/O signal assignment
Access	
PDO Mapping	–
Value range	0...2
Default value	0
Storable	✓

*Sub-index 00h*

Value	Explanation
0	Input signals are used for setting addresses in field bus operation.
1	Available inputs can be freely used.
2	Input signals are assigned fixed functions

**2062h IntSigSr**

This object shows the status of internal monitoring signals for movements.

*Object description*

Index	2062h
Object name	IntSigSr
PDO Mapping	VAR
Object code	Unsigned32

*Value description*

Sub-index	00h, IntSigSr
Explanation	Monitoring signals
Access	read-only
PDO Mapping	–
Value range	–
Default value	0
Storable	–

*Sub-index 00h*

Bit	Significance for bit value 1
2	Position overrun
5	SW_LimP
6	SW_LimN
7	SW_STOP
15	Amplifier not activated
0..1, 3..4, 8..12, 16..31	reserved

**2065h I\_0**

The object gives the phase current when stepping motoris at standstill.

*Object description*

Index	2065h
Object name	I_0
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, I_0
Explanation	Phase current, standstill (100=1Arms)
Access	
PDO Mapping	–
Value range	0...1000
Default value	90
Storable	✓

**2066h I\_acc**

This object gives the phase current for the acceleration and deceleration of the stepping motor.

*Object description*

Index	2066h
Object name	I_acc
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, I_acc
Explanation	Phase current, acceleration / deceleration (100=1Arms)
Access	
PDO Mapping	–
Value range	0...1000
Default value	90
Storable	✓

**2067h I\_const**

This object defines the phase current for stepping motor at a constant speed.

*Object description*

Index	2067h
Object name	I_const
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, I_const
Explanation	Phase current, constant movement (100=1Arms)
Access	
PDO Mapping	–
Value range	0...1000
Default value	90
Storable	✓

**2068h n\_90%**

The object stores the 90% fulcrum of the torque characteristic of a stepping motor. The value gives the motor speed at which 90% of the motor's maximum torque is still available.

*Object description*

Index	2068h
Object name	n_90%
PDO Mapping	VAR
Object code	Unsigned32

*Value description*

Sub-index	00h, n_90%
Explanation	Motor speed with 90% of the standstill momentum [r.p.m]
Access	
PDO Mapping	–
Value range	1...3000
Default value	–
Storable	✓

**2069h n\_50%**

The object gives the 50% fulcrum of the torque characteristic of a stepping motor. The value gives the motor speed at which 50% of the motor's maximum torque is still available.

*Object description*

Index	2069h
Object name	n_50%
PDO Mapping	VAR
Object code	Unsigned32

*Value description*

Sub-index	00h, n_50%
Explanation	Motor speed with 50% of the standstill momentum [r.p.m]
Access	
PDO Mapping	–
Value range	1...3000
Default value	–
Storable	✓

**206Ah SM\_toggle**

This object can be used to switch on and off the minimum movement of a stepping motor which is executed when the output stage is switched on. 0 switches the function off, 1 switches it on.

*Object description*

Index	206Ah
Object name	SM_toggle
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, SM_toggle
Explanation	Short minimal motor movement when switching on the amplifier
Access	
PDO Mapping	–
Value range	0...1
Default value	1
Storable	✓

**206Bh MonitorM**

The functions for monitoring a stepping motor can be turned on and off with this object.

*Object description*

Index	206Bh
Object name	monitorM
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, monitorM
Explanation	Motor monitoring
Access	
PDO Mapping	–
Value range	0...3
Default value	3
Storable	✓



*Sub-index 00h*

Bit	Significance for bit value 1
0	speed monitoring on
1	temperature monitoring on

**2070h    *ActCtrl***

The object displays the activated control parameter set. The parameter set can be changed using the *SetCtrl* (2043h) object.

*Object description*

Index	2070h
Object name	ActCtrl
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, ActCtrl
Explanation	Active controller parameter set
Access	read-only
PDO Mapping	–
Value range	–
Default value	1
Storable	–

*Sub-index 00h*

Value	Explanation
0	reserved
1	Parameter set 1 active
2	Parameter set 2 active

**2071h    *p\_ref***

This object specifies the setpoint of the rotor in the motor.

*Object description*

Index	2071h
Object name	p_ref
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, p_ref
Explanation	Setpoint position of rotor [inc]
Access	read-only
PDO Mapping	–
Value range	–
Default value	–
Storable	–

**2072h    *p\_act***

This object gives the current position of the motor in relation to one revolution.

*Object description*

Index	2072h
Object name	p_act
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, p_act
Explanation	Motor position / rev. [inc]
Access	read-only
PDO Mapping	T_PDO
Value range	–
Default value	–
Storable	–

**2073h    *p\_dif***

This object shows the contouring error. That is the difference between the setpoint and the actual position of the motor.

*Object description*

Index	2073h
Object name	p_dif
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, p_dif
Explanation	Contouring error [inc]
Access	read-only
PDO Mapping	T_PDO
Value range	–
Default value	–
Storable	–

**2074h    *n\_ref***

This object specifies the set speed for the motor.

*Object description*

Index	2074h
Object name	n_ref
PDO Mapping	VAR
Object code	Integer16

*Value description*

Sub-index	00h, n_ref
Explanation	Setpoint speed [rpm]
Access	read-only
PDO Mapping	–
Value range	–
Default value	0
Storable	–

**2075h    *n\_act***

This object gives the actual speed of the motor.

*Object description*

Index	2075h
Object name	n_act
PDO Mapping	VAR
Object code	Integer16

*Value description*

Sub-index	00h, n_act
Explanation	Actual speed [rpm]
Access	read-only
PDO Mapping	T_PDO
Value range	–
Default value	0
Storable	–

**2076h    *I\_ref***

This object gives the set current.

*Object description*

Index	2076h
Object name	I_ref
PDO Mapping	VAR
Object code	Integer16

*Value description*

Sub-index	00h, I_ref
Explanation	Setpoint current [100=1A]
Access	read-only
PDO Mapping	–
Value range	–
Default value	0
Storable	–

**2077h    *I\_act***

This object identifies the current motor current.

*Object description*

Index	2077h
Object name	I_act
PDO Mapping	VAR
Object code	Integer16

*Value description*

Sub-index	00h, I_act
Explanation	Current motor current [100=1A]
Access	read-only
PDO Mapping	T_PDO
Value range	–
Default value	0
Storable	–

**2078h    *p\_abs***

This object shows the absolute position of the motor per revolution.

<i>Object description</i>	Index	2078h
	Object name	p_abs
	PDO Mapping	VAR
	Object code	Unsigned16
<i>Value description</i>	Sub-index	00h, p_abs
	Explanation	Absolute position per motor revolution (modulo value) [inc]
	Access	read-only
	PDO Mapping	T_PDO
	Value range	–
	Default value	0
	Storable	–

**2079h I2tM\_act**

This object gives the values for  $I^2t$  monitoring of the motor in relation to the maximum possible load.

<i>Object description</i>	Index	2079h
	Object name	I2tM_act
	PDO Mapping	VAR
	Object code	Integer16
<i>Value description</i>	Sub-index	00h, I2tM_act
	Explanation	$I^2t$ total motor [%]
	Access	read-only
	PDO Mapping	–
	Value range	–
	Default value	0
	Storable	–

**207Ah I2tPA\_act**

This object gives the values for  $I^2t$  monitoring of the output stage in relation to the maximum possible load.

<i>Object description</i>	Index	207Ah
	Object name	I2tPA_act
	PDO Mapping	VAR
	Object code	Integer16
<i>Value description</i>	Sub-index	00h, I2tPA_act
	Explanation	$I^2t$ total power amplifier [%]
	Access	read-only
	PDO Mapping	–
	Value range	–
	Default value	0
	Storable	–

**207Bh I2tB\_act**

This object gives the values for  $I^2t$  monitoring of the bleed resistor in relation to the maximum possible load.

*Object description*

Index	207Bh
Object name	I2tB_act
PDO Mapping	VAR
Object code	Integer16

*Value description*

Sub-index	00h, I2tB_act
Explanation	$I^2t$ total ballast [%]
Access	read-only
PDO Mapping	—
Value range	—
Default value	0
Storable	—

**207Ch UDC\_act**

This object can be used to monitor the DC line voltage in the device.

*Object description*

Index	207Ch
Object name	UDC_act
PDO Mapping	VAR
Object code	Integer16

*Value description*

Sub-index	00h, UDC_act
Explanation	DC-line voltage [10=1V]
Access	read-only
PDO Mapping	—
Value range	—
Default value	—
Storable	—

**207Fh TM\_act**

This object gives the current motor temperature if measurement data are being recorded.

*Object description*

Index	207Fh
Object name	TM_act
PDO Mapping	VAR
Object code	Integer16

*Value description*

Sub-index	00h, TM_act
Explanation	Temperature of motor [°C]
Access	read-only
PDO Mapping	—
Value range	—
Default value	—
Storable	—

**2080h TPA\_act**

This object gives the current temperature of the amplifier.

*Object description*

Index	2080h
Object name	TPA_act
PDO Mapping	VAR
Object code	Integer16

*Value description*

Sub-index	00h, TPA_act
Explanation	Temperature of power amplifier [°C]
Access	read-only
PDO Mapping	–
Value range	–
Default value	–
Storable	–

**2081h p\_refGear**

This object reports the setpoint values in electronic gear mode. The increments at the gear input are counted.

*Object description*

Index	2081h
Object name	p_refGear
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, p_refGear
Explanation	Setpoint position of electronic gearbox [inc]
Access	read-only
PDO Mapping	T_PDO
Value range	–
Default value	–
Storable	–

**2082h v\_refGear**

This object reports the set speed in electronic gear mode.

*Object description*

Index	2082h
Object name	v_refGear
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, v_refGear
Explanation	Setpoint speed of electronic gearbox [inc/s]
Access	read-only
PDO Mapping	–
Value range	–
Default value	–
Storable	–

**2083h v\_ref**

This object records the rotor position set speed  $p\_ref$  (2071h).

*Object description*

Index	2083h
Object name	v_ref
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, v_ref
Explanation	Speed of the rotor position setpoint value $p\_ref$ [inc/s]
Access	read-only
PDO Mapping	—
Value range	—
Default value	—
Storable	—

**2084h acc\_ref**

This object records the rotor position set acceleration  $p\_ref$  (2071h).

*Object description*

Index	2084h
Object name	acc_ref
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, acc_ref
Explanation	Acceleration of the position controller setpoint $p\_ref$ [rpm*s]
Access	read-only
PDO Mapping	—
Value range	—
Default value	1
Storable	—

**2085h p\_target**

This object gives information about the target position of the movement profile generator.

*Object description*

Index	2085h
Object name	p_target
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, p_target
Explanation	Target position of travel profile generator [usr]
Access	read-only
PDO Mapping	—
Value range	—
Default value	—
Storable	—

**2086h**    ***p\_jerkusr***

This object reports the actual position of the movement profile generator.

*Object description*

Index	2086h
Object name	p_jerkusr
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, p_jerkusr
Explanation	Actual position of travel profile generator [usr]
Access	read-only
PDO Mapping	T_PDO
Value range	—
Default value	—
Storable	—

**2087h**    ***p\_tarOffs***

This object reports the target position of the offset positioning in electronic gear mode.

*Object description*

Index	2087h
Object name	p_tarOffs
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, p_tarOffs
Explanation	Target position of offset positioning in electronic gearbox [inc]
Access	read-only
PDO Mapping	—
Value range	—
Default value	—
Storable	—

**2088h**    ***p\_refOffs***

This object reports the actual position of the offset positioning in electronic gear mode.

*Object description*

Index	2088h
Object name	p_refOffs
PDO Mapping	VAR
Object code	Integer32



*Value description*

Sub-index	00h, p_refOffs
Explanation	Actual position of offset positioning in electronic gearbox [inc]
Access	read-only
PDO Mapping	T_PDO
Value range	–
Default value	–
Storable	–

**2089h** ***p\_actusr***

This object gives the actual position of the motor in user-defined units.

*Object description*

Index	2089h
Object name	p_actusr
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, p_actusr
Explanation	Actual position of motor in operator units [usr]
Access	read-only
PDO Mapping	T_PDO
Value range	–
Default value	–
Storable	–

**208Ah** ***v\_jerkusr***

This object gives the actual speed of the movement profile generator in user-defined units.

*Object description*

Index	208Ah
Object name	v_jerkusr
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, v_jerkusr
Explanation	Actual speed of travel profile generator [usr]
Access	read-only
PDO Mapping	T_PDO
Value range	–
Default value	–
Storable	–

**208Bh** ***n\_refOffs***

The object gives the actual speed of the offset positioning in electronic gear mode.

*Object description*

Index	208Bh
Object name	n_refOffs
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, n_refOffs
Explanation	Actual speed of offset positioning in electronic gearbox [r.p.m.]
Access	read-only
PDO Mapping	–
Value range	–
Default value	–
Storable	–

**208Ch**    ***p\_remaind***

This object can be used to determine the residual value during position normalization of the positional setpoint. *p\_ref* (2071).

*Object description*

Index	208Ch
Object name	p_remaind
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, p_remaind
Explanation	Residual value of position calibration of position setpoint <i>p_ref</i> [inc]
Access	read-only
PDO Mapping	–
Value range	–
Default value	–
Storable	–

**208Dh**    ***v\_target***

This object gives information about the target speed of the profile generator.

*Object description*

Index	208Dh
Object name	v_target
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, v_target
Explanation	Target speed of travel profile generator
Access	read-only
PDO Mapping	–
Value range	–
Default value	–
Storable	–

**2090h *p\_absPTP***

The object starts an absolute positioning operation in manufacturer-specific point-to-point mode with transfer of absolute target position.

*Object description*

Index	2090h
Object name	p_absPTP
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, p_absPTP
Explanation	Start of absolute positioning with transfer of absolute target position value [usr]
Access	
PDO Mapping	R_PDO
Value range	-2147483648...2147483647
Default value	0
Storable	–

This object is mapped in R\_PDO4 for standard settings.

**2091h *StatePTP***

This object gives information on the status of the manufacturer-specific mode, point-to-point.

*Object description*

Index	2091h
Object name	StatePTP
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, StatePTP
Explanation	Acknowledgement: PTP positioning
Access	read-only
PDO Mapping	T_PDO
Value range	–
Default value	0
Storable	–

*Sub-index 00h*

Bit	Significance for bit value 1
15, motion_err	Error in executing operating mode
14, motion_end	Operating mode completed
13, motion_add_info	Setpoint reached
7	Error SW_STOP
6	Error SW_LIMN
5	Error SW_LIMP
3	Error $\overline{\text{STOP}}$
2	Error $\overline{\text{REF}}$
1	Error $\overline{\text{LIMN}}$
0	Error $\overline{\text{LIMP}}$

**2092h**    ***p\_relPTP***

The object starts a relative positioning operation in manufacturer-specific point-to-point mode with transfer of position value.

*Object description*

Index	2092h
Object name	p_relPTP
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, p_relPTP
Explanation	Start of relative positioning with value transfer for travel [usr]
Access	
PDO Mapping	R_PDO
Value range	-2147483648...2147483647
Default value	0
Storable	—

**2093h**    ***continue***

This object can be used to continue an interrupted positioning operation. The value transferred is of no significance.

*Object description*

Index	2093h
Object name	continue
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, continue
Explanation	Continuation of interrupted positioning with transfer of any value
Access	
PDO Mapping	—
Value range	0...65535
Default value	0
Storable	—

**2094h**    ***v\_tarPTP***

With this object, the set speed for a positioning operation in the manufacturer-specific point-to-point mode is set. If no value is set, the device uses the object value from *v\_target0* (205Ch).

*Object description*

Index	2094h
Object name	v_tarPTP
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, v_tarPTP
Explanation	Setpoint speed of PTP positioning [usr]
Access	
PDO Mapping	R_PDO
Value range	-2147483648...2147483647
Default value	–
Storable	–

**2095h    *velocity***

This object starts the manufacturer-specific speedmode. When a new set speed is communicated while speed mode is running, the speed is immediately adjusted.

*Object description*

Index	2095h
Object name	velocity
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, velocity
Explanation	Start of speed change with transfer of setpoint speed [usr]
Access	
PDO Mapping	R_PDO
Value range	-2147483648...2147483647
Default value	0
Storable	–

**2096h    *StateVEL***

This object gives information on the status of the manufacturer-specific speed mode.

*Object description*

Index	2096h
Object name	StateVEL
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, StateVEL
Explanation	Acknowledgement: speed profile mode
Access	read-only
PDO Mapping	–
Value range	–
Default value	–
Storable	–

*Sub-index 00h*

Bit	Significance for bit value 1
15, vel_err	Error in executing operating mode
14, vel_end	Operating mode completed
13, vel_add_info	Set speed reached
7	Error SW_STOP
6	Error SW_LIMN

Bit	Significance for bit value 1
5	Error SW_LIMP
3	Error $\overline{\text{STOP}}$
2	Error $\overline{\text{REF}}$
1	Error $\overline{\text{LIMN}}$
0	Error $\overline{\text{LIMP}}$

**209Ch** *dirEnGear*

This object prevents any movement against the desired direction of travel in electronic gear mode.

*Object description*

Index	209Ch
Object name	dirEnGear
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, dirEnGear
Explanation	Release of movement direction, Reversing the sense of rotation inverts the movement direction
Access	
PDO Mapping	–
Value range	1...3
Default value	3
Storable	–

*Sub-index 00h*

Value	Explanation
1	only enable clockwise direction
2	only enable anti-clockwise direction
3	enable both directions

Direction reversal via the object, *invertDir* (2045), reverses the enable direction.

**209Dh** *startGear*

This object starts the manufacturer-specific electronic gear mode with selection of the processing mode.

*Object description*

Index	209Dh
Object name	startGear
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, startGear
Explanation	Start of electronic gear box processing with selection of processing mode
Access	
PDO Mapping	R_PDO
Value range	0...3
Default value	0
Storable	–

*Sub-index 00h*

Value	Explanation
0	Disable
1	immediate synchronisation
2	synchronisation with compensatory movement
3	reserved

**209Eh    stateGear**

This object gives information on the status of the manufacturer-specific electronic gear mode.

*Object description*

Index	209Eh
Object name	stateGear
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, stateGear
Explanation	Acknowledgement: gear box processing
Access	read-only
PDO Mapping	–
Value range	–
Default value	–
Storable	–

*Sub-index 00h*

Bit	Significance for bit value 1
15, gear_err	Error in executing operating mode
14, gear_end	Operating mode completed
13, gear_sync_window	
7	Error SW_STOP
6	Error SW_LIMN
5	Error SW_LIMP
3	Error $\overline{\text{STOP}}$
2	Error $\overline{\text{REF}}$
1	Error $\overline{\text{LIMN}}$
0	Error $\overline{\text{LIMP}}$

**209Fh**    ***n\_maxGear***

The object sets the maximum speed for clockwise rotation in electronic gear mode.

*Object description*

Index	209Fh
Object name	n_maxGear
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, n_maxGear
Explanation	Max. speed [rpm]
Access	
PDO Mapping	–
Value range	1...12000
Default value	3000
Storable	✓

**20A1h**    ***numGear***

The object gives the gear factor numerator.

*Object description*

Index	20A1h
Object name	numGear
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, numGear
Explanation	Gearbox factor numerator
Access	
PDO Mapping	–
Value range	-2147483648...2147483647
Default value	1
Storable	–

**20A2h**    ***denGear***

The object gives the gear factor denominator.

*Object description*

Index	20A2h
Object name	denGear
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, denGear
Explanation	Gearbox factor denominator
Access	
PDO Mapping	–
Value range	1...2147483647
Default value	1
Storable	–



**20A4h    *p\_absOffs***

This object starts an absolute offset positioning operation with transfer of the position value while gear processing is running.

*Object description*

Index	20A4h
Object name	p_absOffs
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, p_absOffs
Explanation	Start of absolute offset positioning with transfer of position
Access	
PDO Mapping	–
Value range	-2147483648...2147483647
Default value	0
Storable	–

**20A5h    *stateOffs***

This object gives information on the status of a superimposed offset positioning operation in electronic gear mode.

*Object description*

Index	20A5h
Object name	stateOffs
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, stateOffs
Explanation	Acknowledgement: offset positioning
Access	read-only
PDO Mapping	–
Value range	–
Default value	–
Storable	–

*Sub-index 00h*

Bit	Significance for bit value 1
15, offset_motion_err	error in offset movement
14, offset_motion_end	offset processing completed
13, offset_add_info	Offset setpoint reached

**20A6h    *p\_relOffs***

This object starts a relative offset positioning operation with transfer of the distance value.

*Object description*

Index	20A6h
Object name	p_relOffs
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, p_relOffs
Explanation	Start of relative offset positioning with transfer of travel value [inc]
Access	
PDO Mapping	–
Value range	-2147483648...2147483647
Default value	0
Storable	–

**20A7h    *n\_tarOffs***

This object sets the set speed for an offset positioning process.

*Object description*

Index	20A7h
Object name	n_tarOffs
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, n_tarOffs
Explanation	Setpoint speed of offset positioning [inc/s]
Access	
PDO Mapping	–
Value range	-12000...12000
Default value	60
Storable	–

**20A8h    *phomeOffs***

This object sets the reference point for an absolute positioning process to a particular value.

*Object description*

Index	20A8h
Object name	phomeOffs
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, phomeOffs
Explanation	Sizing in offset positioning [inc]
Access	
PDO Mapping	–
Value range	-2147483648...2147483647
Default value	0
Storable	–

**20A9h    *startMan***

This object starts manufacturer-specific manual movement mode with the transfer of the control bits for slow or fast movement in a clockwise or anti-clockwise direction.

Status information about the operating mode is given by the *stateMan* (20F5h) object.

*Object description*

Index	20A9h
Object name	startMan
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, startMan
Explanation	Start of manual travel with transfer of control bits
Access	
PDO Mapping	R_PDO
Value range	0...7
Default value	0
Storable	–

*Sub-index 00h*

Bit	Significance for bit value 1
2	0: slow movement 1: fast movement
1	anti-clockwise rotation
0	clockwise rotation

**20AAh** *typeMan*

This objects allows the user to select manual movement via classic or united inching.

*Object description*

Index	20AAh
Object name	typeMan
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, typeMan
Explanation	Type of manual travel
Access	
PDO Mapping	–
Value range	0...1
Default value	0
Storable	✓

*Sub-index 00h*

Value	Explanation
0	classic inching
1	united inching

**20ABh** *n\_slowMan*

This object stores the speed for slow manual movement.

*Object description*

Index	20ABh
Object name	n_slowMan
PDO Mapping	VAR
Object code	Unsigned32

*Value description*

Sub-index	00h, n_slowMan
Explanation	Speed for slow manual travel [usr]
Access	
PDO Mapping	–
Value range	1...2147483647
Default value	60
Storable	✓

**20ACh**    ***n\_fastMan***

This object stores the speed for fast manual movement.

*Object description*

Index	20ACh
Object name	n_fastMan
PDO Mapping	VAR
Object code	Unsigned32

*Value description*

Sub-index	00h, n_fastMan
Explanation	Speed for fast manual travel [usr]
Access	
PDO Mapping	–
Value range	1...2147483647
Default value	180
Storable	✓

**20ADh**    ***dist\_Man***

This object defines the inching path in order to move a defined distance per inching cycle in united inching.

*Object description*

Index	20ADh
Object name	dist_Man
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, dist_Man
Explanation	Inch travel, defined travel per jog cycle on travel-limited inching [usr]
Access	
PDO Mapping	–
Value range	1...65535
Default value	20
Storable	✓

**20AEh**    ***step\_Man***

This object specifies the distance which the first inching process performs when manual movement is started.

*Object description*

Index	20AEh
Object name	step_Man
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, step_Man
Explanation	Inch travel, defined travel on manual travel start [usr]
Access	
PDO Mapping	–
Value range	0...65535
Default value	20
Storable	✓

**20AFh** *time\_Man*

The object gives the delay for classic inching after the first movement has been completed and before the change is made to continuous movement.

*Object description*

Index	20AFh
Object name	time_Man
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, time_Man
Explanation	Classical waiting time [ms]
Access	
PDO Mapping	–
Value range	1...30000
Default value	500
Storable	✓

**20B0h** *storeTeac*

The object allows the record or list number of a teach-in operation to be selected in order to save the current position value.

*Object description*

Index	20B0h
Object name	storeTeac
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, storeTeac
Explanation	Teach-In processing, select memory address
Access	
PDO Mapping	–
Value range	0...65535
Default value	0
Storable	–

*Sub-index 00h*

Bit	Explanation
0..5	record or list number 0..63 for filing position value, bitcoded example: List number 5=000101

**20B1h** *stateTeac*

The object gives information on the processing status of a teach-in operation.

*Object description*

Index	20B1h
Object name	stateTeac
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, stateTeac
Explanation	Acknowledgement: teach-in processing
Access	read-only
PDO Mapping	–
Value range	–
Default value	–
Storable	–

*Sub-index 00h*

Bit	Significance for bit value 1
15, teach_err	error during teach-in processing
14, teach_end	teach-in processing completed

**20B2h** *memNrTeac*

The list for teach-in processing is selected with this object.

*Object description*

Index	20B2h
Object name	memNrTeac
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, memNrTeac
Explanation	List for teach-in processing
Access	–
PDO Mapping	–
Value range	1...3
Default value	1
Storable	–

*Sub-index 00h*

Value	Explanation
1	List 1 for list processing
2	List 2 for list processing
3	List for position record processing (for TLC4xx devices)

**20B3h** *p\_actTeac*

The object gives the current position of the motor in teach-in processing.

*Object description*

Index	20B3h
Object name	p_actTeac
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, p_actTeac
Explanation	current motor position in teach-in processing [usr]
Access	read-only
PDO Mapping	–
Value range	–
Default value	0
Storable	–

**20B4h** *startList*

This object activates a new list processing operation, and any running list processing operation is at the same time deactivated.

*Object description*

Index	20B4h
Object name	startList
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, startList
Explanation	activate new list processing, running list processing is deactivated beforehand
Access	
PDO Mapping	–
Value range	0...2
Default value	0
Storable	–

*Sub-index 00h*

Value	Explanation
0	do not activate list
1	activate list 1
2	activate list 2

**20B5h** *stateList*

This object gives information on the start and status of list processing.

*Object description*

Index	20B5h
Object name	stateList
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, stateList
Explanation	acknowledgement and status: list processing
Access	read-only
PDO Mapping	–
Value range	–
Default value	0
Storable	–

*Sub-index 00h*

Bit	Significance for bit value 1
15, list_err	error in list processing
14, list_quit	list processing completed
1	list 2 active
0	list 1 active

**20B7h** *cntList1*

This object gives the number of possible list numbers for list 1.

*Object description*

Index	20B7h
Object name	cntList1
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, cntList1
Explanation	List 1: number of available list entries
Access	
PDO Mapping	–
Value range	0...64
Default value	64
Storable	–

**20B8h** *bgnList1*

This object gives the first list number for list 1. The list number must be smaller than the final number *endList1* (20B9h).

*Object description*

Index	20B8h
Object name	bgnList1
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, bgnList1
Explanation	List 1: starting number, first entry for list processing
Access	
PDO Mapping	–
Value range	0...63
Default value	0
Storable	✓



**20B9h    *endList1***

This object gives the last list number for list 1. The list number must be larger than the starting number *bgnList1* (20B8h).

*Object description*

Index	20B9h
Object name	endList1
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, endList1
Explanation	List 1: finishing number, last entry for list processing
Access	
PDO Mapping	–
Value range	0...63
Default value	63
Storable	✓

**20BAh    *actList1***

This object gives the current list number for list1. The list number must lie within the range of list values *bgnList1* (20B8h) and *endList1* (20B9h).

*Object description*

Index	20BAh
Object name	actList1
PDO Mapping	VAR
Object code	Integer16

*Value description*

Sub-index	00h, actList1
Explanation	List 1: current processing number
Access	read-only
PDO Mapping	–
Value range	–
Default value	-1
Storable	–

*Sub-index 00h*

Value	Explanation
-1	no list entry activated
0..63	last activated list entry

**20BDh    *cntList2***

This object gives the number of possible list numbers for list 2.

*Object description*

Index	20BDh
Object name	cntList2
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, cntList2
Explanation	List 2: number of available list entries
Access	
PDO Mapping	–
Value range	0...64
Default value	64
Storable	–

*Sub-index 00h*

Value	Explanation
0..64	number of available entries

**20BEh** *bgnList2*

This object gives the first list number for list 2. The list number must be smaller than the final number *endList2* (20BFh)sein.

*Object description*

Index	20BEh
Object name	bgnList2
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, bgnList2
Explanation	list 2 starting number, first entry for list processing
Access	
PDO Mapping	–
Value range	0...63
Default value	0
Storable	✓

*Sub-index 00h*

Value	Explanation
0..63	list value 0 .. 63

**20BFh** *endList2*

This object gives the last list number for list 2. The list number must be larger than the starting number *bgnList2* (20BEh)sein.

*Object description*

Index	20BFh
Object name	endList2
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, endList2
Explanation	List 2: finishing number, last entry for list processing
Access	
PDO Mapping	–
Value range	0...63
Default value	63
Storable	✓

**20C0h**    ***actList2***

This object gives the current list number for list2. The list number must lie within the range of list values *bgnList2* (20BEh) und *endList2* (20BFh) liegen.

*Object description*

Index	20C0h
Object name	actList2
PDO Mapping	VAR
Object code	Integer16

*Value description*

Sub-index	00h, actList2
Explanation	List 2: current processing number
Access	read-only
PDO Mapping	–
Value range	–
Default value	-1
Storable	–

*Sub-index 00h*

Value	Explanation
-1	no list entry activated yet
0..63	last activated list entry

**20D0h**    ***startRec***

This object starts the manufacturer-specific record processing mode with the transfer of the starting number, encoded ia bits 0..5.

*Object description*

Index	20D0h
Object name	startRec
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, startRec
Explanation	start of record processing with transfer of record number
Access	
PDO Mapping	R_PDO
Value range	0...50
Default value	0
Storable	–

**20D1h**    ***stateRec***

This object gives information on the status of the manufacturer-specific record processing mode.

*Object description*

Index	20D1h
Object name	stateRec
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, stateRec
Explanation	acknowledgement: record processing
Access	read-only
PDO Mapping	–
Value range	–
Default value	0
Storable	–

*Sub-index 00h*

Bit	Significance for bit value 1
15, record_err	Error in executing operating mode
14, record_end	Operating mode completed
7	Error SW_STOP
6	Error SW_LIMN
5	Error SW_LIMP
3	Error $\overline{\text{STOP}}$
2	Error $\overline{\text{REF}}$
1	Error $\overline{\text{LIMN}}$
0	Error $\overline{\text{LIMP}}$

**20D2h UpRamp1**

This object stores the value for the first acceleration ramp for the manufacturer-specific record processing mode.

*Object description*

Index	20D2h
Object name	UpRamp1
PDO Mapping	VAR
Object code	Unsigned32

*Value description*

Sub-index	00h, UpRamp1
Explanation	acceleration ramp selection 1 [usr]
Access	
PDO Mapping	–
Value range	60...2000000
Default value	600
Storable	–

**20D3h DnRamp1**

This object stores the value for the first deceleration ramp for the manufacturer-specific record processing mode.

*Object description*

Index	20D3h
Object name	DnRamp1
PDO Mapping	VAR
Object code	Unsigned32

*Value description*

Sub-index	00h, DnRamp1
Explanation	deceleration ramp selection 1 [usr]
Access	
PDO Mapping	–
Value range	60...2000000
Default value	600
Storable	–

**20D4h    *UpRamp2***

This object stores the value for the second acceleration ramp for the manufacturer-specific record processing mode.

*Object description*

Index	20D4h
Object name	UpRamp2
PDO Mapping	VAR
Object code	Unsigned32

*Value description*

Sub-index	00h, UpRamp2
Explanation	acceleration ramp selection 2 [usr]
Access	
PDO Mapping	–
Value range	60...2000000
Default value	600
Storable	–

**20D5h    *DnRamp2***

This object stores the value for the second deceleration ramp for the manufacturer-specific record processing mode.

*Object description*

Index	20D5h
Object name	DnRamp2
PDO Mapping	VAR
Object code	Unsigned32

*Value description*

Sub-index	00h, DnRamp2
Explanation	deceleration ramp selection 2 [usr]
Access	
PDO Mapping	–
Value range	60...2000000
Default value	600
Storable	–

**20D6h    *UpRamp3***

This object stores the value for the third acceleration ramp for the manufacturer-specific record processing mode.

*Object description*

Index	20D6h
Object name	UpRamp3
PDO Mapping	VAR
Object code	Unsigned32

*Value description*

Sub-index	00h, UpRamp3
Explanation	acceleration ramp selection 3
Access	
PDO Mapping	–
Value range	60...2000000
Default value	600
Storable	–

**20D7h    *DnRamp3***

This object stores the value for the third deceleration ramp for the manufacturer-specific record processing mode.

*Object description*

Index	20D7h
Object name	DnRamp3
PDO Mapping	VAR
Object code	Unsigned32

*Value description*

Sub-index	00h, DnRamp3
Explanation	deceleration ramp selection 3 [usr]
Access	
PDO Mapping	–
Value range	60...2000000
Default value	600
Storable	–

**20E8h    *v\_tar16***

This object gives the set speed for the manufacturer-specific point-to-point mode.

*Object description*

Index	20E8h
Object name	v_tar16
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, v_tar16
Explanation	set speed for manufacturer-specific PTP positioning [usr]
Access	
PDO Mapping	R_PDO
Value range	0...65535
Default value	–
Storable	–

This object is mapped in bytes 2 to 5 of the R\_PDO4 for standard settings.

**20E9h ramp**

This object stores the values for the acceleration and deceleration ramps for manufacturer-specific operating modes.

*Object description*

Index	20E9h
Object name	ramp
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, ramp
Explanation	ramp values for acceleration and deceleration [usr]
Access	
PDO Mapping	R_PDO
Value range	0...65535
Default value	600
Storable	–

This object is mapped in the last two bytes of the R\_PDO4 for standard settings.

**20EBh v\_jerk16**

This object gives the actual speed for the manufacturer-specific point-to-point mode.

*Object description*

Index	20EBh
Object name	v_jerk16
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, v_jerk16
Explanation	actual speed for manufacturer-specific PTP positioning
Access	read-only
PDO Mapping	T_PDO
Value range	–
Default value	–
Storable	–

**20F0h startHome**

This object starts the manufacturer-specific homing mode by transferring a referencing method.

*Object description*

Index	20F0h
Object name	startHome
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, startHome
Explanation	Start of operating mode referencing
Access	
PDO Mapping	R_PDO
Value range	1...8
Default value	3
Storable	–

*Sub-index 00h*

Value	Explanation
1	$\overline{\text{LIMP}}$
2	$\overline{\text{LIMN}}$
3	REFZ anti-clockwise rotation
4	REFZ clockwise rotation
5	$\overline{\text{LIMP}}$ with index pulse
6	$\overline{\text{LIMN}}$ with index pulse
7	REFZ anti-clockwise rotation with index pulse
8	REFZ clockwise rotation with index pulse

**20F1h stateHome**

This object gives information on the status of the manufacturer-specific homing mode.

*Object description*

Index	20F1h
Object name	stateHome
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, stateHome
Explanation	Acknowledgement: referencing
Access	read-only
PDO Mapping	–
Value range	–
Default value	0
Storable	–

*Sub-index 00h*

Bit	Significance for bit value 1
15, ref_err	Error in executing operating mode
14, ref_end	Operating mode completed
7	Error SW_STOP
6	Error SW_LIMN
5	Error SW_LIMP
3	Error $\overline{\text{STOP}}$
2	Error $\overline{\text{REF}}$
1	Error $\overline{\text{LIMN}}$
0	Error $\overline{\text{LIMP}}$



**20F3h    *p\_outHome***

This object specifies the maximum run-out travel formoving out of the activated reference switch area.

*Object description*

Index	20F3h
Object name	p_outHome
PDO Mapping	VAR
Object code	Unsigned32

*Value description*

Sub-index	00h, p_outHome
Explanation	Run-out distance, is automatically approached when reference is found [usr]
Access	
PDO Mapping	–
Value range	0...2147483647
Default value	0
Storable	✓

*Sub-index 00h*

Value	Explanation
0	monitoring of run-out path switched off
> 0	run-out path in user-defined units

**20F4h    *p\_disHome***

This object gives the distance from the switching edge of the limit switch to the reference point.

*Object description*

Index	20F4h
Object name	p_disHome
PDO Mapping	VAR
Object code	Unsigned32

*Value description*

Sub-index	00h, p_disHome
Explanation	Safety distance of switching edge to reference point
Access	
PDO Mapping	–
Value range	0...2147483647
Default value	200
Storable	✓

**20F5h    *stateMan***

This object gives information on the status of the manufacturer-specific manual movement mode.

*Object description*

Index	20F5h
Object name	stateMan
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, stateMan
Explanation	Acknowledgement: manual travel
Access	read-only
PDO Mapping	–
Value range	–
Default value	0
Storable	–

*Sub-index 00h*

Bit	Significance for bit value 1
15, ref_err	Error in executing operating mode
14, ref_end	Operating mode completed
7	Error SW_STOP
6	Error SW_LIMN
5	Error SW_LIMP
3	Error $\overline{\text{STOP}}$
2	Error $\overline{\text{REF}}$
1	Error $\overline{\text{LIMN}}$
0	Error $\overline{\text{LIMP}}$

**20F6h** *startSetp*

A homing operation by means of dimension setting is triggered by this object. In the process, the position value transmitted is defined as the new reference point.

*Object description*

Index	20F6h
Object name	startSetp
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, startSetp
Explanation	Sizing on sizing position (set absolute position) [usr]
Access	–
PDO Mapping	–
Value range	-2147483648...2147483647
Default value	0
Storable	–

**20F8h** *accOffs*

This object specifies the acceleration ramp for offset positioning in electronic gear mode.

*Object description*

Index	20F8h
Object name	accOffs
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, accOffs
Explanation	Acceleration ramp for offset positioning [r.p.m/s]
Access	
PDO Mapping	–
Value range	60...2000000
Default value	600
Storable	–

**20F9h    *decOffs***

This object specifies the deceleration ramp for offset positioning in electronic gear mode.

*Object description*

Index	20F9h
Object name	decOffs
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, decOffs
Explanation	Deceleration ramp for offset positioning [r.p.m/s]
Access	
PDO Mapping	–
Value range	60...2000000
Default value	600
Storable	–

**2100h    *Record\_Type***

This object gives the record type for data records. Sub-index 00h gives information on the number of records, and the following sub-index values 01h..40h correspond to the record number.

*Object description*

Index	2100h
Object name	Record_Type
PDO Mapping	ARRAY
Object code	Unsigned16

*Value description*

Sub-index	00h, number_of_records
Explanation	number of values for the object
Access	
PDO Mapping	–
Value range	1...40
Default value	40
Storable	–

Sub-index	28h, TypeRecord
Explanation	record data type for ALL the following record data entries
Access	
PDO Mapping	–
Value range	1...2
Default value	1
Storable	–

Subindex 01h...40h

Value	Explanation
	PTP records
	speed records

**2101h Record\_Mass**

This object gives the dimension system for PTP positioning with data records. Sub-index 00h gives information on the number of records, and the following sub-index values 01h..40h correspond to the record number.

Object description

Index	2101h
Object name	Record_Mass
PDO Mapping	ARRAY
Object code	Unsigned16

Value description

Sub-index	00h, number_of_records
Explanation	number of values for the object
Access	
PDO Mapping	–
Value range	1...40
Default value	40
Storable	–
Sub-index	28h, MassRecord
Explanation	mass system for PTP record processing
Access	
PDO Mapping	–
Value range	1...2
Default value	1
Storable	–

Subindex 01h...40h

Value	Explanation
	absolute positioning
	relative positioning

**2102h Record\_Position**

This object gives the position or path for PTP positioning with the data records. Sub-index 00h gives information on the number of records, and the following sub-index values 01h..40h correspond to the record number.

Object description

Index	2102h
Object name	Record_Position
PDO Mapping	ARRAY
Object code	Integer32

*Value description*

Sub-index	00h, number_of_records
Explanation	number of values for the object
Access	
PDO Mapping	–
Value range	1...40
Default value	40
Storable	–

---

Sub-index	28h, PosRecord
Explanation	setpoint for PTP record processing [usr]
Access	
PDO Mapping	–
Value range	-2147483648...2147483647
Default value	1
Storable	–

**2103h    *Record\_Velocity***

This object gives the speed for speed mode with the data records. Sub-index 00h gives information on the number of records, and the following sub-index values 01h..40h correspond to the record number.

*Object description*

Index	2103h
Object name	Record_Velocity
PDO Mapping	ARRAY
Object code	Integer32

*Value description*

Sub-index	00h, number_of_records
Explanation	number of values for the object
Access	
PDO Mapping	–
Value range	1...40
Default value	40
Storable	–

---

Sub-index	28h, VelRecord
Explanation	set speed [usr]
Access	
PDO Mapping	–
Value range	-2147483648...2147483647
Default value	1
Storable	–

**2104h    *Record\_Ramp***

This object gives the ramp selection for operation with the data records. Sub-index 00h gives information on the number of records, and the following sub-index values 01h..40h correspond to the record number.

*Object description*

Index	2104h
Object name	Record_Ramp
PDO Mapping	ARRAY
Object code	Unsigned16

*Value description*

Sub-index	00h, number_of_records
Explanation	number of values for the object
Access	
PDO Mapping	–
Value range	1...40
Default value	40
Storable	–

---

Sub-index	28h, RampRecord
Explanation	ramp selection for record
Access	
PDO Mapping	–
Value range	0...3
Default value	0
Storable	–

*Subindex 01h...40h*

Value	Explanation
	ramp 1: <i>UpRamp1</i> (20D2h) / <i>DnRamp1</i> (20D3h)
	ramp 2: <i>UpRamp2</i> (20D4h) / <i>DnRamp2</i> (20D5h)
	ramp 3: <i>UpRamp3</i> (20D6h) / <i>DnRamp3</i> (20D7h)

**2200h** *List1\_Type*

This object specifies the type of list 1. Sub-index 00h gives information on the number of records, and the following sub-index values 01h..40h correspond to the list position number.

*Object description*

Index	2200h
Object name	List1_Type
PDO Mapping	ARRAY
Object code	Unsigned16

*Value description*

Sub-index	00h, number_of_records
Explanation	number of values for the object
Access	
PDO Mapping	–
Value range	1...64
Default value	64
Storable	–

---

Sub-index	40h, typeList1
Explanation	List 1: list type for ALL following list entries
Access	
PDO Mapping	–
Value range	1...2
Default value	1
Storable	✓

*Subindex 01h...40h*

Value	Explanation
1	position signal list
2	position speed list

**2201h List1\_Position**

This object stores the position values from list 1. Sub-index 00h gives information on the number of records, and the following sub-index values 01h..40h correspond to the list position number.

*Object description*

Index	2201h
Object name	List1_Position
PDO Mapping	ARRAY
Object code	Integer32

*Value description*

Sub-index	00h, number_of_records
Explanation	number of values for the object
Access	
PDO Mapping	—
Value range	1...64
Default value	64
Storable	—

Sub-index	40h, posList1
Explanation	List 1: position
Access	
PDO Mapping	—
Value range	-2147483648...2147483647
Default value	0
Storable	✓

**2202h List1\_Signal**

This object stores the signal values from list 1. Sub-index 00h gives information on the number of records, and the following sub-index values 01h..40h correspond to the list position number.

*Object description*

Index	2202h
Object name	List1_Signal
PDO Mapping	ARRAY
Object code	Unsigned16

*Value description*

Sub-index	00h, number_of_records
Explanation	number of values for the object
Access	
PDO Mapping	—
Value range	1...64
Default value	64
Storable	—

Sub-index	40h, signList1
Explanation	List 1: signal state
Access	
PDO Mapping	—
Value range	0...1
Default value	0
Storable	✓

**2203h List1\_Velocity**

This object stores the speed values from list 1. Sub-index00h gives information on the number of records, and the following sub-index values 01h..40h correspond to the list position number.

<i>Object description</i>	Index	2203h
	Object name	List1_Velocity
	PDO Mapping	ARRAY
	Object code	Integer32
<i>Value description</i>	Sub-index	00h, number_of_records
	Explanation	number of values for the object
	Access	
	PDO Mapping	–
	Value range	1...64
	Default value	64
	Storable	–
	Sub-index	40h, vellist1
	Explanation	List 1: setpoint speed
	Access	
<i>Value description</i>	PDO Mapping	–
	Value range	-2147483648...2147483647
	Default value	0
	Storable	✓

**2300h List2\_Type**

This object specifies the type of list 2. Sub-index00h gives information on the number of usable records, and the following sub-index values 01h..40h correspond to the record number.

<i>Object description</i>	Index	2300h
	Object name	List2_Type
	PDO Mapping	ARRAY
	Object code	Unsigned16
<i>Value description</i>	Sub-index	00h, number_of_records
	Explanation	number of values for the object
	Access	
	PDO Mapping	–
	Value range	1...64
	Default value	64
	Storable	–
	Sub-index	40h, typelist2
	Explanation	List 2: list type for ALL following list entries
	Access	
<i>Value description</i>	PDO Mapping	–
	Value range	1...2
	Default value	1
	Storable	✓



*Subindex 01h...40h*

Value	Explanation
1	position signal list
2	position speed list

**2301h List2\_Position**

This object stores the position values from list 2. Sub-index 00h gives information on the number of records, and the following sub-index values 01h..40h correspond to the list position number.

*Object description*

Index	2301h
Object name	List2_Position
PDO Mapping	ARRAY
Object code	Integer32

*Value description*

Sub-index	00h, number_of_records
Explanation	number of values for the object
Access	
PDO Mapping	–
Value range	1...64
Default value	64
Storable	–

Sub-index	40h, posList2
Explanation	List 2: position
Access	
PDO Mapping	–
Value range	-2147483648...2147483647
Default value	0
Storable	✓

**2302h List2\_Signal**

This object stores the signal values from list 2. Sub-index 00h gives information on the number of records, and the following sub-index values 01h..40h correspond to the list position number.

*Object description*

Index	2302h
Object name	List2_Signal
PDO Mapping	ARRAY
Object code	Unsigned16

*Value description*

Sub-index	00h, number_of_records
Explanation	number of values for the object
Access	
PDO Mapping	–
Value range	1...64
Default value	64
Storable	–

Sub-index	40h, signList2
Explanation	List 2: signal state
Access	
PDO Mapping	–
Value range	0...1
Default value	0
Storable	✓

### 2303h **List2\_Velocity**

This object stores the speed values from list 2. Sub-index 00h gives information on the number of records, and the following sub-index values 01h..40h correspond to the list position number.

#### Object description

Index	2303h
Object name	List2_Velocity
PDO Mapping	ARRAY
Object code	Integer32

#### Value description

Sub-index	00h, number_of_records
Explanation	number of values for the object
Access	
PDO Mapping	–
Value range	1...64
Default value	64
Storable	–

Sub-index	40h, velList2
Explanation	List 2: setpoint speed
Access	
PDO Mapping	–
Value range	-2147483648...2147483647
Default value	0
Storable	✓

### 2400h **Error\_field**

This object stores the last 20 error messages.

#### Object description

Index	2400h
Object name	Error-Field
PDO Mapping	ARRAY
Object code	Unsigned16

#### Value description

Sub-index	00h, number_of_records
Explanation	number of values for the object
Access	
PDO Mapping	–
Value range	0...20
Default value	20
Storable	–

Sub-index	14h, error number
Explanation	Coded error number
Access	
PDO Mapping	–
Value range	0...65535
Default value	0
Storable	–

### 603Fh **Error code**

This object gives the error code of the last error. The value corresponds to the lower 16 bits in the object *predefinederror field* (1003h).

#### Object description

Index	603Fh
Object name	error code
PDO Mapping	VAR
Object code	Unsigned16

#### Value description

Sub-index	00h, error code
Explanation	last error
Access	read-only
PDO Mapping	–
Value range	–
Default value	0
Storable	–

### 6040h **controlword**

This object represents the device's control word. Together with the monitoring signals of the device, the bit-coded values of the control word cause the changes of status for controlling the device in accordance with the status machine. The control word is used with the standardized operating modes in accordance with DSP 402, and is mapped in the first two bytes of PDOs R\_PDO1, R\_PDO2 and R\_PDO3.

#### Object description

Index	6040h
Object name	controlword
PDO Mapping	VAR
Object code	Unsigned16

#### Value description

Sub-index	00h, controlword
Explanation	control word for changing operating status
Access	
PDO Mapping	R_PDO
Value range	0...65535
Default value	0
Storable	–

#### Bit coding, sub-index 00h

Change of mode:

Bit	Designation	Significance for bit value 1
11..15	-	not used
9, 10	-	reserved

Bit	Designation	Significance for bit value 1
8	Halt	stop motor
7	Reset fault	reset fault
4..6	-	depends on mode, see below
3	Enable operation	execute operating mode
2	Quick Stop	brake using Quick Stop ramp
1	Disable voltage	switch off power
0	Switch on	switch on ready for operation

The significance of bits 4 to 6 depends on the operating mode currently set.

Operating mode	Bits 4 - 6	Explanation
point-to-point operation in profile position mode	Bit 4: New setpoint	0-> 1: Start positioning operation or prepare follow-on positioning
Bit 5: Change set immediately	only applies for new setpoint 0->1: 0: activate new target values when target position reached 1: activate new targets immediately	
Bit 6: Absolute / relative	0: relative positioning 1: absolute positioning	
homing in homing mode	Bit 4: Homing operation	1: Start homing start
Bits 5, 6: -	not assigned	

The status machine for describing operating states and transitions is described in the chapter on operating modes.

The *driveCtrl* object (2040h) must be used for changes of status when using manufacturer-specific operating modes.

## 6041h **statusword**

This object describes the current operating status of the device. The status word is mapped in the first two bytes of PDOs T\_PDO1, T\_PDO2 and T\_PDO3.

### Object description

Index	6041h
Object name	statusword
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, statusword
Explanation	status word for evaluating the operating status
Access	read-only
PDO Mapping	T_PDO
Value range	–
Default value	–
Storable	–

*Bit coding, sub-index 00h*

Operating states:

Bit	Designation	Explanation
12..13	-	significance dependent on operating mode, see below
11	Internal limit active	0->1: limit switch position passed
10	Target reached	1->0: New target position conveyed 0->1: Target position reached(setpoint = actual value) or motor standstill after stop request
9	Remote	0: field bus operation 1: local operation, SDOs still possible
8	-	-
7	Warning	Warning
6	Switch On disabled	not ready for operation
5	Quick Stop	Quick Stop active
4	Voltage disabled	voltage switched off
3	Fault	fault has occurred
2	Operation enabled	operating mode activated
1	Switched On	ready for operation
0	Ready to switch on	ready to be switched on

The status machine for describing operating states and transitions is described in the chapter on operating modes.

The significance of bits 12 and 13 depends on the operating mode currently set.

Operating mode	Bit 12 , 13	Explanation
point-to-point operation in profile position mode	Bit 12: setpoint acknowledge	0: new position can be accepted 1: new target position received
Bit 13: following error	1: contouring error	
homing in homing mode	Bit 12: Homing attained	1: homing performed
Bit 13: homing error	0:homing error-free	
speed operation in profile velocity mode	Bit 12: Speed=0	0: motor moving 1: motor at standstill
Bit 13: max_slippage error	0: Actual speed within tolerance	

For monitoring operating states when using manufacturer-specific operating modes, the *driveStat* object (2041h) must be used.

**6060h    *Modes of operation***

The object switches to the specified operating mode. The operating mode is changed as soon as the drive has come to a stop.

In order to switch to a manufacturer-specific mode, SIG mode must first be selected. The operating mode can then be initiated with the relevant starting object, e.g. the *startMan* object (20A9h) for manual mode.

*Object description*

Index	6060h
Object name	modes of operation
PDO Mapping	VAR
Object code	Integer8

*Value description*

Sub-index	00h, modes of operation
Explanation	setting the operating mode
Access	write-only
PDO Mapping	R_PDO
Value range	-1...127
Default value	1
Storable	–

*Bit coding, sub-index 00h*

Value	Explanation
-1 (FFh)	SIG mode, manufacturer-specific mode
1	point-to-point operation in profile position mode
3	speed operation in profile velocity mode
6	homing in homing mode

The current operating mode is shown by the *Modes of operation display* object (6061h).

**6061h    *Modes of operation display***

The object displays the current operating mode.

*Object description*

Index	6061h
Object name	modes of operation display
PDO Mapping	VAR
Object code	Integer8

*Value description*

Sub-index	00h, modes of operation display
Explanation	displaying current operating mode
Access	
PDO Mapping	T_PDO
Value range	-1...127
Default value	1
Storable	–

*Bit coding, sub-index 00h*

Value	Explanation
-1 (FFh)	SIG mode, manufacturer-specific mode
1	point-to-point operation in profile position mode

Value	Explanation
3	speed operation in profile velocity mode
6	homing in homing mode

The current operating mode is changed by means of the *Modes of operation* object (6060h).

### 6063h *Position actual value\**

This object gives the current position in increments. If the value is given in physical units, e.g. in mm, it must be converted to increments using the normalization factor of the *position factor* object (6093h) before being conveyed to the object.

The \* sign shows that the object displays position values in increments in contrast to the *Positions actual value* object (6064h) which shows values in user-defined units.

#### Object description

Index	6063h
Object name	position actual value*
PDO Mapping	VAR
Object code	Integer32

#### Value description

Sub-index	00h, position actual value*
Explanation	current position of drive in increments [inc]
Access	read-only
PDO Mapping	–
Value range	–
Default value	–
Storable	–

### 6064h *Position actual value*

This object gives the current position in user-defined units. The normalization factors of the *position factor* object (6093h) can be used to convert between user-defined units and increments.

#### Object description

Index	6064h
Object name	position actual value
PDO Mapping	VAR
Object code	Integer32

#### Value description

Sub-index	00h, position actual value
Explanation	current position of drive [usr]
Access	read-only
PDO Mapping	T_PDO
Value range	–
Default value	–
Storable	–

For standard PDO settings, the positioning controller transmits the current position in bytes 2..5 of T\_PDO2.

A target position is conveyed by the *target position* object (607Ah).

**6068h    *Position window time***

This object gives the period for which the drive must remain in the target position for the status word target reached to be reported.

*Object description*

Index	6068h
Object name	position window time
PDO Mapping	VAR
Object code	Unsigned16

*Value description*

Sub-index	00h, position window time
Explanation	Time, for which control deviations must apply in the standstill window for standstill to be signalled [ms]
Access	
PDO Mapping	–
Value range	0...32767
Default value	16
Storable	–

The object *p\_win* (2011h) specifies the standstill window within which the drive must come to a halt to enable the target position to be reported as having been reached.

**6069h    *Velocity sensor actual value***

This object returns the value of an encoder in increments. For AC actuators, this value corresponds to the current position of the rotor, and for stepping motors the signal from a connected encoder.

*Object description*

Index	6069h
Object name	velocity sensor actual value
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, velocity sensor actual value
Explanation	current speed of drive in [in/sec]
Access	read-only
PDO Mapping	T_PDO
Value range	–
Default value	–
Storable	–

**606Ch    *Velocity actual value***

The object gives information on the current speed of the drive.

*Object description*

Index	606Ch
Object name	velocity actual value
PDO Mapping	VAR
Object code	Integer32



*Value description*

Sub-index	00h, velocity actual value
Explanation	current speed of drive [usr]
Access	read-only
PDO Mapping	T_PDO
Value range	–
Default value	1
Storable	–

**607Ah    *Target position***

The object specifies the new position value for point-to-point positioning in profile position mode. The target position is approached with the set values in the movement profile.

Depending on bit 4 of the *controlword* (6040h), the target position is expressed as an absolute or relative position.

Positioning is initiated via bit 6 in the control word when point-to-point operation is activated. For standard settings, the position value is mapped in bytes 2..5 of the R\_PDO2.

*Object description*

Index	607Ah
Object name	target position
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, target position
Explanation	target position (setpoint) [usr]
Access	
PDO Mapping	R_PDO
Value range	-2147483648...2147483647
Default value	0
Storable	–

When the start command is given for a positioning operation, the drive checks to see whether the specified target position lies within the permissible work area. If this is not the case, the drive remains stationary and signals EMCY fault code FF22h: invalid position value.

**607Dh    *Software position limit***

This object gives the absolute limit switch positions SW\_LimP and SW\_LimN for the area of travel. Within these limits, the drive can be operated in point-to-point mode. The limit switch positions relate to the reference point.

*Object description*

Index	607Dh
Object name	software position limit
PDO Mapping	ARRAY
Object code	Integer32

*Value description*

Sub-index	00h, number of elements
Explanation	number of values for the object
Access	read-only
PDO Mapping	–
Value range	–
Default value	2
Storable	–
Sub-index	01h, min position limit
Explanation	lower software limit switch SW_LimN
Access	
PDO Mapping	–
Value range	-2147483648...2147483647
Default value	-2147483648
Storable	✓
Sub-index	02h, max position limit
Explanation	upper software limit switch SW_LimP
Access	
PDO Mapping	–
Value range	-2147483648...2147483647
Default value	2147483647
Storable	✓

**607Fh    *Max profile velocity***

This object gives the maximum permissible drive speed.

*Object description*

Index	607Fh
Object name	max profile velocity
PDO Mapping	VAR
Object code	Unsigned32

*Value description*

Sub-index	00h, max profile velocity
Explanation	maximum permissible profile speed [usr]
Access	
PDO Mapping	–
Value range	1...12000
Default value	3000
Storable	✓

**6081h    *Profile velocity***

This object gives the drive speed after completion of an acceleration phase. The value may not exceed the maximum speed, as defined by the *Max profile velocity* object (607Fh).

If a threshold is exceeded, the positioning controller sends an SDO error message and aborts the movement command.

*Object description*

Index	6081h
Object name	profile velocity
PDO Mapping	VAR
Object code	Unsigned32

*Value description*

Sub-index	00h, profile velocity
Explanation	speed default for positioning profile [usr]
Access	
PDO Mapping	R_PDO
Value range	-2147483648...2147483647
Default value	–
Storable	–

**6083h    *Profile acceleration***

The object specifies the acceleration after a positioning operation has been commenced in profile position mode. The value for AC actuators is limited via the maximum permitted current, the *positioncontrol parameter set* object (6510h), sub-index 01h/02h/1\_max1/1\_max2.

For manufacturer-specific operating modes, the value for the acceleration ramp is stored in the *acc* object (205E).

*Object description*

Index	6083h
Object name	profile acceleration
PDO Mapping	VAR
Object code	Unsigned32

*Value description*

Sub-index	00h, profile acceleration
Explanation	acceleration default for positioning profile [usr]
Access	
PDO Mapping	R_PDO
Value range	60...2000000
Default value	600
Storable	–

**6084h    *Profile deceleration***

The object specifies the deceleration after a positioning operation has been commenced in profile position mode. The value for AC actuators is limited via the maximum permitted current, the *positioncontrol parameter set* object (6510h), sub-index 01h/02h/1\_max1/1\_max2.

For manufacturer-specific operating modes, the value for the deceleration ramp is stored in the *dec* object (205F).

*Object description*

Index	6084h
Object name	profile deceleration
PDO Mapping	VAR
Object code	Unsigned32

*Value description*

Sub-index	00h, profile deceleration
Explanation	deceleration default for positioning profile [usr]
Access	
PDO Mapping	R_PDO
Value range	60...2000000
Default value	600
Storable	–

**6085h Quick stop deceleration**

The object specifies the deceleration for stopping the drive with Quick Stop. Quick Stop is triggered by means of bit2 =1 in the control word *controlword* (6040h).

The value for AC actuators is limited via the maximum permitted current, the *position control parameter set* object (6510h), sub-index 01h/02h *I\_max1/I\_max2*, for stepping motors via the profile deceleration, the *dec* object (6084h).

*Object description*

Index	6085h
Object name	quick stop deceleration
PDO Mapping	VAR
Object code	Unsigned32

*Value description*

Sub-index	00h, quick stop deceleration
Explanation	Deceleration ramp for quick stop [rev/(min*s)]
Access	
PDO Mapping	–
Value range	60...2000000
Default value	600
Storable	✓

**6086h Motion profile type**

This object specifies the type of selected movement profile.

*Object description*

Index	6086h
Object name	motion profile type
PDO Mapping	VAR
Object code	Integer16

*Value description*

Sub-index	00h, motion profile type
Explanation	selection for movement profile
Access	
PDO Mapping	–
Value range	-1...0
Default value	0
Storable	✓

The positioning controller supports a trapezoidal profile with linear ramps as a movement profile. For stepping motors a motor-optimized ramp can also be set.

**6093h Position factor**

This object stores the factor for position normalization which is used to convert motor increments into user-defined units, e.g. in mm.

*Object description*

Index	6093h
Object name	position factor
PDO Mapping	ARRAY
Object code	Integer32

*Value description*

Sub-index	00h, number of elements
Explanation	number of values for the object
Access	read-only
PDO Mapping	–
Value range	–
Default value	2
Storable	–

Sub-index	01h, pNormNum
Explanation	Position calibration numerator
Access	
PDO Mapping	–
Value range	-2147483648...2147483647
Default value	1
Storable	✓

Sub-index	02h, pNormDen
Explanation	Position calibration denominator
Access	
PDO Mapping	–
Value range	-2147483648...2147483647
Default value	16384
Storable	✓

**6094h    *Velocity factor***

This object stores the factor for speed normalization which is used to convert motor increments into user-defined units, e.g. in m/s.

*Object description*

Index	6094h
Object name	velocity encoder factor
PDO Mapping	ARRAY
Object code	Integer32

*Value description*

Sub-index	00h, number of elements
Explanation	number of values for the object
Access	read-only
PDO Mapping	–
Value range	–
Default value	2
Storable	–

Sub-index	01h, vNormNum
Explanation	Speed calibration numerator
Access	
PDO Mapping	–
Value range	1...2147483647
Default value	1
Storable	✓

Sub-index	02h, vNormDen
Explanation	Speed calibration denominator
Access	
PDO Mapping	–
Value range	1...2147483647
Default value	1
Storable	✓

### 6097h **Acceleration factor**

This object stores the factor for acceleration normalization which is used to convert motor increments into user-defined units, e.g. in mm/s.<sup>2</sup>

#### Object description

Index	6097h
Object name	acceleration factor
PDO Mapping	ARRAY
Object code	Integer32

#### Value description

Sub-index	00h, number of elements
Explanation	number of values for the object
Access	read-only
PDO Mapping	–
Value range	–
Default value	2
Storable	–

Sub-index	01h, aNormNum
Explanation	Acceleration calibration numerator
Access	
PDO Mapping	–
Value range	1...2147483647
Default value	1
Storable	✓

Sub-index	02h, aNormDen
Explanation	Acceleration calibration denominator
Access	
PDO Mapping	–
Value range	1...2147483647
Default value	1
Storable	✓

### 6098h **Homing method**

This object specifies the method for carrying out homing. The positioning controller supports eight standardized methods in accordance with DSP 402 via the *startSetp* object(20F6h), as well as the manufacturer-specific method of dimension setting in which the reference point is defined by the allocation of the current position value.

The standardized methods can also be initiated via the manufacturer-specific object, *startHome* (20F0h).

*Object description*

Index	6098h
Object name	homing method
PDO Mapping	VAR
Object code	Integer8

*Value description*

Sub-index	00h, homing method
Explanation	homing method
Access	
PDO Mapping	–
Value range	0...255
Default value	18
Storable	–

*Homing method*

	Referencing to...
Method 1	Limit switch $\overline{\text{LIMN}}$ with index pulse
Method 2	Limit switch $\overline{\text{LIMP}}$ with index pulse
Method 7	reference switch $\overline{\text{REF}}$ with index pulse, firstmovement in clockwise direction
Method 11	reference switch $\overline{\text{REF}}$ with index pulse, firstmovement in anti-clockwise direction
Method 17	Limit switch $\overline{\text{LIMN}}$ no index pulse
Method 18	Limit switch $\overline{\text{LIMP}}$ no index pulse
Method 23	reference switch $\overline{\text{REF}}$ no index pulse, firstmovement in clockwise direction
Method 27	reference switch $\overline{\text{REF}}$ no index pulse, firstmovement in anti-clockwise direction
Method 35	dimension setting to current position with <i>startSetp</i> object(20F6h)

Homing speeds for searching for the reference point and for travelling to it are determined via the *homing speed* object(6099h).

Homing by methods 1 to 27 is initiated via bit 4 of the controlword *controlword* (6040h). Bits 12 and 13 in the status word *statusword* (6041h) give information on the operating status of the homing operation. You will find details on homing procedures in the chapter on operating modes.

**6099h Homing speeds**

This object defines the speeds for homing operations.

*Object description*

Index	6099h
Object name	homing speeds
PDO Mapping	ARRAY
Object code	Integer32

*Value description*

Sub-index	00h, number of elements
Explanation	number of objects for the value
Access	read-only
PDO Mapping	–
Value range	–
Default value	2
Storable	–
Sub-index	01h, speed during search for switch
Explanation	Speed for search of reference switch [usr]
Access	
PDO Mapping	–
Value range	-2147483648...2147483647
Default value	60
Storable	✓
Sub-index	02h, speed during search for zero
Explanation	Freewheel speed to 'p_outHome' position [usr]
Access	
PDO Mapping	–
Value range	-2147483648...2147483647
Default value	6
Storable	✓

**60F4h** *Following error actual value*

This object gives the current contouring error in user-defined units.

The interval after which the positioning controller reports a contouring error, can be set using the *following error timeout* object (6066h).

*Object description*

Index	60F4h
Object name	following error actual value
PDO Mapping	VAR
Object code	Integer32

*Value description*

Sub-index	00h, following error actual value
Explanation	current contouring error
Access	
PDO Mapping	–
Value range	-2147483648...2147483647
Default value	0
Storable	–

The current contouring error can be read in increments via the *p\_dif* object (2073h). The maximum permissible contouring error is set by means of the *p\_maxDiff* object (2010h), and the largest contouring error recorded can be determined via the *p\_difPeak* object (2013h).

**60FBh** *position control parameter set*

This object stores the values of both control parametersets for positioning controllers for driving AC actuators. The settings can be optimi-



zed during set-up with the operating software TL CT. The control parameters and the optimization steps are described in the manual.

*Object description*

Index	60FBh
Object name	position control parameter set
PDO Mapping	RECORD
Object code	Unsigned16

*Value description*

Sub-index	00h, number of elements
Explanation	number of values for the object
Access	read-only
PDO Mapping	–
Value range	–
Default value	24
Storable	–

Sub-index	01h, I_max1
Explanation	Parameter set 1: Current limit in all operating modes apart from manual and quick stop [100=1A]
Access	
PDO Mapping	–
Value range	0...29999
Default value	1000
Storable	✓

Sub-index	02h, I_max2
Explanation	Parameter set 2: Current limit in all operating modes apart from manual and quick stop (100=1A)
Access	
PDO Mapping	–
Value range	0...29999
Default value	1000
Storable	✓

Sub-index	03h, n_max1
Explanation	Parameter set 1: Max. speed [rpm]
Access	
PDO Mapping	–
Value range	0...12000
Default value	6000
Storable	✓

Sub-index	04h, n_max2
Explanation	Parameter set 2: Max. speed [rpm]
Access	
PDO Mapping	–
Value range	0...12000
Default value	6000
Storable	✓

Sub-index	05h, KPn1
Explanation	Parameter set 1: Speed controller P-factor [1000=A*min/rev]
Access	
PDO Mapping	–
Value range	0...32767
Default value	10
Storable	✓

---

Sub-index	06h, KPn2
Explanation	Parameter set 2: Speed controller P-factor (1000=1Amin/rev)
Access	
PDO Mapping	–
Value range	0...32767
Default value	10
Storable	✓

---

Sub-index	07h, TNn1
Explanation	Parameter set 1: Speed controller integral time I-factor [100=1ms]
Access	
PDO Mapping	–
Value range	0...32767
Default value	500
Storable	✓

---

Sub-index	08h, TNn2
Explanation	Parameter set 2: Speed controller integral time I-factor (100=1ms)
Access	
PDO Mapping	–
Value range	0...32767
Default value	500
Storable	✓

---

Sub-index	09h, TVn1
Explanation	Parameter set 1: Speed controller derivative time D-factor [100=1ms]
Access	
PDO Mapping	–
Value range	0...32767
Default value	0
Storable	✓

---

Sub-index	0Ah, TVn2
Explanation	Parameter set 2: Speed controller derivative time D-factor (100=1ms)
Access	
PDO Mapping	–
Value range	0...32767
Default value	0
Storable	✓

Sub-index	0Bh, KFPn1
Explanation	Parameter set 1: Speed controller feed forward control P-factor [100=1A*min/rev]
Access	
PDO Mapping	–
Value range	0...32767
Default value	0
Storable	✓

---

Sub-index	0Ch, KFPn2
Explanation	Parameter set 2: Speed controller feed forward control P-factor (100=1A*min/rev)
Access	
PDO Mapping	–
Value range	0...32767
Default value	0
Storable	✓

---

Sub-index	0Dh, KFDn1
Explanation	Parameter set 1: Speed controller feed forward control D-factor [10 000=1mAs*min/rev]
Access	
PDO Mapping	–
Value range	0...32767
Default value	0
Storable	✓

---

Sub-index	0Eh, KFDn2
Explanation	Parameter set 2: Speed controller feed forward control D-factor (10 000=1mAs*min/rev)
Access	
PDO Mapping	–
Value range	0...32767
Default value	0
Storable	✓

---

Sub-index	0Fh, KPp1
Explanation	Parameter set 1: Position controller P-factor [10=1/s]
Access	
PDO Mapping	–
Value range	0...32767
Default value	14
Storable	✓

---

Sub-index	10h, KPp2
Explanation	Parameter set 2: Position controller P-factor [1/s]
Access	
PDO Mapping	–
Value range	0...32767
Default value	14
Storable	✓

Sub-index	11h, TVp1
Explanation	Parameter set 1: Position controller derivative time D-factor (100=1ms)
Access	
PDO Mapping	–
Value range	0...32767
Default value	0
Storable	✓
Sub-index	12h, TVp2
Explanation	Parameter set 2: Position controller derivative time D-factor (100=1ms)
Access	
PDO Mapping	–
Value range	0...32767
Default value	0
Storable	✓
Sub-index	13h, KFPp1
Explanation	Parameter set 1: Position controller feed forward control speed [-]
Access	
PDO Mapping	–
Value range	0...32767
Default value	100
Storable	✓
Sub-index	14h, KFPp2
Explanation	Parameter set 2: Position controller feed forward control speed [-]
Access	
PDO Mapping	–
Value range	0...32767
Default value	100
Storable	✓
Sub-index	15h, KFAp1
Explanation	Parameter set 1: Speed controller feed forward control acceleration (10 000=1As*min/rev)
Access	
PDO Mapping	–
Value range	0...32767
Default value	0
Storable	✓
Sub-index	16h, KFAp2
Explanation	Parameter set 2: Speed controller feed forward control acceleration (10 000=1mAs*min/rev)
Access	
PDO Mapping	–
Value range	0...32767
Default value	0
Storable	✓

Sub-index	17h, Filt_nRef1
Explanation	Parameter set 1: Filter time constant reference variable filter of the setpoint speed (100=1ms)
Access	
PDO Mapping	–
Value range	0...32767
Default value	0
Storable	✓

---

Sub-index	18h, Filt_nRef2
Explanation	Parameter set 2: Filter time constant reference variable filter of the setpoint speed (100=1ms)
Access	
PDO Mapping	–
Value range	0...32767
Default value	0
Storable	✓

## 60FDh **Digital inputs**

The object gives the signal states of the device's digital inputs.

### Object description

Index	60FDh
Object name	digital inputs
PDO Mapping	VAR
Object code	Unsigned32

### Value description

Sub-index	00h, digital inputs
Explanation	signal status of digital inputs
Access	read-only
PDO Mapping	T_PDO
Value range	–
Default value	–
Storable	–

### Bit coding, sub-index 00h

The input states are given bit coded. The value 1 corresponds to the signal state High and signifies: input triggered

Bit	input	Bit	input
31, 30	-		
29	D32		
28	D16		
27	D8		
26	D4		
25	D2		
24	D1		
23	Teach-In		
22	Start/ CAPTURE 2		

Bit	input	Bit	input
21	FAULT_RE SET/ CAPTURE 1		
20	AUTOM	15..4	reserved
19	Enable	3	$\overline{\text{STOP}}$
18	MAN_FAS T	2	$\overline{\text{REF}}$
17	MAN_P	1	$\overline{\text{LIMP}}$
16	MAN_N	0	$\overline{\text{LIMN}}$

**60FEh** *Digital outputs*

The object gives the signal states of the device's digital outputs.

*Object description*

Index	60FEh
Object name	digital outputs
PDO Mapping	RECORD
Object code	Unsigned32

*Value description*

Sub-index	00h, number of elements
Explanation	number of values for the object
Access	read-only
PDO Mapping	–
Value range	–
Default value	2
Storable	–

Sub-index	01h, physical outputs
Explanation	signal interface outputs
Access	
PDO Mapping	R_PDO
Value range	0...4294967295
Default value	0
Storable	–

Sub-index	02h, bitmask
Explanation	enabling and disabling device outputs
Access	
PDO Mapping	–
Value range	0...4294967295
Default value	0
Storable	–

*Bit coding, sub-index 01h*

The output states are given bit coded. The value 1 corresponds to the signal state High and signifies: output triggered.

Bit	output
31..23	-
22	Alarm/ TRIGGER

Bit	output
21	ACTIVE_C ON
20	RDY_TSO
19	FUNCT_O UT, Fault, Q3
18	Q2
17	Q1
16	Q0
15..0	-

*Bit coding, sub-index 02h* If output triggering via sub-index 01h is selected, the bit mask must be enabled to allow the outputs to be triggered.

Value	Explanation
0	do not switch outputs
1	switch outputs

## 60FFh **Traget velocity**

This object initiates speed mode or changes the speed in Profile velocity mode by conveying a new set speed in user-defined units. When the value is transferred, the current ramp settings of the movement profile are adopted.

This object is mapped in bytes 2 to 5 of the R\_PDO2 for standard settings.

### *Object description*

Index	60FFh
Object name	target velocity
PDO Mapping	VAR
Object code	Integer32

### *Value description*

Sub-index	00h, target velocity
Explanation	set speed
Access	
PDO Mapping	R_PDO
Value range	-2147483648...2147483647
Default value	0
Storable	–

## 6404h **Motor manufacturer**

The object gives the manufacturer's name of the connected motor.

### *Object description*

Index	6404h
Object name	motor manufacturer
PDO Mapping	VAR
Object code	Visible String16

*Value description*

Sub-index	00h, motor manufacturer
Explanation	motor manufacturer
Access	read-only
PDO Mapping	–
Value range	–
Default value	"SIG Positec BL"
Storable	✓

**6410h    *Motor data***

The object gives the technical data and settings for the connected AC actuator.

*Object description*

Index	6410h
Object name	motor data
PDO Mapping	RECORD
Object code	User Defined

*Value description*

Sub-index	00h, number of elements
Explanation	number of values for the object
Access	read-only
PDO Mapping	–
Value range	–
Default value	7
Storable	–

Sub-index	01h, TypeM
Explanation	Motor type, consecutive numbers
Access	
PDO Mapping	–
Value range	-2147483648...2147483647
Default value	–
Storable	✓

Sub-index	05h, n_nomM
Explanation	Nominal motor speed [rpm]
Access	
PDO Mapping	–
Value range	0...12000
Default value	3000
Storable	✓

Sub-index	06h, I_maxM
Explanation	Max. motor current [100=1A]
Access	
PDO Mapping	–
Value range	0...32767
Default value	1000
Storable	✓



Sub-index	07h, I_nomM
Explanation	Nominal motor current [100=1A]
Access	
PDO Mapping	–
Value range	0...32767
Default value	100
Storable	✓

---

Sub-index	08h, M_nomM
Explanation	Nominal torque [Ncm]
Access	
PDO Mapping	–
Value range	0...32767
Default value	100
Storable	✓

---

Sub-index	28h, nameM
Explanation	Motor name
Access	
PDO Mapping	–
Value range	–
Default value	–
Storable	✓

---

Sub-index	29h, I_0M
Explanation	Motor continuous current at standstill (100=1A)
Access	
PDO Mapping	–
Value range	1...32767
Default value	100
Storable	✓

*Sub-index 01h:* Negative values signify a resolver motor, positive a Sincodermotor.

## 6510h **Drive data**

This object gives the technical data and settings for the positioning controller's power amplifier.

### *Object description*

Index	6510h
Object name	drive data
PDO Mapping	RECORD
Object code	User Defined

### *Value description*

Sub-index	00h, number of elements
Explanation	number of values for the object
Access	read-only
PDO Mapping	–
Value range	–
Default value	2
Storable	–

Sub-index	01h, I_maxPA
Explanation	Peak current of the unit [100=1A]
Access	read-only
PDO Mapping	–
Value range	–
Default value	1000
Storable	–

---

Sub-index	02h, I_nomPA
Explanation	Nominal current of the unit [100=1A]
Access	read-only
PDO Mapping	–
Value range	–
Default value	300
Storable	–

---

## Index

### Numerics

1st receive PDO mapping 3-17  
 1st receive PDO parameter 3-15  
 1st transmit PDO mapping 3-17  
 1st transmit PDO parameter 3-15  
 2nd receive PDO mapping 3-17  
 2nd receive PDO parameter 3-15  
 2nd transmit PDO mapping 3-18  
 2nd transmit PDO parameter 3-15  
 3rd receive PDO mapping 3-17  
 3rd receive PDO parameter 3-15  
 3rd transmit PDO mapping 3-18  
 3rd transmit PDO parameter 3-15  
 4th receive PDO mapping 3-17  
 4th receive PDO parameter 3-15  
 4th transmit PDO mapping 3-18  
 4th transmit PDO parameter 3-16

### A

Absolute positioning 5-12, 5-18  
*acc* 6-5  
*acc\_ref* 5-25  
*acc\_type* 6-5  
 acceleration factor 5-16, 5-20, 6-4  
*accOffs* 5-32  
*ActCtrl* 5-24  
*action\_st* 6-9  
 Activating PDOs 3-16  
*actList1* 6-1  
*actList2* 6-1  
 acyclical transmission 3-21  
 Addressing 4-1  
*AnalogIn* 6-10

### B

*bgnList1* 6-1  
*bgnList2* 6-1  
 bleed resistor control TL\_BRC 6-9  
 bleed resistor, external 6-9  
 Boot-Up message 3-22  
 broadcast transmission 3-4  
 Bus Arbitration 3-3  
 Bus termination, see termination

### C

CAN field bus  
     field bus devices 2-1  
     Operating modes 2-2  
     operating modes 2-1  
 CAN Message 3-2  
 CANopen

- Documentation on 1-5
- Make-up of a message 3-3
- CANopen example
  - Positioning with PDO1, PDO2 5-16, 5-21
  - switching operating status 5-7
- Capturing position values 6-7
- ccd (command code), see command code
- cntList1* 6-1
- cntList2* 6-1
- COB Identifier (COB-ID) 3-3
- COB-ID emergency* 3-23
- Command code of an SDO message 3-8
- Communication cycle period* 3-20
- Communication objects, Overview 3-2
- Communication profile 3-1
- Communication relationship
  - Client-Server 3-5
  - Master-Slave 3-5
  - Producer-Consumer 3-6
- Connection monitoring 3-26
- continue* 5-29
- Contouring error
  - Monitoring function 6-9
- controlword*
  - Bit 4 - setting the type of positioning 5-12, 5-18
- Cyclical transmission 3-21
- D**
  - dec* 6-5
  - decOffs* 5-32
  - default* 6-10
  - denGear* 5-31
  - digital outputs* 6-10
  - digital\_inputs* 6-10
  - direction of rotation, invert 6-6
  - Direction of rotation, inverting 6-6
  - dirEnGear* 5-31
  - dist\_Man* 5-28
  - Dynamic PDO mapping 3-17
- E**
  - eeprSave* 6-10
  - EMC measures 4-1
  - EMCY message 3-22
  - endList1* 6-1
  - endList2* 6-1
  - Error code
    - EMCY message 3-22
  - error code* 6-9, 7-4
  - error field* 3-23
  - Error register* 7-4
  - Error register of an EMCY message 3-22
  - Errors

- Diagnosis via field bus 7-3
- Establishing field bus communication, Diagnostics 7-1
- Reporting objects 7-3
- Signals from the device 7-3

**F**

- Fault response 5-5
- Filt\_jerk* 6-5
- FltSig* 6-9
- FltSig\_SR* 6-9

**G**

- Guard time* 3-26

**H**

- Heartbeat 3-27
- homing speeds* 5-33
- homing, Function 5-22

**I**

- I\_act* 5-25
- I\_maxMan* 5-27
- I\_maxSTOP* 6-6
- I\_maxSTOP* in Profile position mode 5-15
- I\_ref* 5-25
- I2tB\_act* 5-25
- I2tM\_act* 5-25
- I2tPA\_act* 5-25
- Index 3-1
- Initiating speed mode 5-18
- Installation
  - Addressing 4-1
  - Connection of the Twin Line unit 4-3
  - Device installation 4-1
- Interface signal
  - TRIGGER 6-1
- interface signal
  - STOP 6-8
- Interrogating current position 5-12, 5-18
- IntSigSr* 6-8, 6-9
- invertDir* 6-7
- IO\_mode* 5-24

**J**

- jerk filter 6-5

**L**

- Life guarding, see connection monitoring
- Life time factor* 3-26
- Limit switches
  - monitoring function 6-8
  - software limit switches 6-8
- List control 6-1
- List1\_Position* 6-2
- List1\_Signal* 6-2
- List1\_Type* 6-2, 6-3

*List1\_Velocity* 6-2  
*List2\_Position* 6-2  
*List2\_Signal* 6-2  
*List2\_Type* 6-2, 6-3  
*List2\_Velocity* 6-2

**M**

Master  
     NMT Master 3-23  
*max profile velocity* 5-14, 5-19  
*memNrTeac* 6-3  
 Monitoring parameters 6-9  
*motion profile type* 5-15  
 Motor stop 6-9  
 motor-start-PDO 3-21

**N**

*n\_act* 5-25  
*n\_fastMan* 5-27  
*n\_max0* 6-5  
*n\_maxGear* 5-31  
*n\_ref* 5-25  
*n\_refOffs* 5-26  
*n\_slowMan* 5-27  
*n\_start0* 6-5  
*n\_tarOffs* 5-32  
 Network management services, see NMT  
 Network management, see NMT 3-2  
 NMT  
     NMT Services 3-23  
     NMT Status machine 3-24  
*NMT error control* 3-27  
 node address 3-4  
 Node guarding, see connection monitoring  
 Node-ID, see node address 3-4  
 Normalization factors 6-4  
 normalization, residual value of 6-4  
*number of PDOs supported* 3-13  
*numGear* 5-31

**O**

Object dictionary  
     Set-up 3-1  
*OnIAuto* 5-24  
 Operating mode  
     homing 5-22  
     initiating 5-10  
     monitoring 5-10  
     Overview 5-1  
     Point-to-point operation as "profile position mode" 5-12  
     Speed operation as "profile velocity mode" 5-18  
     Status monitoring with *x\_add\_info*, *x\_end*, *x\_err* 5-11  
 Operating status  
     monitor 5-7

Overview of all states 5-4

## P

*p\_abs* 5-25

*p\_absOffs* 5-31

*p\_absPTP* 3-14, 5-29

*p\_act* 5-25

*p\_actTeac* 6-3

*p\_actusr* 3-15, 5-26

*p\_dif* 5-25

*p\_disHome* 5-33

*p\_jerkusr* 5-26

*p\_maxDiff* 6-9

*p\_outHome* 5-33

*p\_ref* 5-25

*p\_refGear* 5-25

*p\_refOffs* 5-26

*p\_relOffs* 5-32

*p\_relPTP* 5-29

*p\_remaind* 5-26, 6-4

*p\_target* 5-26

*p\_tarOffs* 5-26

*p\_win* 6-6

*p\_winTime* 6-6

Parameter groups 5-25

PBDP+interface, connection 4-3

PDO mapping 3-17

    changing entries 3-18

    objects 3-18

    significance of sub-index values 3-18

PDO mapping, static and dynamic 3-17

PDO time intervals 3-16

*phomeOffs* 5-32

*position actual value* 5-12

*position control parameter set* 5-15

*position factor* 5-16, 6-4

Positioning drive

    documentation on 1-5

    Documentation on the 1-5

Positioning limits 6-8

Positioning mode

    Initiating positioning 5-12

    Setting relative and absolute positioning 5-12, 5-18

positioning mode

    Interrogating current position 5-12, 5-18

Potential equalization lines 4-1

Pre-operational, NMT status 3-24

Process Data Object PDO 3-11

Profibus+DP

    Connecting the Twin Line unit 4-3

*profile acceleration* 5-14, 5-19

*profile deceleration* 5-14, 5-19

*profile velocity* 5-14

**Q**

*quick stop deceleration* im Profile position mode 5-15

*quick stop deceleration* im Profile velocity mode 5-20

Quick Stop function 6-6

*quick\_stop\_deceleration* 6-6

**R**

*ramp* 3-14

Relative positioning 5-12, 5-18

Repeaters 4-4

Reset application, NMT status 3-24

Residual value 6-4

rror 7-4

RTR-Bit, Requesting data 3-12

**S**

SDO message

Evaluating error messages 7-4

Index and sub-index 3-8

overview 3-8

Service address 8-1

Service Data Object SDO 3-7

*SetCtrl* 5-24

signal interface, addressing via 4-1

Signals from the device 7-3

*SignEnabl* 6-8

*SignLevel* 6-8

*SignQstop* 6-6

Slave

determine NMT status 3-27

NMT Slave 3-23

software limit switches 6-8

*software position limit* 6-8

*software position limit* im Profile position mode 5-15

Standardized objects 3-2

Standstill window 6-6

Start remote node, NMT change of status 3-24

*startGear* 5-31

*startHome* 5-33

*startList* 6-1

*startMan* 5-27

*startSetpin* homing operating mode 5-33

*stateGear* 5-31

*stateHome* 5-33

*stateList* 6-1

*stateMan* 5-27

*stateOffs* 5-31

*statePTP* 3-15, 5-29

*stateTeac* 6-3

*stateVEL* 5-28

static PDO mapping 3-17

Status machine

Changing and monitoring operating states 5-3



- Flow chart 5-4
- NMT Status Machine 3-24
- Status word 5-7
- statusword*
  - bits 0..3, 5, 6 - monitoring operating states 5-7
  - Bits 10..15 - monitoring operating mode 5-10
- stepMan* 5-28
- Stopped, NMT status 3-24
- storeTeac* 6-3
- Sub-index 3-1
- SW\_Enabl* 6-8
- SW\_LimN* 6-8
- SW\_LimN* im Profile position mode 5-15
- SW\_LimP* 6-8
- SW\_LimP* im Profile position mode 5-15
- SYNC object 3-20
- Synchronization 3-20
- Synchronous window length* 3-20

**T**

- target position* 5-14
- Teach-In
  - Overview 6-3
- Termination 4-4
- The Client-Server relationship 3-5
- The Master-Slave relationship 3-5
- The Producer-Consumer relationship 3-6
- time interval „inhibit time“ 3-16
- time intervall „event timer“ 3-16
- time\_Man* 5-28
- TL\_BRC* 6-9
- TM\_act* 5-25
- TPA\_act* 5-25
- Transition
  - Overview of all transitions 5-5
- trigger channels 6-7
- TrigLevl* 6-7
- TrigPact1* 6-7
- TrigPact2* 6-7
- TrigPref1* 6-7
- TrigPref2* 6-7
- TrigSign* 6-7
- TrigStart* 6-7
- TrigStat* 6-7
- TrigType* 6-7
- typeMan* 5-27

**U**

- UDC\_act* 5-25

**V**

- v\_jerkusr* 3-15, 5-26
- v\_ref* 5-25
- v\_refGear* 5-25

*v\_target* 5-26  
*v\_target0* 6-5  
*v\_targetr* 3-14  
*v\_tarPTP* 5-29  
*velocity* 5-28  
*velocity actual value* 5-18  
*velocity encoder factor* 5-16, 5-20  
*velocity factor* 6-4

**X**

*xMode* 5-25  
*xMode\_act* 5-25