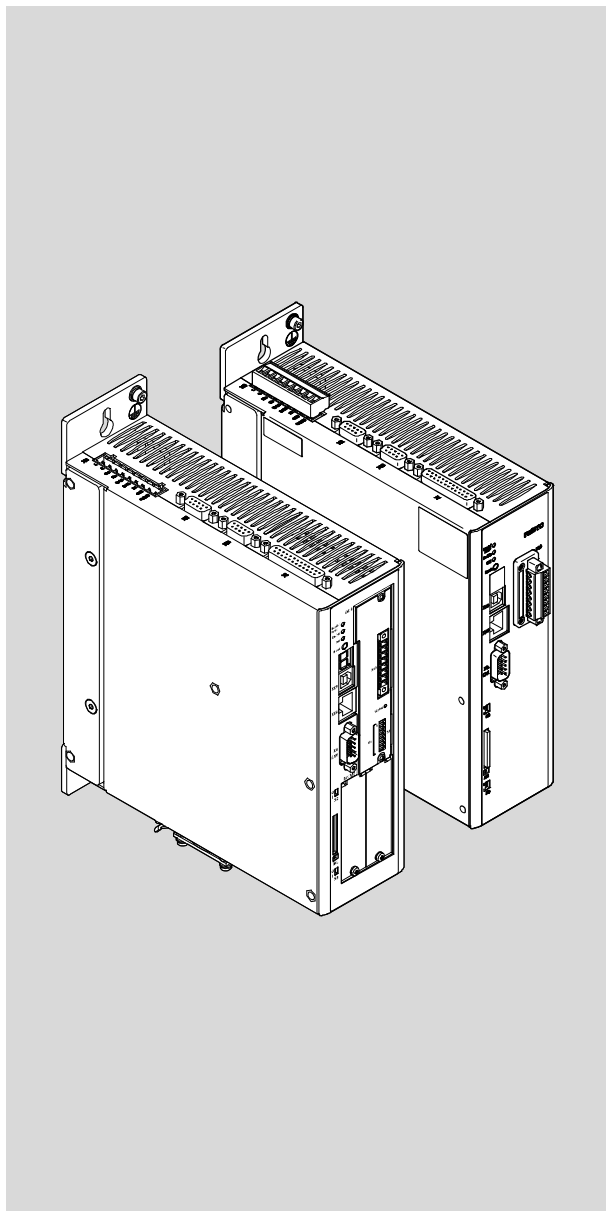


CiA 402 for motor controller

CMMP-AS-...-M3/-M0



FESTO

Description

Device profile
CiA 402

for motor controller
CMMP-AS-...-M3
via fieldbus:
– CANopen
– EtherCAT
with interface
CAMC-EC

for motor controller
CMMP-AS-...-M0
via fieldbus:
– CANopen

8046795
1510b

Translation of the original instructions
GDCP-CMMP-M3/-M0-C-CO-EN

CANopen®, CiA®, EthetCAT®, TwinCAT® are registered trademarks of the respective trademark owners in certain countries.

Identification of hazards and instructions on how to prevent them:



Danger

Immediate dangers which can lead to death or serious injuries



Warning

Hazards that can cause death or serious injuries



Caution

Hazards that can cause minor injuries or serious material damage

Other symbols:



Note

Material damage or loss of function



Recommendations, tips, references to other documentation



Essential or useful accessories



Information on environmentally sound usage

Text designations:

- Activities that may be carried out in any order
- 1. Activities that should be carried out in the order stated
- General lists
- ➔ Result of an action/References to more detailed information

Table of Contents – CMMP-AS-...-M3/-M0

Instructions on this documentation	7
Target group	7
Service	7
Information on the version	7
Documentation	8
1 Fieldbus interfaces	9
2 CANopen [X4]	10
2.1 General information on CANopen	10
2.2 Cabling and pin assignment	11
2.2.1 Pin allocations	11
2.2.2 Cabling instructions	11
2.3 Configuration of CANopen stations on the CMMP-AS-...-M3	13
2.3.1 Setting of the node number	14
2.3.2 Setting of the transmission rate with DIP switches	15
2.3.3 Activation of CANopen communication with DIP switches	15
2.3.4 Setting the physical units (factor group)	15
2.4 Configuration of CANopen participants on the CMMP-AS-...-M0	16
2.4.1 Setting the node number via DINs and FCT	17
2.4.2 Setting the transmission rate via DINs or FCT	17
2.4.3 Setting the protocol (data profile) via DINs or FCT	18
2.4.4 Activation of CANopen communication via DINs or FCT	18
2.4.5 Setting the physical units (factor group)	19
2.5 Configuration CANopen master	19
3 CANopen access procedure	20
3.1 Introduction	20
3.2 SDO Access	21
3.2.1 SDO Sequences for Reading and Writing	22
3.2.2 SDO Error Messages	23
3.2.3 Simulation of SDO access	24
3.3 PDO Message	25
3.3.1 Description of the Objects	26
3.3.2 Objects for PDO Parametrisation	29
3.3.3 Activation of PDOs	34
3.4 SYNC message	35
3.5 EMERGENCY Message	36
3.5.1 Overview	36
3.5.2 Structure of the EMERGENCY Message	37

3.5.3	Description of the Objects	37
3.6	Network Management (NMT Service)	39
3.7	Bootup	41
3.7.1	Overview	41
3.7.2	Structure of the Bootup Message	41
3.8	Heartbeat (Error Control Protocol)	42
3.8.1	Overview	42
3.8.2	Structure of the Heartbeat Message	42
3.8.3	Description of the Objects	42
3.9	Nodeguarding (Error Control Protocol)	43
3.9.1	Overview	43
3.9.2	Structure of the Nodeguarding Messages	43
3.9.3	Description of the Objects	44
3.9.4	Object 100Dh: life_time_factor	45
3.9.5	Table of Identifiers	45
4	EtherCAT with CoE	46
4.1	Overview	46
4.2	EtherCat-Interface CAMC-EC	46
4.3	Installing the EtherCAT interface in the controller	48
4.4	Pin allocation and cable specifications	48
4.5	Configuration of EtherCAT participants	50
4.5.1	Setting of the physical units of measure (Factor Group)	50
4.6	CANopen communication interface	51
4.6.1	Configuration of the Communication Interface	51
4.6.2	New and revised objects under CoE	54
4.6.3	Objects not supported under CoE	61
4.7	Communication Finite State Machine	62
4.7.1	Differences between the finite state machines of CANopen and EtherCAT	64
4.8	SDO Frame	65
4.9	PDO Frame	66
4.10	Error Control	68
4.11	Emergency Frame	68
4.12	XML Device Description File	69
4.12.1	Fundamental structure of the device description file	69
4.12.2	Receive PDO configuration in the RxPDO node	71
4.12.3	Transmit PDO configuration in the TxPDO node	73
4.12.4	Initialisation commands via the “Mailbox” node	73
4.13	Synchronisation (Distributed Clocks)	74
5	Setting parameters	75
5.1	Loading and Saving Parameter Sets	75

5.2	Compatibility settings	78
5.3	Conversion factors (factor group)	80
5.4	Output stage parameter	90
5.5	Current Regulator and Motor Adjustment	96
5.6	Speed Control	103
5.7	Position Controller (Position Control Function)	105
5.8	Setpoint value limitation	116
5.9	Encoder Adjustments	118
5.10	Incremental Encoder Emulation	122
5.11	Setpoint/Actual Value Activation	124
5.12	Analogue inputs	127
5.13	Digital inputs and outputs	129
5.14	Limit Switch/Reference Switch	134
5.15	Sampling of Positions	137
5.16	Brake Activation	140
5.17	Device Information	141
5.18	Error Management	148
6	Device Control	150
6.1	Status Diagram (State Machine)	150
6.1.1	Overview	150
6.1.2	Status diagram of the motor controller (state machine)	151
6.1.3	Control word (controlword)	156
6.1.4	Read-out of the motor controller status	159
6.1.5	Status words (statuswords)	160
6.1.6	Description of the additional objects	170
7	Operating modes	173
7.1	Setting the operating mode	173
7.1.1	Overview	173
7.1.2	Description of the Objects	173
7.2	Operating mode reference travel (homing mode)	175
7.2.1	Overview	175
7.2.2	Description of the Objects	176
7.2.3	Reference Travel Processes	180
7.2.4	Control of Reference Travel	184
7.3	Positioning Operating Mode (Profile Position Mode)	186
7.3.1	Overview	186
7.3.2	Description of the objects	187
7.3.3	Description of function	190
7.4	Synchronous position specification (interpolated position mode)	193
7.4.1	Overview	193

7.4.2	Description of the Objects	193
7.4.3	Description of function	199
7.5	Speed Adjustment Operating Mode (Profile Velocity Mode)	201
7.5.1	Overview	201
7.5.2	Description of the Objects	203
7.6	Speed ramps	209
7.7	Torque Regulation Operating Mode (Profile Torque Mode)	212
7.7.1	Overview	212
7.7.2	Description of the Objects	213
A	Technical appendix	218
A.1	Technical Data Interface EtherCAT	218
A.1.1	General	218
A.1.2	Operating and environmental conditions	218
B	Diagnostic messages	219
B.1	Explanations on the diagnostic messages	219
B.2	Error codes via CiA 301/402	220
B.3	Diagnostic messages with instructions for fault clearance	223
Index	282

Instructions on this documentation

This documentation describes the device profile CiA 402 (DS 402) for the motor controllers CMMP-AS-...-M3/-M0 conforming to the section “Information on the version” through the fieldbus interfaces:

- CANopen – interface [X4] integrated into the motor controller.
- EtherCAT – optional interface CAMC-EC in the slot Ext2, only for CMMP-AS-...-M3.

This provides you with supplementary information about control, diagnostics and parametrisation of the motor controllers via the fieldbus.

- Unconditionally observe the general safety regulations for the CMMP-AS-...-M3/-M0.



The general safety regulations for the CMMP-AS-...-M3/-M0 can be found in the hardware description, GDCP-CMMP-AS-M3-HW-... or GDCP-CMMP-AS-M0-HW-..., see Tab. 2.

Target group

This description is intended exclusively for technicians trained in control and automation technology, who have experience in installation, commissioning, programming and diagnosing of positioning systems.

Service

Please consult your regional Festo contact if you have any technical problems.

Information on the version

This description refers to the following versions:

Motor controller	Version
CMMP-AS-...-M3	Motor controller CMMP-AS-...-M3 from Rev 01
	FCT plug-in CMMP-AS from Version 2.0.x.
CMMP-AS-...-M0	Motor controller CMMP-AS-...-M0 from Rev 01
	FCT plug-in CMMP-AS from Version 2.0.x.

Tab. 1 Versions



This description does not apply to the older variants CMMP-AS-.... Use the assigned CANopen description for the motor controller CMMP-AS for these variants.



Note

With newer firmware versions, check whether there is a newer version of this description available: → www.festo.com/sp

Documentation

You will find additional information on the motor controller in the following documentation:

User documentation on the motor controller CMMP-AS-...-M3/-M0	
Name, type	Contents
Hardware description, GDCP-CMMP-M3-HW-...	Mounting and installation of the motor controller CMMP-AS-...- M3 for all variants/output classes (1-phase, 3-phase), pin assignments, error messages, maintenance.
Description of functions, GDCP-CMMP-M3-FW-...	Functional description (firmware) CMMP-AS-...- M3 , instructions on commissioning.
Hardware description, GDCP-CMMP-M0-HW-...	Mounting and installation of the motor controller CMMP-AS-...- M0 for all variants/output classes (1-phase, 3-phase), pin assignments, error messages, maintenance.
Description of functions, GDCP-CMMP-M0-FW-...	Functional description (firmware) CMMP-AS-...- M0 , instructions on commissioning.
Description of FHPP, GDCP-CMMP-M3/-M0-C-HP-...	Control and parameterisation of the motor controller via the FHPP Festo profile. <ul style="list-style-type: none"> – Motor controller CMMP-AS-...-M3 with the following fieldbuses: CANopen, Modbus TCP, PROFINET, PROFIBUS, EtherNet/IP, DeviceNet, EtherCAT. – Motor controller CMMP-AS-...-M0 with fieldbuses CANopen, Modbus TCP.
Description of CiA 402 (DS 402), GDCP-CMMP-M3/-M0-C-CO-...	Control and parameterisation of the motor controller via the device profile CiA 402 (DS402) <ul style="list-style-type: none"> – Motor controller CMMP-AS-...-M3 with the following fieldbuses: CANopen and EtherCAT. – Motor controller CMMP-AS-...-M0 with fieldbus CANopen.
Description of CAM editor, P.BE-CMMP-CAM-SW-...	Cam disc function (CAM) of the motor controller CMMP-AS-...- M3/-M0 .
Description of the safety module, GDCP-CAMC-G-S1-...	Functional safety engineering for the motor controller CMMP-AS-...- M3 with the safety function STO.
Description of the safety module, GDCP-CAMC-G-S3-...	Functional safety engineering for the motor controller CMMP-AS-...- M3 with the safety functions STO, SS1, SS2, SOS, SLS, SSR, SSM, SBC.
Description of the safety function STO, GDCP-CMMP-AS-M0-S1-...	Functional safety engineering for the motor controller CMMP-AS-...- M0 with the integrated safety function STO.
Description for exchange and project conversion GDCP-CMMP-M3/-M0-RP-...	Motor controller CMMP-AS-...- M3/-M0 as a replacement device for previous motor controller CMMP-AS. Changes to the electrical installation and description of project conversion.
Help for the FCT plug-in CMMP-AS	User interface and functions of the CMMP-AS plug-in for the Festo Configuration Tool. ➔ www.festo.com/sp

Tab. 2 Documentation on the motor controller CMMP-AS-...-M3/-M0

1 Fieldbus interfaces

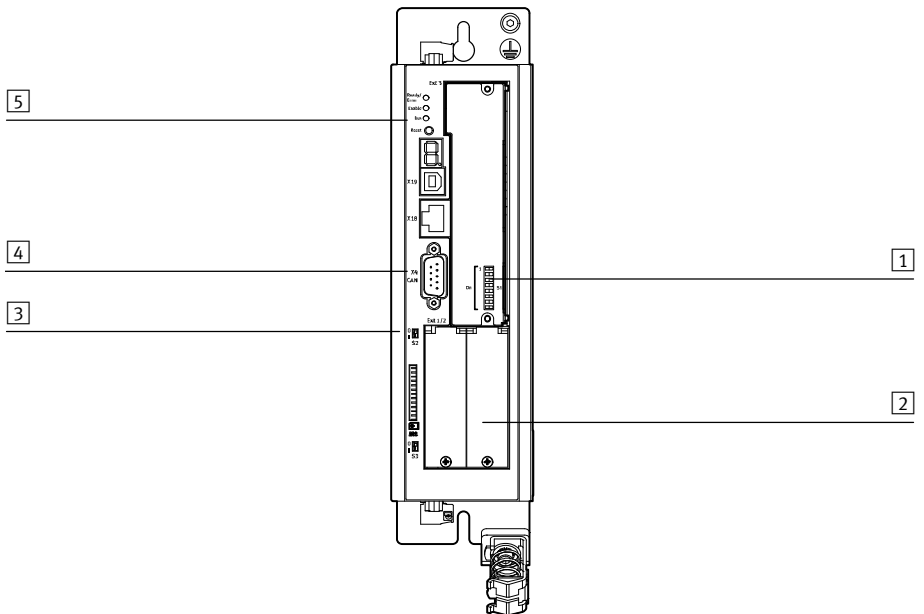
Control and parameterisation via CiA 402 for the CMMP-AS-...-M3/-M0 is supported correspondingly through the fieldbus interfaces. Tab. 1.1 The CANopen interface is integrated into the motor controller; through interfaces, the motor controller can be extended with additional fieldbus interfaces. The fieldbus is configured with the DIP switches [S1].

Fieldbus	Interface	Description
CANopen	[X4] – integrated	→ Chapter 2
EtherCAT	Interface CAMC-EC	→ Chapter 4

Tab. 1.1 Fieldbus interfaces for CiA 402

M0

The motor controllers CMMP-AS-...-M0 are only equipped with the CANopen fieldbus interface and do not feature any slots for interfaces, switches or safety modules.



- 1) DIP switches [S1] for fieldbus settings on the switch or safety module in slot Ext3
- 2) Slots Ext1/Ext2 for interfaces
- 3) CANopen terminating resistor [S2]
- 4) CANopen interface [X4]
- 5) CAN-LED

Fig. 1.1 Motor controller CMMP-AS-...-M3: Front view, example with micro switch module in Ext3

2 CANopen [X4]

2.1 General information on CANopen

CANopen is a standard worked out by the “CAN in Automation” association. Numerous device manufacturers are organised in this network. This standard has largely replaced the current manufacturer-specific CAN protocols. As a result, the end user has a non-proprietary communication interface. The following manuals, among others, can be obtained from this association:

CiA Draft Standard 201 ... 207:

These documents cover the general basic principles and embedding of CANopen into the OSI layered architecture. The relevant points of this book are presented in this CANopen manual, so procurement of DS 201 ... 207 is generally not necessary.

CiA Draft Standard 301:

This book describes the fundamental design of the object directory of a CANopen device and access to it. The statements of DS201 ... 207 are also made concrete. The elements of the object directory needed for the CMMP motor controller families and the related access methods are described in this manual. Procurement of DS 301 is recommended but not unconditionally necessary.

CiA Draft Standard 402:

This book deals with the specific implementation of CANopen in drive controllers. Although all implemented objects are also briefly documented and described in this CANopen manual, the user should have this book available.

Source address: → www.can-cia.de

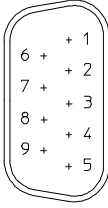
The CANopen implementation of the motor controller is based on the following standards:

- | | | | |
|---|----------------------------------|---------------|------------------|
| 1 | CiA Draft Standard 301, | Version 4.02, | 13 February 2002 |
| 2 | CiA Draft Standard Proposal 402, | Version 2.0, | 26 July 2002 |

2.2 Cabling and pin assignment

2.2.1 Pin allocations

The CAN interface is already integrated in the motor controller CMMP-AS-...-M3/-M0 and thus is always available. The CAN bus connection is designed as a 9-pole DSUB plug in accordance with standards.

[X4]	Pin no.	Designation	Value	Description
	1	–	–	Unused
	6	CAN-GND	–	Ground
	2	CAN-L	–	Negative CAN signal (dominant low)
	7	CAN-H	–	Positive CAN signal (dominant high)
	3	CAN-GND	–	Ground
	8	–	–	Unused
	4	–	–	Unused
	9	–	–	Unused
	5	CAN shield	–	Screening

Tab. 2.1 Pin assignment for CAN-interface [X4]



CAN bus cabling

When cabling the motor controller via the CAN bus, you should unconditionally observe the following information and instructions to obtain a stable, trouble-free system.

If cabling is improperly done, malfunctions can occur on the CAN bus during operation.

These can cause the motor controller to shut off with an error for safety reasons.

Termination

A terminating resistor (120 Ω) can, if required, be switched by means of DIP switch S2 = 1 (CAN Term) on the basic unit.

2.2.2 Cabling instructions

The CAN bus offers a simple, fail-safe ability to network all the components of a system together. But a requirement for this is that all of the following instructions on cabling are observed.

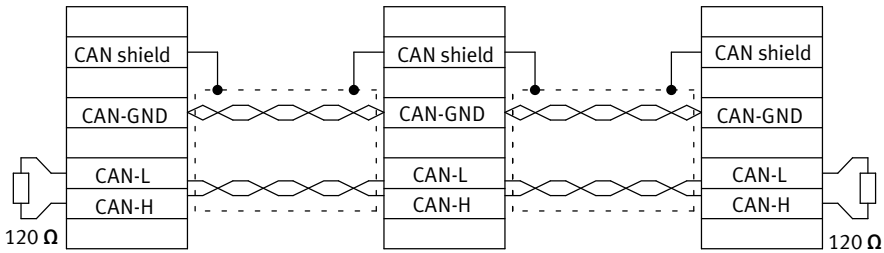


Fig. 2.1 Cabling example

- The individual nodes of the network are connected point-to-point to each other, so the CAN cable is looped from controller to controller (→ Fig. 2.1).
- A terminating resistor of exactly $120\ \Omega \pm 5\%$ must be available at both ends of the CAN cable. Such a terminating resistor is often already integrated into CAN cards or PLCs, which must be taken into account correspondingly.
- A screened cable with precisely two twisted conductor pairs must be used for the cabling. One twisted pair is used for connecting CAN-H and CAN-L. The conductors of the other pair are used together for CAN-GND. The cable screening is connected to the CAN shield connection at all nodes. (A table with the technical data of usable cables is located at the end of this chapter.)
- The use of adapters is not recommended for CAN bus cabling. If this is unavoidable, then metallic plug housings should be used to connect the cable screening.
- To keep the disturbance coupling as low as possible, motor cables should not be laid parallel to signal lines. Motor cables must conform to specifications. Motor cables must be correctly shielded and earthed.
- For additional information on design of trouble-free CAN bus cabling, refer to the Controller Area Network protocol specification, Version 2.0 from Robert Bosch GmbH, 1991.

Characteristic		Value
Wire pairs	–	2
Wire cross section	[mm ²]	≥ 0.22
Screening	–	Yes
Loop resistance	[Ω / m]	< 0.2
Surge impedance	[Ω]	100...120

Tab. 2.2 Technical data, CAN bus cable

2.3 Configuration of CANopen stations on the CMMP-AS-...-M3

M3

This section is only applicable for the motor controller CMMP-AS-...-M3.

Several steps are required in order to produce an operational CANopen interface. Some of these settings should or must be carried out before the CANopen communication is activated. This section provides an overview of the steps required by the slave for parametrisation and configuration. As some parameters are only effective after saving and reset, we recommend that commissioning with the FCT without connection to the CANopen bus should be carried out first.



Instructions on commissioning with the Festo Configuration Tool can be found in the Help for the device-specific FCT plug-in.

When designing the CANopen interface, the user must therefore make these determinations. Only then should parameterisation of the fieldbus connection take place on both pages. We recommend that parameterisation of the slave should be executed first. Then the master should be configured.

We recommend the following procedure:

1. Setting of the offset of the node number, bit rate and activation of the bus communication via DIP switches.



The status of the DIP switches is read once at Power- ON / RESET.
The CMMP-AS takes over changes to the switch setting in ongoing operation only at the next RESET or restart

2. Parametrisation and commissioning with the Festo Configuration Tool (FCT).

In particular on the Application Data page:

- CANopen control interface (Mode Selection tab)

In addition, the following settings on the fieldbus page:

- Basic address of the node number
- Protocol CANopen DS 402 (Operating parameter tab)
- Physical units (Factor Group tab)



Observe that parameterisation of the CANopen function remains intact after a reset only if the parameter set of the motor controller was saved.

While the FCT device control is active, CAN communication is automatically deactivated.

3. Configuration of the CANopen master → Sections 2.5 and 3.

2.3.1 Setting of the node number

Each device in the network must be assigned a unique node number.

The node number can be set via the DIP switches 1 ... 5 on the module in slot Ext3 and in the program FCT.



The resulting node number consists of the base address (FCT) and the offset (DIP switches).

Permissible values for the node number lie in the range 1 ... 127.

Setting of the offset of the node number with DIP switches

Setting of the node number can be made with DIP switch 1 ... 5. The offset of the node number set via DIP switches 1 ... 5 is displayed in the program FCT on the Fieldbus page in the Operating Parameters tab.

DIP switches	Value		Example	
	ON	OFF		Value
	1	0	ON	1
	2	0	ON	2
	3	0	OFF	0
	4	0	ON	8
	5	0	ON	16
Total 1 ... 5 = Offset	1 ... 31 ¹⁾			27

1) The value 0 for the offset is interpreted in connection with a base address 0 as node number 1.

A node number greater than 31 must be set with the FCT.

Tab. 2.3 Setting of the offset of the node number

Setting the base address of the node number with FCT

With the Festo Configuration Tool (FCT), the node number is set as base address on the Fieldbus page in the Operating Parameters tab.

Default setting = 0 (that means offset = node number).



If a node number is assigned simultaneously via DIP switches 1...5 and in the FCT program, the resulting node number consists of the sum of the base address and the offset. If this sum is greater than 127, the value is automatically limited to 127.

2.3.2 Setting of the transmission rate with DIP switches

The transmission rate must be set with DIP switches 6 and 7 on the module in slot Ext3. The status of the DIP switches is read one time at Power On/RESET. The CMMP-AS-...-M3 takes over changes to the switch setting in ongoing operation only at the next RESET.

Transmission rate		DIP switch 6	DIP switch 7
125	[kbit/s]	OFF	OFF
250	[kbit/s]	ON	OFF
500	[kbit/s]	OFF	ON
1	[Mbps]	ON	ON

Tab. 2.4 Setting of the transmission rate

2.3.3 Activation of CANopen communication with DIP switches

When the node number and transmission rate have been set, CANopen communication can be activated. Please note that the above-mentioned parameters can only be revised when the protocol is deactivated.

CANopen communication	DIP switch 8
Deactivated	OFF
Enabled	ON

Tab. 2.5 Activation of CANopen communication

Please observe that CANopen communication can only be activated after the parameter set (the FCT project) has been saved and a Reset carried out.



If another fieldbus interface is plugged into Ext1 or Ext2 (→ chapter 1), CANopen communication is activated with DIP switch 8 instead of via [X4] of the corresponding fieldbus.

2.3.4 Setting the physical units (factor group)

In order for a fieldbus master to exchange position, speed and acceleration data in physical units (e.g. mm, mm/s, mm/s²) with the motor controller, it must be parameterised via the factor group → section 5.3.

Parameterisation can be carried out via FCT or the fieldbus.

2.4 Configuration of CANopen participants on the CMMP-AS-...-M0

M0

This section is only applicable for the motor controller CMMP-AS-...-M0.

Several steps are required in order to produce an operational CANopen interface. Some of these settings should or must be carried out before the CANopen communication is activated. This section provides an overview of the steps required by the slave for parameterisation and configuration.



Instructions on commissioning with the Festo Configuration Tool can be found in the Help for the device-specific FCT plug-in.

When designing the CANopen interface, the user must therefore make these determinations. Only then should parameterisation of the fieldbus connection take place on both pages. We recommend that parameterisation of the slave should be executed first. Then the master should be configured.

The CAN-bus-specific parameters can be set on two paths. These paths are separated from one another and are accessed via the option “Fieldbus parameterisation via DINs” on the “Application data” page in the FCT.

The option “Fieldbus parameterisation via DINs” is active on delivery and after a reset to the factory settings. Parameterisation with FCT for activation of the CAN bus is thus not absolutely necessary.

The following parameters can be set via the DINs or FCT:

Parameters	Setting via	
	DIN	FCT
Node number	0 ... 3 ¹⁾	“Fieldbus” page, operating parameters.
Transmission rate (bit rate)	12, 13 ¹⁾	Activation of the CAN bus is performed automatically by FCT (dependent on device control):
Input/activation	8	
Protocol (data profile)	9 ²⁾	<ul style="list-style-type: none"> – Device control by FCT → CAN deactivated – Device control released → CAN activated

1) Only transferred in the event of inactive CAN communication

2) Only transferred after a device RESET

Tab. 2.6 Overview of settings for CAN parameters via DINs or FCT

2.4.1 Setting the node number via DINs and FCT

Each device in the network must be assigned a unique node number.

The node number can be set via the digital inputs DIN0 ... DIN3 **and** in the FCT programme.



Permissible values for the node number lie in the range 1 ... 127.

Setting the offset of the node number via DINs

The node number can be set via the circuitry of the digital inputs DIN0 ... DIN3. The offset of the node number set via the digital inputs is displayed in the FCT programme on the “Fieldbus” panel in the “Operating parameters” tab.

DINs	Value		Example	
	High	Low		Value
0	1	0	High	1
1	2	0	High	2
2	4	0	Low	0
3	8	0	High	8
Total 0 ... 3 = node number 0 ... 15				11

Tab. 2.7 Setting the node number

Setting the base address of the node number via FCT

The base address of the node number can be set via FCT on the “Fieldbus” panel in the “Operating parameters” tab.

The resulting node number is dependent on the option “Fieldbus parameterisation via DINs” on the “Application data” page. If this option is activated, the node number is determined by adding the base address in the FCT to the offset via the digital inputs DIN0...3.

If the option is deactivated, the base address in the FCT corresponds to the resulting node number.

2.4.2 Setting the transmission rate via DINs or FCT

The transmission rate can be set via the digital inputs DIN12 and DIN13 **or** in the FCT.

Setting the transmission rate via DINs

Transmission rate		DIN 12	DIN 13
125	[Kbit/s]	Low	Low
250	[Kbit/s]	High	Low
500	[Kbit/s]	Low	High
1	[Mbit/s]	High	High

Tab. 2.8 Setting the transmission rate

Setting the transmission rate via FCT

The transmission rate can be set via FCT on the “Fieldbus” panel in the “Operating parameters” tab. The option “Fieldbus parameterisation via DINs” must be deactivated beforehand on the “Application data” panel. When this option is deactivated, DIN12 and DIN13 can be parameterised freely again. Optionally, however, AIN1 and AIN2 can also be parameterised with the FCT.

2.4.3 Setting the protocol (data profile) via DINs or FCT

The protocol (data profile) can be set via the digital input DIN9 or the FCT.

Setting the protocol (data profile) via DINs

Protocol (data profile)	DIN 9
CiA 402 (DS 402)	Low
FHPP	High

Tab. 2.9 Activating the protocol (data profile)

Setting the protocol (data profile) via FCT

The protocol is set via FCT on the “Fieldbus” page in the “Operating parameters” tab.

2.4.4 Activation of CANopen communication via DINs or FCT

When the node number, transmission rate and protocol (data profile) have been set, CANopen communication can be activated.

Activation of CANopen communication via DIN

CANopen communication	DIN 8
Deactivated	Low
Enabled	High

Tab. 2.10 Activation of CANopen communication



The device does not need to be reset again for activation via digital input. The CAN bus is activated immediately after a level change (Low → High) at DIN8.

Activation of CANopen communication via FCT

CANopen communication is automatically activated by the FCT if the option “Fieldbus parameterisation via DINs” is deactivated.



The CAN bus is switched off for as long as the device control remains with FCT.

2.4.5 Setting the physical units (factor group)

In order for a fieldbus master to exchange position, speed and acceleration data in physical units (e.g. mm, mm/s, mm/s²) with the motor controller, it must be parameterised via the factor group → section 5.3.

Parameterisation can be carried out via FCT or the fieldbus.

2.5 Configuration CANopen master

You can use an EDS file to configure the CANopen master.

The EDS file is included on the CD-ROM supplied with the motor controller.



You will find the most current version under → www.festo.com/sp

EDS files	Description
CMMP-AS-...-M3.eds	Motor controller CMMP-AS-...- M3 with protocol “CiA402 (DS 402)”
CMMP-AS-...-M0.eds	Motor controller CMMP-AS-...- M0 with protocol “CiA402 (DS 402)”

Tab. 2.11 EDS files for CANopen

3 CANopen access procedure

3.1 Introduction

CANopen makes available a simple and standardised possibility to access the parameters of the motor controller (e.g. the maximum motor current). To achieve this, a unique number (index and subindex) is assigned to each parameter (CAN object). The totality of all adjustable parameters is designated an object directory.

For access to the CAN objects through the CAN bus, there are fundamentally two methods available: a confirmed access type, in which the motor controller acknowledges each parameter access (via so-called SDOs), and an unconfirmed access type, in which no acknowledgement is made (via so-called PDOs).

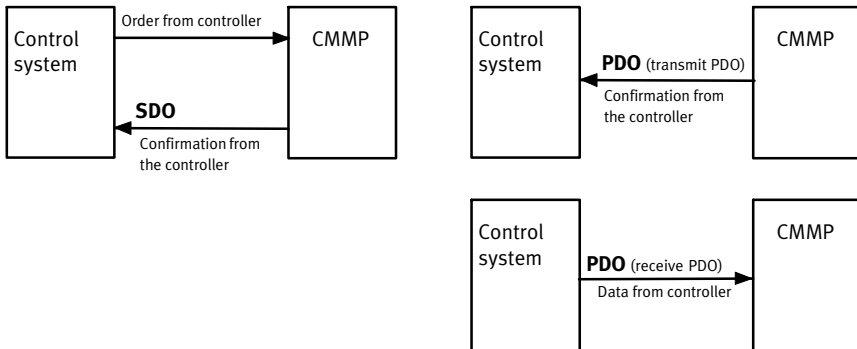


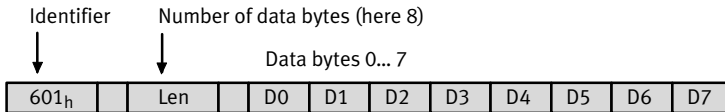
Fig. 3.1 Access procedure

As a rule, the motor controller is parametrised and also controlled via SDO access. In addition, other types of messages (so-called communication objects), which are sent either by the motor controller or the higher-level controller, are defined for special application cases:

Communication objects		
SDO	Service Data Object	Used for normal parametrisation of the motor controller.
PDO	Process Data Object	Fast exchange of process data (e.g. actual speed) possible
SYNC	Synchronisation Message	Synchronisation of multiple CAN nodes
EMCY	Emergency message	Transmission of error messages
NMT	Network management	Network service: All CAN nodes can be worked on simultaneously, for example.
HEART-BEAT	Error Control Protocol	Monitoring of the communications participants through regular messages.

Tab. 3.1 Communication objects

Every message sent on the CAN bus contains a type of address which is used to determine the bus participant for which the message is meant. This number is designated the identifier. The lower the identifier, the greater the priority of the message. Identifiers are established for the above-named communication objects. The following sketch shows the basic design of a CANopen message:



3.2 SDO Access

The Service Data Objects (SDO) permit access to the object directory of the motor controller. This access is especially simple and clear. It is therefore recommended to build up the application at first only with SDOs and only later to convert to the faster but also more complicated Process Data Objects (PDOs).

SDO access always starts from the higher-order controller (Host). This either sends the motor controller a write command to modify a parameter in the object directory, or a read command to read out a parameter. For each command, the host receives an answer that either contains the read-out value or – in the case of a write command – serves as an acknowledgement.

For the motor controller to recognise that the command is meant for it, the host must send the command with a specific identifier. This identifier is made up of the base 600_h + node number of the applicable motor controller. The motor controller answers correspondingly with the identifier 580_h + node number.

The design of the commands or answers depends on the data type of the object to be read or written, since either 1, 2 or 4 data bytes must be sent or received. The following data types are supported:

Data type	Size and algebraic sign	Range
UINT8	8 bit value without algebraic sign	0 ... 255
INT8	8 bit value with algebraic sign	-128 ... 127
UINT16	16 bit value without algebraic sign	0 ... 65535
INT16	16 bit value with algebraic sign	-32768 ... 32767
UINT32	32 bit value without algebraic sign	0 ... (2 ³² -1)
INT32	32 bit value with algebraic sign	-(2 ³¹) ... (2 ³² -1)

Tab. 3.2 Supported data types

3.2.1 SDO Sequences for Reading and Writing

To read out or describe objects of these number types, the following listed sequences are used. The commands for writing a value into the motor controller begin with a different identifier, depending on the data type. The answer identifier, in contrast, is always the same. Read commands always start with the same identifier, and the motor controller answers differently, depending on the data type returned. All numbers are kept in hexadecimal form.

Identifier	8 bits	16 bit	32 bit
Task identifier	2F _h	2B _h	23 _h
Response identifier	4F _h	4B _h	43 _h
Response identifier in case of error	–	–	80 _h

Tab. 3.3 SDO – response/task identifier

EXAMPLE			
UINT8/INT8	Reading of Obj. 6061_00 _h Return data: 01 _h	Writing of Obj. 1401_02 _h Data: EF _h	
Command	40 _h 61 _h 60 _h 00 _h	2F _h 01 _h 14 _h 02 _h EF _h	
Response:	4F _h 61 _h 60 _h 00 _h 01 _h	60 _h 01 _h 14 _h 02 _h	
UINT16/INT16	Reading of Obj. 6041_00 _h Return data: 1234 _h	Writing of Obj. 6040_00 _h Data: 03E8 _h	
Command	40 _h 41 _h 60 _h 00 _h	2B _h 40 _h 60 _h 00 _h E8 _h 03 _h	
Response:	4B _h 41 _h 60 _h 00 _h 34 _h 12 _h	60 _h 40 _h 60 _h 00 _h	
UINT32/INT32	Reading of Obj. 6093_01 _h Return data: 12345678 _h	Writing of Obj. 6093_01 _h Data: 12345678 _h	
Command	40 _h 93 _h 60 _h 01 _h	23 _h 93 _h 60 _h 01 _h 78 _h 56 _h 34 _h 12 _h	
Response:	43 _h 93 _h 60 _h 01 _h 78 _h 56 _h 34 _h 12 _h	60 _h 93 _h 60 _h 01 _h	

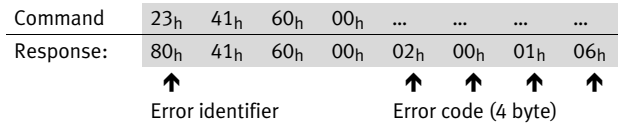


Caution

The acknowledgement from the motor controller must always be waited for! Only when the motor controller has acknowledged the request may additional requests be sent.

3.2.2 SDO Error Messages

In case of an error when reading or writing (for example, because the written value is too large), the motor controller answers with an error message instead of the acknowledgement:



Error code	Significance
F3 F2 F1 F0	
05 03 00 00 _h	Protocol error: Toggle bit was not revised
05 04 00 01 _h	Protocol error: Client / server command specifier invalid or unknown
06 06 00 00 _h	Access faulty due to a hardware problem ¹⁾
06 01 00 00 _h	Access type is not supported.
06 01 00 01 _h	Read access to an object that can only be written
06 01 00 02 _h	Write access to an object that can only be read
06 02 00 00 _h	The addressed object does not exist in the object directory
06 04 00 41 _h	The object must not be entered into a PDO (e.g. ro-object in RPDO)
06 04 00 42 _h	The length of the objects entered in the PDO exceeds the PDO length
06 04 00 43 _h	General parameter error
06 04 00 47 _h	Overflow of an internal variable / general error
06 07 00 10 _h	Protocol error: Length of the service parameter does not agree
06 07 00 12 _h	Protocol error: Length of the service parameter is too large
06 07 00 13 _h	Protocol error: Length of the service parameter is too small
06 09 00 11 _h	The addressed subindex does not exist
06 09 00 30 _h	The data exceed the range of values of the object
06 09 00 31 _h	The data are too large for the object
06 09 00 32 _h	The data are too small for the object
06 09 00 36 _h	Upper limit is less than lower limit
08 00 00 20 _h	Data cannot be transmitted or stored ¹⁾
08 00 00 21 _h	Data cannot be transmitted or stored, since the controller is working locally
08 00 00 22 _h	Data cannot be transmitted or stored, since the controller for this is not in the correct state ²⁾
08 00 00 23 _h	There is no object dictionary available ³⁾

- 1) Returned in accordance with CiA 301 in case of incorrect access to store_parameters / restore_parameters.
- 2) "Status" should be understood generally here: It may be a problem of the incorrect operating mode or a technology module that is not available or the like.
- 3) This error is returned, for example, when another bus system controls the motor controller or the parameter access is not permitted.

3.2.3 Simulation of SDO access

The firmware of the motor controller offers the possibility to simulate SDO access. In this way, after being written through the CAN bus, objects in the test phase can be read and checked through the CI terminal of the parametrisation software.

The syntax of the commands is:

	Read commands	Write commands
	↓ Main index (hex)	
UINT8/INT8	↓ Sub-index (hex)	
Command	? XXXX SU	= XXXX SU: WW
Response:	= XXXX SU: WW	= XXXX SU: WW
UINT16/INT16	↑ 8 bit data (hex)	
Command	? XXXX SU	= XXXX SU: WWWW
Response:	= XXXX SU: WWWW	= XXXX SU: WWWW
UINT32/INT32	↑ 16 bit data (hex)	
Command	? XXXX SU	= XXXX SU:
Response:	= XXXX SU: WWWWXXXX	= XXXX SU: WWWWXXXX
	↑ 32 bit data (hex)	

Note that the commands are entered as characters without any blanks.

Read error

Command ? XXXX SU
 Response: ! FFFFFFFF
 ↑ 32 bit error code
 F3 F2 F1 F0 in accordance with
 chap.

Write error

= XXXX SU: WWWWXXXX¹⁾
 ! FFFFFFFF
 ↑ 32 bit error code
 F3 F2 F1 F0 in accordance with
 chap.

1) In case of error, the response is built up the same for all 3 write commands (8, 16, 32 bit).

The commands are entered as characters without any blanks.



Caution

Never use these test commands in applications!
 Access only serves test purposes and is not appropriate for real-time-capable communication.
 In addition, the syntax of the test commands can be revised at any time.

3.3 PDO Message

With Process Data Objects (PDOs), data can be transmitted in an event-driven manner or cyclically. The PDO thereby transmits one or more previously established parameters. Other than with an SDO, there is no acknowledgement when a PDO is transmitted. After PDO activation, all recipients must therefore be able to process any arriving PDOs at any time. This normally means a significant software effort in the host computer. This disadvantage is offset by the advantage that the host computer does not need to cyclically request parameters transmitted by a PDO, which leads to a strong reduction in CAN bus capacity utilisation.

EXAMPLE

The host computer would like to know when the motor controller has completed a positioning from A to B.

When SDOs are used, it must frequently, such as every millisecond, request the statusword object, which uses up bus capacity.

When a PDO is used, the motor controller is parametrised at the start of the application in such a way that, with every change in the statusword object, a PDO containing the statusword object is deposited.

Instead of constantly requesting, the host computer thus automatically receives a corresponding message as soon as the event occurs.

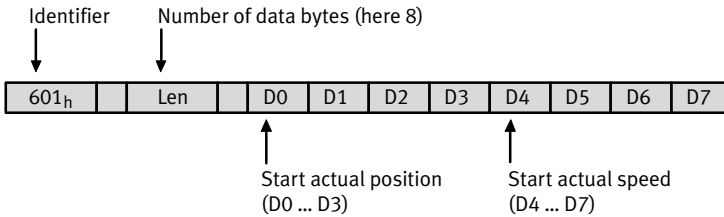
A distinction is made between the following types of PDOs:

Type	Path	Comment
Transmit PDO	Motor controller → Host	Motor controller sends PDO when a certain event occurs.
Receive PDO	Host → Motor controller	Motor controller evaluates PDO when a certain event occurs.

Tab. 3.4 PDO types

The motor controller has four transmit and four receive PDOs.

Almost all objects of the object directory can be entered (mapped) into the PDOs; that is, the PDO contains all data, e.g. the actual speed, the actual position, or the like. The motor controller must first be told which data have to be transmitted, since the PDO only contains reference data and no information about the type of parameter. In the example below, the actual position is transmitted in the data bytes 0 ... 3 of the PDO and the actual speed in the bytes 4 ... 7.



In this way, almost any desired data telegrams can be defined. The following chapters describe the settings necessary for this.

3.3.1 Description of the Objects

Object	Comment
COB_ID_used_by_PDO	In the object COB_ID_used_by_PDO, the identifier in which the respective PDO is sent or received is entered. If bit 31 is set, the respective PDO is deactivated. This is the presetting for all PDOs. The COB-ID may only be revised if the PDO is deactivated, that is, bit 31 is set. A different identifier than is currently set in the controller may therefore only be written if bit 31 is simultaneously set. The set bit 30 shows when the identifier is read that the object cannot be requested by a remote frame. This bit is ignored during writing and is always set during reading.
number_of_mapped_objects	This object specifies how many objects should be mapped into the corresponding PDO. The following limitations must be observed: A maximum of 4 objects can be mapped per PDO A PDO may have a maximum of 64 bits (8 byte).
first_mapped_object ... fourth_mapped_object	For each object contained in the PDO, the motor controller must be told the corresponding index, sub-index and length. The stated length must agree with the stated length in the object dictionary. Parts of an object cannot be mapped. The mapping information has the following format → Tab. 3.6
transmission_type and inhibit_time	Which event results in sending (transmit PDO) or evaluation (receive PDO) of a message can be determined for each PDO. → Tab. 3.7

Object	Comment
transmit_mask_high and transmit_mask_low	If “change” is selected as the transmission_type, the TPDO is always sent when at least 1 bit of the TPDO changes. But frequently it is necessary that the TPDO should only be sent when certain bits have changed. For that reason, the TPDO can be equipped with a mask: Only the bits of the TPDO that are set to “1” in the mask are used to evaluate whether the PDO has changed. Since this function is manufacturer-specific, all bits of the masks are set as default value.

Tab. 3.5 Description of the Objects

xxx_mapped_object		
Main index (hex)	[bit]	16
Sub-index (hex)	[bit]	8
Length of the object (hex)	[bit]	8

Tab. 3.6 Format of the mapping information

To simplify the mapping, the following procedure is established:

1. The number of mapped objects is set to 0.
2. The parameters first_mapped_object ... fourth_mapped_object may be described (The overall length of all objects is not relevant in this time).
3. The number of mapped objects is set to a value between 1 ... 4. The length of all these objects must now not exceed 64 bits.

Value	Significance	Permitted with
01 _h – F0 _h	SYNC message The numerical value specifies how many SYNC messages have to be received before the PDO – is sent (T-PDO) or – evaluated (R-PDO).	TPDOs RPDOs
FE _h	Cyclical The transfer PDO is cyclically updated and sent by the motor controller. The time period is set by the object inhibit_time. Receive PDOs, in contrast, are evaluated immediately after reception.	TPDOs (RPDOs)
FF _h	Change The transfer PDO is sent when at least 1 bit has changed in the data of the PDO. With inhibit_time, the minimum interval between sending two PDOs can also be established in 100 µs steps.	TPDOs

Tab. 3.7 Type of transmission

The use of all other values is not permitted.

EXAMPLE

The following objects should be transmitted in one PDO:

Name of the object	Index_Subindex	Significance
statusword	6041 _h _00 _h	Controller regulation
modes_of_operation_display	6061 _h _00 _h	Operating mode
digital_inputs	60FD _h _00 _h	Digital inputs

The first transmit PDO (TPDO 1) should be used, which should always be sent whenever one of the digital inputs changes, but at a maximum of every 10 ms. As an identifier for this PDO, 187_h should be used.

- Deactivating PDO
If the PDO is active, it must first be deactivated.
Writing the identifier with set bit 31 (PDO is deactivated):
→ cob_id_used_by_pdo = C0000187_h
- Deleting number of objects
Set the number of objects to zero in order to be able to change the object mapping.
→ number_of_mapped_objects = 0
- Parametrisation of objects that are to be mapped
The above-listed objects must be combined into a 32 bit value:

Index = 6041 _h	Sub-index = 00 _h	Length = 10 _h	→ first_mapped_object = 60410010 _h
Index = 6061 _h	Sub-index = 00 _h	Length = 08 _h	→ second_mapped_object = 60610008 _h
Index = 60FD _h	Sub-index = 00 _h	Length = 20 _h	→ third_mapped_object = 60FD0020 _h
- Parametrisation of number of objects
The PDO should contain 3 objects
→ number_of_mapped_objects = 3_h
- Parametrisation of transmission type
The PDO should be sent when changes (to the digital inputs) are sent.
To ensure that only changes to the digital inputs result in transmission, the PDO is masked so that only the 16 bits of the object 60FD_h “come through”.
The PDO should be sent no more than every 10 ms (100D100 μs).
→ transmission_type = FF_h
→ transmit_mask_high = 00FFFF00_h
→ transmit_mask_low = 00000000_h
→ inhibit_time = 64_h
- Parametrisation of identifiers
The PDO should be sent with identifier 187_h.
Write the new identifier and activate the PDO through deletion of bit 31:
→ cob_id_used_by_pdo = 40000187_h



Observe that parametrisation of the PDOs may generally only be changed when the network status (NMT) is not operational. → Chapter 3.3.3

3.3.2 Objects for PDO Parametrisation

The motor controllers of the CMMP series have available a total of 4 transmit and 4 receive PDOs. The individual objects for parametrisation of these PDOs are the same for all 4 TPDOs and all 4 RPDOs respectively. For that reason, only the parameter description of the first TPDO is explicitly listed. The meaning can also be used for the other PDOs, which are listed in table form in the following:

Index	1800_h
Name	transmit_pdo_parameter_tpdo1
Object Code	RECORD
No. of Elements	3

Sub-Index	01_h
Description	cob_id_used_by_pdo_tpdo1
Data Type	UINT32
Access	rw
PDO Mapping	no
Units	–
Value Range	181 _h ... 1FF _h , bit 30 and 31 may be set
Default Value	C0000181 _h

Sub-Index	02_h
Description	transmission_type_tpdo1
Data Type	UINT8
Access	rw
PDO Mapping	no
Units	–
Value Range	0 ... 8C _h , FE _h , FF _h
Default Value	FF _h

Sub-Index	03_h
Description	inhibit_time_tpdo1
Data Type	UINT16
Access	rw
PDO Mapping	no
Units	100 μs (i.e. 10 = 1ms)
Value Range	–
Default Value	0

Index	1A00_h
Name	transmit_pdo_mapping_tpdo1
Object Code	RECORD
No. of Elements	4

Sub-Index	00_h
Description	number_of_mapped_objects_tpdo1
Data Type	UINT8
Access	rw
PDO Mapping	no
Units	–
Value Range	0 ... 4
Default Value	→ Table

Sub-Index	01_h
Description	first_mapped_object_tpdo1
Data Type	UINT32
Access	rw
PDO Mapping	no
Units	–
Value Range	–
Default Value	→ Table

Sub-Index	02_h
Description	second_mapped_object_tpdo1
Data Type	UINT32
Access	rw
PDO Mapping	no
Units	–
Value Range	–
Default Value	→ Table

Sub-Index	03_h
Description	third_mapped_object_tpdo1
Data Type	UINT32
Access	rw
PDO Mapping	no
Units	–
Value Range	–
Default Value	→ Table

Sub-Index	04_h
Description	fourth_mapped_object_tpdo1
Data Type	UINT32
Access	rw
PDO Mapping	no
Units	–
Value Range	–
Default Value	→ Table



Observe that the object groups `transmit_pdo_parameter_xxx` and `transmit_pdo_mapping_xxx` can only be written when the PDO is deactivated (bit 31 in `cob_id_used_by_pdo_xxx` set)

1. Transmit PDO

Index	Comment	Type	Acc.	Default Value
1800 _{h_00} _h	number of entries	UINT8	ro	03 _h
1800 _{h_01} _h	COB-ID used by PDO	UINT32	rw	C0000181 _h
1800 _{h_02} _h	transmission type	UINT8	rw	FF _h
1800 _{h_03} _h	inhibit time (100 μs)	UINT16	rw	0000 _h
1A00 _{h_00} _h	number of mapped objects	UINT8	rw	01 _h
1A00 _{h_01} _h	first mapped object	UINT32	rw	60410010 _h
1A00 _{h_02} _h	second mapped object	UINT32	rw	00000000 _h
1A00 _{h_03} _h	third mapped object	UINT32	rw	00000000 _h
1A00 _{h_04} _h	fourth mapped object	UINT32	rw	00000000 _h

2. Transmit PDO

Index	Comment	Type	Acc.	Default Value
1801 _{h_00} _h	number of entries	UINT8	ro	03 _h
1801 _{h_01} _h	COB-ID used by PDO	UINT32	rw	C0000281 _h
1801 _{h_02} _h	transmission type	UINT8	rw	FF _h
1801 _{h_03} _h	inhibit time (100 μs)	UINT16	rw	0000 _h
1A01 _{h_00} _h	number of mapped objects	UINT8	rw	02 _h
1A01 _{h_01} _h	first mapped object	UINT32	rw	60410010 _h
1A01 _{h_02} _h	second mapped object	UINT32	rw	60610008 _h
1A01 _{h_03} _h	third mapped object	UINT32	rw	00000000 _h
1A01 _{h_04} _h	fourth mapped object	UINT32	rw	00000000 _h

3. Transmit PDO

Index	Comment	Type	Acc.	Default Value
1802 _h _00 _h	number of entries	UINT8	ro	03 _h
1802 _h _01 _h	COB-ID used by PDO	UINT32	rw	C0000381 _h
1802 _h _02 _h	transmission type	UINT8	rw	FF _h
1802 _h _03 _h	inhibit time (100 µs)	UINT16	rw	0000 _h
1A02 _h _00 _h	number of mapped objects	UINT8	rw	02 _h
1A02 _h _01 _h	first mapped object	UINT32	rw	60410010 _h
1A02 _h _02 _h	second mapped object	UINT32	rw	60640020 _h
1A02 _h _03 _h	third mapped object	UINT32	rw	00000000 _h
1A02 _h _04 _h	fourth mapped object	UINT32	rw	00000000 _h

4. Transmit PDO

Index	Comment	Type	Acc.	Default Value
1803 _h _00 _h	number of entries	UINT8	ro	03 _h
1803 _h _01 _h	COB-ID used by PDO	UINT32	rw	C0000481 _h
1803 _h _02 _h	transmission type	UINT8	rw	FF _h
1803 _h _03 _h	inhibit time (100 µs)	UINT16	rw	0000 _h
1A03 _h _00 _h	number of mapped objects	UINT8	rw	02 _h
1A03 _h _01 _h	first mapped object	UINT32	rw	60410010 _h
1A03 _h _02 _h	second mapped object	UINT32	rw	606C0020 _h
1A03 _h _03 _h	third mapped object	UINT32	rw	00000000 _h
1A03 _h _04 _h	fourth mapped object	UINT32	rw	00000000 _h

tpdo_1_transmit_mask

Index	Comment	Type	Acc.	Default Value
2014 _h _00 _h	number of entries	UINT8	ro	02 _h
2014 _h _01 _h	tpdo_1_transmit_mask_low	UINT32	rw	FFFFFFFF _h
2014 _h _02 _h	tpdo_1_transmit_mask_high	UINT32	rw	FFFFFFFF _h

tpdo_2_transmit_mask

Index	Comment	Type	Acc.	Default Value
2015 _h _00 _h	number of entries	UINT8	ro	02 _h
2015 _h _01 _h	tpdo_2_transmit_mask_low	UINT32	rw	FFFFFFFF _h
2015 _h _02 _h	tpdo_2_transmit_mask_high	UINT32	rw	FFFFFFFF _h

tpdo_3_transmit_mask

Index	Comment	Type	Acc.	Default Value
2016 _h _00 _h	number of entries	UINT8	ro	02 _h
2016 _h _01 _h	tpdo_3_transmit_mask_low	UINT32	rw	FFFFFFFF _h
2016 _h _02 _h	tpdo_3_transmit_mask_high	UINT32	rw	FFFFFFFF _h

tpdo_4_transmit_mask

Index	Comment	Type	Acc.	Default Value
2017 _h _00 _h	number of entries	UINT8	ro	02 _h
2017 _h _01 _h	tpdo_4_transmit_mask_low	UINT32	rw	FFFFFFFF _h
2017 _h _02 _h	tpdo_4_transmit_mask_high	UINT32	rw	FFFFFFFF _h

1. Receive PDO

Index	Comment	Type	Acc.	Default Value
1400 _h _00 _h	number of entries	UINT8	ro	02 _h
1400 _h _01 _h	COB-ID used by PDO	UINT32	rw	C0000201 _h
1400 _h _02 _h	transmission type	UINT8	rw	FF _h
1600 _h _00 _h	number of mapped objects	UINT8	rw	01 _h
1600 _h _01 _h	first mapped object	UINT32	rw	60400010 _h
1600 _h _02 _h	second mapped object	UINT32	rw	00000000 _h
1600 _h _03 _h	third mapped object	UINT32	rw	00000000 _h
1600 _h _04 _h	fourth mapped object	UINT32	rw	00000000 _h

2. Receive PDO

Index	Comment	Type	Acc.	Default Value
1401 _h _00 _h	number of entries	UINT8	ro	02 _h
1401 _h _01 _h	COB-ID used by PDO	UINT32	rw	C0000301 _h
1401 _h _02 _h	transmission type	UINT8	rw	FF _h
1601 _h _00 _h	number of mapped objects	UINT8	rw	02 _h
1601 _h _01 _h	first mapped object	UINT32	rw	60400010 _h
1601 _h _02 _h	second mapped object	UINT32	rw	60600008 _h
1601 _h _03 _h	third mapped object	UINT32	rw	00000000 _h
1601 _h _04 _h	fourth mapped object	UINT32	rw	00000000 _h

3. Receive PDO

Index	Comment	Type	Acc.	Default Value
1402 _h _00 _h	number of entries	UINT8	ro	02 _h
1402 _h _01 _h	COB-ID used by PDO	UINT32	rw	C0000401 _h
1402 _h _02 _h	transmission type	UINT8	rw	FF _h
1602 _h _00 _h	number of mapped objects	UINT8	rw	02 _h
1602 _h _01 _h	first mapped object	UINT32	rw	60400010 _h
1602 _h _02 _h	second mapped object	UINT32	rw	607A0020 _h
1602 _h _03 _h	third mapped object	UINT32	rw	00000000 _h
1602 _h _04 _h	fourth mapped object	UINT32	rw	00000000 _h

4. Receive PDO

Index	Comment	Type	Acc.	Default Value
1403 _h _00 _h	number of entries	UINT8	ro	02 _h
1403 _h _01 _h	COB-ID used by PDO	UINT32	rw	C0000501 _h
1403 _h _02 _h	transmission type	UINT8	rw	FF _h
1603 _h _00 _h	number of mapped objects	UINT8	rw	02 _h
1603 _h _01 _h	first mapped object	UINT32	rw	60400010 _h
1603 _h _02 _h	second mapped object	UINT32	rw	60FF0020 _h
1603 _h _03 _h	third mapped object	UINT32	rw	00000000 _h
1603 _h _04 _h	fourth mapped object	UINT32	rw	00000000 _h

3.3.3 Activation of PDOs

For the motor controller to send or receive PDOs, the following points must be met:

- The object number_of_mapped_objects must not equal zero.
- In the object cob_id_used_for_pdos, bit 31 must be deleted.
- The communication status of the motor controller must be operational (➔ chapter 3.6, Network Management: NMT-Service)

To parametrise PDOs, the following points must be met:

- The communication status of the motor controller must not be operational.

3.4 SYNC message

Several devices of a system can be synchronised with each other. To do this, one of the devices (usually the higher-order controller) periodically sends out synchronisation messages. All connected controllers receive these messages and use them for treatment of the PDOs (→ chapter 3.3).



The identifier on which the motor controller receives the SYNC message is set permanently to 080h. The identifier can be read via the object `cob_id_sync`.

Index	1005h
Name	cob_id_sync
Object Code	VAR
Data Type	UINT32

Access	rw
PDO Mapping	no
Units	--
Value Range	80000080h, 00000080h
Default Value	00000080h

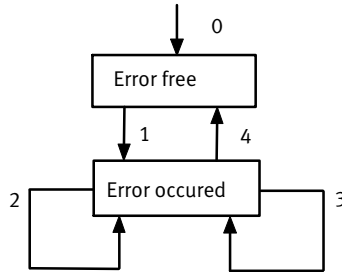
3.5 EMERGENCY Message

The motor controller monitors the function of its major assemblies. These include the power supply, output stage, angle transmitter evaluation and the slots Ext1 ... Ext3. In addition, the motor (temperature, angle encoder) and limit switch are checked. Incorrect parameter setting can also result in error messages (division by zero, etc.).

When an error occurs, the error number is shown in the motor controller's display. If several error messages occur simultaneously, the message with the highest priority (lowest number) is always shown in the display.

3.5.1 Overview

When an error occurs or an error acknowledgment is carried out, the controller transmits an EMERGENCY message. The identifier of this message is made up of the identifier 80_n and the node number of the relevant controller.



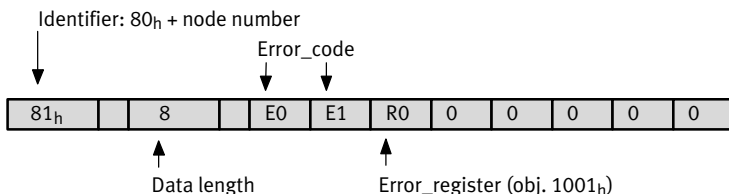
After a reset, the controller is in the status Error free (which it might leave again immediately, because an error is on hand from the beginning). The following status transitions are possible:

No.	Cause	Significance
0	Initialisation completed	
1	Error occurs	No error is present and an error occurs. An EMERGENCY telegram with the error code of the occurring error is sent.
2	Error acknowledgment	An error acknowledgment (→ chap. 6.1.5) is attempted, but not all causes are eliminated.
3	Error occurs	An error is present and an additional error occurs. An EMERGENCY telegram with the error code of the new error is sent.
4	Error acknowledgment	An error acknowledgment is attempted, and all causes are eliminated. An EMERGENCY telegram with the error code 0000 is sent.

Tab. 3.8 Possible status transitions

3.5.2 Structure of the EMERGENCY Message

When an error occurs, the motor controller transmits an EMERGENCY message. The identifier of this message is made up of the identifier 80_h and the node number of the relevant motor controller. The EMERGENCY message consists of eight data bytes, whereby the first two bytes contain an error_code, which is listed in the following table. An additional error code is in the third byte (object 1001_h). The remaining five bytes contain zeros.



error_register (R0)		
Bit	M/O1)	Significance
0	M	generic error: Error is present (Or-link of the bits 1 ... 7)
1	O	current: I ² t error
2	O	voltage: voltage monitoring error
3	O	temperature: motor overtemperature
4	O	communication error: (overrun, error state)
5	O	–
6	O	reserved, fix = 0
7	O	reserved, fix = 0
Values: 0 = no error; 1 = error present		

1) M = required / O = optional

Tab. 3.9 Bit assignment error_register

The error codes as well as the cause and measures can be found in chapter B “Diagnostic messages”.

3.5.3 Description of the Objects

Object 1003_h: pre_defined_error_field

The respective error_code of the error messages is also stored in a four-stage error memory. This is structured like a shift register, so that the last occurring error is always stored in the object 1003_h_01_h (standard_error_field_0). Through read access on the object 1003_h_00_h (pre_defined_error_field), it can be determined how many error messages are currently stored in the error memory. The error memory is cleared by writing the value 00_h into the object 1003_h_00_h (pre_defined_error_field_0). To be able to reactivate the output stage of the motor controller after an error, an error acknowledgement → chapter 6.1: Status Diagram (State Machine) must also be performed.

Index	1003_h
Name	pre_defined_error_field
Object Code	ARRAY
No. of Elements	4
Data Type	UINT32

Sub-Index	01_h
Description	standard_error_field_0
Access	ro
PDO Mapping	no
Units	–
Value Range	–
Default Value	–

Sub-Index	02_h
Description	standard_error_field_1
Access	ro
PDO Mapping	no
Units	–
Value Range	–
Default Value	–

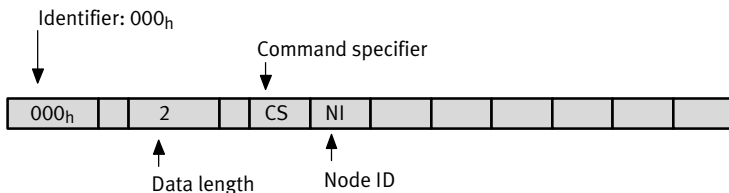
Sub-Index	03_h
Description	standard_error_field_2
Access	ro
PDO Mapping	no
Units	–
Value Range	–
Default Value	–

Sub-Index	04_h
Description	standard_error_field_3
Access	ro
PDO Mapping	no
Units	–
Value Range	–
Default Value	–

3.6 Network Management (NMT Service)

All CANopen equipment can be triggered via the Network Management. Reserved for this is the identifier with the top priority (000_h). By means of NMT, commands can be sent to one or all controllers. Each command consists of two bytes, whereby the first byte contains the command specifier (CS) and the second byte the node ID (NI) of the addressed controller. Through the node ID zero, all nodes in the network can be addressed simultaneously. It is thus possible, for example, that a reset is triggered in all devices simultaneously. The controllers do not acknowledge the NMT commands. Successful completion can only be determined indirectly (e.g. through the switch-on message after a reset).

Structure of the NMT Message:



For the NMT status of the CANopen node, statuses are established in a status diagram. Changes in statuses can be triggered via the CS byte in the NMT message. These are largely oriented on the target status.

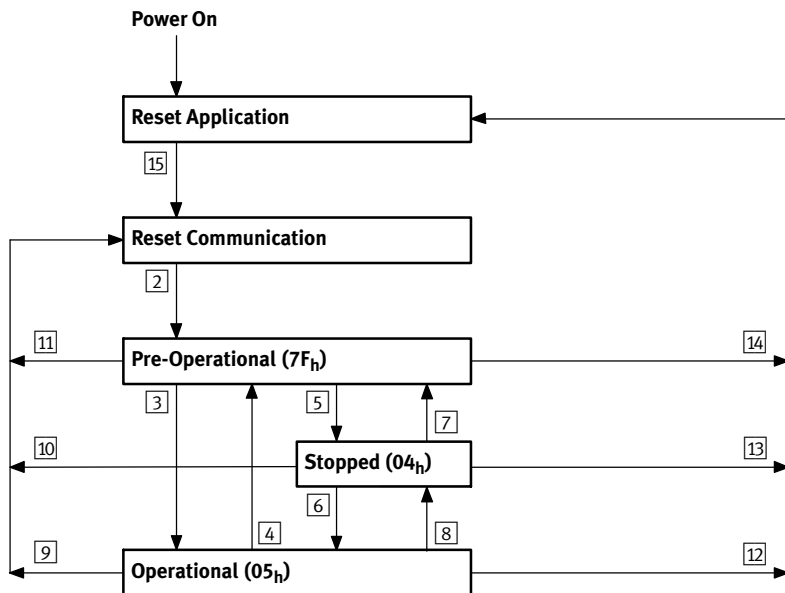


Fig. 3.2 Status diagram

Transition	Significance	CS	Target status	
2	Bootup	--	Pre-Operational	7F _h
3	Start Remote Node	01 _h	Operational	05 _h
4	Enter Pre-Operational	80 _h	Pre-Operational	7F _h
5	Stop Remote Node	02 _h	Stopped	04 _h
6	Start Remote Node	01 _h	Operational	05 _h
7	Enter Pre-Operational	80 _h	Pre-Operational	7F _h
8	Stop Remote Node	02 _h	Stopped	04 _h
9	Reset Communication	82 _h	Reset Communication ¹⁾	
10	Reset Communication	82 _h	Reset Communication ¹⁾	
11	Reset Communication	82 _h	Reset Communication ¹⁾	
12	Reset Application	81 _h	Reset Application ¹⁾	
13	Reset Application	81 _h	Reset Application ¹⁾	
14	Reset Application	81 _h	Reset Application ¹⁾	

1) The final target status is pre-operational (7F_h), since the transitions 15 and 2 are automatically performed by the controller.

Tab. 3.10 NMT state machine

All other status transitions are performed automatically by the controller, e.g. because the initialisation is completed.

In the NI parameter, the node number of the controller must be specified or zero if all nodes in the network are to be addressed (broadcast). Depending on the NMT status, certain communication objects cannot be used: For example, it is absolutely necessary to place the NMT status to operational so that the controller sends PDOs.

Name	Significance	SDO	PDO	NMT
Reset Application	No Communication. All CAN objects are reset to their reset values (application parameter set)	–	–	–
Reset Communication	No communication: The CAN controller is newly initialised.	–	–	–
Initialising	Status after hardware reset. Resetting of the CAN node, Sending of the bootup message	–	–	–
Pre-Operational	Communication via SDOs possible; PDOs not active (no sending/evaluating)	X	–	X
Operational	Communication via SDOs possible; all PDOs active (sending/evaluating)	X	X	X
Stopped	No communication except for heartbeating	–	–	X

Tab. 3.11 NMT state machine



NMT telegrams must not be sent in a burst (one immediately after another)!
 At least twice the position controller cycle time must lie between two consecutive NMT messages on the bus (also for different nodes!) for the controller to process the NMT messages correctly.



If necessary, the NMT command “Reset Application” is delayed until an ongoing saving procedure is completed, since otherwise the saving procedure would remain incomplete (defective parameter set).
 The delay can be in the range of a few seconds.



The communication status must be set to operational for the controller to transmit and receive PDOs.

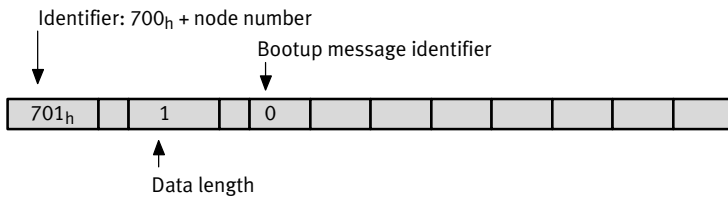
3.7 Bootup

3.7.1 Overview

After the power supply is switched on or after a reset, the controller reports via a Bootup message that the initialisation phase is ended. The controller is then in the NMT status preoperational (→ chapter 3.6, Network Management (NMT Service))

3.7.2 Structure of the Bootup Message

The Bootup message is structured almost identically to the following Heartbeat message. Only a zero is sent instead of the NMT status.



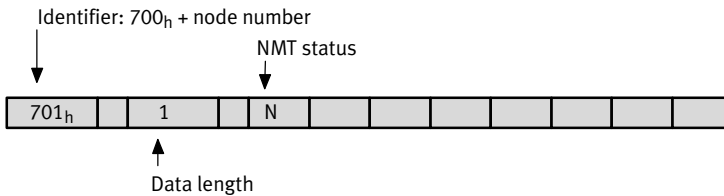
3.8 Heartbeat (Error Control Protocol)

3.8.1 Overview

The so-called Heartbeat protocol can be activated to monitor communication between slave (drive) and master: Here, the drive sends messages cyclically to the master. The master can check whether these messages occur cyclically and introduce corresponding measures if they do not. Since both Heartbeat and Nodeguarding telegrams (→ chap. 3.9) are sent with the identifier 700_h + node number, both protocols can be active at the same time. If both protocols are activated simultaneously, only the Heartbeat protocol is active.

3.8.2 Structure of the Heartbeat Message

The Heartbeat telegram is transmitted with the identifier 700_h + node number. It contains only 1 byte of user data, the NMT status of the controller (→ chapter 3.6, Network Management (NMT Service)).



N	Significance
04 _h	Stopped
05 _h	Operational
7F _h	Pre-Operational

3.8.3 Description of the Objects

Object 1017_h: producer_heartbeat_time

To activate then Heartbeat function, the time between two Heartbeat telegrams can be established via the object producer_heartbeat_time.

Index	1017_h
Name	producer_heartbeat_time
Object Code	VAR
Data Type	UINT16

Access	rw
PDO	no
Units	ms
Value Range	0 ... 65535
Default Value	0

The producer_heartbeat_time can be stored in the parameter record. If the controller starts with a producer_heartbeat_time not equal to zero, the bootup message is considered to be the first Heartbeat.

The controller can only be used as a so-called Heartbeat producer. The object 1016_h (consumer_heartbeat_time) is therefore implemented only for compatibility reasons and always returns 0.

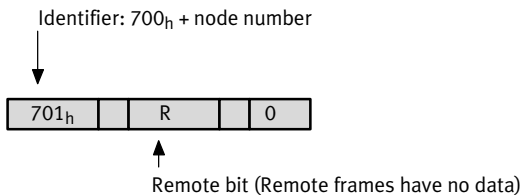
3.9 Nodeguarding (Error Control Protocol)

3.9.1 Overview

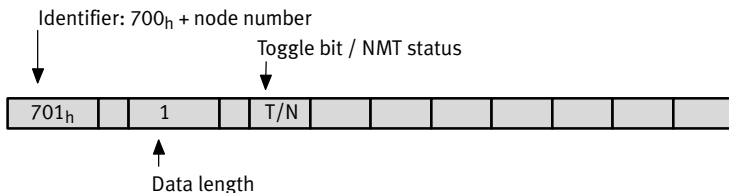
The so-called Nodeguarding protocol can also be used to monitor communication between slave (drive) and master. In contrast to the Heartbeat protocol, master and slave monitor each other: The master queries the drive cyclically about its NMT status. In every response of the controller, a specific bit is inverted (toggled). If these responses are not made or the controller always responds with the same toggle bit, the master can react accordingly. Likewise, the drive monitors the regular arrival of the Nodeguarding requests from the master: If messages are not received for a certain time period, the controller triggers error 12-4. Since both Heartbeat and Nodeguarding telegrams (→ chapter 3.8) are sent with the identifier 700_h + node number, both protocols cannot be active simultaneously. If both protocols are activated simultaneously, only the Heartbeat protocol is active.

3.9.2 Structure of the Nodeguarding Messages

The master's request must be sent as a so-called remote frame with the identifier 700_h + node number. In the case of a remote frame, a special bit is also set in the telegram, the remote bit. Remote frames have no data.



The response of the controller is built up analogously to the Heartbeat message. It contains only 1 byte of user data, the toggle bit and the NMT status of the controller (→ chapter 3.6).



The first data byte (T/N) is constructed in the following way:

Bit	Value	Name	Significance
7	80 _h	toggle_bit	Changes with every telegram
0 ... 6	7F _h	nmt_state	04 _h Stopped 05 _h Operational 7F _h Pre-Operational

The monitoring time for the master's requests can be parametrised. Monitoring begins with the first received remote request of the master. From this time on, the remote requests must arrive before the monitoring time has passed, since otherwise error 12-4 is triggered.

The toggle bit is reset through the NMT command Reset Communication. It is therefore deleted in the first response of the controller.

3.9.3 Description of the Objects

Object 100C_h: guard_time

To activate the Nodeguarding monitoring, the maximum time between two remote requests of the master is parametrised. This time is established in the controller from the product of guard_time (100C_h) and life_time_factor (100D_h). It is therefore recommended to write the life_time_factor with 1 and then specify the time directly via the guard_time in milliseconds.

Index	100C_h
Name	guard_time
Object Code	VAR
Data Type	UINT16

Access	rw
PDO Mapping	no
Units	ms
Value Range	0 ... 65535
Default Value	0

3.9.4 Object 100D_h: life_time_factor

The life_time_factor should be written with 1 in order to specify the guard_time directly.

Index	100D_h
Name	life_time_factor
Object Code	VAR
Data Type	UINT8

Access	rw
PDO Mapping	no
Units	–
Value Range	0.1
Default Value	0

3.9.5 Table of Identifiers

The following table gives an overview of the identifiers used:

Object type	Identifier (hexadecimal)	Comment
SDO (Host to controller)	600 _h + node number	
SDO (Controller to host)	580 _h + node number	
TPDO1	180 _h	Standard values. Can be revised if needed.
TPDO2	280 _h	
TPDO3	380 _h	
TPDO4	480 _h	
RPDO1	200 _h	
RPDO2	300 _h	
RPDO3	400 _h	
RPDO4	500 _h	
SYNC	080 _h	
EMCY	080 _h + node number	
HEARTBEAT	700 _h + node number	
NODEGUARDING	700 _h + node number	
BOOTUP	700 _h + node number	
NMT	000 _h	

4 EtherCAT with CoE

M3

This section is only applicable for the motor controller CMMP-AS-...-M3.

4.1 Overview

This part of the documentation describes the connection and configuration of the motor controller CMMP-AS-...-M3 in an EtherCAT network. It is intended for people who are already familiar with this motor controller series and CANopen CiA 402.

The fieldbus system EtherCAT means “Ethernet for Controller and Automation Technology” and is managed and supported by the international EtherCAT Technology Group (ETG) organisation; it is designed as an open technology, which is standardised by the International Electrotechnical Commission (IEC). EtherCAT is a fieldbus system based on Ethernet and can be handled like a fieldbus, thanks to high speed, flexible topology (line, tree, star) and simple configuration. The EtherCAT protocol is transported with a special standardised Ethernet type directly in the Ethernet frame in accordance with IEEE802.3. The slaves can broadcast, multicast and communicate laterally. For EtherCAT, the data exchange is based on a pure hardware machine.

Abbreviation	Significance
ESC	EtherCAT Slave Controller
PDI	Process Data Interface
CoE	CANopen-over-EtherCAT protocol

Tab. 4.1 EtherCAT-specific abbreviations

4.2 EtherCat-Interface CAMC-EC

The EtherCAT interface CAMC-EC allows the CMMP-AS-...-M3 motor controller to be connected to the EtherCAT fieldbus system. Communication over the EtherCAT interface (IEEE 802.3u) takes place with an EtherCAT standard cabling and is possible between the CMMP-AS-...-M3 from Revision 01 and the FCT parameterisation software from Version 2.0.



Festo supports the CoE protocol (CANopen over EtherCAT) with the CMMP-AS-...-M3.

EtherCAT CAMC-EC interface characteristics

The EtherCAT interface has the following features:

- Can be mechanically fully integrated into the CMMP-AS-...-M3 series motor controllers
- EtherCAT conforming to IEEE-802.3u (100Base-TX) with 100Mbps (full-duplex)
- Star and line topology
- Plug connector: RJ45
- Electrically isolated EtherCAT interface
- Communication cycle: 1 ms
- Max. 127 slaves
- EtherCAT slave implementation based on the Beckhoff FPGA ESC20
- Support of the “Distributed Clocks” feature for time-synchronous setpoint value transfer
- LED displays for ready status and link detect

Connection and display elements of the EtherCAT interface

The following elements can be found on the front plate of the EtherCAT interface:

- **LED 1** (two-colour LED) for:
 - EtherCAT communication (yellow)
 - “Connection active on Port 1” (red)
 - Run (green)
- **LED 2** (red) for displaying “Connection active on Port 2”
- Two RJ45 sockets.

The following figure shows the position and numbering of the sockets:

- 1 LED2
- 2 LED1
- 3 RJ45 socket [X1]
- 4 RJ45 socket [X2]

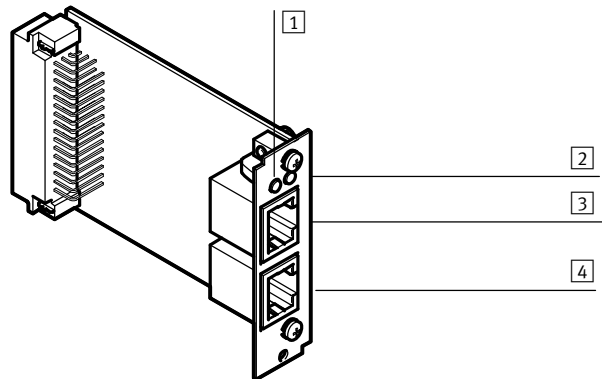


Fig. 4.1 Connection and display elements at the EtherCAT interface



The EtherCAT interface can only be operated in option slot Ext2. Operation of other interface modules in option slot Ext1, except with CAMC-D-8E8A interface, is then no longer possible.

4.3 Installing the EtherCAT interface in the controller



Note

Before performing mounting and installation work, observe the safety instructions in the hardware description GDCP-CMMP-M3-HW-...

Use an appropriate Phillips screwdriver to remove the front cover over option slot Ext2 of the CMMP-AS-...-M3 motor controller. The EtherCAT interface is now plugged into the open slot (Ext2) so that the printed circuit board slides into the lateral guides of the slot. The interface is pushed in up to the stop. Then screw the interface to the motor controller housing using the Phillips screw.

4.4 Pin allocation and cable specifications

RJ45 sockets	Function
[X1] (RJ45 socket on top)	Uplink to the master or a previous station of a series connection (e.g. multiple motor controllers)
[X2] (RJ45 socket underneath)	End of a series connection or connection of additional downstream stations

Tab. 4.2 Design of plug connectors [X1] and [X2]

	Pin	Specification	
	1	Receiver signal- (RX-)	Wire pair 3
	2	Receiver signal+ (RX+)	Wire pair 3
	3	Transmission signal- (TX-)	Wire pair 2
	4	–	Wire pair 1
	5	–	Wire pair 1
	6	Transmission signal+ (TX+)	Wire pair 2
	7	–	Wire pair 4
	8	–	Wire pair 4

Tab. 4.3 Allocation of the plug connectors [X1] and [X2]

Value	Function
EtherCAT interface, signal level	0 ... 2.5 V DC
EtherCAT interface, differential voltage	1.9 ... 2.1 V DC

Tab. 4.4 EtherCAT interface specification

Type and design of cable

Shielded twisted-pair STP, Cat.5 cables must be used for cabling. Star and line topologies are supported. The network structure must conform to the 5-4-3 rule. A maximum of 10 hubs can be cabled in series. The EtherCAT interface contains a hub. The total cable length is restricted to 100 m.



Errors due to inappropriate bus cables

Due to the very high possible transmission rates, we recommend that you use only standardised cables and plug connectors that at a minimum conform to the category 5 (CAT5) in accordance with EN 50173 or ISO/IEC 11801.

When setting up the EtherCAT network, you must unconditionally follow the advice in the relevant literature or the subsequent information and instructions in order to maintain a stable, trouble-free system. If the system is not cabled properly, EtherCAT bus malfunctions can occur during operation. These can cause the CMMP-AS-...-M3 motor controller to shut off with an error for safety reasons.

Bus termination

No external bus terminations are required. The EtherCAT technology module monitors its two ports and terminates the bus automatically (loop-back function).

4.5 Configuration of EtherCAT participants

Several steps are required in order to produce an operational EtherCAT interface. This section provides an overview of the steps required for parameterisation and configuration of the slave.



Note: Parameterisation and commissioning of the motor controller with the EtherCAT control interface is possible only with the connected master.



Notes on commissioning with the Festo Configuration Tool can be found in the Help for the device-specific FCT plug-in.

When designing the EtherCAT interface, the user must therefore make these determinations. Only then should parameterisation of the fieldbus connection take place on both pages. We recommend that the slave parameters should be set first. The master should be configured thereafter. With correct parameterisation, the application is ready immediately without communication faults.

We recommend the following procedure:

1. Activation of the bus communication.

EtherCAT communication is automatically started through the CMMP-AS...-M3 if it detects after switch-on that an EtherCAT interface is plugged in.

Communication cannot be deactivated by flipping DIL switch 8.

2. Parameterisation and commissioning with the Festo Configuration Tool (FCT).

In addition, the following settings on the fieldbus page:

- physical units of measure (Factor Group tab)



Observe that the parameterisation of the EtherCAT function only remains intact after a reset if the parameter set of the motor controller was saved.

3. Configuration of the EtherCAT master → following sections.

4.5.1 Setting of the physical units of measure (Factor Group)

In order for a fieldbus master to exchange position, speed and acceleration data in physical units (e.g. mm, mm/s, mm/s²) with the motor controller, it must be parameterised via the factor group → section 5.3.

Parameterisation can be carried out via either FCT or the fieldbus.

4.6 CANopen communication interface

User protocols are tunnelled via EtherCAT. For the CANopen-over-EtherCAT protocol (CoE) supported by the CMMP-AS-...-M3, most objects for the communication layer are supported by EtherCAT in accordance with CiA 301. This primarily involves objects for setting up communication between masters and slaves.

For the CANopen Motion Profile in accordance with CiA 402, most objects which can also be operated via the standard CANopen fieldbus are supported. In general, the following services and object groups are supported by the EtherCAT CoE implementation in the motor controller CMMP-AS-...-M3:

Services/object groups		Function
SDO	Service Data Object	Used for normal parametrisation of the motor controller.
PDO	Process Data Object	Fast exchange of process data (e.g. actual speed) possible.
EMCY	Emergency Message	Transmission of error messages.

Tab. 4.5 Supported services and object groups

The individual objects which can be addressed via the CoE protocol in the motor controller CMMP-AS-...-M3 are internally forwarded to the existing CANopen implementation and processed there. However, some new CANopen objects are added under the CoE implementation under EtherCAT, which are required for special connection via CoE. This is the result of the revised communication interface between the EtherCAT protocol and the CANopen protocol. A so-called Sync Manager is used to control the transmission of PDOs and SDOs via the two EtherCAT transfer types (mailbox and process data protocol).

This Sync Manager and the necessary configuration steps for operation of the CMMP-AS-...-M3 under EtherCAT-CoE are described in chapter 4.6.1 “Configuration of the Communication Interface”. The additional objects are described in chapter 4.6.2 “New and revised objects under CoE”.

Also, some CANopen objects of the CMMP-AS-...-M3, which are available under a normal CANopen connection, are not supported via a CoE connection over EtherCAT.

A list of the CANopen objects not supported under CoE is provided in chapter 4.6.3 “Objects not supported under CoE”.

4.6.1 Configuration of the Communication Interface

As already described in the previous chapter, the EtherCAT protocol uses two different transfer types for transmission of the device and user protocols, such as the CANopen-over-EtherCAT protocol (CoE) used by the CMMP-AS-...-M3. These two transfer types are the mailbox telegram protocol for non-cyclic data and the process data telegram protocol for transmission of cyclic data.

These two transfer types are used for the different CANopen transfer types for the CoE protocol. They are used as follows:

Telegram protocol	Description	Reference
Mailbox	This transfer type is used to transmit the Service Data Objects (SDOs) defined under CANopen. They are transmitted to EtherCAT in SDO frames.	→ chapter 4.8 “SDO Frame”
Process Data	This transfer type is used to transmit the Process Data Objects (PDOs) defined under CANopen, which are used to exchange cyclic data. They are transmitted to EtherCAT in PDO frames.	→ chapter 4.9 “PDO Frame”

Tab. 4.6 Telegram protocol – description

In general, these two transfer types allow all PDOs and SDOs to be used exactly as they are defined for the CANopen protocol for CMMP-AS-...-M3.

However, parametrisation of PDOs and SDOs for sending objects via EtherCAT is different from the settings which must be made under CANopen. In order to link the CANopen objects to be exchanged via PDO or SDO transfers between masters and slaves into the EtherCAT protocol, a so-called Sync Manager is implemented under EtherCAT.

This Sync Manager is used to link the data of the PDOs and SDOs to be sent to the EtherCAT telegrams. To accomplish this, the Sync Manager provides multiple Sync channels which can each implement a CANopen data channel (Receive SDO, Transmit SDO, Receive PDO or Transmit PDO) on the EtherCAT telegram.

The figure shows how the Sync Manager is linked to the system:

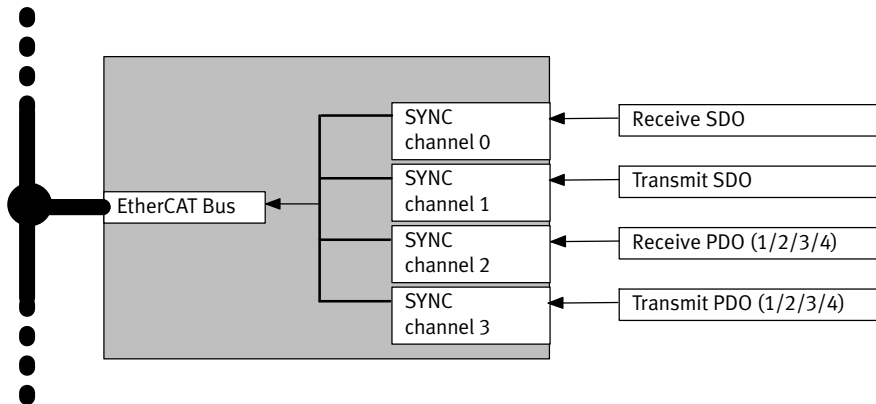


Fig. 4.2 Sample mapping of the SDOs and PDOs to the Sync channels

All objects are sent via so-called Sync channels. The data from these channels are automatically linked to the EtherCAT data flow and transmitted. The EtherCAT implementation in the motor controller CMMP-AS-...-M3 supports four such Sync channels.

For this reason, additional mapping of the SDOs and PDOs to the Sync channels is required compared with CANopen. This occurs via the so-called Sync Manager objects (objects 1C00_h and 1C10_h ... 1C13_h → chapter 4.6.2). These objects are described in more detail below.

These Sync channels are permanently allocated to the individual transfer types and cannot be changed by the user. The allocation is as follows:

- Sync channel 0: Mailbox telegram protocol for incoming SDOs (Master => Slave)
- Sync channel 1: Mailbox telegram protocol for outgoing SDOs (Master <= Slave)
- Sync channel 2: Process data telegram protocol for incoming PDOs (Master => Slave).
The object 1C12_h must be observed here.
- Sync channel 3: Process data telegram protocol for outgoing PDOs (Master <= Slave).
The object 1C13_h must be observed here.

The parametrisation of the individual PDOs is set via objects 1600_h to 1603_h (Receive PDOs) and 1A00_h to 1A03_h (Transmit PDOs). Parametrisation of the PDOs is carried out as described in chapter 3 “CANopen access procedure”.

The Sync channels can only be set and the PDOs only configured in the “Pre-Operational” status.



It is not intended to parameterise the slave under EtherCAT. The device description files are available for this purpose. They prescribe the total parametrisation, including PDO parametrisation, which is used by the master during initialisation.

All changes to the parametrisation should therefore not be made by hand, but in the device description files. For this purpose, sections of the device description files that are important for the user are described in more detail in section 4.12.



The Sync channels described here are NOT the same as the Sync telegrams familiar from CANopen. CANopen Sync telegrams can still be transmitted as SDOs via the SDO interface implemented under CoE, but do not directly influence the Sync channels described above.

4.6.2 New and revised objects under CoE

The following table contains an overview of the indices and subindices used for CANopen-compatible communication objects, which are inserted in the range from 1000_h to 1FFF_h for the EtherCAT fieldbus system. These primarily replace the communication parameters in accordance with CiA 301.

Object	Significance	Permitted with
1000 _h	Device type	Device control identifier
1018 _h	Identity object	Vendor ID, product code, revision, serial number
1100 _h	EtherCAT fixed station address	Fixed address assigned to the slave during initialisation by the master
1600 _h	1. RxPDO Mapping	Identifier of the 1st Receive PDO
1601 _h	2. RxPDO Mapping	Identifier of the 2nd Receive PDO
1602 _h	3. RxPDO Mapping	Identifier of the 3rd Receive PDO
1603 _h	4. RxPDO Mapping	Identifier of the 4th Receive PDO
1A00 _h	1. TxPDO Mapping	Identifier of the 1st Transmit PDO
1A01 _h	2. TxPDO Mapping	Identifier of the 2nd Transmit PDO
1A02 _h	3. TxPDO Mapping	Identifier of the 3rd Transmit PDO
1A03 _h	4. TxPDO Mapping	Identifier of the 4th Transmit PDO
1C00 _h	Sync Manager Communication Type	Object for configuring the individual Sync channels (SDO or PDO Transfer)
1C10 _h	Sync Manager PDO Mapping for Sync Channel 0	Assignment of the Sync channel 0 to a PDO/SDO (Channel 0 is always reserved for Mailbox Receive SDO Transfer)
1C11 _h	Sync Manager PDO Mapping for Sync Channel 1	Assignment of the Sync channel 1 to a PDO/SDO (Channel 1 is always reserved for Mailbox Send SDO Transfer)
1C12 _h	Sync Manager PDO Mapping for Sync Channel 2	Assignment of the Sync channel 2 to a PDO (Channel 2 is reserved for Receive PDOs)
1C13 _h	Sync Manager PDO Mapping for Sync Channel 3	Assignment of the Sync channel 3 to a PDO (Channel 3 is reserved for Transmit PDOs)

Tab. 4.7 New and revised communication objects

The subsequent chapters describe the objects 1C00_h and 1C10_h...1C13_h in more detail, as they are only defined and implemented under the EtherCAT CoE protocol and therefore are not documented in the CANopen manual for the motor controller CMMP-AS-...-M3.



The motor controller CMMP-AS-...-M3 with the EtherCAT interface supports four Receive PDOs (RxPDO) and four Transmit PDOs (TxPDO).

Objects 1008_h, 1009_h and 100A_h are not supported by the CMMP-AS-...-M3, as plain text strings cannot be read from the motor controller.

Object 1100_h - EtherCAT fixed station address

This object allocates a unique address to the slave during the initialisation phase. The object has the following significance:

Index	1100_h
Name	EtherCAT fixed station address
Object Code	Var
Data Type	uint16
Access	ro
PDO Mapping	no
Value Range	0 ... FFFF _h
Default Value	0

Object 1C00_h - Sync Manager Communication Type

This object allows the transfer type for the various channels of the EtherCAT Sync Manager to be read. As the CMMP-AS-...-M3 only supports the first four Sync channels under the EtherCAT CoE protocol, the following objects are “read only”.

The Sync Manager for the CMMP-AS-...-M3 is configured as a result. The objects have the following significance:

Index	1C00_h
Name	Sync Manager Communication Type
Object Code	Array
Data Type	uint8

Sub-Index	00_h
Description	Number of Used Sync Manager Channels
Access	ro
PDO Mapping	no
Value Range	4
Default Value	4

Sub-Index	01_h
Description	Communication Type Sync Channel 0
Access	ro
PDO Mapping	no
Value Range	2: Mailbox Transmit (Master => Slave)
Default Value	2: Mailbox Transmit (Master => Slave)

Sub-Index	02_h
Description	Communication Type Sync Channel 1
Access	ro
PDO Mapping	no
Value Range	2: Mailbox Transmit (Master <= Slave)
Default Value	2: Mailbox Transmit (Master <= Slave)

Index	03_h
Description	Communication Type Sync Channel 2
Access	ro
PDO Mapping	no
Value Range	0: unused 3: Process Data Output (RxPDO / Master => Slave)
Default Value	3

Sub-Index	04_h
Description	Communication Type Sync Channel 3
Access	ro
PDO Mapping	no
Value Range	0: unused 4: Process Data Input (TxPDO/Master <= Slave)
Default Value	4

Object 1C10_h - Sync Manager Channel 0 (Mailbox Receive)

This object allows a PDO to be configured for Sync channel 0. As Sync channel 0 is always allocated to the mailbox telegram protocol, the user cannot change this object. The object therefore always has the following values:

Index	1C10_h
Name	Sync Manager Channel 0 (Mailbox Receive)
Object Code	Array
Data Type	uint8

Sub-Index	00_h
Description	Number of assigned PDOs
Access	ro
PDO Mapping	no
Value Range	0 (no PDO assigned to this channel)
Default Value	0 (no PDO assigned to this channel)



The name “Number of assigned PDOs” assigned by the EtherCAT specification for Sub-index 0 of these objects is confusing here, as Sync Manager channels 0 and 1 are always allocated through the mailbox telegram. SDOs are always transmitted in this telegram type under EtherCAT CoE. Sub-index 0 of these two objects is therefore unused.

Object 1C11_h - Sync Manager Channel 1 (Mailbox Send)

This object allows a PDO to be configured for Sync channel 1. As Sync channel 1 is always allocated to the mailbox telegram protocol, the user cannot change this object. The object therefore always has the following values:

Index	1C11_h
Name	Sync Manager Channel 1 (Mailbox Send)
Object Code	Array
Data Type	uint8

Sub-Index	00_h
Description	Number of assigned PDOs
Access	ro
PDO Mapping	no
Value Range	0 (no PDO assigned to this channel)
Default Value	0 (no PDO assigned to this channel)

Object 1C12_h - Sync Manager Channel 2 (Process Data Output)

This object allows a PDO to be configured for Sync channel 2. Sync channel 2 is permanently assigned for the reception of Receive PDOs (Master ⇒ Slave). In this object, the number of PDOs assigned to this Sync channel must be set under sub-index 0.

The object number of the PDO to be allocated to the channel is subsequently entered in sub-indices 1 to 4. Only the object numbers of the previously configured Receive PDOs can be used for this (object 1600_h ... 1603_h).

In the current implementation, the data of the objects below is not evaluated further by the firmware of the motor controller.

The CANopen configuration of the PDOs is used for evaluation under EtherCAT.

Index	1C12_h
Name	Sync Manager Channel 2 (Process Data Output)
Object Code	Array
Data Type	uint8

Sub-Index	00_h
Description	Number of assigned PDOs
Access	rw
PDO Mapping	no
Value Range	0: no PDO assigned to this channel 1: one PDO assigned to this channel 2: two PDOs assigned to this channel 3: three PDOs assigned to this channel 4: four PDOs assigned to this channel
Default Value	0 :no PDO assigned to this channel

Sub-Index	01_h
Description	PDO Mapping object Number of assigned RxPDO
Access	rw
PDO Mapping	no
Value Range	1600 _h : first Receive PDO
Default Value	1600 _h : first Receive PDO

Sub-Index	02_h
Description	PDO Mapping object Number of assigned RxPDO
Access	rw
PDO Mapping	no
Value Range	1601 _h : second Receive PDO
Default Value	1601 _h : second Receive PDO

Sub-Index	03_h
Description	PDO Mapping object Number of assigned RxPDO
Access	rw
PDO Mapping	no
Value Range	1602 _h : third Receive PDO
Default Value	1602 _h : third Receive PDO

Sub-Index	04_h
Description	PDO Mapping object Number of assigned RxPDO
Access	rw
PDO Mapping	no
Value Range	1603 _h : fourth Receive PDO
Default Value	1603 _h : fourth Receive PDO

Object 1C13_h - Sync Manager Channel 3 (Process Data Input)

This object allows a PDO to be configured for Sync channel 3. Sync channel 3 is permanently assigned for sending Transmit PDOs (Master <= Slave). In this object, the number of PDOs assigned to this Sync channel must be set under sub-index 0.

The object number of the PDO to be allocated to the channel is subsequently entered in sub-indices 1 to 4. Only the object numbers of the previously configured Transmit PDOs can be used for this (1A00_h to 1A03_h).

Index	1C13_h
Name	Sync Manager Channel 3 (Process Data Input)
Object Code	Array
Data Type	uint8

Sub-Index	00_h
Description	Number of assigned PDOs
Access	rw
PDO Mapping	no
Value Range	0: no PDO assigned to this channel 1: one PDO assigned to this channel 2: two PDOs assigned to this channel 3: three PDOs assigned to this channel 4: four PDOs assigned to this channel
Default Value	0: no PDO assigned to this channel

Sub-Index	01_h
Description	PDO Mapping object Number of assigned TxPDO
Access	rw
PDO Mapping	no
Value Range	1A00 _h : first Transmit PDO
Default Value	1A00 _h : first Transmit PDO

Sub-Index	02_h
Description	PDO Mapping object Number of assigned TxPDO
Access	rw
PDO Mapping	no
Value Range	1A01 _h : second Transmit PDO
Default Value	1A01 _h : second Transmit PDO

Sub-Index	03_h
Description	PDO Mapping object Number of assigned TxPDO
Access	rw
PDO Mapping	no
Value Range	1A02 _h : third Transmit PDO
Default Value	1A02 _h : third Transmit PDO

Sub-Index	04_h
Description	PDO Mapping object Number of assigned TxPDO
Access	rw
PDO Mapping	no
Value Range	1A03 _h : fourth Transmit PDO
Default Value	1A03 _h : fourth Transmit PDO

4.6.3 Objects not supported under CoE

When connecting the CMMP-AS-...-M3 under “CANopen over EtherCAT”, some CANopen objects, which are available under a direct connection of the CMMP-AS-...-M3 via CiA 402, are not supported. These objects are listed in the table below:

Identifier	Name	Significance
1008 _h	Manufacturer Device Name (String)	Device name (object is not available)
1009 _h	Manufacturer Hardware Version (String)	HW version (object is not available)
100A _h	Manufacturer Software Version (String)	SW version (object is not available)
6089 _h	position_notation_index	Specifies the number of decimal places for displaying the position values in the controller. The object is only available as a data container. The firmware is not evaluated further.
608A _h	position_dimension_index	Specifies the unit for displaying the position values in the controller. The object is only available as a data container. The firmware is not evaluated further.
608B _h	velocity_notation_index	Specifies the number of decimal places for displaying the velocity values in the controller. The object is only available as a data container. The firmware is not evaluated further.
608C _h	velocity_dimension_index	Specifies the unit for displaying the velocity values in the controller. The object is only available as a data container. The firmware is not evaluated further.
608D _h	acceleration_notation_index	Specifies the number of decimal places for displaying the acceleration values in the controller. The object is only available as a data container. The firmware is not evaluated further.
608E _h	acceleration_dimension_index	Specifies the unit for displaying the acceleration values in the controller. The object is only available as a data container. The firmware is not evaluated further.

Tab. 4.8 Unsupported communication objects

4.7 Communication Finite State Machine

As in almost all fieldbus interfaces for motor controllers, the connected slave (in this case the motor controller CMMP-AS-...-M3) must first be initialised by the master before it can be used by the master in an application. For this purpose, a finite state machine is defined for communication, to specify a fixed sequence of actions for this initialisation process.

A finite state machine is also defined for the EtherCAT interface. Changes between the individual statuses of the finite state machine may only occur between specific statuses, and are always initiated by the master. Slaves may not implement status changes independently. The individual statuses and the permitted status changes are described in the following tables and figures.

Status	Description
Power ON	The device has been switched on. It initialises itself and switches directly to the "Init" status.
Init	In this status, the EtherCAT fieldbus is synchronised by the master. This includes setting up the asynchronous communication between master and slave (mailbox telegram protocol). There is no direct communication between the master and slave yet. The configuration starts, saved values are loaded. When all devices are connected to the bus and configured, the status switches to "Pre-Operational".
Pre-Operational	In this status, asynchronous communication between the master and slave is active. The master uses this status to set up possible cyclic communication via PDOs and use acyclic communication for necessary parametrisation. If this status runs without errors, the master switches to the "Safe-Operational" status.
Safe-Operational	This status is used to set all equipment connected to the EtherCAT bus to a safe status. The slave sends up-to-date actual values to the master but ignores new setpoint values from the master and uses safe default values instead. If this status runs without errors, the master switches to the "Operational" status.
Operational	In this status, both acyclic and cyclic communication are active. Masters and slaves exchange target and actual value data. In this status, the CMMP-AS-...-M3 can be enabled and travel via the CoE protocol.

Tab. 4.9 Statuses of communication finite state machine

Only transitions in accordance with Fig. 4.3 are permitted between the individual statuses of the communication finite state machine:

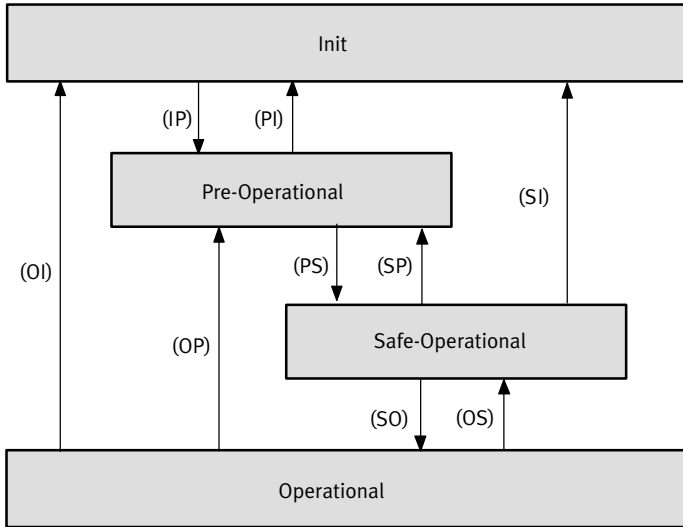


Fig. 4.3 Communication finite state machine

The transitions are described individually in the following table.

Status transition	Status
IP	Start of acyclic communication (mailbox telegram protocol)
PI	Stop of acyclic communication (mailbox telegram protocol)
PS	Start Inputs Update: start of cyclic communication (process data telegram protocol) Slave sends actual values to master. The slave ignores setpoint values from the master and uses internal default values.
SP	Stop Input Update: stop of cyclic communication (process data telegram protocol). The slave no longer sends actual values to the master.
SO	Start Output Update: The slave evaluates up-to-date setpoint specifications from the master.
OS	Stop Output Update: The slave ignores setpoint values from the master and uses internal default values.
OP	Stop Output Update, Stop Input Update: stop of cyclic communication (process data telegram protocol). The slave no longer sends actual values to the master, and the master no longer sends setpoint values to the slave.

Status transition	Status
SI	Stop Input Update, Stop Mailbox Communication: Stop of cyclic communication (process data telegram protocol) and stop of acyclic communication (mailbox telegram protocol). The slave no longer sends actual values to the master, and the master no longer sends setpoint values to the slave.
OI	Stop Output Update, Stop Input Update, Stop Mailbox Communication: Stop of cyclic communication (process data telegram protocol) and stop of acyclic communication (mailbox telegram protocol). The slave no longer sends actual values to the master, and the master no longer sends setpoint values to the slave.

Tab. 4.10 Status transitions



In the EtherCAT finite state machine, the “Bootstrap” status is also specified in addition to the statuses listed here. This status is not implemented for the motor controller CMMP-AS...-M3.

4.7.1 Differences between the finite state machines of CANopen and EtherCAT

When operating the CMMP-AS...-M3 via the EtherCAT-CoE protocol, the EtherCAT finite state machine is used instead of the CANopen NMT finite state machine. This differs from the CANopen finite state machine in several aspects. These different characteristics are listed below:

- No direct transition from Pre-Operational after Power On
- No Stopped status, direct transition to the INIT status
- Additional status: Safe-Operational

The following table compares the different statuses:

EtherCAT State	CANopen NMT State
Power ON	Power-On (initialisation)
Init	Stopped
Safe-Operational	–
Operational	Operational

Tab. 4.11 Comparison of the statuses for EtherCAT and CANopen

4.8 SDO Frame

All data of an SDO transfer are transmitted via SDO frames in CoE. These frames have the following structure:

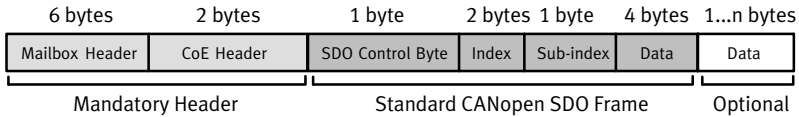


Fig. 4.4 SDO Frame: telegram structure

Element	Description
Mailbox Header	Data for mailbox communication (length, address and type)
CoE Header	Identifier of the CoE service
SDO Control Byte	Identifier for a read or write command
Index	Main index of the CANopen communication object
Sub-index	Sub-index of the CANopen communication object
Data	Data content of the CANopen communication object
Data (optional)	Additional optional data. This option is not supported by the motor controller CMMP-AS-...-M3, as only standard CANopen objects can be addressed. The maximum size of these objects is 32 bits.

Tab. 4.12 SDO Frame: elements

In order to transmit a standard CANopen object via one of these SDO frames, the actual CANopen SDO frame is packaged in an EtherCAT SDO frame and transmitted.

Standard CANopen SDO frames can be used for:

- Initialisation of the SDO download
- Download of the SDO segment
- Initialisation of the SDO upload
- Upload of the SDO segment
- Abort of the SDO transfer
- SDO upload expedited request
- SDO upload expedited response
- SDO upload segmented request (max. 1 segment with 4 bytes of user data)
- SDO upload segmented response (max. 1 segment with 4 bytes of user data)



All above-mentioned transfer types are supported by the motor controller CMMP-AS-...-M3.

As the use of the CoE implementation of the CMMP-AS-...-M3 only allows the standard CANopen objects to be addressed, whose size is restricted to 32 bits (4 bytes), only transfer types with a maximum data length of up to 32 bits (4 bytes) are supported.

4.9 PDO Frame

Process Data Objects (PDO) are used for cyclic transmission of setpoint values and actual values between master and slave. They must be configured in the “Pre-Operational” status by the master before the slave is operated. They are then transmitted in PDO frames. These PDO frames have the following structure:

All data of a PDO transfer are transmitted via PDO frames in CoE. These frames have the following structure:

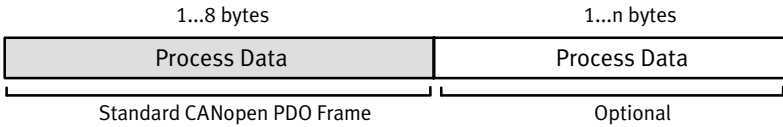


Fig. 4.5 PDO Frame: telegram structure

Element	Description
Process Data	Data content of the PDO (Process Data Object)
Process Data (optional)	Optional data content of additional PDOs

Tab. 4.13 PDO Frame: elements

To transmit a PDO via the EtherCAT-CoE protocol, in addition to the PDO configuration (PDO Mapping), the Transmit and Receive PDOs must be assigned to a transmission channel of the Sync Manager (→ chapter 4.6.1 “Configuration of the Communication Interface”). The data exchange of PDOs for the motor controller CMMP-AS-...-M3 takes place exclusively via the EtherCAT process data telegram protocol.



The transfer of CANopen process data (PDOs) via acyclic communication (mailbox telegram protocol) is not supported by the motor controller CMMP-AS-...-M3.

As all data exchanged via the EtherCAT CoE protocol are forwarded directly to the internal CANopen implementation in the motor controller CMMP-AS-...-M3, the PDO mapping is also implemented as described in chapter 3.3 “PDO Message”. The figure below depicts this process:

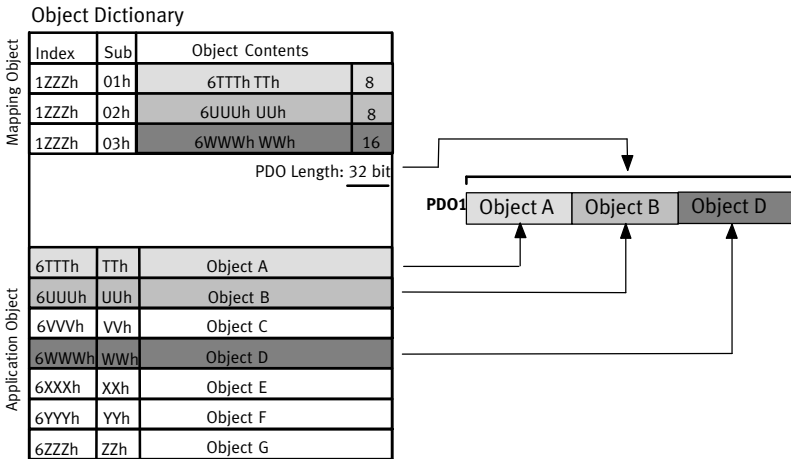


Fig. 4.6 PDO Mapping

The simple forwarding of the data received via CoE to the CANopen protocol implemented in CMMP-AS-...-M3 means that the “Transmission Types” of the PDOs available for the CANopen protocol for the CMMP-AS-...-M3 can be used in addition to CANopen object mapping for the PDOs to be parameterised.

The motor controller CMMP-AS-...-M3 also supports the “Sync Message” transmission type. However, the Sync Message does not have to be sent via EtherCAT.

It is used either for the arrival of the telegram or the hardware synchronisation pulse of the “Distributed Clocks” mechanism (see below) for data transfer.

The EtherCAT interface for CMMP-AS-...-M3 supports synchronisation via the “Distributed Clocks” mechanism specified under EtherCAT by means of the use of FPGA module ESC20. The current regulator of the motor controller CMMP-AS-...-M3 is synchronised to this pulse, and the correspondingly configured PDOs are evaluated or sent.

The motor controller CMMP-AS-...-M3 with the EtherCAT interface supports the following functions:

- Cyclic PDO frame telegram via the process data telegram protocol.
- Synchronous PDO frame telegram via the process data telegram protocol.

The motor controller CMMP-AS-...-M3 with the EtherCAT interface supports four Receive PDOs (RxPDO) and four Transmit PDOs (TxPDO).

4.10 Error Control

The EtherCAT CoE implementation for the motor controller CMMP-AS-...-M3 monitors the following error statuses of the EtherCAT fieldbus:

- FPGA is not ready when the system is started.
- A bus error has occurred.
- An error has occurred on the mailbox channel. The following errors are monitored in this case:
 - An unknown service is requested.
 - A protocol other than CANopen over EtherCAT (CoE) is to be used.
 - An unknown Sync Manager is addressed.

All of these errors are defined as corresponding error codes for the motor controller CMMP-AS-...-M3. If one of the above-mentioned errors occurs, it is transmitted to the controller via a “Standard Emergency Frame”. See also Chapter 4.11 “Emergency Frame” and Chapter B “Diagnostic messages”.

The CMMP-AS-...-M3/-M0 motor controller with the EtherCAT interface supports the following function:

- Application Controller determines a defined error message number as a result of an event (Error Control Frame telegram from the controller).

4.11 Emergency Frame

The master and slaves exchange error messages via the EtherCAT CoE emergency frame. The CoE emergency frames are used for direct transfer of the “Emergency Messages” defined under CANopen. The CANopen telegrams are simply tunnelled through the CoE emergency frames, as is the case for SDO and PDO transmission.

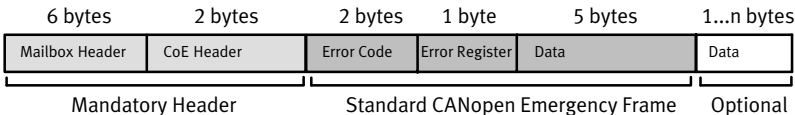


Fig. 4.7 Emergency Frame: telegram structure

Element	Description
Mailbox Header	Data for mailbox communication (length, address and type)
CoE Header	Identifier of the CoE service
Error Code	Error Code of the CANopen EMERGENCY Message → Chapter 3.5.2
Error Register	Error Register of the CANopen EMERGENCY Message → Tab. 3.9
Data	Data content of the CANopen EMERGENCY Message
Data (optional)	Additional optional data. As only the standard CANopen emergency frames are supported in the CoE implementation for the motor controller CMMP-AS-...-M3, the “Data (optional)” field is not supported.

Tab. 4.14 Emergency Frame: elements

As the “Emergency Messages” received and sent via CoE are simply forwarded to the CANopen protocol implemented in the motor controller, all error messages can be looked up in the chapter B.

4.12 XML Device Description File

In order to connect EtherCAT slave devices simply to an EtherCAT master, there must be a description file for every EtherCAT slave device. This description file is comparable to the EDS files for the CANopen fieldbus system or the GSD files for Profibus. In contrast to the latter, the EtherCAT description file is in the XML format, as is often used for internet and web applications, and contains information on the following features of the EtherCAT slave devices:

- Information on the device manufacturer
- Name, type and version number of the device
- Type and version number of the protocol to be used for this device (e.g. CANopen over Ethernet, ...)
- Parametrisation of the device and configuration of the process data

This file contains the complete parametrisation of the slave, including the parametrisation of the Sync Manager and the PDOs. For this reason, the configuration of the slave can be changed using this file. Festo has created a device description file for the CMMP-AS-...-M3 motor controller. It can be downloaded from the Festo homepage. Its contents will now be described in more detail to permit users to adapt this file to their application.

The available device description file supports both the CiA 402 profile and the FHPP profile via separately selectable modules.

The XML files are included on the CD-ROM supplied with the motor controller.

XML file	Description
Festo_CMMP-AS_V4p0_FHPP.xml	Motor controller CMMP-AS-...-M3 with protocol "FHPP"
Festo_CMMP-AS_V4p0_CIA402_IP7.xml	Motor controller CMMP-AS-...-M3 with protocol "CiA 402"

Tab. 4.15 XML file



You will find the most current version under → www.festo.com/sp

4.12.1 Fundamental structure of the device description file

The EtherCAT device description file is in the XML format. This format has the advantage that it can be read and edited in a standard text editor. An XML file always describes a tree structure. It defines the individual branches via nodes. These nodes have a start and end marking. Each node can contain any number of sub-nodes.

EXAMPLE: Rough explanation of the fundamental structure of an XML file:

```

<EtherCATInfo Version="0.2">
  <Vendor>
    <Id>#x1D</Id>
    <Name>Festo SE</Name>
    <ImageData16x14>424DD60200.....</ImageData16x14>
  </Vendor>
  <Descriptions>
    <Groups>
      <Group SortOrder="1">
        <Type>Festo Electric-Drives</Type>
        <Name LcId="1033">Festo Electric-Drive</Name>
      </Group>
    </Groups>
    <Devices>
      <Device Physics="YY">
        </Device>
      </Devices>
    </Descriptions>
  </EteherCATInfo>

```

The following brief rules must be observed for the structure of an XML file:

- Each node must have a unique name.
- Each node opens with <node name> and closes with </node name>.

The device description file for the motor controller CMMP-AS-...-M3 under EtherCAT CoE is divided into the following sub-items:

Node name	Significance	Adaptable
Vendor	This node contains the name and the ID of the manufacturer of the device to which this description file belongs. It also contains the binary code of a bitmap with the manufacturer's logo.	No
Description	This sub-item contains the actual device description including the configuration and initialisation.	Partially
Group	This node contains the assignment of the device to a device group. These groups are specified and may not be changed by the user.	No
Devices	This sub-item contains the actual description of the device.	Partially

Tab. 4.16 Nodes of the device description file

The following table describes only the subnodes of the "Description" node that are required for parameterisation of the motor controller CMMP-AS-...-M3 under CoE. All other nodes are fixed and may not be changed by the user.

Node name	Significance	Adaptable
RxPDO Fixed=...	This node contains the PDO Mapping and the assignment of the PDO to the Sync Manager for Receive PDOs.	Yes
TxPDO Fixed=...	This node contains the PDO Mapping and the assignment of the PDO to the Sync Manager for Transmit PDOs.	Yes
Mailbox	This node allows commands to be defined that are transmitted to the slave via SDO transfers by the master during the phase transition from “Pre-Operational” to “Operational”.	Yes

Tab. 4.17 Subnode of the “Descriptions” node

As only the nodes from the table above are important for users to adapt the device description file, they are described in detail in the following chapters. The remaining content of the device description file is fixed and may not be changed by the user.

**Important:**

If changes are made to nodes and content other than the RxPDO, TxPDO and Mailbox nodes in the device description file, error-free operation of the device can no longer be guaranteed.

4.12.2 Receive PDO configuration in the RxPDO node

The RxPDO node is used to specify the mapping for the Receive PDOs and their assignment to a channel of the Sync Manager. A typical entry in the device description file for the motor controller CMMP-AS-...-M3 can be as follows:

```
<RxPDO Fixed="1" Sm="2">
  <Index>#x1600</Index>
  <Name>Outputs</Name>
  <Entry>
    <Index>#x6040</Index>
    <SubIndex>0</SubIndex>
    <BitLen>16</BitLen>
    <Name>Controlword</Name>
    <DataType>UINT</DataType>
  </Entry>
  <Entry>
    <Index>#x6060</Index>
    <SubIndex>0</SubIndex>
    <BitLen>8</BitLen>
    <Name>Mode_Of_Operation</Name>
    <DataType>USINT</DataType>
  </Entry>
</RxPDO>
```

As the example above shows, the entire mapping of the Receive PDO is described in detail in such entries. The first large block specifies the object number of the PDO and its type. This is followed by a list of all CANopen objects which are to be mapped to the PDO.

The table below describes the individual entries in greater detail:

Node name	Significance	Adaptable
RxPDO Fixed="1" Sm="2"	This node describes the properties of the Receive PDO directly and its assignment to the Sync Manager. The Fixed="1" entry indicates that the object mapping cannot be revised. The Sm="2" entry indicates that the PDO is to be allocated to Sync channel 2 of the Sync Manager.	No
Index	This entry contains the object number of the PDO. The first Receive PDO under object number 0x1600 is configured here.	Yes
Name	The name indicates that this PDO is a Receive PDO (outputs) or a Transmit PDO (inputs). This value must always be set to "Output" for a Receive PDO.	No
Entry	Each Entry node contains a CANopen object to be mapped to the PDO. An Entry node contains the index and sub-index of the CANopen object to be mapped, as well as their name and data type.	Yes

Tab. 4.18 Elements of the node "RxPDO"

The sequence and mapping of the individual CANopen objects for the PDO correspond to the sequence in which they are specified via the "Entry" entries in the device description file. The individual sub-items of an "Entry" node are specified in the following table:

Node name	Significance	Adaptable
Index	This entry specifies the index of the CANopen object to be mapped to the PDO.	Yes
Sub-index	This entry specifies the sub-index of the CANopen object to be mapped.	Yes
BitLen	This entry specifies the size of the object to be mapped in bits. This entry must always correspond to the type of the object to be mapped. Allowed: 8 Bit / 16 Bit / 32 Bit.	Yes
Name	This entry specifies the name of the object to be mapped as a string.	Yes
Data Type	This entry specifies the data type of the object to be mapped. This can be taken from the respective description for the individual CANopen objects.	Yes

Tab. 4.19 Elements of the node "Entry"

4.12.3 Transmit PDO configuration in the TxPDO node

The TxPDO node is used to specify the mapping for the Transmit PDOs and their assignment to a channel of the Sync Manager. The configuration corresponds to that of the Receive PDOs from section 4.12.2 “Receive PDO configuration in node RxPDO” with the difference that the node “Name” of the PDO must be set to “Inputs”, not “Outputs”.

4.12.4 Initialisation commands via the “Mailbox” node

The “Mailbox” node in the device description file is used to describe CANopen objects via the master in the slave during the initialisation phase. The commands and objects to be described there are specified via special entries. These entries specify the phase transition at which this value is to be written. Furthermore, such entries contain the object number (index and sub-index) as well as the data value to be written and comments.

A typical entry has the following form:

```
<InitCmd>
  <Transition>PS</Transition>
  <Index#>x6060</Index>
  <SubIndex>0</SubIndex>
  <Data>03</Data>
  <Comment>velocity mode</Comment>
</InitCmd>
```

In the example above, status transition PS from “Pre-Operational” to “Safe Operational” sets the operating mode in the CANopen object “modes_of_operation” to “Speed adjustment”. The individual sub-nodes have the following significance:

Node name	Significance	Adaptable
Transition	Name of the status transition for which this command should be executed (→ chapter 4.7 “Communication Finite State Machine”)	Yes
Index	Index of the CANopen object to be written	Yes
Sub-index	Sub-index of the CANopen object to be written	Yes
Data	Data value to be written, as a hexadecimal value	Yes
Comment	Comment on this command	Yes

Tab. 4.20 Elements of the node “InitCmd”



Important:

In a device description file for the motor controller CMMP-AS-...-M3, some entries in this section are already assigned. These entries must remain as they are and may not be changed by the user.

4.13 Synchronisation (Distributed Clocks)

Time synchronisation is implemented via so-called “Distributed Clocks” in EtherCAT. Each EtherCAT slave receives a real-time clock, which is synchronised in all slaves by the clock master during the initialisation phase. The clocks in all slaves are then adjusted during operation. The clock master is the first slave in the network.

This provides a uniform time base in the entire system with which the individual slaves can synchronise. The Sync telegrams provided for this purpose under CANopen are unnecessary under CoE.

The FPGA ESC20 used in the motor controller CMMP-AS-...-M3 supports Distributed Clocks. This facilitates extremely precise time synchronisation. The cycle time of the EtherCAT Frame must exactly match the cycle time t_p of the controller-internal interpolator. If necessary, the interpolator time must be adjusted via the object included in the device description file.

In the present implementation, synchronous transfer of PDO data and synchronisation of the controller-internal PLL to the synchronous data framework of the EtherCAT Frame can be implemented even without Distributed Clocks. For this purpose, the firmware uses the arrival of the EtherCAT Frame as a time base.

The following restrictions apply:

- The master must be able to send the EtherCAT frames with an extremely low jitter ($< 70\%$ of the set current controller cycle time).
- The cycle time of the EtherCAT Frame must exactly match the cycle time t_p of the controller-internal interpolator.
- The Ethernet must be available exclusively for the EtherCAT Frame. It may be necessary to synchronise other telegrams to the grid, as they may not block the bus.

5 Setting parameters

Before the motor controller can carry out the desired task (torque regulation, speed adjustment, positioning), numerous parameters of the motor controller must be adapted to the motor used and the specific application. The sequence in the subsequent chapters should be followed thereby. After setting of the parameters, device control and use of the various operating modes are explained.



The display of the motor controller shows an “A” (Attention) if the motor controller has not been parametrised appropriately yet. If the motor controller is supposed to be parametrised completely over CANopen, you must write over the object 6510_h_C0_h in order to suppress this display (→ page 146).

Besides the parameters described in depth here, the object directory of the motor controller contains other parameters that have to be implemented in accordance with CANopen. But they normally do not include any information that can be used in designing an application with a motor controller CMMP-AS-...-M3/-M0 in a sensible way. If required, read about this in the specifications of CiA.

5.1 Loading and Saving Parameter Sets

Overview

The motor controller has three parameter sets:

- **Current parameter set**
This parameter set is located in the volatile memory (RAM) of the motor controller. It can be read and written on as desired with the parametrisation software or via the CAN bus. When the motor controller is switched on, the application parameter set is copied into the current parameter set.
- **Default parameter set**
This is the parameter set of the motor controller provided standard by the manufacturer and is unchangeable. Through a write process into the CANopen object 1011_h_01_h (restore_all_default_parameters), the default parameter set can be copied into the current parameter set. This copying process is only possible when the output stage is switched off.
- **Application parameter set**
The current parameter set can be stored in the non-volatile flash memory. The storage process can be triggered with a read access to the CANopen object 1010_h_01_h (save_all_parameters). When the motor controller is switched on, the application parameter set is automatically copied into the current parameter set.

The following diagram illustrates the connections between the individual parameter sets.

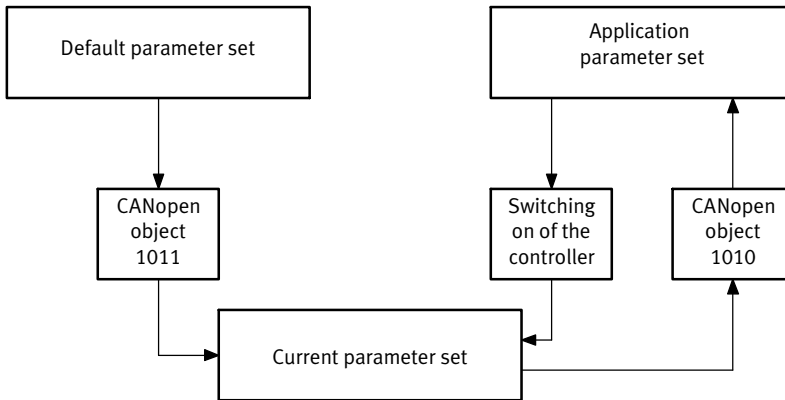


Fig. 5.1 Connections between parameter sets

Two different concepts are conceivable for administering parameter sets:

1. The parameter set is created with the parametrisation software and transmitted completely into the individual controllers. With this procedure, only the objects accessible via CANopen have to be put in via the CAN bus. A disadvantage here is that the parameterisation software is needed for each start-up of a new machine or in case of a repair (controller exchange).
2. This variant is based on the fact that most application-specific parameter sets deviate from the default parameter set in only a few parameters. This makes it possible for the current parameter set to be newly constructed via the CAN bus each time the system is switched on. To do this, the higher level controller first loads the default parameter set (call-up of the CANopen object 1011_h_01_h (restore_all_default_parameters)). After that, only the differing objects are transmitted. The entire procedure lasts less than 1 second per controller. An advantage is that this procedure also works for unparametrised controllers, so that the commissioning of new systems or replacement of individual controllers is unproblematic, and parametrisation software is not required for this.



Warning

Before the output stage is switched on for the first time, make sure the controller really includes the parameters you want.

An incorrectly parametrised controller can turn out of control and cause personal injury or property damage.

Description of the Objects**Object 1011_h: restore_default_parameters**

Index	1011_h
Name	restore_parameters
Object Code	ARRAY
No. of Elements	1
Data Type	UINT32

Sub-Index	01_h
Description	restore_all_default_parameters
Access	rw
PDO mapping	no
Units	–
Value Range	64616F6C _h (“load”)
Default Value	1 (read access)

Signature	MSB			LSB
ASCII	d	a	o	l
Hex	64 _h	61 _h	6F _h	6C _h

Tab. 5.1 Example for ASCII text “load”

The object 1011_h_01_h (restore_all_default_parameters) makes it possible to put the current parameter set into a defined state. To achieve this, the default parameter set is copied into the current parameter set. The copying process is triggered by a write access to this object, whereby the string “load” must be transferred as a record in hexadecimal form.

This command is only carried out with a deactivated output stage. Otherwise, the SDO error “Data cannot be transmitted or stored, since the motor controller for this is not in the correct state” is generated. If the incorrect identifier is sent, the error “Data cannot be transmitted or stored” is generated. If the object is accessed by reading, a 1 is returned to show that resetting to default values is supported. The CAN communication parameters (node no., baud rate and operating mode) as well as numerous angle encoder settings (some of which require a reset to become effective) remain unchanged.

Object 1010_h: store_parameters

Index	1010_h
Name	store_parameters
Object Code	ARRAY
No. of Elements	1
Data Type	UINT32

Sub-Index	01 _h
Description	save_all_parameters
Access	rw
PDO mapping	no
Units	–
Value Range	65766173 _h (“save”)
Default Value	1

Signature	MSB			LSB
ASCII	e	v	a	s
Hex	65 _h	76 _h	61 _h	73 _h

Tab. 5.2 Example for ASCII text “save”

If the default parameter set should also be taken over into the application parameter set, the object 1010_h_01_h (save_all_parameters) must also be called up.

If the object is written via an SDO, the default behaviour is that the SDO is answered immediately. The answer thus does not reflect the end of the storage procedure. But this behaviour can be revised through the object 6510_h_F0_h (compatibility_control).

5.2 Compatibility settings

Overview

In order to remain compatible with earlier CANopen implementations (e.g. also in other device families) and still be able to execute changes and corrections compared to CiA 402 and CiA 301, the object compatibility_control was introduced. In the default parameter set, this object delivers 0, that is, compatibility with earlier versions. For new applications, we recommend setting the defined bits to permit as much agreement as possible with the named standards.

Description of the Objects

Objects treated in this chapter

Index	Object	Name	Type	Attr.
6510_F0 _h	VAR	compatibility_control	UINT16	rw

Object 6510_h_F0_h: compatibility_control

Sub-Index	F0 _h
Description	compatibility_control
Data Type	UINT16
Access	rw
PDO mapping	no
Units	–
Value Range	0 ... 1FF _h , → Table
Default Value	0

Bit	Value	Name	
0	0001 _h	homing_method_scheme	The bit has the same significance as bit 2 and is available for compatibility reasons. If bit 2 is set, this bit is also set and vice versa.
1	0002 _h	Reserved	The bit is reserved. It must not be set.
2	0004 _h	homing_method_scheme	If this bit is set, the homing methods 32 ... 35 are numbered in accordance with CiA 402; otherwise, the numbering is compatible with earlier implementations. (→ also chap. 7.2.3). If this bit is set, bit 0 is also set and vice versa.
3	0008 _h	Reserved	The bit is reserved. It must not be set.
4	0010 _h	response_after_save	If this bit is set, the answer to save_all_parameters is not sent until saving is completed. This can take several seconds, which might result in a time-out in the controller. If the bit is deleted, answering takes place immediately; but it should be considered that the saving procedure has not been completed yet.
5	0020 _h	Reserved	The bit is reserved. It must not be set.
6	0040 _h	homing_to_zero	Up to now, homing under CANopen consists of only two phases (search travel and creep travel). The drive then does not travel to the determined zero position (which may have been shifted to the reference position found through homing_offset, for example). If this bit is set, this standard behaviour is revised and the drive performs travel to zero after homing → chap. 7.2 Operating mode reference travel (homing mode).
7	0080 _h	device_control	If this bit is set, bit 4 of the status word (voltage_enabled) is output in accordance with CiA 402 v2.0. In addition, the status FAULT_REACTION_ACTIVE is distinguishable from the FAULT status. → see chapter 6
8	0100 _h	Reserved	The bit is reserved. It must not be set.

5.3 Conversion factors (factor group)

Overview

Motor controllers are used in a number of applications: as direct drive, with following gear, for linear drive, etc. To permit easy parametrisation, the motor controller can be parametrised with the help of the factor group so that the user can specify or read out all variables, such as speed, directly in the desired units at the output (e.g. with a linear axis position value in millimetres and speeds in millimetres per second). The motor controller then uses the factor group to calculate the entries in its internal units of measurement. For each physical variable, (position, speed and acceleration), there is a conversion factor available to adapt the user units to the own application. The units set through the factor group are generally designated position_units, speed_units or acceleration_units. The following sketch illustrates the function of the factor group:

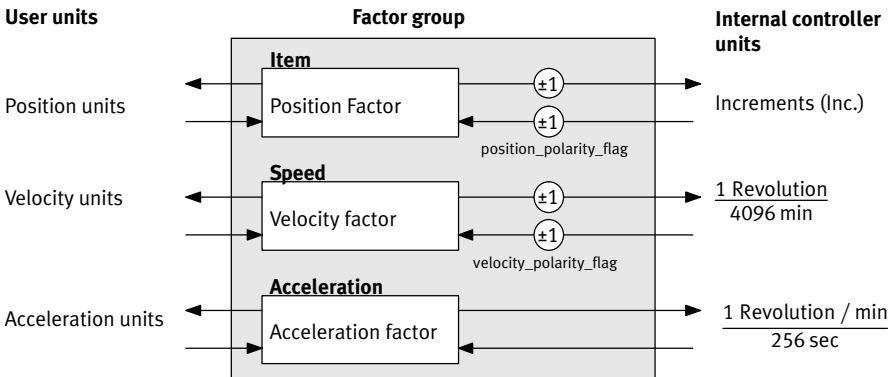


Fig. 5.2 Factor group

All parameters are stored in the motor controller in its internal units and only converted with the help of the factor group when being written in or read out.

For that reason, the factor group should be set before the first parameter setting and not changed during parameter setting.

The factor group is set to the following units by default:

Size	Measurement file	Unit	Explanation
Length	Position units	Increments	65536 increments per revolution
Speed	Velocity units	min ⁻¹	Revolutions per minute
Acceleration	Acceleration units	(min ⁻¹)/s	Rotational speed increase per second

Tab. 5.3 Factor group default settings

Description of the Objects

Objects treated in this chapter

Index	Object	Name	Type	Attr.
607E _h	VAR	polarity	UINT8	rw
6093 _h	ARRAY	position_factor	UINT32	rw
6094 _h	ARRAY	velocity_encoder_factor	UINT32	rw
6097 _h	ARRAY	acceleration_factor	UINT32	rw

Object 6093_h: position_factor

The object position_factor converts all units of length of the application from position_units into the internal unit increments (65536 increments correspond to 1 revolution). It consists of numerator and denominator.

Motor with gearing

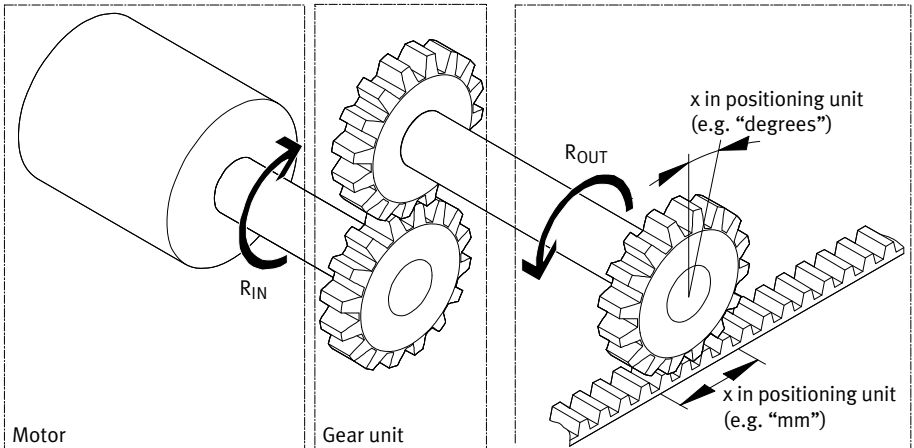


Fig. 5.3 Calculating the positioning units

Index	6093_h
Name	position_factor
Object Code	ARRAY
No. of Elements	2
Data Type	UINT32

Sub-Index	01_h
Description	numerator
Access	rw
PDO mapping	yes
Units	–
Value Range	–
Default Value	1

Sub-Index	02_h
Description	divisor
Access	rw
PDO mapping	yes
Units	–
Value Range	–
Default Value	1

The following parameters are involved in the calculation formula of the position_factor:

Parameters	Description
gear_ratio	Gear ratio between revolutions at the drive-in (RIN) and revolutions at the drive-out (ROUT)
feed_constant	Ratio between revolutions at the drive-out (ROUT) and movement in position_units (e.g. 1 R = 360 degrees)

Tab. 5.4 Position factor parameters

The position_factor is calculated using the following formula:

$$\text{position_factor} = \frac{\text{numerator}}{\text{divisor}} = \frac{\text{Gear ratio} * \text{Increments/Revolution}}{\text{Feed constant}}$$

The position_factor must be written to the motor controller separated into numerators and denominators. This can make it necessary to bring the fraction up to whole integers by expanding it accordingly.



The position_factor must not be larger than 2²⁴

EXAMPLE

First, the desired unit (column 1) and the desired number of decimal places (dp) have to be specified, along with the application's gear ratio and its feed constant (if applicable). The feed constant is then displayed in the desired positioning units (column 2). Finally all values can be entered into the formula and the fraction can be calculated:

Position factor calculation sequence				
Position units	Feed constant	Gear ratio	Formula	Result shortened
Degree, 1 DP → 1/10 degree (°/10)	1 R _{OUT} = 3600 $\frac{\circ}{10}$	1/1	$\frac{1}{1} * 65536 \text{ Inc}$ = $\frac{65536 \text{ Inc}}{3600 \frac{\circ}{10}}$	num : 4096 div : 225

Fig. 5.4 Position factor calculation sequence

Examples of calculating the position factor				
Position units ¹⁾	Feed constant ²⁾	Gear ratio ³⁾	Formula ⁴⁾	Result shortened
Increments, 0 DP → Inc.	1 R _{OUT} = 65536 Ink	1/1	$\frac{1}{1} * 65536 \text{ Inc}$ = $\frac{1 \text{ Inc}}{1 \text{ Inc}}$	num : 1 div : 1
Degree, 1 DP → 1/10 degree (°/10)	1 R _{OUT} = 3600 $\frac{\circ}{10}$	1/1	$\frac{1}{1} * 65536 \text{ Inc}$ = $\frac{65536 \text{ Inc}}{3600 \frac{\circ}{10}}$	num : 4096 div : 225
Rev., 2 DP → 1/100 Rev. (^R /100)	1 R _{OUT} = 100 $\frac{R}{100}$	1/1	$\frac{1}{1} * 65536 \text{ Inc}$ = $\frac{65536 \text{ Inc}}{100 \frac{1}{100}}$	num : 16384 div : 25
		2/3	$\frac{2}{3} * 65536 \text{ Inc}$ = $\frac{131072 \text{ Inc}}{300 \frac{1}{100}}$	num : 32768 div : 75
mm, 1 DP → 1/10 mm (mm/10)	1 R _{OUT} = 631.5 $\frac{\text{mm}}{10}$	4/5	$\frac{4}{5} * 65536 \text{ Inc}$ = $\frac{2621440 \text{ Inc}}{631.5 \frac{\text{mm}}{10}}$	num: 524288 div: 6315

- 1) Desired unit at the drive-out
- 2) Positioning units per revolution at the drive-out (R_{OUT}). Feed constant of the drive * 10^{-DP} (points after the decimal)
- 3) Revolutions at the drive in per revolutions at the drive-out (R_{IN} per R_{OUT})
- 4) Insert values into equation.

Tab. 5.5 Examples of calculating the position factor

6094_h: velocity_encoder_factor

The object velocity_encoder_factor converts all speed values of the application from speed_units into the internal unit revolutions per 4096 minutes. It consists of numerator and denominator.

Index	6094_h
Name	velocity_encoder_factor
Object Code	ARRAY
No. of Elements	2
Data Type	UINT32

Sub-Index	01_h
Description	numerator
Access	rw
PDO mapping	yes
Units	–
Value Range	–
Default Value	1000 _h

Sub-Index	02_h
Description	divisor
Access	rw
PDO mapping	yes
Units	–
Value Range	–
Default Value	1

Calculation of the velocity_encoder_factor is in principle made up of two parts: a conversion factor from internal length units into position_units, and a conversion factor from internal time units into user-defined time units (e.g. from seconds into minutes). The first part corresponds to the calculation of the position_factor; for the second part, an additional factor is added to the calculation:

Parameters	Description
time_factor_v	The ratio between the internal time unit and the user-defined time unit. (e.g. 1 min = $1/4096$ 4096 min)
gear_ratio	Gear ratio between revolutions at the drive-in (R_{IN}) and revolutions at the drive-out (R_{OUT})
feed_constant	Ratio between revolutions at the drive-out (R_{OUT}) and movement in position_units (e.g. 1 R = 360 degrees)

Tab. 5.6 Speed factor parameters

The calculation of the velocity_encoder_factor uses the following equation:

$$\text{velocity_encoder_factor} = \frac{\text{numerator}}{\text{divisor}} = \frac{\text{gear_ratio} * \text{time_factor_v}}{\text{feedconstant}}$$



The velocity_encoder_factor must not be greater than 2²⁴

Like the position_factor, the velocity_encoder_factor also has to be written to the motor controller separated into numerators and denominators. This can make it necessary to bring the fraction up to whole integers by expanding it accordingly.

EXAMPLE

First, the desired unit (column 1) and the desired number of decimal places (dp) have to be specified, along with the application's gear ratio and its feed constant (if applicable). The feed constant is then displayed in the desired positioning units (column 2). Then the desired time unit is converted into the time unit of the motor controller (column 3).

Finally all values can be entered into the formula and the fraction can be calculated:

Speed units	Feed const.	Time Constant	Gear	Formula	Result shortened
mm/s, 1 DP → 1/10 mm/s (mm/10s)	63.15 $\frac{\text{mm}}{\text{R}}$ ⇒ 1 R _{OUT} = 631.5 $\frac{\text{mm}}{10}$	1 $\frac{1}{\text{s}}$ = 60 $\frac{1}{\text{min}}$ = 60 * 4096 $\frac{1}{4096 \text{ min}}$	4/5	$\frac{4}{5} * \frac{60 * 4096 \frac{1}{4096 \text{ min}}}{1 \frac{1}{\text{s}}}$ = 631.5 $\frac{\text{mm}}{10}$	$\frac{1966080 \frac{1}{4096 \text{ min}}}{6315 \frac{\text{mm}}{10\text{s}}}$ num: 131072 div: 421

Fig. 5.5 Velocity factor calculation sequence

Examples of calculating the speed factor					
Speed units ¹⁾	Feed const. ²⁾	Time Constant ³⁾	Gear 4)	Formula ⁵⁾	Result shortened
R/min, 0 DP → R/min	$1 R_{OUT} =$ $1 R_{OUT}$	$1 \frac{1}{min} =$ $4096 \frac{1}{4096 min}$	1/1	$\frac{1}{1} * \frac{4096 \frac{1}{4096 min}}{1 \frac{1}{min}} =$ $\frac{4096 \frac{1}{4096 min}}{1 \frac{1}{min}}$	num: 4096 div: 1
R/min, 2 DP → 1/100 R/min (R/100 min)	$1 R_{OUT} =$ $100 \frac{R}{100}$	$1 \frac{1}{min} =$ $4096 \frac{1}{4096 min}$	2/3	$\frac{2}{3} * \frac{4096 \frac{1}{4096 min}}{1 \frac{1}{min}} =$ $\frac{8192 \frac{1}{4096 min}}{100 \frac{1}{100}} =$ $\frac{8192 \frac{1}{4096 min}}{300 \frac{1}{100 min}}$	num: 2048 div: 75
°/s, 1 DP → 1/10 °/s (°/10 s)	$1 R_{OUT} =$ $3600 \frac{°}{10}$	$1 \frac{1}{s} =$ $60 \frac{1}{min} =$ $60 * 4096 \frac{1}{4096 min}$	1/1	$\frac{1}{1} * \frac{60 * 4096 \frac{1}{4096 min}}{1 \frac{1}{s}} =$ $\frac{245760 \frac{1}{4096 min}}{3600 \frac{°}{10}} =$ $\frac{245760 \frac{1}{4096 min}}{3600 \frac{°}{10 s}}$	num: 1024 div: 15
mm/s, 1 DP → 1/10 mm/s (mm/10 s)	$63.15 \frac{mm}{R}$ ⇒ $1 R_{OUT} =$ $631.5 \frac{mm}{10}$	$1 \frac{1}{s} =$ $60 \frac{1}{min} =$ $60 * 4096 \frac{1}{4096 min}$	4/5	$\frac{4}{5} * \frac{60 * 4096 \frac{1}{4096 min}}{1 \frac{1}{s}} =$ $\frac{1966080 \frac{1}{4096 min}}{631.5 \frac{mm}{10}} =$ $\frac{1966080 \frac{1}{4096 min}}{6315 \frac{mm}{10 s}}$	num: 131072 div: 421

- 1) Desired unit at the drive-out
- 2) Positioning units per revolution at the drive-out (R_{OUT}). Feed constant of the drive * 10^{-DP} (points after the decimal)
- 3) Time factor_v: desired time unit per internal time unit
- 4) Gear factor: R_{IN} per R_{OUT}
- 5) Insert values into equation.

Tab. 5.7 Examples of calculating the speed factor

6097_h: acceleration_factor

The object acceleration_factor converts all acceleration values of the application from acceleration_units into the internal unit Revolutions per minute per 256 seconds. It consists of numerator and denominator.

Index	6097_h
Name	acceleration_factor
Object Code	ARRAY
No. of Elements	2
Data Type	UINT32

Sub-Index	01_h
Description	numerator
Access	rw
PDO mapping	yes
Units	–
Value Range	–
Default Value	100 _h

Sub-Index	02_h
Description	divisor
Access	rw
PDO mapping	yes
Units	–
Value Range	–
Default Value	1

Calculation of the acceleration_factor is also made up of two parts: a conversion factor from internal units of length into position_units, and a conversion factor from internal time units squared into the user-defined time units squared (e.g. from seconds² into minutes²). The first part corresponds to the calculation of the position_factor; for the second part, an additional factor is added:

Parameters	Description
time_factor_a	The ratio between the internal time unit squared and the user-defined time unit squared. (e.g. $1 \text{ min}^2 = 1 \text{ min} \times 1 \text{ min} = 60 \text{ s} \times 1 \text{ min} = \frac{60}{256} 256 \text{ min} \times \text{s}$).
gear_ratio	Gear ratio between revolutions at the input shaft (R_{IN}) and revolutions at the output shaft (R_{OUT}).
feed_constant	Ratio between revolutions at the drive-out (R_{OUT}) and movement in position_units (e.g. $1 \text{ R} = 360 \text{ degrees}$)

Tab. 5.8 Acceleration factor parameter

Calculation of the acceleration_factor uses the following equation:

$$\text{acceleration_factor} = \frac{\text{numerator}}{\text{divisor}} = \frac{\text{gear_ratio} * \text{time_factor_a}}{\text{feed_constant}}$$

The acceleration_factor is also written into the motor controller separated by numerator and denominator, so it may have to be expanded.

EXAMPLE

First, the desired unit (column 1) and the desired number of decimal places (dp) have to be specified, along with the application's gear ratio and its feed constant (if applicable). The feed constant is then displayed in the desired positioning units (column 2). Then the desired time unit is converted into the time unit of the motor controller (column 3). Finally all values can be entered into the formula and the fraction can be calculated:

Process of calculating the acceleration factor

Units of acceleration	Feed const.	Time Constant	Gear	Formula	Result shortened
mm/s ² , 1 DP → 1/10 mm/s ² (mm/10s ²)	63.15 $\frac{\text{mm}}{\text{R}}$ ⇒ 1 R _{OUT} = 631.5 $\frac{\text{mm}}{10}$	$1 \frac{1}{\text{s}^2} =$ $60 \frac{1}{\text{min} * \text{s}} =$ $60 * 256 \frac{1}{256 * \text{s}}$	4/5	$4 \frac{1}{5} * \frac{60 * 256 \frac{1}{256 \text{ min} * \text{s}}}{1 \frac{1}{\text{s}^2}} =$ $\frac{631,5 \frac{\text{mm}}{10}}{1} =$ $6315 \frac{\text{mm}}{10 \text{s}^2}$	num: 8192 div: 421

Examples of calculating the acceleration factor

Units of acceleration ¹⁾	Feed const. ²⁾	Time Constant ³⁾	Gear ⁴⁾	Formula ⁵⁾	Result shortened
R/min, 0 DP → R/min s	1 R _{OUT} = 1 R _{OUT}	$1 \frac{1}{\text{min} * \text{s}} =$ $256 \frac{1}{256 * \text{s}}$	1/1	$1 * \frac{256 \frac{1}{256 \text{ min} * \text{s}}}{1 \frac{1}{\text{min} * \text{s}}} =$ $\frac{256 \frac{1}{256 * \text{s}}}{1 \frac{1}{\text{min}}}$	num: 256 div: 1
°/s ² , 1 DP → 1/10 °/s ² (°/10s ²)	1 R _{OUT} = 3600 $\frac{°}{10}$	$1 \frac{1}{\text{s}^2} =$ $60 \frac{1}{\text{min} * \text{s}} =$ $60 * 256 \frac{1}{256 * \text{s}}$	1/1	$1 * \frac{60 * 256 \frac{1}{256 \text{ min} * \text{s}}}{1 \frac{1}{\text{s}^2}} =$ $\frac{15360 \frac{1}{256 * \text{s}}}{3600 \frac{°}{10}} =$ $\frac{3600 \frac{°}{10}}{1} =$ $3600 \frac{1}{10 \text{s}^2}$	num: 64 div: 15
R/min ² , 2 DP → 1/100 R/min ² (R/100 min ²)	1 R _{OUT} = 100 $\frac{\text{R}}{100}$	$1 \frac{1}{\text{min}^2} =$ $60 \frac{1}{\text{min} * \text{s}} =$ $\frac{256 \frac{1}{\text{min}}}{60 \frac{1}{256 * \text{s}}}$	2/3	$2 \frac{1}{3} * \frac{256 \frac{1}{256 \text{ min} * \text{s}}}{60 \frac{1}{\text{min}^2}} =$ $\frac{512 \frac{1}{256 \text{s}}}{100 \frac{1}{100}} =$ $\frac{18000 \frac{1}{100 \text{ min}^2}}{1}$	num: 32 div: 1125
mm/s ² , 1 DP → 1/10 mm/s ² (mm/10s ²)	63.15 $\frac{\text{mm}}{\text{R}}$ ⇒ 1 R _{OUT} = 631.5 $\frac{\text{mm}}{10}$	$1 \frac{1}{\text{s}^2} =$ $60 \frac{1}{\text{min} * \text{s}} =$ $60 * 256 \frac{1}{256 * \text{s}}$	4/5	$4 \frac{1}{5} * \frac{60 * 256 \frac{1}{256 \text{ min} * \text{s}}}{1 \frac{1}{\text{s}^2}} =$ $\frac{122880 \frac{1}{256 \text{s}}}{631,5 \frac{\text{mm}}{10}} =$ $\frac{6315 \frac{\text{mm}}{10 \text{s}^2}}{1}$	num: 8192 div: 421

- 1) Desired unit at the drive-out
- 2) Positioning units per revolution at the drive-out (R_{OUT}). Feed constant of the drive * 10^{-DP} (points after the decimal)
- 3) Time factor_v: desired time unit per internal time unit
- 4) Gear factor: R_N per R_{OUT}
- 5) Insert values into equation.

Tab. 5.9 Examples of calculating the acceleration factor

Object 607E_h: polarity

The algebraic sign of the position and speed values of the motor controller can be set with the corresponding `polarity_flag`. This can serve to invert the motor's direction of rotation with the same setpoint values.

In most applications, it makes sense to set the `position_polarity_flag` and the `velocity_polarity_flag` to the same value.

Setting of the `polarity_flag` influences only parameters when reading and writing. Parameters already present in the motor controller are not changed.

Index	607E_h
Name	polarity
Object Code	VAR
Data Type	UINT8

Access	rw
PDO mapping	yes
Units	–
Value Range	40 _h , 80 _h , C0 _h
Default Value	0

Bit	Value	Name	Meaning
6	40 _h	<code>velocity_polarity_flag</code>	0: multiply by 1 (default) 1: multiply by -1 (inverse)
7	80 _h	<code>position_polarity_flag</code>	0: multiply by 1 (default) 1: multiply by -1 (inverse)

5.4 Output stage parameter

Overview

The mains voltage is fed in to the output stage via a precharging circuit. When the power supply is switched on, the starting current is limited and charging is monitored. After precharging of the intermediate circuit, the charging circuit is bridged. This status is a requirement for issuing controller enable. The rectified mains voltage is smoothed with the condensers of the intermediate circuit. From the intermediate circuit, the motor is powered via the IGBTs. The output stage contains a series of safety functions, which can be partially parametrised:

- Controller enable logic (software and hardware enable)
- Overcurrent monitoring
- Overvoltage / undervoltage monitoring of the intermediate circuit
- Power partial monitoring

Description of the objects

Index	Object	Name	Type	Attr.
6510 _h	RECORD	Drive_data		

Object 6510_h_10_h: enable_logic

For the output stage of the motor controller to be activated, the digital inputs output stage enable and controller enable must be set: The output stage enable has a direct effect on the control signals of the power transistors and would also be able to interrupt them in case of a defective microprocessor. The removal of the output stage enable with running motor thus ensures that the motor runs out unbraked or is only stopped by the holding brake, if on hand. The controller enable is processed by the microcontroller of the motor controller. Depending on the operating mode, the motor controller reacts differently after removal of this signal:

- Positioning mode and speed-regulated mode
The motor is braked with a defined brake ramp after removal of the signal. The output stage is only switched off when the motor speed lies below 10 min⁻¹ and the holding brake, if on hand, has activated.
- Torque-controlled mode
The output stage is switched off immediately after removal of the signal. At the same time, a holding brake, if on hand, is activated. And so the motor runs out unbraked or is stopped only by the holding brake, if on hand.



Warning

Dangerous voltage!

Both signals do not guarantee that the motor is voltage-free.

When operating the motor controller over the CAN bus, the two digital inputs output stage enable and controller enable can be placed together onto 24 V, and the enable can be controlled via the CAN bus. For this, the object 6510_h_10_h (enable_logic) must be set to two. For safety reasons, this takes place automatically with activation of CANopen (also after a reset of the motor controller).

Index	6510_h
Name	drive_data
Object Code	RECORD
No. of Elements	51

Sub-Index	10_h
Description	enable_logic
Data Type	UINT16
Access	rw
PDO mapping	no
Units	–
Value Range	0 ... 2
Default Value	0

Value	Meaning
0	Digital inputs output stage enable + controller enable
1	Digital inputs output stage enable + controller enable + parametrisation interface
2	Digital inputs output stage enable + controller enable + CAN

Object 6510_h_30_h: pwm_frequency

The switching losses of the output stage are proportional to the switching frequency of the power transistors. With the devices of the CMMP family, cutting the normal pulse-width modulation frequency in half can produce more output. But the current ripple factor caused by the output stage rises. Reversing is only possible when the output stage is switched off.

Sub-Index	30_h
Description	pwm_frequency
Data Type	UINT16
Access	rw
PDO mapping	no
Units	–
Value Range	0, 1
Default Value	0

Value	Meaning
0	Standard output stage frequency
1	Half output stage frequency

Object 6510_h_3A_h: enable_enhanced_modulation

The extended sine modulation can be activated with the object `enable_enhanced_modulation`. It permits better utilization of the intermediate circuit voltage and thus about 14 %-higher speeds. A disadvantage in certain applications is that the control behaviour and smooth running of the motor can be slightly worse at very low speeds. Write access is possible only when the output stage is switched off. To accept the change, the parameter set must be backed up and a reset performed.

Sub-Index	3A_h
Description	enable_enhanced_modulation
Data Type	UINT16
Access	rw
PDO mapping	no
Units	–
Value Range	0, 1
Default Value	0

Value	Meaning
0	Extended sine modulation OFF
1	Extended sine modulation ON



Activation of the extended sine modulation is only effective after a reset. As a result, the parameter set must be saved (`save_all_parameters`) and then a reset performed.

Object 6510_h_31_h: power_stage_temperature

The temperature of the output stage can be read via the object `power_stage_temperature`. If the temperature specified in object 6510_h_32_h (`max_power_stage_temperature`) is exceeded, the output stage shuts off and an error message is output.

Sub-Index	31_h
Description	power_stage_temperature
Data Type	INT16
Access	ro
PDO mapping	yes
Units	°C
Value Range	–
Default Value	–

Object 6510_h_32_h: max_power_stage_temperature

The temperature of the output stage can be read via the object 6510_h_31_h (power_stage_temperature). If the temperature specified in the object max_power_stage_temperature is exceeded, the output stage shuts off and an error message is output.

Sub-Index	32_h
Description	max_power_stage_temperature
Data Type	INT16
Access	ro
PDO mapping	no
Units	°C
Value Range	100
Default Value	device-dependent

Object 6510_h_33_h: nominal_dc_link_circuit_voltage

Through the object nominal_dc_link_circuit_voltage, the nominal device voltage can be read in millivolt.

Sub-Index	33_h
Description	nominal_dc_link_circuit_voltage
Data Type	UINT32
Access	ro
PDO mapping	no
Units	mV
Value Range	–
Default Value	device-dependent

Object 6510_h_34_h: actual_dc_link_circuit_voltage

Via the object actual_dc_link_circuit_voltage, the current voltage of the intermediate circuit can be read in millivolts.

Sub-Index	34_h
Description	actual_dc_link_circuit_voltage
Data Type	UINT32
Access	ro
PDO mapping	yes
Units	mV
Value Range	–
Default Value	–

Object 6510_h_35_h: max_dc_link_circuit_voltage

The object max_dc_link_circuit_voltage specifies at which intermediate circuit voltage or higher the output stage is switched off immediately for safety reasons and an error message is sent.

Sub-Index	35_h
Description	max_dc_link_circuit_voltage
Data Type	UINT32
Access	ro
PDO mapping	no
Units	mV
Value Range	–
Default Value	device-dependent

Object 6510_h_36_h: min_dc_link_circuit_voltage

The motor controller has an undervoltage monitor. This can be activated through the object 6510_h_37_h (enable_dc_link_undervoltage_error). The object 6510_h_36_h (min_dc_link_circuit_voltage) specifies up to which lower intermediate circuit voltage the motor controller should work. Below this voltage, the error E 02-0 is triggered if it was activated with the subsequent object.

Sub-Index	36_h
Description	min_dc_link_circuit_voltage
Data Type	UINT32
Access	rw
PDO mapping	no
Units	mV
Value Range	0 ... 1000000
Default Value	0

Object 6510_h_37_h: enable_dc_link_undervoltage_error

The undervoltage monitor can be activated with the object enable_dc_link_undervoltage_error. Specify in the object 6510_h_36_h (min_dc_link_circuit_voltage) up to which lower intermediate circuit voltage the motor controller should work.

Sub-Index	37_h
Description	enable_dc_link_undervoltage_error
Data Type	UINT16
Access	rw
PDO mapping	no
Units	–
Value Range	0, 1
Default Value	0

Value	Meaning
0	Undervoltage error OFF (reaction WARNING)
1	Undervoltage error ON (reaction CONTROLLER ENABLE OFF)

The error 02-0 is activated through modification of the error response. Reactions causing the drive to stop are returned as ON, all others as OFF. When overwriting with 0, the error response WARNING is set; when overwriting with 1, the error response CONTROLLER ENABLE OFF.

→ also 5.18, Error Management.

Object 6510_h 40_h: nominal_current

The device nominal current can be read with the object nominal_current. It is simultaneously the upper limit value that can be written into the object 6075_h (motor_rated_current).

Sub-Index	40 _h
Description	nominal_current
Data Type	UINT32
Access	ro
PDO mapping	no
Units	mA
Value Range	–
Default Value	device-dependent



Other values may be displayed due to a power derating, dependent on the controller cycle time and the output stage clock frequency.

Object 6510_h 41_h: peak_current

The device peak current can be read with the object peak_current. It is simultaneously the upper limit value, which can be written into the object 6073_h (max_current).

Sub-Index	41 _h
Description	peak_current
Data Type	UINT32
Access	ro
PDO mapping	no
Units	mA
Value Range	–
Default Value	device-dependent



The values apply for a current regulator cycle time of 125 µs.



Other values may be displayed due to a power derating, dependent on the controller cycle time and the output stage clock frequency.

5.5 Current Regulator and Motor Adjustment



Caution

Incorrect settings of the current regulator parameters and current limits can destroy the motor and, possibly, also the motor controller within a very short time!

Overview

The parameter set of the motor controller must be adapted for the connected motor and the set of cables used. Affected are the following parameters:

Parameters	Dependencies
Nominal current	Dependent on the motor
Overload capacity	Dependent on the motor
Number of poles	Dependent on the motor
Current regulator	Dependent on the motor
Direction of rotation	Dependent on the motor and on the phase sequence in the motor and angle transmitter cable
Offset Angle	Dependent on the motor and on the phase sequence in the motor and angle transmitter cable

Please observe that the direction of rotation and offset angle also depend on the cable set used. The parameter sets therefore work only with identical cabling.



Caution

If the phase sequence is distorted in the motor or angle encoder cable, the result may be positive feedback, so the speed in the motor cannot be regulated. The motor can turn uncontrollably!

Description of the Objects

Index	Object	Name	Type	Attr.
6075 _h	VAR	motorRatedCurrent	UINT32	rw
6073 _h	VAR	maxCurrent	UINT16	rw
604D _h	VAR	poleNumber	UINT8	rw
6410 _h	RECORD	motorData		rw
60F6 _h	RECORD	torqueControlParameters		rw

Affected objects from other chapters

Index	Object	Name	Type	Chapter
2415 _h	RECORD	currentLimitation		5.8 Setpoint value limitation

Object 6075_h: motorRatedCurrent

This value can be taken from the motor rating plate and is entered in milliamperes. The effective value (RMS) is always assumed. No current can be specified above the motor controller nominal current (6510_{h_40h}: nominalCurrent).

Index	6075_h
Name	motorRatedCurrent
Object Code	VAR
Data Type	UINT32

Access	rw
PDO mapping	yes
Units	mA
Value Range	0 ... nominalCurrent
Default Value	296



If the object 6075_h (motorRatedCurrent) is written over with a new value, the object 6073_h (maxCurrent) must always be parametrised again.

Object 6073_h: maxCurrent

As a rule, servo motors may be overloaded for a certain time period. With this object, the maximum permissible motor current is set as a factor. It refers to the nominal motor current (object 6075_h: motorRatedCurrent) and is set in thousandths. The range of values is limited upward by the maximum controller current (object 6510_{h_41h}: peakCurrent). Many motors may be overloaded briefly by a factor of 4. In this case, the value 4000 is written into this object.



The object 6073_h (maxCurrent) may only be written over if the object 6075_h (motorRatedCurrent) was previously overwritten.

Index	6073_h
Name	maxCurrent
Object Code	VAR
Data Type	UINT16

Access	rw
PDO mapping	yes
Units	per thousands of rated current
Value Range	–
Default Value	2023

Object 604D_h: pole_number

The number of poles of the motor can be found in the motor data sheet or the parametrisation software. The number of poles is always even. The number of pole pairs is often specified instead of the number of pins. The number of poles then equals twice the number of pole pairs.

This object is not revised through restore_default_parameters.

Index	604D_h
Name	pole_number
Object Code	VAR
Data Type	UINT8

Access	rw
PDO mapping	yes
Units	–
Value Range	2 ... 254
Default Value	4 (after INIT!)

Object 6410_h_03_h: iit_time_motor

As a rule, servo motors may be overloaded for a certain time period. This object specifies how long current can flow through the connected motor with the current specified in the object 6073_h (max_current). After expiration of the I²t time, to protect the motor the current is automatically limited to the value set in object 6075_h (motor_rated_current). The standard setting is two seconds and is valid for most motors.

Index	6410_h
Name	motor_data
Object Code	RECORD
No. of Elements	5

Sub-Index	03_h
Description	iit_time_motor
Data Type	UINT16
Access	rw
PDO mapping	no
Units	ms
Value Range	0 ... 100000
Default Value	2000

Object 6410h_04h: iit_ratio_motor

Through the object iit_ratio_motor, the current extent of utilisation of the I²t limitation can be read in thousandths.

Sub-Index	04h
Description	iit_ratio_motor
Data Type	UINT16
Access	ro
PDO mapping	no
Units	thousandths
Value Range	–
Default Value	–

Object 6510h_38h: iit_error_enable

The object iit_error_enable establishes how the motor controller behaves when the I²t limitation occurs. Either it is only displayed in the status word, or error E 31-0 is triggered.

Index	6510h
Name	drive_data
Object Code	RECORD
No. of Elements	51

Sub-Index	38h
Description	iit_error_enable
Data Type	UINT16
Access	rw
PDO mapping	no
Units	–
Value Range	0, 1
Default Value	0

Value	Meaning	
0	I ² t error OFF	(Priority WARNING)
1	I ² t error ON	(Priority CONTROLLER ENABLE OFF)

The error E 31-0 is activated by changing the error response. Reactions causing the drive to stop are returned as ON, all others as OFF. When overwriting with 0, the error response WARNING is set; when overwriting with 1, the error response CONTROLLER ENABLE OFF. → chapter 5.18, Error Management.

Object 6410_h_10_h: phase_order

In the phase sequence (phase_order), twisting between motor cable and angle encoder cable are taken into account. It can be taken from the parameterisation software.

Sub-Index	10_h
Description	phase_order
Data Type	INT16
Access	rw
PDO mapping	yes
Units	–
Value Range	0, 1
Default Value	0

Value	Meaning
0	Right
1	Left-hand

Object 6410_h_11_h: encoder_offset_angle

The servo motors used have permanent magnets on the rotor. These generate a magnetic field, whose orientation toward the stator depends on the rotor position. For electronic commutation, the motor controller must always set the electromagnetic field of the stator in the correct angle to this permanent magnet field. To do this, it constantly determines the rotor position with an angle encoder (resolver, etc.).

The orientation of the angle encoder to the permanent magnetic field must be entered in the object resolver_offset_angle. This angle can be determined with the parameterisation software. The angle determined with the parameterisation software lies in the range of ±180°. It must be calculated as follows:

$$\text{encoder_offset_angle} = \text{Offset_angle_of_the_angle_encoder} * \frac{32767}{180^\circ}$$

This object is not revised through restore_default_parameters.

Index	6410_h
Name	motor_data
Object Code	RECORD
No. of Elements	5

Sub-Index	11_h
Description	encoder_offset_angle
Data Type	INT16
Access	rw
PDO mapping	yes
Units	...
Value Range	-32767 ... 32767
Default Value	E000 _h (-45°) (according to factory setting)

Object 6410_h 14_h: motor_temperature_sensor_polarity

This object is used to determine whether a normally closed contact or a normally open contact is used as a digital motor temperature sensor.

Sub-Index	14_h
Description	motor_temperature_sensor_polarity
Data Type	INT16
Access	rw
PDO mapping	yes
Units	–
Value Range	0, 1
Default Value	0

Value	Meaning
0	N/C contact
1	N/O contact

Object 6510_h 2E_h: motor_temperature

With this object, the current motor temperature can be read if an analogue temperature sensor is connected. Otherwise, the object is undefined.

Index	6510_h
Name	drive_data
Object Code	RECORD
No. of Elements	51

Sub-Index	2E_h
Description	motor_temperature
Data Type	INT16
Access	ro
PDO mapping	yes
Units	°C
Value Range	–
Default Value	–

Object 6510_h 2F_h: max_motor_temperature

If the motor temperature defined in this object is exceeded, a reaction takes place in accordance with error management (error 03-0, over-temperature motor analogue). If a reaction that stops the drive is parametrised, an emergency message is sent.

For parametrisation of error management → chap. 5.18, Error Management.

Sub-Index	2F_h
Description	max_motor_temperature
Data Type	INT16
Access	rw
PDO mapping	no
Units	°C
Value Range	20 ... 300
Default Value	100

Object 60F6_h: torque_control_parameters

The data of the current controller must be taken from the parametrisation software. The following calculations must be observed:

Amplification of the current controller must be multiplied by 256. With an amplification of 1.5 in the Current Regulator menu of the parametrisation software, the value $384 = 180_{\text{h}}$ must be written in the object torque_control_gain.

The current controller time constant is specified in the parametrisation software in milliseconds. To transfer this time constant into the object torque_control_time, it must previously be converted into microseconds. With a specified time of 0.6 milliseconds, the corresponding value 600 is entered in the object torque_control_time.

Index	60F6_h
Name	torque_control_parameters
Object Code	RECORD
No. of Elements	2

Sub-Index	01_h
Description	torque_control_gain
Data Type	UINT16
Access	rw
PDO mapping	no
Units	256 = "1"
Value Range	0 ... 32*256
Default Value	3*256 (768)

Sub-Index	02_h
Description	torque_control_time
Data Type	UINT16
Access	rw
PDO mapping	no
Units	µs
Value Range	104 ... 64401
Default Value	1020

5.6 Speed Control

Overview

The parameter set of the motor controller must be adapted for the application. In particular, the amplification is strongly dependent on dimensions that may be connected to the motor. The data must be optimally determined during commissioning of the system with the help of the parametrisation software.



Caution

Incorrect setting of the speed regulator parameters can result in strong vibrations and possibly destroy parts of the system!

Description of the objects

Index	Object	Name	Type	Attr.
60F9 _h	RECORD	velocity_control_parameters		rw
2073 _h	VAR	velocity_display_filter_time	UINT32	rw

Object 60F9_h: velocity_control_parameters

The data of the speed controller must be taken from the parametrisation software. The following calculations must be observed:

Amplification of the speed regulator must be multiplied by 256.

With an amplification of 1.5 in the “Speed Regulator” menu of the parametrisation software, the value 384 = 180_h must be written in the object velocity_control_gain.

The speed regulator time constant is specified in the parametrisation software in milliseconds. To transfer this time constant into the velocity_control_time object, it must previously be converted into microseconds. With a specified time of 2.0 milliseconds, the corresponding value 2000 is entered in the object velocity_control_time.

Index	60F9_h
Name	velocity_control_parameter_set
Object Code	RECORD
No. of Elements	3

Sub-Index	01_h
Description	velocity_control_gain
Data Type	UINT16
Access	rw
PDO mapping	no
Units	256 = Gain 1
Value Range	20 ... 64*256 (16384)
Default Value	256

Sub-Index	02_h
Description	velocity_control_time
Data Type	UINT16
Access	rw
PDO mapping	no
Units	μs
Value Range	1 ... 32000
Default Value	2000

Sub-Index	04_h
Description	velocity_control_filter_time
Data Type	UINT16
Access	rw
PDO mapping	no
Units	μs
Value Range	1 ... 32000
Default Value	400

Object 2073_h: velocity_display_filter_time

The filter time constant of the display speed actual value filter can be set via the object velocity_display_filter_time.

Index	2073_h
Name	velocity_display_filter_time
Object Code	VAR
Data Type	UINT32

Access	rw
PDO mapping	no
Units	μs
Value Range	1000 ... 50000
Default Value	20000



Please observe that the object velocity_actual_value_filtered is used for spinning protection. With very large filter times, a spinning error is detected only with a corresponding delay.

5.7 Position Controller (Position Control Function)

Overview

This chapter describes all parameters required for the position controller. The position setpoint value (`position_demand_value`) of the curve generator is applied to the input of the position controller. In addition, the actual position value (`position_actual_value`) is added from the angle encoder (resolver, increment generator, etc.). The actions of the position controller can be influenced by parameters. It is possible to limit the output variable (`control_effort`) to keep the position control circuit stable. The output variable is supplied to the speed regulator as the speed setpoint value. All input and output variables of the position controller are converted in the Factor Group from the application-specific units into the respective internal units of the regulator.

The following subfunctions are defined in this chapter:

1. Contouring error (Following_Error)

The deviation of the actual position value (`position_actual_value`) from the position setpoint value (`position_demand_value`) is designated as contouring error. If this contouring error is greater than specified in the contouring error window (`following_error_window`) for a specific time period, bit 13 `following_error` is set in the object statusword. The permissible time period can be specified via the object `following_error_time_out`.

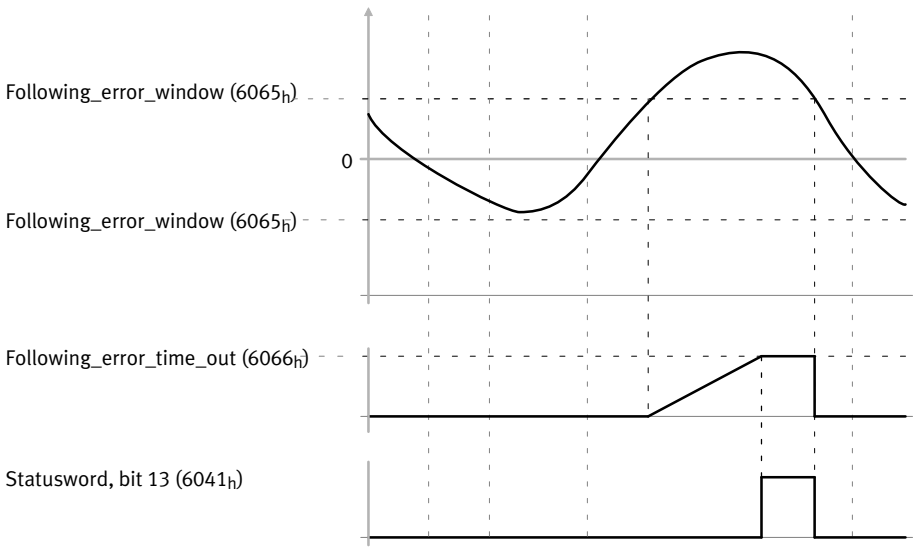


Fig. 5.6 Contouring error – functional overview

2. Position reached (position_reached)

This function offers the possibility of defining a position window around the target position (target_position). If the actual position of the drive is located within this range for a specific time – the position_window_time – the related bit 10 (target_reached) is set in the statusword.

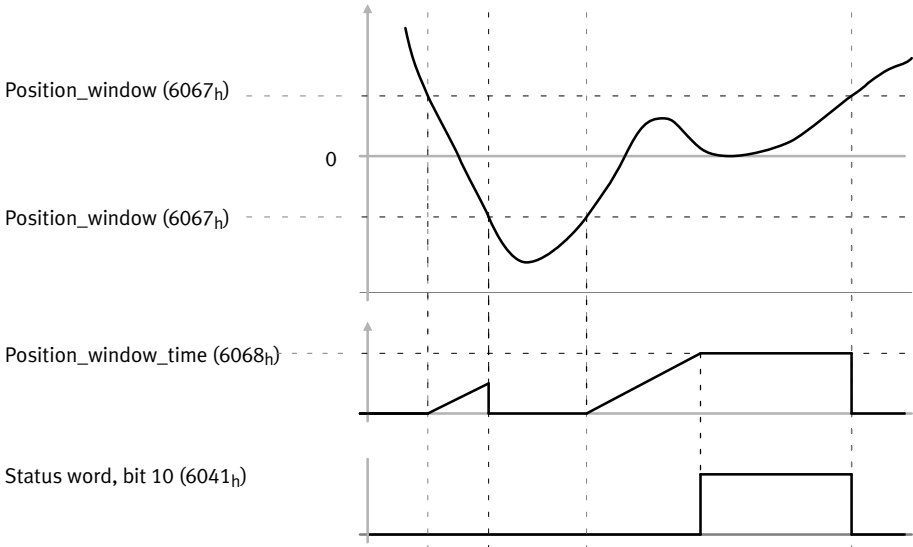


Fig. 5.7 Position reached – functional overview

Description of the objects

Objects treated in this chapter

Index	Object	Name	Type	Attr.
202D _h	VAR	position_demand_sync_value	INT32	ro
2030 _h	VAR	set_position_absolute	INT32	wo
6062 _h	VAR	position_demand_value	INT32	ro
6063 _h	VAR	position_actual_value_s ¹⁾	INT32	ro
6064 _h	VAR	position_actual_value	INT32	ro
6065 _h	VAR	following_error_window	UINT32	rw
6066 _h	VAR	following_error_time_out	UINT16	rw
6067 _h	VAR	position_window	UINT32	rw
6068 _h	VAR	position_window_time	UINT16	rw
607B _h	ARRAY	position_range_limit	INT32	rw
60F4 _h	VAR	following_error_actual_value	INT32	ro
60FA _h	VAR	control_effort	INT32	ro
60FB _h	RECORD	position_control_parameter_set		rw
6510 _h _20 _h	VAR	position_range_limit_enable	UINT16	rw
6510 _h _22 _h	VAR	position_error_switch_off_limit	UINT32	rw

1) In increments

Affected objects from other chapters

Index	Object	Name	Type	Chapter
607A _h	VAR	target_position	INT32	7.3 Positioning operating mode
607C _h	VAR	home_offset	INT32	7.2 Homing run
607D _h	VAR	software_position_limit	INT32	7.3 Positioning operating mode
607E _h	VAR	polarity	UINT8	5.3 Conversion factors
6093 _h	VAR	position_factor	UINT32	5.3 Conversion factors
6094 _h	ARRAY	velocity_encoder_factor	UINT32	5.3 Conversion factors
6096 _h	ARRAY	acceleration_factor	UINT32	5.3 Conversion factors
6040 _h	VAR	controlword	INT16	6.1.3 Control word (controlword)
6041 _h	VAR	statusword	UINT16	6.1.5 Status words (statuswords)

Object 60FB_h: position_control_parameter_set

The parameter set of the motor controller must be adapted for the application. The data of the position controller must be optimally determined during commissioning using the parametrisation software.

**Caution**

Incorrect setting of the position regulator parameters can result in strong vibrations and possibly destroy parts of the system!

The position controller compares the target location with the actual location and, from the difference, creates a correction speed (object 60FA_h: control_effort), which is fed to the speed regulator, taking into account the gain and possibly the integrator.

The position controller is relatively slow, compared to the current and speed regulator. Therefore, the controller works internally with activation, so the stabilisation work for the position controller is minimised and the controller can rapidly stabilise.

A proportional link normally suffices as position controller. Amplification of the position controller must be multiplied by 256. With an amplification of 1.5 in the “Current Regulator” menu of the parametrisation software, the value 384 must be written in the object position_control_gain.

The position controller normally does not need an integrator. Then the value zero must be written into the object position_control_time . Otherwise, the time constant of the position controller must be converted into microseconds. With a time of 4.0 milliseconds, the corresponding value 4000 is entered in the object position_control_time.

Since the position controller already converts the smallest position deviations into appreciable correction speeds, in the case of a brief disturbance (e.g. brief jamming of the system) it would lead to very major stabilisation processes with very large correction speeds. This can be avoided if the output of the position controller is sensibly limited via the object position_control_v_max (e.g. 500 min⁻¹).

The size of a position deviation up to which the position controller will not intervene (dead area) can be defined with the object position_error_tolerance_window. This can be used for stabilisation, such as when there is play in the system.

Index	60FB_h
Name	position_control_parameter_set
Object Code	RECORD
No. of Elements	4

Sub-Index	01_h
Description	position_control_gain
Data Type	UINT16
Access	rw
PDO mapping	no
Units	256 = "1"
Value Range	0 ... 64*256 (16384)
Default Value	102

Sub-Index	02_h
Description	position_control_time
Data Type	UINT16
Access	ro
PDO mapping	no
Units	μs
Value Range	0
Default Value	0

Sub-Index	04_h
Description	position_control_v_max
Data Type	UINT32
Access	rw
PDO mapping	no
Units	speed units
Value Range	0 ... 131072 min ⁻¹
Default Value	500 min ⁻¹

Sub-Index	05_h
Description	position_error_tolerance_window
Data Type	UINT32
Access	rw
PDO mapping	no
Units	position units
Value Range	1 ... 65536 (1 R)
Default Value	2 (1/32768 R)

Object 6062_h: position_demand_value

The current position setpoint value can be read out via this object. The curve generator feeds this into the position controller.

Index	6062_h
Name	position_demand_value
Object Code	VAR
No. of Elements	INT32

Access	ro
PDO mapping	yes
Units	position units
Value Range	–
Default Value	–

Object 202D_h: position_demand_sync_value

The target position of the synchronisation encoder can be read out via this object. This is defined through the object 2022_h synchronization_encoder_select (→ chap. 5.11). This object is specified in user-defined increments.

Index	202D_h
Name	position_demand_sync_value
Object Code	VAR
Data Type	INT32

Access	ro
PDO mapping	no
Units	position units
Value Range	–
Default Value	–

Object 6063_h: position_actual_value_s (increments)

The actual position can be read out via this object. The angle encoder feeds this to the position controller. This object is specified in increments.

Index	6063_h
Name	position_actual_value_s
Object Code	VAR
Data Type	INT32

Access	ro
PDO mapping	yes
Units	increments
Value Range	–
Default Value	–

Object 6064_h: position_actual_value (user-defined units)

The actual position can be read out via this object. The angle encoder feeds this to the position controller. This object is specified in user-defined increments.

Index	6064_h
Name	position_actual_value
Object Code	VAR
Data Type	INT32

Access	ro
PDO mapping	yes
Units	position units
Value Range	–
Default Value	–

Object 6065_h: following_error_window

The object following_error_window (contouring error window) defines a symmetrical range around the position setpoint value (position_demand_value) . If the actual position value (position_actual_value) is outside the contouring error window (following_error_window), a contouring error occurs and bit 13 is set in the object status word. The following can cause a contouring error:

- the drive is blocked
- the positioning speed is too high
- the acceleration values are too large
- the object following_error_window has too small a value
- the position controller is not correctly parametrised

Index	6065_h
Name	following_error_window
Object Code	VAR
Data Type	UINT32

Access	rw
PDO mapping	yes
Units	position units
Value Range	–
Default Value	9101 (9101/65536 R = 50°)

Object 6066_h: following_error_time_out

If a contouring error longer than defined in this object occurs, the related bit 13 following_error is set in the statusword.

Index	6066_h
Name	following_error_time_out
Object Code	VAR
Data Type	UINT16

Access	rw
PDO mapping	yes
Units	ms
Value Range	0 ... 27314
Default Value	0

Object 60F4_h: following_error_actual_value

The current contouring error can be read out via this object. This object is specified in user-defined increments.

Index	60F4_h
Name	following_error_actual_value
Object Code	VAR
Data Type	INT32

Access	ro
PDO mapping	yes
Units	position units
Value Range	–
Default Value	–

Object 60FA_h: control_effort

The output variable of the position controller can be read via this object. This value is internally fed to the speed regulator as setpoint value.

Index	60FA_h
Name	control_effort
Object Code	VAR
Data Type	INT32

Access	ro
PDO mapping	yes
Units	speed units
Value Range	–
Default Value	–

Object 6067_h: position_window

With the object **position_window**, a symmetrical area is defined around the target position (target_position). If the actual position value (position_actual_value) lies within this area for a certain time, the target position (target_position) is considered reached.

Index	6067_h
Name	position_window
Object Code	VAR
Data Type	UINT32

Access	rw
PDO mapping	yes
Units	position units
Value Range	–
Default Value	1820 (1820/65536 R = 10°)

Object 6068_h: position_window_time

If the actual position of the drive is located within the positioning window (position_window) for as long as defined in this object, the related bit 10 target_reached is set in the status word.

Index	6068_h
Name	position_window_time
Object Code	VAR
Data Type	UINT16

Access	rw
PDO mapping	yes
Units	ms
Value Range	–
Default Value	0

Object 6510_h_22_h: position_error_switch_off_limit

In the object position_error_switch_off_limit, the maximum tolerance between the setpoint and actual position can be entered. In contrast to the above contouring error message, if exceeded the output stage is immediately switched off and an error triggered. The motor thus runs out unbraked (unless a holding brake is on hand).

Index	6510_h
Name	drive_data
Object Code	RECORD
No. of Elements	51

Sub-Index	22_h
Description	position_error_switch_off_limit
Data Type	UINT32
Access	rw
PDO mapping	no
Units	position units
Value Range	0 ... 2 ³² -1
Default Value	0

Value	Meaning
0	Contouring error limit value OFF (reaction: NO ACTION)
> 0	Contouring error limit value ON (reaction: SWITCH OUTPUT STAGE OFF IMMEDIATELY)

The error 17-0 is activated through modification of the error response. The reaction SWITCH OUTPUT STAGE OFF IMMEDIATELY is returned as ON, all others as OFF. When overwriting with 0, the error response NO ACTION is set; when overwriting with a value greater than 0, the error response SWITCH OUTPUT STAGE OFF IMMEDIATELY is set. ➔ chapter 5.18 Error Management.

Object 607B_h: position_range_limit

The object group position_range_limit includes two sub-parameters that limit the numeric range of the position values. If one of these limits is exceeded, the position actual value automatically jumps to the other limit. This permits parametrisation of so-called round axes. To be specified are the limits that should correspond physically to the same position, for example 0° and 360°.

For these limits to become effective, a round axis must be selected via the object 6510_{h_20h} (position_range_limit_enable).

Index	607B_h
Name	position_range_limit
Object Code	ARRAY
No. of Elements	2
Data Type	INT32

Sub-Index	01_h
Description	min_position_range_limit
Access	rw
PDO mapping	yes
Units	position units
Value Range	–
Default Value	–

Sub-Index	02_h
Description	max_position_range_limit
Access	rw
PDO mapping	yes
Units	position units
Value Range	–
Default Value	–

Object 6510_{h_20h}: position_range_limit_enable

Through the object position_range_limit_enable, the range limits defined through the object 607B_h can be activated. Various modes are possible:

If the mode “shortest path” is selected, positioning is always executed along the physically shortest distance. To do this, the drive adapts to the prefix of the travel speed. In both “fixed direction of rotation” modes, positioning only takes place in the direction specified in the mode.

Index	6510_h
Name	drive_data
Object Code	RECORD
No. of Elements	51

Sub-Index	20_h
Description	position_range_limit_enable
Data Type	UINT16
Access	rw
PDO mapping	no
Units	–
Value Range	0 ... 5
Default Value	0

Value	Meaning
0	Off
1	Shortest path (for compatibility reasons)
2	Shortest distance
3	Reserved
4	Fixed direction of rotation “positive”
5	Fixed direction of rotation, “negative”

Object 2030_h: set_position_absolute

Through the object set_position_absolute, the readable actual position can be displaced without changing the physical position. The drive does not make any movement.

If an absolute encoder system is connected, the position shift is stored in the encoder if the encoder system permits it. The position shift thus remains intact even after a reset. This storage operation runs in the background independently of this object. All parameters belonging to the encoder storage are saved with their current values.

Index	2030_h
Name	set_position_absolute
Object Code	VAR
Data Type	INT32

Access	wo
PDO mapping	no
Units	position units
Value Range	–
Default Value	–

5.8 Setpoint value limitation

Description of the objects

Objects treated in this chapter

Index	Object	Name	Type	Attr.
2415 _h	RECORD	current_limitation		rw
2416 _h	RECORD	speed_limitation		rw

Object 2415_h: current_limitation

With the object group `current_limitation`, the maximum peak current for the motor can be limited in the operating modes `profile_position_mode`, `interpolated_position_mode`, `homing_mode` und `velocity_mode`, which makes a torque-limited speed operation possible. The setpoint value source of the limit torque is specified via the object `limit_current_input_channel`. Here, a choice can be made between specification of a direct setpoint value (fixed value) or specification via an analogue input. Depending on the source chosen, either the limit torque (source = fixed value) or the scaling factor for the analogue inputs (source = analogue input) is specified via the object `limit_current`. In the first case, the torque-proportional current, in mA, is limited directly; in the second case, the current that should correspond to a voltage of 10 V is specified, in mA.

Index	2415_h
Name	current_limitation
Object Code	RECORD
No. of Elements	2

Sub-Index	01_h
Description	limit_current_input_channel
Data Type	UINT8
Access	rw
PDO mapping	no
Units	–
Value Range	0 ... 4
Default Value	0

Sub-Index	02_h
Description	limit_current
Data Type	INT32
Access	rw
PDO mapping	no
Units	mA
Value Range	–
Default Value	0

Value	Meaning
0	No limitation
1	AIN0
2	AIN1
3	AIN2
4	Fieldbus (selector B)

Object 2416_h: speed_limitation

With the object group `speed_limitation`, the maximum speed of the motor can be limited in the operating mode `profile_torque_mode`, which makes a speed-limited torque mode possible. The setpoint value source of the limit speed is specified via the object `limit_speed_input_channel`. Here, a choice can be made between specification of a direct setpoint value (fixed value) or specification via an analogue input. Depending on the source chosen, either the limit torque (fixed value) or the scaling factor for the analogue inputs (source = analogue input) is specified via the object `limit_speed`. In the first case, the limit is at the specified speed; in the second case, the speed is specified that should correspond to a present voltage of 10 V.

Index	2416_h
Name	speed_limitation
Object Code	RECORD
No. of Elements	2

Sub-Index	01_h
Description	limit_speed_input_channel
Data Type	UINT8
Access	rw
PDO mapping	no
Units	–
Value Range	0 ... 4
Default Value	0

Sub-Index	02_h
Description	limit_speed
Data Type	INT32
Access	rw
PDO mapping	no
Units	speed units
Value Range	–
Default Value	–

Value	Meaning
0	No limitation
1	AIN0
2	AIN1
3	AIN2
4	Fieldbus (selector B)

5.9 Encoder Adjustments

Overview

This chapter describes the configuration of the angle encoder input [X2A], [X2B] and incremental input [X10].



Caution

Incorrect angle encoder settings can cause the drive to turn uncontrollably and possibly destroy parts of the system.

Description of the objects

Objects treated in this chapter

Index	Object	Name	Type	Attr.
2024 _h	RECORD	encoder_x2a_data_field		ro
2024 _{h_01h}	VAR	encoder_x2a_resolution	UINT32	ro
2024 _{h_02h}	VAR	encoder_x2a_numerator	INT16	rw
2024 _{h_03h}	VAR	encoder_x2a_divisor	INT16	rw
2025 _h	RECORD	encoder_x10_data_field		ro
2025 _{h_01h}	VAR	encoder_x10_resolution	UINT32	rw
2025 _{h_02h}	VAR	encoder_x10_numerator	INT16	rw
2025 _{h_03h}	VAR	encoder_x10_divisor	INT16	rw
2025 _{h_04h}	VAR	encoder_x10_counter	UINT32	ro
2026 _h	RECORD	encoder_x2b_data_field		ro
2026 _{h_01h}	VAR	encoder_x2b_resolution	UINT32	rw
2026 _{h_02h}	VAR	encoder_x2b_numerator	INT16	rw
2026 _{h_03h}	VAR	encoder_x2b_divisor	INT16	rw
2026 _{h_04h}	VAR	encoder_x2b_counter	UINT32	ro

Object 2024_h: encoder_x2a_data_field

The record encoder_x2a_data_field summarises parameters that are necessary for operation of the angle encoder at the plug [X2A].

Since the numerous angle encoder settings only become effective after a reset, selection and adjustment of the encoders should take place through the parametrisation software. Under CANopen, the following settings can be read or changed:

The object encoder_x2a_resolution specifies how many increments are generated by the encoder per revolution or unit of length. Since only resolvers that are always evaluated using 16 bit can be connected at the input [X2A], 65536 is always returned here. With the object encoder_x2a_numerator and encoder_x2a_divisor, a gear unit (also with prefix) if necessary between motor shaft and encoder can be taken into account.

Index	2024_h
Name	encoder_x2a_data_field
Object Code	RECORD
No. of Elements	3

Sub-Index	01_h
Description	encoder_x2a_resolution
Data Type	UINT32
Access	ro
PDO mapping	no
Units	Increments (4 * number of lines)
Value Range	–
Default Value	65536

Sub-Index	02_h
Description	encoder_x2a_numerator
Data Type	INT16
Access	rw
PDO mapping	no
Units	–
Value Range	–32768 ... 32767 (except 0)
Default Value	1

Sub-Index	03_h
Description	encoder_x2a_divisor
Data Type	INT16
Access	rw
PDO mapping	no
Units	–
Value Range	1 ... 32767
Default Value	1

Object 2026_h: encoder_x2b_data_field

The record encoder_x2a_data_field summarises parameters that are necessary for operation of the angle encoder at the plug [X2A].

The object encoder_x2b_resolution specifies how many increments are generated by the encoder per revolution (for incremental encoders, this equals four times the number of lines or periods per revolution).

The object encoder_x2b_counter delivers the currently counted number of increments. And so it delivers values between 0 and the set increment number-1.

With the objects encoder_x2b_numerator and encoder_x2b_divisor, a gear unit connected between the motor shaft and the encoder connected to [X2B] can be taken into account.

Index	2026_h
Name	encoder_x2b_data_field
Object Code	RECORD
No. of Elements	4

Sub-Index	01_h
Description	encoder_x2b_resolution
Data Type	UINT32
Access	rw
PDO mapping	no
Units	Increments (4 * number of lines)
Value Range	Dependent on the encoder used
Default Value	Dependent on the encoder used

Sub-Index	02_h
Description	encoder_x2b_numerator
Data Type	INT16
Access	rw
PDO mapping	no
Units	–
Value Range	–32768 ... 32767
Default Value	1

Sub-Index	03_h
Description	encoder_x2b_divisor
Data Type	INT16
Access	rw
PDO mapping	no
Units	–
Value Range	1 ... 32767
Default Value	1

Sub-Index	04_h
Description	encoder_x2b_counter
Data Type	UINT32
Access	ro
PDO mapping	yes
Units	Increments (4 * number of lines)
Value Range	0 ... (encoder_x2b_resolution -1)
Default Value	–

Object 2025_h: encoder_x10_data_field

The record encoder_X10_data_field summarises parameters that are necessary for operation of the incremental input [X10]. A digital incremental encoder or emulated incremental signals, for example from another CMMP, can be optionally connected here. The input signals over [X10] can optionally be used as a setpoint value or an actual value. More information on this can be found in chapter 5.11. How many increments are generated by the encoder per revolution or unit of length must be specified in the object encoder_X10_resolution. This corresponds to four times the number of lines. The object encoder_X10_counter supplies the currently counted incremental number (between 0 and the set increment number-1).

With the object encoder_X10_numerator and encoder_X10_divisor, a gear unit (also with prefix), if necessary, between motor shaft and encoder can be taken into account.

With use of the X10 signal as an actual value, this corresponds to a gear unit between the motor and the actual value encoder connected to [X10], which is mounted on the drive-out. With use of the X10 signal as setpoint value, gear ratios between master and slave can be implemented herewith.

Index	2025_h
Name	encoder_x10_data_field
Object Code	RECORD
No. of Elements	4

Sub-Index	01_h
Description	encoder_x10_resolution
Data Type	UINT32
Access	rw
PDO mapping	no
Units	Increments (4 * number of lines)
Value Range	Dependent on the encoder used
Default Value	Dependent on the encoder used

Sub-Index	02_h
Description	encoder_x10_numerator
Data Type	INT16
Access	rw
PDO mapping	no
Units	–
Value Range	–32768 ... 32767 (except 0)
Default Value	1

Sub-Index	03_h
Description	encoder_x10_divisor
Data Type	INT16
Access	rw
PDO mapping	no
Units	–
Value Range	1 ... 32767
Default Value	1

Sub-Index	04_h
Description	encoder_x10_counter
Data Type	UINT32
Access	ro
PDO mapping	yes
Units	Increments (4 * number of lines)
Value Range	0 ... (encoder_x10_resolution - 1)
Default Value	–

5.10 Incremental Encoder Emulation

Overview

This object group makes it possible to parametrise the incremental encoder output [X11]. As a result, master-slave applications, in which the output of the master [X11] is connected to the input of the slave [X10], can hereby be parametrised under CANopen.

Description of the objects

Objects treated in this chapter

Index	Object	Name	Type	Attr.
2028 _h	VAR	encoder_emulation_resolution	INT32	rw
201A _h	RECORD	encoder_emulation_data		ro
201A _h _01 _h	VAR	encoder_emulation_resolution	INT32	rw
201A _h _02 _h	VAR	encoder_emulation_offset	INT16	rw

Object 201A_h: encoder_emulation_data

The object record encoder_emulation_data encapsulates all setting options for the incremental encoder output [X11]:

Through the object encoder_emulation_resolution, the output increment number (= fourfold number of lines) can be freely set as a multiple of 4. In a master-slave application, this must correspond to the encoder_X10_resolution of the slave to achieve a ratio of 1:1.

With the object encoder_emulation_offset, the position of the output zero pulse can be shifted compared to the zero position of the actual value encoder.

Index	201A_h
Name	encoder_emulation_data
Object Code	RECORD
No. of Elements	2

Sub-Index	01_h
Description	encoder_emulation_resolution
Data Type	INT32
Access	rw
PDO mapping	no
Units	(4 * number of lines)
Value Range	4 * (1 ... 8192)
Default Value	4096

Sub-Index	02_h
Description	encoder_emulation_offset
Data Type	INT16
Access	rw
PDO mapping	no
Units	32767 = 180°
Value Range	-32768 ... 32767
Default Value	0

Object 2028_h: encoder_emulation_resolution

The object encoder_emulation_resolution is available only for compatibility reasons. It corresponds to the object 201A_h_01_h.

Index	2028_h
Name	encoder_emulation_resolution
Object Code	VAR
Data Type	INT32

Access	rw
PDO mapping	no
Units	→ 201A _h _01 _h
Value Range	→ 201A _h _01 _h
Default Value	→ 201A _h _01 _h

5.11 Setpoint/Actual Value Activation

Overview

With the help of the following objects, the source for the setpoint value and the source for the actual value can be revised. As standard, the motor controller uses the input for the motor encoder [X2A] or [X2B] as actual value for the position controller. With use of an external position controller, e.g. behind a gear unit, the position value supplied via [X10] can be activated as an actual value for the position controller. In addition, it is possible via [X10] to activate incoming signals (e.g. of a second controller) as additional setpoint values, through which synchronous operating modes are enabled.

Description of the objects

Objects treated in this chapter

Index	Object	Name	Type	Attr.
2021 _h	VAR	position_encoder_selection	INT16	rw
2022 _h	VAR	synchronisation_encoder_selection	INT16	rw
2023 _h	VAR	synchronisation_filter_time	UINT32	rw
202F _h	RECORD	synchronisation_selector_data		ro
202F _h _07 _h	VAR	synchronisation_main	UINT16	rw

Object 2021_h: position_encoder_selection

The object **position_encoder_selection** specifies the encoder input that is used for regulation of the actual position (actual position encoder). This value can be revised in order to switch to position control through an external encoder (connected to the drive-out). Switching is possible thereby between [X10] and the encoder input ([X2A]/[X2B]) selected as commutation encoder. If one of the encoder inputs [X2A]/[X2B] is selected as position actual value encoder, it must be the one used as commutation encoder. If the other encoder is selected, the commutation encoder is switched to automatically.

Index	2021_h
Name	position_encoder_selection
Object Code	VAR
Data Type	INT16

Access	rw
PDO mapping	no
Units	–
Value Range	0 ... 2 (→ table)
Default Value	0

Value	Measurement file
0	[X2A]
1	[X2B]
2	[X10]



Selection as position actual value encoder is only possible between the encoder input [X10] and the respective commutation encoder [X2A] or [X2B]. It is not possible to use the configuration [X2A] as commutation encoder and [X2B] as position actual value encoder, or vice versa.

Object 2022_h: synchronisation_encoder_selection

The object synchronisation_encoder_selection specifies the encoder input that is used as synchronisation setpoint value. Dependent on the operating mode, this corresponds to a position setpoint (profile position mode) or a speed setpoint (profile velocity mode).

Only [X10] can be used as synchronisation input. As a result, selection can be made between [X10] and no input. The same input should not be selected as synchronisation setpoint as for the actual value encoder.

Index	2022 _h
Name	synchronisation_encoder_selection
Object Code	VAR
Data Type	INT16

Access	rw
PDO mapping	no
Units	–
Value Range	-1, 2 (→ table)
Default Value	2

Value	Measurement file
-1	No encoder / undefined
2	[X10]

Object 202F_h: synchronisation_selector_data

Activation of a synchronous setpoint can take place through the object synchronisation_main. For the synchronous setpoint to be calculated at all, bit 0 must be set. Bit 1 permits activation of the synchronous position through starting of a position record. Currently, only 0 can be parametrised, so the synchronous position is always switched on. Through bit 8, it can be established that homing should take place without activation of the synchronous position so that the master and slave can be referenced separately.

Index	202F_h
Name	synchronisation_selector_data
Object Code	RECORD
No. of Elements	1

Sub-Index	07_h
Description	synchronisation_main
Data Type	UINT16
Access	rw
PDO mapping	no
Units	–
Value Range	→ Table
Default Value	–

Bit	Value	Meaning
0	0001 _h	0: Synchronisation inactive 1: Synchronisation active
1	0002 _h	“Flying saw” not possible
8	0100 _h	0: Synchronisation during homing 1: No synchronisation during homing

Object 2023_h: synchronisation_filter_time

Through the object synchronisation_filter_time, the filter time constant of a PT1 filter is established, with which the synchronisation speed is smoothed. This can be necessary especially with a low number of lines, since even small changes in the input value here can correspond to high speeds. On the other hand, the drive might no longer be in a position to follow a dynamic input signal quickly enough if filter times are high.

Index	2023_h
Name	synchronisation_filter_time
Object Code	VAR
Data Type	UINT32

Access	rw
PDO mapping	no
Units	µs
Value Range	10 ... 50000
Default Value	600

5.12 Analogue inputs

Overview

The motor controllers of the series CMMP-AS-...-M3/-M0 have three analogue inputs through which setpoint values can be specified to the motor controller, for example. For all these analogue inputs, the subsequent objects offer the possibility to read the current input voltage (analog_input_voltage) and set an offset (analog_input_offset).

Description of the objects

Index	Object	Name	Type	Attr.
2400 _h	ARRAY	analog_input_voltage	INT16	ro
2401 _h	ARRAY	analog_input_offset	INT32	rw

2400_h: analog_input_voltage (input voltage)

The object group analog_input_voltage delivers the current input voltage of the respective channel in millivolts, taking into account the offset.

Index	2400_h
Name	analog_input_voltage
Object Code	ARRAY
No. of Elements	3
Data Type	INT16

Sub-Index	01_h
Description	analog_input_voltage_ch_0
Access	ro
PDO mapping	no
Units	mV
Value Range	–
Default Value	–

Sub-Index	02_h
Description	analog_input_voltage_ch_1
Access	ro
PDO mapping	no
Units	mV
Value Range	–
Default Value	–

Sub-Index	03_h
Description	analog_input_voltage_ch_2
Access	ro
PDO mapping	no
Units	mV
Value Range	–
Default Value	–

Object 2401_h: analog_input_offset (offset for analogue inputs)

Through the object group **analog_input_offset**, the offset voltage in millivolts for the respective inputs can be set or read. With help of the offset, direct voltage that may be present can be compensated. A positive offset compensates thereby for a positive input voltage.

Index	2401_h
Name	analog_input_offset
Object Code	ARRAY
No. of Elements	3
Data Type	INT32

Sub-Index	01_h
Description	analog_input_offset_ch_0
Access	rw
PDO mapping	no
Units	mV
Value Range	–10000 ... 10000
Default Value	0

Sub-Index	02_h
Description	analog_input_offset_ch_1
Access	rw
PDO mapping	no
Units	mV
Value Range	–10000 ... 10000
Default Value	0

Sub-Index	03_h
Description	analog_input_offset_ch_2
Access	rw
PDO mapping	no
Units	mV
Value Range	–10000 ... 10000
Default Value	0

5.13 Digital inputs and outputs

Overview

All digital inputs of the motor controller can be read via the CAN bus, and almost all digital outputs can be set as desired. Moreover, status messages can be assigned to the digital outputs of the motor controller.

Description of the objects

Objects treated in this chapter

Index	Object	Name	Type	Attr.
60FD _h	VAR	digital_inputs	UINT32	ro
60FE _h	ARRAY	digital_outputs	UINT32	rw
2420 _h	RECORD	digital_output_state_mapping		ro
2420 _h _01 _h	VAR	dig_out_state_mapp_dout_1	UINT8	rw
2420 _h _02 _h	VAR	dig_out_state_mapp_dout_2	UINT8	rw
2420 _h _03 _h	VAR	dig_out_state_mapp_dout_3	UINT8	rw

Object 60FD_h: digital_inputs

The digital inputs can be read via the object 60FD_h:

Index	60FD_h
Name	digital_inputs
Object Code	VAR
Data Type	UINT32

Access	ro
PDO mapping	yes
Units	–
Value Range	according to the following table
Default Value	0

Bit	Value	Meaning
0	00000001 _h	Negative limit switch
1	00000002 _h	Positive limit switch
2	00000004 _h	Reference switch
3	00000008 _h	Interlock (controller or output stage enable missing)
16 ... 23	00FF0000 _h	Digital inputs of the CAMC-D-8E8A
24 ... 27	0F000000 _h	DIN0 ... DIN3
28	10000000 _h	DIN 8
29	20000000 _h	DIN 9

Object 60FE_h: digital_outputs

The digital outputs can be triggered via the object 60FE_h. To do this, which of the digital outputs should be triggered must be specified in the object digital_outputs_mask. The selected outputs can then be set as desired via the object digital_outputs_data. It should be noted that a delay of up to 10 ms may occur in triggering the digital outputs. When the outputs are really set can be determined by reading back the object 60FE_h.

Index	60FE_h
Name	digital_outputs
Object Code	ARRAY
No. of Elements	2
Data Type	UINT32

Sub-Index	01_h
Description	digital_outputs_data
Access	rw
PDO mapping	yes
Units	–
Value Range	–
Default Value	(dependent on the status of the brake)

Sub-Index	02_h
Description	digital_outputs_mask
Access	rw
PDO mapping	yes
Units	–
Value Range	–
Default Value	00000000 _h

Bit	Value	Meaning
0	00000001 _h	1 = Energize brake
16 ... 23	0E000000 _h	Digital outputs of the CAMC-D-8E8A
25 ... 27	0E000000 _h	DOUT1 ... DOUT3

**Caution**

If brake triggering is enabled via digital_output_mask, deletion of bit 0 in digital_output_data causes the holding brake to be manually ventilated! In case of hanging axes, this can result in sagging of the axes.

Object 2420_h: digital_output_state_mapping

Through the object group `digital_outputs_state_mapping`, various status messages of the motor controller can be output over the digital outputs.

For the integrated digital outputs of the motor controller, a separate sub-index is available for each output for this purpose. As a result, for each output, a byte is available into which the function number must be entered.

If such a function has been assigned to a digital output and the output is then switched on or off directly over `digital_outputs` (60FE_h), the object `digital_outputs_state_mapping` is also set to OFF (0) or ON (12).

Index	2420_h
Name	digital_outputs_state_mapping
Object Code	RECORD
No. of Elements	5

Sub-Index	01_h
Description	dig_out_state_mapp_dout_1
Data Type	UINT8
Access	rw
PDO mapping	no
Units	–
Value Range	0 ... 44, → table
Default Value	0

Sub-Index	02_h
Description	dig_out_state_mapp_dout_2
Data Type	UINT8
Access	rw
PDO mapping	no
Units	–
Value Range	0 ... 44, → table
Default Value	0

Sub-Index	03_h
Description	dig_out_state_mapp_dout_3
Data Type	UINT8
Access	rw
PDO mapping	no
Units	–
Value Range	0 ... 44, → table
Default Value	0

Value	Measurement file
0	Off (output is low)
1	Position $X_{\text{setpoint}} = X_{\text{dest}}$
2	Position $X_{\text{act}} = X_{\text{dest}}$
3	Reserved
4	Remaining path trigger active
5	Reference run active
6	Declared Velocity Reached
7	I ² t motor reached
8	Contouring error
9	Undervoltage in intermediate circuit
10	Holding brake released
11	Output stage switched on
12	On (output is high)
13	Common error active
14	At least one setpoint lock active
15	Linear motor identified
16	Homing position valid
17	Common status: ready for controller enable
18	Position Trigger 1
19	Position Trigger 2
20	Position Trigger 3
21	Position Trigger 4
22 ... 25	Reserved
26	Alternative target reached
27	Active if position set running
28	Declared torque reached
29	Position $x_{\text{set}} = x_{\text{target}}$ (also with linking for at least 10 ms)
30	Ack signal (active low) as handshake to start position
31	Destination reached with handshake for the dig. Start, is not set as long as START is on HIGH level.
32	Cam disc active
33	CAM-IN movement in operation
34	CAM-CHANGE, like CAM-IN, but change to a new curve
35	CAM-OUT movement in operation
36	Level of digital output stage enable, that is level at DIN4 (high if DIN4 high)
37	Reserved
38	CAM active without CAM-IN or CAM-CHANGE movement
39	Speed actual value in the window for rest
40	Teach acknowledge
41	Saving process (SAVE!, Save Positions) in operation
42	STO active
43	STO is requested
44	Motion Complete (MC)

Sub-Index	11_h
Description	dig_out_state_mapp_ea88_0_low
Data Type	UINT32
Access	rw
PDO mapping	no
Units	–
Value Range	0 ... FFFFFFFF _h , → table
Default Value	0

Bit	Mask	Name	Measurement file
0 ... 7	000000FF _h	EA88_0_dout_0_mapping	Function for CAMC-D-8E8A 0 DOUT1
8 ... 15	0000FF00 _h	EA88_0_dout_1_mapping	Function for CAMC-D-8E8A 0 DOUT2
16 ... 23	00FF0000 _h	EA88_0_dout_2_mapping	Function for CAMC-D-8E8A 0 DOUT3
24 ... 31	FF000000 _h	EA88_0_dout_3_mapping	Function for CAMC-D-8E8A 0 DOUT4

Sub-Index	12_h
Description	dig_out_state_mapp_ea88_0_low
Data Type	UINT32
Access	rw
PDO mapping	no
Units	–
Value Range	0 ... FFFFFFFF _h , → table
Default Value	0

Bit	Mask	Name	Measurement file
0 ... 7	000000FF _h	EA88_0_dout_4_mapping	Function for CAMC-D-8E8A 0 DOUT5
8 ... 15	0000FF00 _h	EA88_0_dout_5_mapping	Function for CAMC-D-8E8A 0 DOUT6
16 ... 23	00FF0000 _h	EA88_0_dout_6_mapping	Function for CAMC-D-8E8A 0 DOUT7
24 ... 31	FF000000 _h	EA88_0_dout_7_mapping	Function for CAMC-D-8E8A 0 DOUT8

5.14 Limit Switch/Reference Switch

Overview

Proximity switches (limit switch) or reference switches (homing switch) can be used alternatively for definition of the reference position of the motor controller. Further information on the possible homing methods can be found in chapter 7.2, Operating mode reference travel (homing mode).

Description of the objects

Index	Object	Name	Type	Attr.
6510 _h	RECORD	drive_data		rw

Object 6510_h_11h: limit_switch_polarity

The polarity of the limit switches can be programmed via the object 6510_h_11h (limit_switch_polarity). For normally closed limit switches, a “0” must be entered in this object, whereas a “1” must be entered when normally open contacts are used.

Index	6510_h
Name	drive_data
Object Code	RECORD
No. of Elements	51

Sub-Index	11_h
Description	limit_switch_polarity
Data Type	INT16
Access	rw
PDO mapping	no
Units	–
Value Range	0, 1
Default Value	1

Value	Meaning
0	N/C contact
1	N/O contact

Object 6510_h_12_h: limit_switch_selector

Through the object 6510_h_12_h (limit_switch_selector), assignment of the limit switches (negative, positive) can be exchanged without having to make changes to the cabling. To exchange the assignment of the limit switches, a One must be entered.

Sub-Index	12_h
Description	limit_switch_selector
Data Type	INT16
Access	rw
PDO mapping	no
Units	–
Value Range	0, 1
Default Value	0

Value	Meaning
0	DIN6 = E0 (limit switch negative) DIN7 = E1 (limit switch positive)
1	DIN6 = E1 (limit switch positive) DIN7 = E0 (limit switch negative)

Object 6510_h_14_h: homing_switch_polarity

The polarity of the reference switch can be programmed through the object 6510_h_14_h (homing_switch_polarity). For a normally closed reference switch, a zero is entered in this object, whereas a “1” is entered when normally open contacts are used.

Sub-Index	14_h
Description	homing_switch_polarity
Data Type	INT16
Access	rw
PDO mapping	no
Units	–
Value Range	0, 1
Default Value	1

Value	Meaning
0	N/C contact
1	N/O contact

Object 6510_h_13_h: homing_switch_selector

The object 6510_h_13_h (homing_switch_selector) determines whether DIN8 or DIN9 should be used as reference switch.

Sub-Index	13_h
Description	homing_switch_selector
Data Type	INT16
Access	rw
PDO mapping	no
Units	–
Value Range	0, 1
Default Value	0

Value	Meaning
0	DIN 9
1	DIN 8

Object 6510_h_15_h: limit_switch_deceleration

The object limit_switch_deceleration establishes the deceleration used in braking when the limit switch is reached during normal operation (limit switch emergency stop ramp).

Sub-Index	15_h
Description	limit_switch_deceleration
Data Type	INT32
Access	rw
PDO mapping	no
Units	acceleration units
Value Range	0 ... 3000000 min ⁻¹ /s
Default Value	2000000 min ⁻¹ /s

5.15 Sampling of Positions

Overview

The CMMP family offers the possibility to save the actual position value on the rising or falling edge of a digital input. This position value can then be read out for calculation within a controller, for example.

All necessary objects are summarised in the record `sample_data`: The object `sample_mode` establishes the type of sampling: whether only a one-time sample event should be recorded or continuous sampling take place. Through the object `sample_status`, the controller can query whether a sample event has occurred. This is signaled by a set bit, which can also be displayed in the status word if the object `sample_status_mask` is set correspondingly.

The object `sample_control` controls enabling of the sample event, and the stamped positions can ultimately be read via the objects `sample_position_rising_edge` and `sample_position_falling_edge`.

Which digital input is used can be established with the parametrisation software under Controller – I/O configuration – Digital inputs – Sample input.

Description of the objects

Objects treated in this chapter

Index	Object	Name	Type	Attr.
204A _h	RECORD	sample_data		ro
204A _h _01 _h	VAR	sample_mode	UINT16	rw
204A _h _02 _h	VAR	sample_status	UINT8	ro
204A _h _03 _h	VAR	sample_status_mask	UINT8	rw
204A _h _04 _h	VAR	sample_control	UINT8	wo
204A _h _05 _h	VAR	sample_position_rising_edge	INT32	ro
204A _h _06 _h	VAR	sample_position_falling_edge	INT32	ro

Object 204A_h: sample_data

Index	204A_h
Name	sample_data
Object Code	RECORD
No. of Elements	6

The following object can be used to select whether the position should be determined for each occurrence of a sample event (continuous sampling) or sampling should be blocked after a sample event until sampling is approved again. Observe that even just one bounce can trigger both edges!

Sub-Index	01_h
Description	sample_mode
Data Type	UINT16
Access	rw
PDO mapping	no
Units	–
Value Range	0 ... 1, → table
Default Value	0

Value	Measurement file
0	Continuous sampling
1	Autolock sampling

The following object displays a new sample event.

Sub-Index	02_h
Description	sample_status
Data Type	UINT8
Access	ro
PDO mapping	yes
Units	–
Value Range	0 ... 3, → table
Default Value	0

Bit	Value	Name	Description
0	01 _h	falling_edge_occurred	= 1: New sample position (falling edge)
1	02 _h	rising_edge_occurred	= 1: New sample position (rising edge)

The bits of the object `sample_status` can be established with the following object, which should also result in setting bit 15 of the status word. Through this, the information “sample event occurred” is available in the status word, which will normally be transmitted anyway, so the controller has to read the object `sample_status` only to determine which edge has occurred, if applicable.

Sub-Index	03_h
Description	sample_status_mask
Data Type	UINT8
Access	rw
PDO mapping	yes
Units	–
Value Range	0 ... 1, → table
Default Value	0

Bit	Value	Name	Description
0	01 _h	rising_edge_visible	If rising_edge_occurred = 1 → Status word bit 15 = 1
1	02 _h	falling_edge_visible	If falling_edge_occurred = 1 → Status word bit 15 = 1

Setting of the respective bit in sample_control resets the corresponding status bit in sample_status and also releases the sampling again in the case of “Autolock” sampling.

Sub-Index	04_h
Description	sample_control
Data Type	UINT8
Access	wo
PDO mapping	yes
Units	–
Value Range	0 ... 1, → table
Default Value	0

Bit	Value	Name	Description
0	01 _h	falling_edge_enable	Sampling with falling edge
1	02 _h	rising_edge_enable	Sampling with rising edge

The following objects contain the sampled positions.

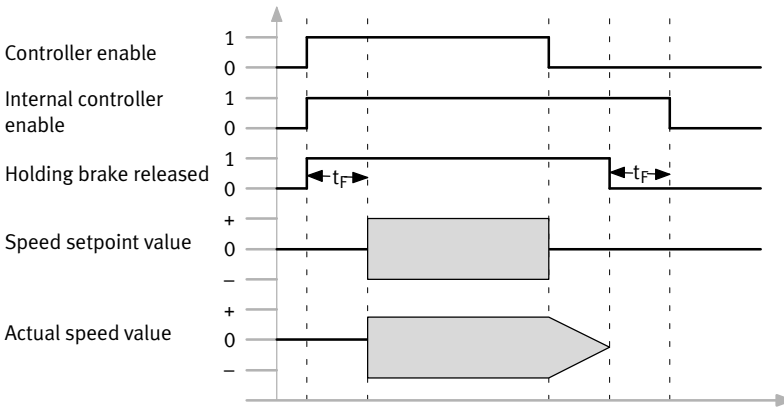
Sub-Index	05_h
Description	sample_position_rising_edge
Data Type	INT32
Access	ro
PDO mapping	yes
Units	position units
Value Range	–
Default Value	–

Sub-Index	06_h
Description	sample_position_falling_edge
Data Type	INT32
Access	ro
PDO mapping	yes
Units	position units
Value Range	–
Default Value	–

5.16 Brake Activation

Overview

Using the subsequent objects, it can be parametrised how the motor controller controls a holding brake integrated into the motor, if necessary. The holding brake is always enabled as soon as controller enable is switched on. For holding brakes with high mechanical inertia, a time delay can be parametrised so that the holding brake takes effect before the output stage is switched off (sagging of vertical axes). This delay is parametrised through the object `brake_delay_time`. As can be seen from the sketch, when the controller enable is switched on, the speed setpoint value is approved only after the `brake_delay_time`, and when the the controller enable is switched off, turning off the regulation is delayed by this time.



t_F : Travel start delay

Fig. 5.8 Function of brake delay (with speed adjustment / positioning)

Description of the objects

Index	Object	Name	Type	Attr.
6510 _h	RECORD	drive_data		rw

Object 6510_h_18_h: brake_delay_time

This brake delay time can be parametrised through the object `brake_delay_time`.

Index	6510_h
Name	drive_data
Object Code	RECORD
No. of Elements	51

Sub-Index	18_h
Description	brake_delay_time
Data Type	UINT16
Access	rw
PDO mapping	no
Units	ms
Value Range	0 ... 32000
Default Value	0

5.17 Device Information

Index	Object	Name	Type	Attr.
1018 _h	RECORD	identity_object		rw
6510 _h	RECORD	drive_data		rw

Via numerous CAN objects, the most varied of information, such as motor controller type, firmware used, etc. can be read out of the device.

Description of the objects

Object 1018_h: identity_object

Through the identity_object established in the CiA 301, the motor controller can be uniquely identified in a CANopen-network. For this purpose, the manufacturer code (vendor_id), a unique product code (product_code), the revision number of the CANopen implementation (revision_number) and the serial number of the device (serial_number) can be read out.

Index	1018_h
Name	identity_object
Object Code	RECORD
No. of Elements	4

Sub-Index	01_h
Description	vendor_id
Data Type	UINT32
Access	ro
PDO mapping	no
Units	–
Value Range	0000001D
Default Value	0000001D

Sub-Index	02_h
Description	product_code
Data Type	UINT32
Access	ro
PDO mapping	no
Units	–
Value Range	–
Default Value	device-dependent

Sub-Index	03_h
Description	revision_number
Data Type	UINT32
Access	ro
PDO mapping	no
Units	MMMMSSSS _h (M: main version, S: sub version)
Value Range	–
Default Value	–

Sub-Index	04_h
Description	serial_number
Data Type	UINT32
Access	ro
PDO mapping	no
Units	–
Value Range	–
Default Value	–

Object 6510_h_A0_h: drive_serial_number

The serial number of the controller can be read via the object drive_serial_number. This object serves to achieve compatibility with earlier versions.

Index	6510_h
Name	drive_data
Object Code	RECORD
No. of Elements	51

Sub-Index	A0_h
Description	drive_serial_number
Data Type	UINT32
Access	ro
PDO mapping	no
Units	–
Value Range	–
Default Value	–

Object 6510_h_A1_h: drive_type

Through the object drive_type, the device type of the controller can be read. This object serves to achieve compatibility with earlier versions.

Sub-Index	A1_h
Description	drive_type
Data Type	UINT32
Access	ro
PDO mapping	no
Units	–
Value Range	→ 1018 _h _02 _h , product_code
Default Value	→ 1018 _h _02 _h , product_code

Object 6510_h_A9_h: firmware_main_version

The main version number of the firmware (product stage) can be read out via the object firmware_main_version.

Sub-Index	A9_h
Description	firmware_main_version
Data Type	UINT32
Access	ro
PDO mapping	no
Units	MMMMSSSS _h (M: main version, S: sub version)
Value Range	–
Default Value	–

Object 6510_h_AA_h: firmware_custom_version

The version number of the customer-specific variants of the firmware can be read out via the object firmware_custom_version.

Sub-Index	AA_h
Description	firmware_custom_version
Data Type	UINT32
Access	ro
PDO mapping	no
Units	MMMMSSSS _h (M: main version, S: sub version)
Value Range	–
Default Value	–

Object 6510_h_AD_h: km_release

Through the version number of the km_release, firmware statuses of the same product stage can be differentiated.

Sub-Index	AD_h
Description	km_release
Data Type	UINT32
Access	ro
PDO mapping	no
Units	–
Value Range	MMMMSSSS _h (M: main version, S: sub version)
Default Value	–

Object 6510_h_AC_h: firmware_type

Through the object firmware_type can be read for which device family and which angle encoder type the loaded firmware is appropriate.

Sub-Index	AC_h
Description	firmware_type
Data Type	UINT32
Access	ro
PDO mapping	no
Units	–
Value Range	00000F2 _h
Default Value	00000F2 _h

Object 6510_h_B0_h: cycletime_current_controller

Through the object cycletime_current_controller, the cycle time of the current regulator in micro-seconds can be read.

Sub-Index	B0_h
Description	cycletime_current_controller
Data Type	UINT32
Access	ro
PDO mapping	no
Units	μs
Value Range	–
Default Value	0000007D _h

Object 6510_h_B1_h: cycletime_velocity_controller

Through the object cycletime_velocity_controller, the cycle time of the speed regulator in micro-seconds can be read.

Sub-Index	B1_h
Description	cycletime_velocity_controller
Data Type	UINT32
Access	ro
PDO mapping	no
Units	μs
Value Range	–
Default Value	000000FA _h

Object 6510_h_B2_h: cycletime_position_controller

Through the object cycletime_position_controller, the cycle time of the position regulator in micro-seconds can be read.

Sub-Index	B2_h
Description	cycletime_position_controller
Data Type	UINT32
Access	ro
PDO mapping	no
Units	μs
Value Range	–
Default Value	000001F4 _h

Object 6510_h_B3_h: cycletime_trajectory_generator

Through the object cycletime_trajectory_generator, the cycle time of the position controller in microseconds can be read.

Sub-Index	B3_h
Description	cycletime_trajectory_generator
Data Type	UINT32
Access	ro
PDO mapping	no
Units	µs
Value Range	–
Default Value	000003E8 _h

Object 6510_h_C0_h: commissioning_state

The commissioning_state object is written by the parameterisation software if certain parameterisations have been executed (e.g. nominal current). After delivery and after restore_default_parameter, this object includes a “0”. In this case, an “A” is displayed on the 7-segments display of the motor controller to point out that this device has not been parameterised yet. If the motor controller is parameterised completely under CANopen, at least one bit in this object must be set to suppress the display “A”. Of course, if required it is also possible to use this object in order to notice the status of the controller parameterisation. Observe in this case that the parameterisation software also accesses this object.

Sub-Index	C0_h
Description	commissioning_state
Data Type	UINT32
Access	rw
PDO mapping	no
Units	–
Value Range	–
Default Value	0

Value	Meaning
0	Nominal current valid
1	Maximum current valid
2	Pole number of the motor valid
3	Offset angle / direction of rotation valid
4	Reserved
5	Offset angle / direction of rotation of Hall encoder valid
6	Reserved
7	Absolute position of encoder system valid
8	Current regulator parameters valid
9	Reserved
10	Physic. units valid
11	Speed regulator valid
12	Position controller valid
13	Safety parameters valid
14	Reserved
15	Limit switch polarity valid
16 ... 31	Reserved



Caution

This object contains no information about whether the motor controller has been correspondingly parametrised correctly for the motor and application, but only whether the named points after shipment have been parametrised at all at least once.



“A” in the 7-segments display

Observe that at least one bit in the commissioning_state object has to be set in order to suppress the “A” on the 7-segments display of your motor controller.

5.18 Error Management

Overview

The motor controllers of the CMMP family offer the option to change the error response of individual events, e.g. the occurrence of a contouring error. Through this, the motor controller reacts differently if a certain event occurs: Dependent on the setting, braking down can occur and the output stage is shut off immediately, but also just a warning can be shown on the display.

For every event, a minimum reaction is intended by the manufacturer, which cannot be fallen below. And so “critical” errors, such as 60-0 short circuit output stage, are not reparametrised, since here an immediate switch-off is necessary to protect the motor controller from possible destruction.

If a lower error response is entered than is permissible for the respective error, the value is limited to the lowest permissible error response. A list of all error numbers is found in chapter B “Diagnostic messages”.

Description of the objects

Objects treated in this chapter

Index	Object	Name	Type	Attr.
2100 _h	RECORD	error_management		ro
2100_01 _h	VAR	error_number	UINT8	rw
2100_02 _h	VAR	error_reaction_code	UINT8	rw
200F _h	VAR	last_warning_code	UINT16	ro

Object 2100_h: error_management

Index	2100_h
Name	error_management
Object Code	RECORD
No. of Elements	2

The main error number whose response should be changed must be specified in the object error_number. The main error number is normally specified before the hyphen (e.g. error 08-2, main error number 8). For possible error numbers, → also chap. 3.5.

Sub-Index	01_h
Description	error_number
Data Type	UINT8
Access	rw
PDO mapping	no
Units	–
Value Range	1 ... 96
Default Value	1

The response of the error can be changed in the object `error_reaction_code`. If the response falls below the manufacturer's minimum response, it is limited to this minimum. The response actually set can be determined by reading back.

Sub-Index	02_h
Description	error_reaction_code
Data Type	UINT8
Access	rw
PDO mapping	no
Units	–
Value Range	0, 1, 3, 5, 7, 8
Default Value	dependent on error_number

Value	Meaning
0	No action
1	Entry in the buffer
3	Warning on the 7-segments display and in the status word
5	Controller enable off
7	Braking with maximum current
8	Output stage off

Object 200F_h: last_warning_code

Warnings are special events of the drive (e.g. a following error), which in contrast to an error should not result in stopping of the drive. Warnings are displayed on the 7-segments display of the controller and after that automatically reset by the controller.

The last warning that occurred can be read via the following object: Bit 15 displays thereby whether the warning is currently still active.

Index	200F_h
Name	last_warning_code
Object Code	VAR
Data Type	UINT16

Access	ro
PDO mapping	yes
Units	–
Value Range	–
Default Value	–

Bit	Value	Meaning
0 ... 3	000F _h	Sub-number of the warning
4 ... 11	0FF0 _h	Main number of the warning
15	8000 _h	Warning is active

6 Device Control

6.1 Status Diagram (State Machine)

6.1.1 Overview

The following chapter describes how the motor controller can be regulated under CANopen, that is, how the output stage is switched on or an error is acknowledged, for example.

Under CANopen, the entire control of the motor controller is achieved through two objects: The host can control the motor controller through the controlword, while the status of the motor controller can be read back in the statusword object. The following terms are used to explain controller regulation:

Term	Description
Status: (state)	The motor controller is in different statuses, depending on whether the output stage is switched on or an error has occurred, for example. The statuses defined under CANopen are presented in the course of the chapter. Example: SWITCH_ON_DISABLED
Status transition (state transition)	Just as with the statuses, it is also defined under CANopen how to go from one status to another (e.g. to acknowledge an error). Status transitions are triggered by the host by setting bits in the controlword or internally through the motor controller, when it recognises an error, for example.
Command (command)	To trigger status transitions, certain combinations of bits must be set in the controlword. Such a combination is designated a command. Example: Enable Operation
Status diagram (state machine)	The statuses and status transitions together form the status diagram, that is, the overview of all conditions and the transitions possible from there.

Tab. 6.1 Controller regulation terms

6.1.2 Status diagram of the motor controller (state machine)

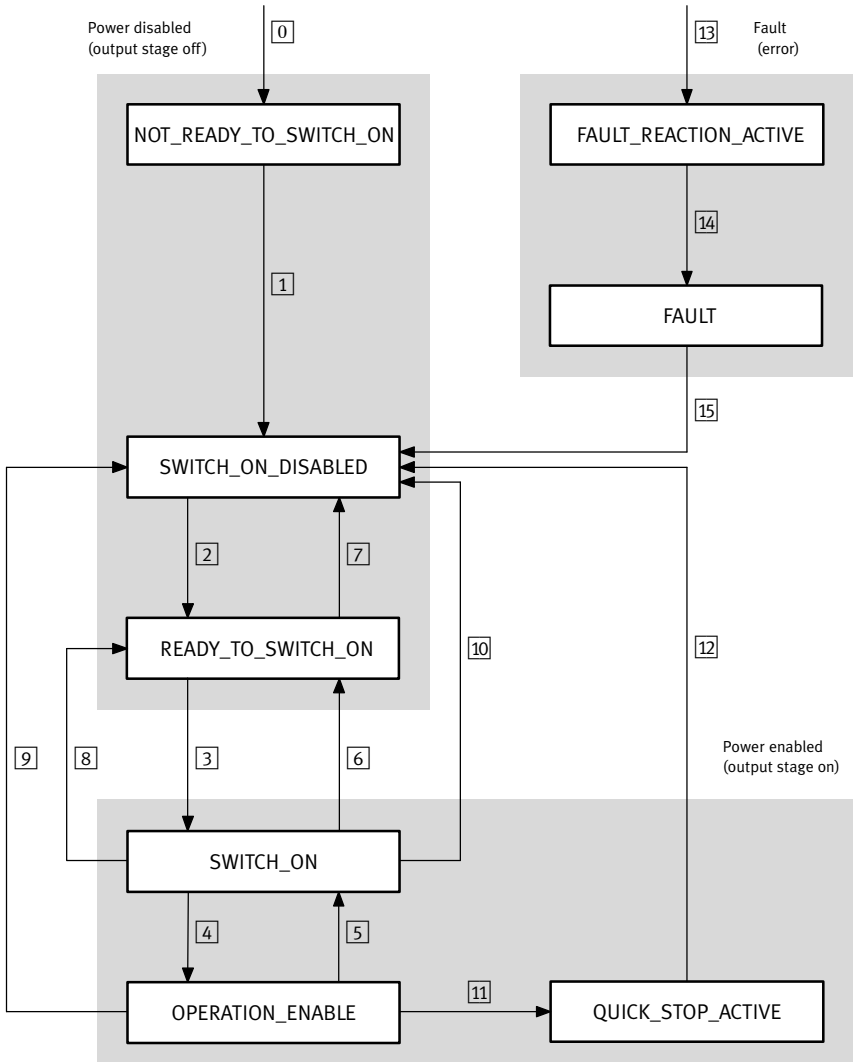


Fig. 6.1 Status diagram of the motor controller

The status diagram can be roughly divided into three areas: “Power Disabled” means that the output stage is switched off and “Power Enabled” that the output stage is switched on. The statuses needed for error handling are summarised in the “Fault” area.

The most important statuses of the motor controller are shown highlighted in the diagram. After it is switched on, the motor controller initialises itself and then reaches the status SWITCH_ON_DISABLED. In this status, the CAN communication is fully operational and the motor controller can be parametrised (e.g. the “speed adjustment” operating mode can be set). The output stage is switched off and the shaft is thus freely rotatable. Through the status transitions [2], [3], [4] – which corresponds in principle to CAN controller enable – the status OPERATION_ENABLE is reached. In this status, the output stage is switched on and the motor is controlled in accordance with the set operating mode. Always make sure beforehand that the drive is correctly parametrised and a corresponding setpoint value is equal to zero. The status transition [9] corresponds to removal of enable, i.e. a motor that is still running would run out uncontrolled.

If an error occurs (regardless from which status), the system ultimately branches into the FAULT status. Depending on the severity of the error, certain actions, such as emergency braking, can still be performed before (FAULT_REACTION_ACTIVE).

In order to perform the named status transitions, certain bit combinations must be set in the controlword (see below). The lower 4 bits of the controlword are jointly evaluated in order to trigger a status transition.

In the following, only the most important status transitions [2], [3], [4], [9] and [15] are explained at first. A table of all possible statuses and status transitions are found at the end of this chapter.

The following table contains the desired status transition in the 1st column and in the 2nd column the requirements necessary for it (usually a command through the host, here depicted with frame). How this command is generated, i.e. which bits in the controlword must be set, is visible in the 3rd column (x = not relevant).

No.	Is performed when	Bit combination (controlword)					Action
		Bit	3	2	1	0	
[2]	Output stage and controller enable prev. + command Shutdown	Shutdown =	x	1	1	0	None
[3]	Command Switch On	Switch On =	x	1	1	1	Switching on the output stage
[4]	Command Enable Operation	Enable Operation =	1	1	1	1	Control in accordance with set operating mode
[9]	Command Disable Voltage	Disable Voltage =	x	x	0	x	Output stage is blocked. Motor rotates freely.
[15]	Error eliminated + Fault Reset command	Fault Reset =	Bit 7 =			0 → 1	Acknowledge Error

Tab. 6.2 Most important status transitions of the motor controller

EXAMPLE

After the motor controller has been parametrised, the motor controller should be “enabled”, that is, the output stage switched on:

1. The motor controller is in the status SWITCH_ON_DISABLED
2. The motor controller should be in the status OPERATION_ENABLE
3. According to the status diagram (Fig. 6.1) the transitions 2, 3 and 4 must be executed.
4. From Tab. 6.2 follows:

Transition 2: controlword = 0006h

New status: READY_TO_SWITCH_ON¹⁾

Transition 3: controlword = 0007h

New status: SWITCHED_ON¹⁾

Transition 4: controlword = 000Fh

New status: OPERATION_ENABLE¹⁾

Instructions:

1. The example assumes that no further bits are set in the controlword (for the transitions, only the bits 0 ... 3 are important).
2. The transitions 3 and 4 can be combined by immediately setting the controlword to 000Fh. For the status transition 2, the set bit 3 is not relevant.

1) The Host must wait until the status in the statusword can be read back. This is explained in detail below.

Status diagram: statuses

The following table lists all statuses and their meaning:

Name	Significance
NOT_READY_TO_SWITCH_ON	The motor controller performs a self-test. The CAN communication does not work yet.
SWITCH_ON_DISABLED	The motor controller has completed its self-test. CAN communication is possible.
READY_TO_SWITCH_ON	The motor controller waits until the digital inputs “output stage” and “controller enable” are at 24 V. (Controller enable logic “Digital input and CAN”).
SWITCHED_ON ¹⁾	The output stage is switched on.
OPERATION_ENABLE ¹⁾	Voltage to the motor is on, and the motor is regulated corresponding to the operating mode.
QUICKSTOP_ACTIVE ¹⁾	The Quick Stop Function is being executed (→ quick_stop_option_code). Voltage to the motor is on, and the motor is regulated according to the quick stop function.
FAULT_REACTION_ACTIVE ¹⁾	An error has occurred. With critical errors, the system immediately switches into the Fault status. Otherwise, the action specified in the fault_reaction_option_code is carried out. Voltage to the motor is on, and the motor is regulated according to the fault reaction function.
FAULT	An error has occurred. No voltage is applied to the motor.

1) The output stage is switched on.

Status diagram: status transitions

The following table lists all statuses and their meaning:

No.	Is performed when	Bit combination (controlword)					Action
		Bit	3	2	1	0	
0	Switched on or reset occurs	Internal transition					Perform self-test
1	Self-test successful	Internal transition					Activation of CAN communication
2	Output stage and controller enable prev. + command Shutdown	Shutdown	x	1	1	0	–
3	Command Switch On	Switch On	x	1	1	1	Switching on the output stage
4	Command Enable Operation	Enable Operation	1	1	1	1	Control in accordance with set operating mode
5	Command Disable Operation	Disable Operation	0	1	1	1	Output stage is blocked. Motor rotates freely.
6	Command Shutdown	Shutdown	x	1	1	0	Output stage is blocked. Motor rotates freely.
7	Command Quick Stop	Quick Stop	x	0	1	x	–
8	Command Shutdown	Shutdown	x	1	1	0	Output stage is blocked. Motor rotates freely.
9	Command Disable Voltage	Disable Voltage	x	x	0	x	Output stage is blocked. Motor rotates freely.
10	Command Disable Voltage	Disable Voltage	x	x	0	x	Output stage is blocked. Motor rotates freely.
11	Command Quick Stop	Quick Stop	x	0	1	x	Braking is initiated in accordance with <code>quick_stop_option_code</code> .
12	Braking ended without command Disable Voltage	Disable Voltage	x	x	0	x	Output stage is blocked. Motor rotates freely.
13	Error occurred	Internal transition					For uncritical errors, reaction in accordance with <code>fault_reaction_option_code</code> . With critical errors, transition 14 follows
14	Error handling is ended	Internal transition					Output stage is blocked. Motor rotates freely.
15	Error eliminated + Fault Reset command	Fault Reset	Bit 7 =			0 → 1	Acknowledge error (with rising edge)

**Caution****Output stage blocked ...**

... means that the power semiconductors (transistors) can no longer be actuated. If this status is taken with a turning motor, it runs out unbraked. If a mechanical motor brake is present, it is automatically actuated.

The signal does not guarantee that the motor is really voltage-free.

**Caution****Output stage enabled ...**

... means that the motor is actuated and controlled corresponding to the selected operating mode. An existing mechanical motor brake will be released automatically. In case of a defect or incorrect parametrisation (motor current, number of poles, resolver offset angle, etc.), this can result in uncontrolled behaviour of the drive.

6.1.3 Control word (controlword)

Object 6040_h: controlword

With the controlword, the current status of the motor controller can be revised or a specific action (e.g. start of homing) triggered directly. The function of bits 4, 5, 6 and 8 depends on the current operating mode (modes_of_operation) of the motor controller, which is explained after this chapter.

Index	6040_h
Name	controlword
Object Code	VAR
Data Type	UINT16

Access	rw
PDO Mapping	yes
Units	–
Value Range	–
Default Value	0

Bit	Value	Function
0	0001 _h	Control of the status transitions. (These bits are evaluated together)
1	0002 _h	
2	0004 _h	
3	0008 _h	
4	0010 _h	new_set_point/start_homing_operation/enable_ip_mode
5	0020 _h	change_set_immediately
6	0040 _h	absolute/relative
7	0080 _h	reset_fault
8	0100 _h	halt
9	0200 _h	reserved – set to 0
10	0400 _h	reserved – set to 0
11	0800 _h	reserved – set to 0
12	1000 _h	reserved – set to 0
13	2000 _h	reserved – set to 0
14	4000 _h	reserved – set to 0
15	8000 _h	reserved – set to 0

Tab. 6.3 Bit assignment of the controlword

As already comprehensively described, status transitions can be carried out with the bits 0 ... 3. The commands necessary for this are presented again here in an overview. The Fault Reset command is generated by bit 7 through a positive edge change (from 0 to 1).

Command:	Bit 7	Bit 3	Bit 2	Bit 1	Bit 0
	0008 _h	0008 _h	0004 _h	0002 _h	0001 _h
Shutdown	x	x	1	1	0
Switch On	x	x	1	1	1
Disable Voltage	x	x	x	0	x
Quick Stop	x	x	0	1	x
Disable Operation	x	0	1	1	1
Enable Operation	x	1	1	1	1
Fault Reset	0 → 1	x	x	x	x

Tab. 6.4 Overview of all commands (x = not relevant)



As some status modifications require a certain amount of time, all status modifications triggered via the controlword must be read back via the statusword. Only when the requested status can also be read in the statusword may a further command be written via the controlword.

The remaining bits of the controlwords are explained in the following. Some bits have different significance, depending on the operating mode (`modes_of_operation`), i.e. whether the motor controller is speed- or torque-controlled, for example:

controlword		
Bit	Function	Description
4	Dependent on <code>modes_of_operation</code>	
	<code>new_set_point</code>	In the Profile Position Mode: A rising edge signals to the motor controller that a new positioning task should be undertaken. → for this, see unconditionally chapter 7.3.
	<code>start_homing_operation</code>	In the Homing Mode: A rising edge causes the parametrised reference travel to start. A falling edge interrupts a running reference travel prematurely.
	<code>enable_ip_mode</code>	In the Interpolated Position Mode: This bit must be set when the interpolation data records are supposed to be evaluated. It is acknowledged through the bit <code>ip_mode_active</code> in the statusword. → unconditionally also chapter 7.4.
5	<code>change_set_immediately</code>	Only in the Profile Position Mode: If this bit is not set, any positioning tasks currently running will be worked off before any new one is begun. If the bit is set, an ongoing positioning is interrupted immediately and replaced by the new positioning task. → for this, see unconditionally chapter 7.3.
6	<code>relative</code>	Only in the Profile Position Mode: If the bit is set, the motor controller obtains the target position (<code>target_position</code>) of the current positioning task relative to the setpoint position (<code>position_demand_value</code>) of the position controller.
7	<code>reset_fault</code>	In the transition from zero to one, the motor controller tries to acknowledge the existing errors. This is only successful if the cause of the error has been resolved.

controlword		
Bit	Function	Description
8	halt	In the Profile Position Mode: If the bit is set, the ongoing positioning is interrupted. Braking is with the profile_deceleration. After the process is ended, the bit target_reached is set in the statusword. Deletion of the bit has no effect.
		In the Profile Velocity Mode: If the bit is set, the speed is reduced to zero. Braking is with the profile_deceleration. Deletion of the bit causes the motor controller to accelerate again.
		In the Profile Torque Mode: If the bit is set, the torque is lowered to zero. This occurs with the torque_slope. Deletion of the bit causes the motor controller to accelerate again.
		In the Homing Mode: If the bit is set, the ongoing reference travel is interrupted. Deletion of the bit has no effect.

Tab. 6.5 controlword bit 4 ... 8

6.1.4 Read-out of the motor controller status

Just as various status transitions can be triggered via the combination of several bits of the controlwords, the status of the motor controller can be read out via the combination of various bits of the statusword.

The following table lists the possible statuses of the status diagram as well as the related bit combination, with which they are displayed in the statusword.

Status	Bit 6	Bit 5	Bit 3	Bit 2	Bit 1	Bit 0	Mask	Value
	0040 _h	0020 _h	0008 _h	0004 _h	0002 _h	0001 _h		
Not_Ready_To_Switch_On	0	x	0	0	0	0	004F _h	0000 _h
Switch_On_Disabled	1	x	0	0	0	0	004F _h	0040 _h
Ready_to_Switch_On	0	1	0	0	0	1	006F _h	0021 _h
Switched_On	0	1	0	0	1	1	006F _h	0023 _h
OPERATION_ENABLE	0	1	0	1	1	1	006F _h	0027 _h
QUICK_STOP_ACTIVE	0	0	0	1	1	1	006F _h	0007 _h
Fault_Reaction_Active	0	x	1	1	1	1	004F _h	000F _h
Fault	0	x	1	1	1	1	004F _h	0008 _h
FAULT (in accordance with CiA402) ¹⁾	0	x	1	0	0	0	004F _h	0008 _h

Tab. 6.6 Device status (x = not relevant)

EXAMPLE

The above example shows which bits in the controlword need to be set in order to enable the motor controller. Now the newly written status should be read out of the statusword:

Transition from SWITCH_ON_DISABLED to OPERATION_ENABLE:

1. Write status transition **2** into the controlword.
2. Wait until the status READY_TO_SWITCH_ON is displayed in the statusword.

Transition 2: controlword = 0006_h

Wait until (statusword & 006F_h) = 0021_h¹⁾

3. Status transition **3** and **4** can be written combined into the controlword.
4. Wait until the status OPERATION_ENABLE is displayed in the statusword.

Transition 3+4: controlword = 000F_h

Wait until (statusword & 006F_h) = 0027_h¹⁾

Note:

The example assumes that no further bits are set in the controlword (for the transitions, only the bits 0 ... 3 are important).

- 1) To identify the statuses, bits that are not set must also be evaluated (see table). For that reason, the statusword must be masked correspondingly.

6.1.5 Status words (statuswords)

Object 6041_h: statusword

Index	6041_h
Name	statusword
Object Code	VAR
Data Type	UINT16

Access	ro
PDO Mapping	yes
Units	–
Value Range	–
Default Value	–

Bit	Value	Function
0	0001 _h	Status of the motor controller (→ Tab. 6.6). (These bits must be evaluated together.)
1	0002 _h	
2	0004 _h	
3	0008 _h	
4	0010 _h	voltage_enabled
5	0020 _h	Status of the motor controller (→ Tab. 6.6).
6	0040 _h	
7	0080 _h	warning
8	0100 _h	drive_is_moving
9	0200 _h	remote
10	0400 _h	target_reached
11	0800 _h	internal_limit_active
12	1000 _h	set_point_acknowledge/speed_0/homing_attained/ip_mode_active
13	2000 _h	following_error/homing_error
14	4000 _h	manufacturer_statusbit
15	8000 _h	Drive referenced

Tab. 6.7 Bit allocation in the status word



All bits of the statusword are unbuffered. They represent the current device status.

Besides the motor controller status, various events are displayed in the statusword, i.e. a specific event, such as following error, is assigned to each bit. The individual bits have the following significance thereby:

statusword		
Bit	Function	Description
4	voltage_enabled	This bit is set when the output stage transistors are switched on. If bit 7 is set in the object 6510 _h _F0 _h (compatibility_control), (→ chap. 5.2) the following applies: This bit is set if the output stage transistors are switched on.

Tab. 6.8 statusword bit 4



Warning

In case of a defect, the motor can still be powered.

statusword		
Bit	Function	Description
5	quick_stop	If the bit is deleted, the drive carries out a Quick Stop in accordance with quick_stop_option_code.
7	warning	This bit shows that a warning is active.
8	drive_is_moving	This bit is set independently of modes_of_operation when the current actual speed (velocity_actual_value) of the drive is outside the related tolerance window (velocity_threshold).
9	remote	This bit shows that the output stage of the motor controller can be enabled via the CAN network. It is set when the controller enable logic is correspondingly set via the object enable_logic.
10	Dependent on modes_of_operation.	
	target_reached	In the Profile Position Mode: The bit is set when the current target position is reached and the current position (position_actual_value) is located in the parametrised position window (position_window). It is also set when the drive comes to a standstill with Stop bit set. It is deleted as soon as a new target is specified.
		In the Profile Velocity Mode The bit is set when the speed (velocity_actual_value) of the drive is in the tolerance window (velocity_window, velocity_window_time).
11	internal_limit_active	This bit shows that the I ² t limitation is active.
12	Dependent on modes_of_operation.	
	set_point_acknowledge	In the Profile Position Mode This bit is set when the motor controller has recognised the set bit new_set_point in the controlword. It is deleted again after the bit new_set_point in the controlword has been set to zero. ➔ for this, see unconditionally also chapter 7.3
	speed_0	In the Profile Velocity Mode This bit is set when the current actual speed (velocity_actual_value) of the drive is within the related tolerance window (velocity_threshold).
	homing_attained	In the Homing Mode: This bit is set when homing has been ended without error.
	ip_mode_active	In the Interpolated Position Mode: This bit shows that interpolation is active and the interpolation data records are being evaluated. It is set when requested by the bit enable_ip_mode in the controlword. ➔ unconditionally also chapter 7.4.

statusword		
Bit	Function	Description
13	Dependent on <code>modes_of_operation</code> .	
	<code>following_error</code>	In the Profile Position Mode: his bit is set when the current actual position (<code>position_actual_value</code>) differs from the target position (<code>position_demand_value</code>) so much that the difference lies outside the parametrised tolerance window (<code>following_error_window</code> , <code>following_error_time_out</code>).
	<code>homing_error</code>	In the Homing Mode: This bit is set when the reference travel has been interrupted (Halt bit), both limit switches respond simultaneously or the limit switch search run that has already been performed is greater than the specified positioning space (<code>min_position_limit</code> , <code>max_position_limit</code>).
14	<code>manufacturer_statusbit</code>	Manufacturer-specific The significance of this bit can be configured: It can be set when any bit of the <code>manufacturer_statusword_1is</code> set or reset. → on this also chap. 6.1.5 Object 2000 _h .
15	Drive referenced	The bit is set when the controller is referenced. This is the case if either homing has been successfully performed or no homing is needed due to the connected encoder system (e.g. with an absolute encoder).

Tab. 6.9 statusword bit 5 ... 15

Object 2000h: manufacturer_statuswords

The object group `manufacturer_statuswords` was introduced, in order to map additional controller statuses which do not need to be present in the status word, which is queried often. The object group `manufacturer_statuswords` was extended for the safety module.

Index	2000_h
Name	manufacturer_statuswords
Object Code	RECORD
No. of Elements	2

Sub-Index	00_h
Description	manufacturer_statuswords
Data Type	UINT8
Access	ro
PDO Mapping	no
Units	–
Value Range	–
Default Value	1

Sub-Index	01_h
Description	manufacturer_statusword_1
Data Type	UINT32
Access	ro
PDO Mapping	yes
Units	–
Value Range	–
Default Value	–

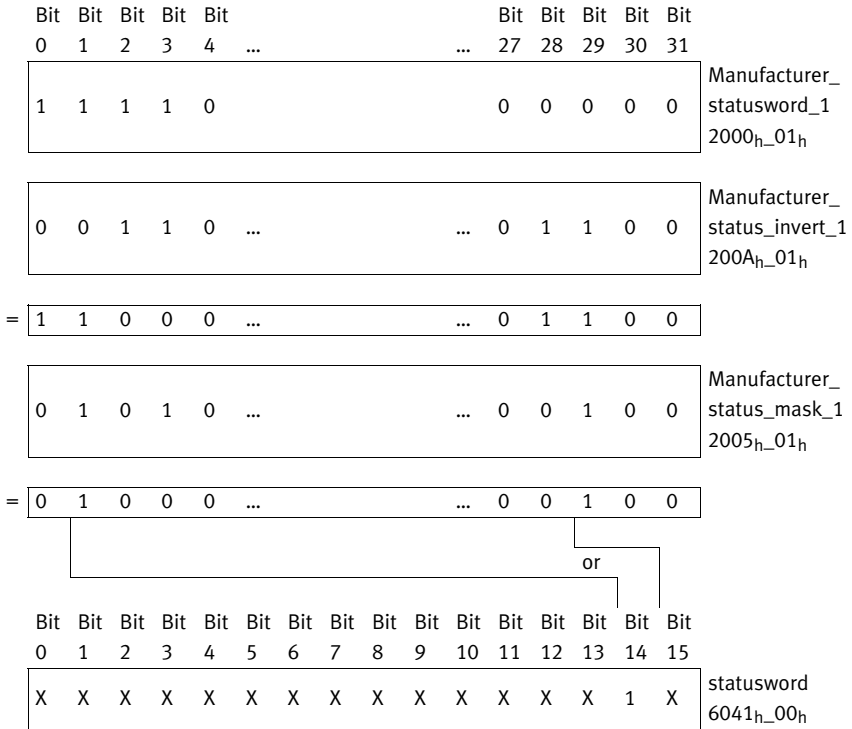
manufacturer_statusword_1		
Bit	Signal	Description
Bit 0	IS_REFERENCED	Drive is referenced
Bit 1	COMMUTATION_VALID	Commutation valid
Bit 2	READY_FOR_ENABLE	The bit is set if all conditions to enable the controller are present and only the controller enable itself is lacking. The following conditions must be present: <ul style="list-style-type: none"> – The drive is error free. – The intermediate circuit is loaded. – Angle encoder analysis is ready. No processes (e.g. serial transmission) are active that prevent enabling. – No blocking process is active (e.g. the automatic motor parameter identification). – STO is not active or a safety function is active, which permits enabling.
Bit 3	IPO_IN_TARGET	Positioning generator has completed the profile.
Bit 4 ... 7	CAM	Reserved and used for the cam.
Bit 8	SAFE_STANDSTILL	“Safe Stop” “H” on the 7-segment display. Use by safety module CAMC-G-S1.
Bit 9 ... 11	–	Reserved for extensions.

manufacturer_statusword_1		
Bit	Signal	Description
Bit 12	VOUT_PS_EN	Displays that the drive can be switched on (no limitations by safety module).
Bit 13	VOUT_WARN	Corresponds to VOUT_WARN (VOUT41) of the safety module. There is at least one error, whose error response is parameterised as “Warning”.
Bit 14	VOUT_SCV	Corresponds to VOUT_SCV (VOUT 42) of the safety module. At least one safety condition was violated.
Bit 15	VOUT_ERROR	Corresponds to VOUT_ERROR (VOUT 43) of the safety module. An internal fault was ascertained.
Bit 16	VOUT_SAVE_STAT	Corresponds to VOUT_SSR (VOUT 44) of the safety module. The bit is set when a safety function was requested in the safety module and the safe state has been reached.
Bit 17	VOUT_SFR	Corresponds to VOUT_SFR (VOUT 45) of the safety module. The bit is set when at least one safety function is requested in the safety module. The bit remains set until all the requests have been reset.
Bit 18	VOUT_SERVICE	No parameters present, parameter invalid or parameterisation procedure is running (not supported by CAMC-G-S1). Status is assumed when the safety module was replaced with another type.
Bit 19	VOUT_READY	Normal status: VOUT_READY= NOT(VOUT_SFR)
Bit 20 ... 31	–	Reserved.

Tab. 6.10 Bit assignment manufacturer_statusword_1

With the help of the objects manufacturer_status_masks and manufacturer_status_invert, one or more bits of the manufacturer_statuswords are shown in bit 14 (manufacturer_statusbit) of the statusword (6041_h). All bits of the manufacturer_statusword_1 can be inverted through the corresponding bit in manufacturer_status_invert_1. As a result, bits can also be monitored on the “Reset” status. After inverting, the bits are masked, i.e. only if the corresponding bit is set in manufacturer_status_mask_1 is the bit evaluated further. If at least one bit is set after masking, bit 14 of the statusword is also set.

The following illustration shows this through an example:



EXAMPLE

- a) Bit 14 of the statusword is supposed to be set if the drive is referenced.
 Drive referenced is bit 0 of the manufacturer_statusword_1
 manufacturer_status_invert = 0x00000000
 manufacturer_status_mask = 0x00000001 (bit 0)
- b) Bit 14 of the statusword is supposed to be set if the drive has no valid commutation position.
 Valid commutation position is bit 1 of the manufacturer_statusword_1.
 This bit must be inverted so it will be set if the commutation information is invalid:
 manufacturer_status_invert = 0x00000002 (bit 1)
 manufacturer_status_mask = 0x00000002 (bit 1)
- c) Bit 14 of the statusword is supposed to be set if the drive is not ready for release OR the drive is referenced.
 Valid commutation position is bit 2 of the manufacturer_statusword_1.
 Drive referenced is bit 0. Bit 2 must be inverted so that it will be set if the drive is not ready for release:
 manufacturer_status_invert = 0x00000004 (bit 2)
 manufacturer_status_mask = 0x00000005 (bit 2, bit 0)

Object 2005_h: manufacturer_status_masks

This object group establishes which set bits of the manufacturer_statuswords are shown in the statusword. → for this also chapter 6.1.5.

Index	2005_h
Name	manufacturer_status_masks
Object Code	RECORD
No. of Elements	1

Sub-Index	01 _h
Description	manufacturer_status_mask_1
Data Type	UINT32
Access	rw
PDO Mapping	yes
Units	–
Value Range	–
Default Value	0x00000000

Object 200A_h: manufacturer_status_invert

This object group establishes which bits of the manufacturer_statuswords are shown inverted in the statusword. → for this also chapter 6.1.5.

Index	200A_h
Name	manufacturer_status_invert
Object Code	RECORD
No. of Elements	1

Sub-Index	01 _h
Description	manufacturer_status_invert_1
Data Type	UINT32
Access	rw
PDO Mapping	yes
Units	–
Value Range	–
Default Value	0x00000000

Object 2600_h: FSM_VOUT

These objects map the status of the VOUT (0..64).

Index	2600_h
Name	FSM_vout
Object Code	RECORD
No. of Elements	2

Sub-Index	01_h
Description	FSM_vout_0_31
Data Type	UINT32
Access	ro
PDO Mapping	yes
Units	–
Value Range	–
Default Value	–

Bits 0..31 = VOUT0..31 of the safety module

Sub-Index	02_h
Description	FSM_vout_32_63
Data Type	UINT32
Access	ro
PDO Mapping	yes
Units	–
Value Range	–
Default Value	–

Bits 0..31 = VOUT32..63 of the safety module

Object 2602h: FSM_IO

Read the level at the inputs of the safety module

Index	2602_h
Name	FSM_io
Object Code	RECORD
No. of Elements	1

Sub-Index	01_h
Description	FSM_dig_io
Data Type	UINT32
Access	ro
PDO Mapping	yes
Units	–
Value Range	–
Default Value	–

FSM_dig_io

Bit	Signal	Description
Bit 0	LOUT48	Logical status DIN40 A/B
Bit 1	LOUT49	Logical status DIN41 A/B
Bit 2	LOUT50	Logical status DIN42 A/B
Bit 3	LOUT51	Logical status DIN43 A/B
Bit 4	LOUT52	Logical status DIN44
Bit 5	LOUT53	Logical status DIN45; mode selector switch (1 of 3)
Bit 6	LOUT54	Logical status DIN46; mode selector switch (1 of 3)
Bit 7	LOUT55	Logical status DIN47; mode selector switch (1 of 3)
Bit 8	LOUT56	Error acknowledgment via DIN48
Bit 9	LOUT57	Restart via DIN49

FSM_dig_io		
Bit	Signal	Description
Bit 10	LOUT58	Logical status, two-handed control device (pair of 2 x DIN4x)
Bit 11	LOUT59	Feedback, holding brake
Bit 12 ... 15	LOUT60 ... 63	Not assigned
Bit 16	LOUT64	Status of the output DOUT40
Bit 17	LOUT65	Status of the output DOUT41
Bit 18	LOUT66	Status of the output DOUT42
Bit 19	LOUT67	Status of the signal relay
Bit 20	LOUT68	Brake control
Bit 21	LOUT69	Status of the SS1 control signal
Bit 22 ... 31	LOUT70 ...	Not assigned

Tab. 6.11 Bit assignment FSM_dig_io

6.1.6 Description of the additional objects

Objects treated in this chapter

Index	Object	Name	Type	Attr.
605B _h	VAR	shutdown_option_code	INT16	rw
605C _h	VAR	disable_operation_option_code	INT16	rw
605A _h	VAR	quick_stop_option_code	INT16	rw
605E _h	VAR	fault_reaction_option_code	INT16	rw

Object 605B_h: shutdown_option_code

The object shutdown_option_code specifies how the motor controller behaves with status transition 8 (from OPERATION ENABLE to READY TO SWITCH ON). The object shows the implemented behaviour of the motor controller. It cannot be changed.

Index	605B_h
Name	shutdown_option_code
Object Code	VAR
Data Type	INT16

Access	rw
PDO Mapping	no
Units	–
Value Range	0
Default Value	0

Value	Significance
0	The output stage is switched off, and the motor is freely rotatable

Object 605C_h: disable_operation_option_code

The object `disable_operation_option_code` specifies how the motor controller behaves with status transition 5 (from OPERATION ENABLE to SWITCH ON). The object shows the implemented behaviour of the motor controller. It cannot be changed.

Index	605C_h
Name	disable_operation_option_code
Object Code	VAR
Data Type	INT16

Access	rw
PDO Mapping	no
Units	–
Value Range	-1
Default Value	-1

Value	Significance
-1	Braking with quickstop_deceleration

Object 605A_h: quick_stop_option_code

The parameter `quick_stop_option_code` specifies how the motor controller behaves with a Quick Stop. The object shows the implemented behaviour of the motor controller. It cannot be changed.

Index	605A_h
Name	quick_stop_option_code
Object Code	VAR
Data Type	INT16

Access	rw
PDO Mapping	no
Units	–
Value Range	2
Default Value	2

Value	Significance
2	Braking with quickstop_deceleration

Object 605E_h: fault_reaction_option_code

The object `fault_reaction_option_code` specifies how the motor controller behaves with an error (fault). Since the error response in the CMMP series is dependent on the respective error, this object cannot be parametrised and always returns 0. To change the error response of the individual errors → chapter 5.18, error management.

Index	605E_h
Name	fault_reaction_option_code
Object Code	VAR
Data Type	INT16

Access	rw
PDO Mapping	no
Units	–
Value Range	0
Default Value	0

7 Operating modes

7.1 Setting the operating mode

7.1.1 Overview

The motor controller can be placed into a number of operating modes. Only some are specified in detail under CANopen:

- Torque-controlled mode
- Controlled RPM mode
- Homing
- Positioning mode
- Synchronous position specification
- Cyclic synchronous position specification (only for EtherCAT)
- Profile torque mode
- Profile velocity mode
- Homing mode
- Profile position mode
- Interpolated position mode
- cyclic synchronous position mode

7.1.2 Description of the Objects

Objects treated in this chapter

Index	Object	Name	Type	Attr.
6060 _h	VAR	modes_of_operation	INT8	wo
6061 _h	VAR	modes_of_operation_display	INT8	ro

Object 6060_h: modes_of_operation

The object modes_of_operation sets the operating mode of the motor controller.

Index	6060_h
Name	modes_of_operation
Object Code	VAR
Data Type	INT8

Access	rw
PDO mapping	yes
Units	–
Value Range	1, 3, 4, 6, 7
Default Value	–

Value	Significance
1	Profile Position Mode (position controller with positioning mode)
3	Profile Velocity Mode (speed regulator with setpoint value ramp)
4	Profile Torque Mode (torque controller with setpoint value ramp)
6	Homing Mode (reference travel)
7	Interpolated Position Mode
8	Cyclic Synchronous Position Mode (only for EtherCAT)



The current operating mode can only be read in the object `modes_of_operation_display`! Since a change in operating mode can take some time, one must wait until the newly selected mode appears in the object `modes_of_operation_display`.

Object 6061_h: `modes_of_operation_display`

In the object `modes_of_operation_display`, the current operating mode of the motor controller can be read. If an operating mode is set via the object 6060_h, besides the actual operating mode, the setpoint value activations (setpoint value selector) needed for operation of the motor controller under CANopen must also be made. These are:

Selector	Profile Velocity Mode	Profile Torque Mode
A	Speed setpoint value (fieldbus 1)	Torque setpoint value (fieldbus 1)
B	Torque limitation, if necessary	Speed limitation, if necessary
C	Speed setpoint value (synchronous speed)	inactive

In addition, the setpoint value ramp is always switched on. Only if these activations are set in the stated way will one of the CANopen operating modes be returned. If these settings are revised, with the parametrisation software, for example, a respective “user” operating mode is returned to show that the selectors have been changed.

Index	6061_h
Name	<code>modes_of_operation_display</code>
Object Code	VAR
Data Type	INT8

Access	ro
PDO mapping	yes
Units	–
Value Range	see table
Default Value	3

Value	Significance
-1	Invalid operating mode or change in operating mode
-11	User Position Mode
-13	User Velocity Mode
-14	User Torque Mode
1	Profile Position Mode (position controller with positioning mode)
3	Profile Velocity Mode (speed regulator with setpoint value ramp)
4	Profile Torque Mode (torque controller with setpoint value ramp)
6	Homing Mode (reference travel)
7	Interpolated Position Mode
8	Cyclic Synchronous Position Mode (only for EtherCAT)



The operating mode can only be set via the object `modes_of_operation`. Since a change in operating mode can take some time, one must wait until the newly selected mode appears in the object `modes_of_operation_display`. During this time, “Invalid operating mode” (-1) may be displayed briefly.

7.2 Operating mode reference travel (homing mode)

7.2.1 Overview

This chapter describes how the motor controller searches for the initial position (also called point of reference, reference point or zero point). There are various methods to determine this position, whereby either the limit switch can be used at the end of the positioning range or a reference switch (zero-point switch) within the possible travel distance. To achieve a reproducibility that is as large as possible, the zero pulse of the angle encoder used (resolver, incremental encoder, etc.) can be included with some methods.

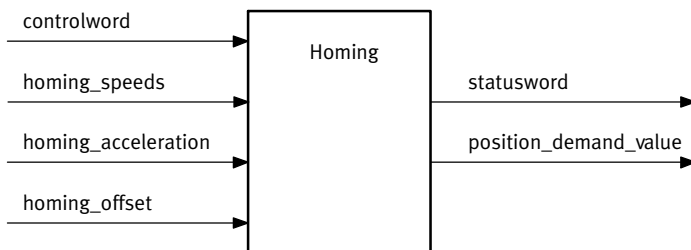


Fig. 7.1 Reference travel

The user can determine the speed, acceleration and type of reference travel. With the object `home_offset`, the zero position of the drive can be displaced to any position desired.

There are two reference travel speeds. The higher search velocity (`speed_during_search_for_switch`) is used to find the limit switch or the reference switch. Then, to exactly determine the position of the switch edge, the system switches to crawl speed (`speed_during_search_for_zero`).

If the drive should not be homed again, but only the position set to a prespecified value, the object `2030h` (`set_position_absolute`) can be used → page 115.



The drive to the zero position under CANopen is normally not a component of the reference travel. If all required variables are known to the motor controller (e.g. because it already knows the position of the zero pulse), no physical movement is performed.

This behaviour can be revised through the object `6510h-F0h` (`compatibility_control`, → chap. 5.2) so that travel to zero is always executed.

7.2.2 Description of the Objects

Objects treated in this chapter

Index	Object	Name	Type	Attr.
607C _h	VAR	home_offset	INT32	rw
6098 _h	VAR	homing_method	INT8	rw
6099 _h	ARRAY	homing_speeds	UINT32	rw
609A _h	VAR	homing_acceleration	UINT32	rw
2045 _h	VAR	homing_timeout	UINT16	rw

Affected objects from other chapters

Index	Object	Name	Type	Chapter
6040 _h	VAR	controlword	UINT16	6.1.3 Control word (controlword)
6041 _h	VAR	statusword	UINT16	6.1.5 Status words (statuswords)

Object 607C_h: home_offset

The object home_offset establishes the shift of the zero position compared to the determined reference position.

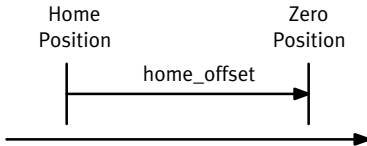


Fig. 7.2 Home Offset

Index	607C_h
Name	home_offset
Object Code	VAR
Data Type	INT32

Access	rw
PDO mapping	yes
Units	position units
Value Range	–
Default Value	0

Object 6098_h: homing_method

A series of different methods are provided for reference travel. Through the object homing_method, the variant needed for the application can be selected. There are four possible homing signals: the negative and positive limit switch, the reference switch and the (periodic) zero pulse of the angle encoder. In addition, the motor controller can perform reference travel to the negative or positive stop completely without any additional signal. If a method for homing is determined through the object homing_method, the following settings are made:

- The reference source (neg./pos. limit switch, reference switch, neg./pos. stop)
- The direction and process of homing
- The type of evaluation of the zero pulse by the angle encoder used

Index	6098_h
Name	homing_method
Object Code	VAR
Data Type	INT8

Access	rw
PDO mapping	yes
Units	
Value Range	-18, -17, -2, -1, 1, 2, 7, 11, 17, 18, 23, 27, 32, 33, 34, 35
Default Value	17

Value	Motion	Objective	Point of reference for zero
-27	Negative	Stop or limit switch	Reference switch
-23	Positive	Stop or limit switch	Reference switch
-18	Positive	Stop	Stop
-17	Negative	Stop	Stop
-2	Positive	Stop	Zero pulse
-1	Negative	Stop	Zero pulse
1	Negative	Limit switch	Zero pulse
2	Positive	Limit switch	Zero pulse
7	Positive	Reference switch	Zero pulse
11	Negative	Reference switch	Zero pulse
17	Negative	Limit switch	Limit switch
18	Positive	Limit switch	Limit switch
23	Positive	Reference switch	Reference switch
27	Negative	Reference switch	Reference switch
33	Negative	Zero pulse	Zero pulse
34	Positive	Zero pulse	Zero pulse
35		No travel	Current actual position

The homing_method can only be set when homing is not active. Otherwise, an error message (→ chapter 3.5) is returned.

The process of the individual methods is described in detail in chapter 7.2.3.

Object 6099_h: homing_speeds

This object determines the speeds used during the reference travel.

Index	6099_h
Name	homing_speeds
Object Code	ARRAY
No. of Elements	2
Data Type	UINT32

Sub-Index	01_h
Description	speed_during_search_for_switch
Access	rw
PDO mapping	yes
Units	speed units
Value Range	–
Default Value	100 min ⁻¹

Sub-Index	02_h
Description	speed_during_search_for_zero
Access	rw
PDO mapping	yes
Units	speed units
Value Range	–
Default Value	10 min ⁻¹



If bit 6 is set in the object compatibility_control, (→ chap. 5.2), after homing travel to zero is performed.

If this bit is set and the object speed_during_search_for_switch is written, both the speed for the switch search and the speed for the travel to zero are written.

Object 609A_h: homing_acceleration

The object homing_acceleration determines the acceleration that is used during homing for all acceleration and braking processes.

Index	609A_h
Name	homing_acceleration
Object Code	VAR
Data Type	UINT32

Access	rw
PDO mapping	yes
Units	acceleration units
Value Range	–
Default Value	1000 min ⁻¹ /s

Object 2045_h: homing_timeout

Homing can be monitored for maximum execution time. In addition, the maximum execution time can be specified with the object homing_timeout. If this time is exceeded without homing having been ended, error 11-3.

Index	2045_h
Name	homing_timeout
Object Code	VAR
Data Type	UINT16

Access	rw
PDO mapping	no
Units	ms
Value Range	0 (off), 1 ... 65535
Default Value	60000

7.2.3 Reference Travel Processes

The various reference travel methods are depicted in the following illustrations.

Homing methods		
hex	dec	Description
01h	1	<p>Negative limit switch with index pulse ¹⁾</p> <ol style="list-style-type: none"> 1. If negative limit switch inactive: Run at search velocity in negative direction to the negative limit switch. 2. Travel at crawling velocity in positive direction until the limit switch becomes inactive, then continue to the first index pulse. This position is taken as the homing point. 3. If this is parameterised: travel at positioning velocity to the axis zero point.
02h	2	<p>Positive limit switch with index pulse ¹⁾</p> <ol style="list-style-type: none"> 1. If positive limit switch inactive: Run at search velocity in positive direction to the positive limit switch. 2. Travel at crawling velocity in negative direction until the limit switch becomes inactive, then continue to the first index pulse. This position is taken as the homing point. 3. If this is parameterised: travel at positioning velocity to the axis zero point.
07h	7	<p>Reference switch in positive direction with index pulse ¹⁾</p> <ol style="list-style-type: none"> 1. If reference switch inactive: Travel at search velocity in positive direction to the reference switch. If the stop or limit switch is approached: Travel at search velocity in positive direction to the reference switch. 2. Travel at crawling velocity in negative direction until the reference switch becomes inactive, then continue to the first index pulse. This position is taken as the homing point. 3. If this is parameterised: travel at positioning velocity to the axis zero point.

1) Only possible for motors with encoder/resolver with index pulse.
 2) Limit switches are ignored during travel to the stop.
 3) Since the axis is not to remain at the stop, the travel to the axis zero point must be parameterised and the axis zero point offset must be $\neq 0$.

Homing methods			
hex	dec	Description	
0B	11	<p>Reference switch in negative direction with index pulse ¹⁾</p> <ol style="list-style-type: none"> If reference switch inactive: Travel at search velocity in negative direction to the reference switch. If the stop or limit switch is approached: Travel at search velocity in positive direction to the reference switch. Travel at crawling velocity in positive direction until the reference switch becomes inactive, then continue to the first index pulse. This position is taken as the homing point. If this is parameterised: travel at positioning velocity to the axis zero point. 	
11h	17	<p>Negative limit switch</p> <ol style="list-style-type: none"> If negative limit switch inactive: Run at search velocity in negative direction to the negative limit switch. Travel at crawling velocity in positive direction until the limit switch becomes inactive. This position is taken as the homing point. If this is parameterised: travel at positioning velocity to the axis zero point. 	
12h	18	<p>Positive limit switch</p> <ol style="list-style-type: none"> If positive limit switch inactive: Run at search velocity in positive direction to the positive limit switch. Travel at crawling velocity in negative direction until the limit switch becomes inactive. This position is taken as the homing point. If this is parameterised: travel at positioning velocity to the axis zero point. 	

- 1) Only possible for motors with encoder/resolver with index pulse.
- 2) Limit switches are ignored during travel to the stop.
- 3) Since the axis is not to remain at the stop, the travel to the axis zero point must be parameterised and the axis zero point offset must be ≠ 0.

Homing methods			
hex	dec	Description	
17h	23	<p>Reference switch in positive direction</p> <ol style="list-style-type: none"> If reference switch inactive: Travel at search velocity in positive direction to the reference switch. If the stop or limit switch is approached: Travel at search velocity in positive direction to the reference switch. Travel at crawling velocity in negative direction until the reference switch becomes inactive. This position is taken as the homing point. If this is parameterised: travel at positioning velocity to the axis zero point. 	
18h	27	<p>Reference switch in negative direction</p> <ol style="list-style-type: none"> If reference switch inactive: Travel at search velocity in negative direction to the reference switch. If the stop or limit switch is approached: Travel at search velocity in positive direction to the reference switch. Travel at crawling velocity in positive direction until the reference switch becomes inactive. This position is taken as the homing point. If this is parameterised: travel at positioning velocity to the axis zero point. 	
21h	33	<p>Index pulse in a negative direction ¹⁾</p> <ol style="list-style-type: none"> Travel at crawling velocity in negative direction until the index pulse. This position is taken as the homing point. If this is parameterised: travel at positioning velocity to the axis zero point. 	
22h	34	<p>Index pulse in a positive direction ¹⁾</p> <ol style="list-style-type: none"> Travel at crawling velocity in positive direction up to the index pulse. This position is taken as the homing point. If this is parameterised: travel at positioning velocity to the axis zero point. 	

- 1) Only possible for motors with encoder/resolver with index pulse.
- 2) Limit switches are ignored during travel to the stop.
- 3) Since the axis is not to remain at the stop, the travel to the axis zero point must be parameterised and the axis zero point offset must be $\neq 0$.

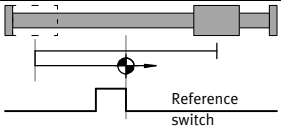
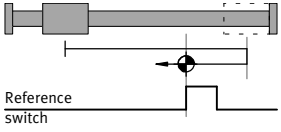
Homing methods			
hex	dec	Description	
23h	35	<p>Current position</p> <ol style="list-style-type: none"> 1. The current position is taken as the reference position. 2. If this is parameterised: travel at positioning velocity to the axis zero point. <p>If the drive should not be homed again, but only the position set to a prespecified value, the object 2030h (set_position_absolute) can be used. → page 115.</p>	
FFh	-1	<p>Negative stop with index pulse 1) 2)</p> <ol style="list-style-type: none"> 1. Travel at search velocity in negative direction to the stop. 2. Travel at crawling velocity in positive direction until the next index pulse. This position is taken as the homing point. 3. If this is parameterised: travel at positioning velocity to the axis zero point. 	
FEh	-2	<p>Positive stop with index pulse 1) 2)</p> <ol style="list-style-type: none"> 1. Travel at search velocity in positive direction to the stop. 2. Travel at crawling velocity in negative direction until the next index pulse. This position is taken as the homing point. 3. If this is parameterised: travel at positioning velocity to the axis zero point. 	
EFh	-17	<p>Negative stop 1) 2) 3)</p> <ol style="list-style-type: none"> 1. Travel at search velocity in negative direction to the stop. This position is taken as the homing point. 2. If this is parameterised: travel at positioning velocity to the axis zero point. 	
EEh	-18	<p>Positive stop 1) 2) 3)</p> <ol style="list-style-type: none"> 1. Travel at search velocity in positive direction to the stop. This position is taken as the homing point. 2. If this is parameterised: travel at positioning velocity to the axis zero point. 	

1) Only possible for motors with encoder/resolver with index pulse.

2) Limit switches are ignored during travel to the stop.

3) Since the axis is not to remain at the stop, the travel to the axis zero point must be parameterised and the axis zero point offset must be $\neq 0$.

Homing methods		
hex	dec	Description
E9h	-23	<p>Reference switch in positive direction with travel to stop or limit switch.</p> <ol style="list-style-type: none"> 1. Run at search velocity in positive direction to stop or limit switch. 2. Travel at search velocity in negative direction to the reference switch. 3. Travel at crawling velocity in negative direction until the reference switch becomes inactive. This position is taken as the homing point. 4. If this is parameterised: travel at positioning velocity to the axis zero point.
E5h	-27	<p>Reference switch in negative direction with travel to stop or limit switch</p> <ol style="list-style-type: none"> 1. Run at search velocity in negative direction to stop or limit switch. 2. Travel at search velocity in positive direction to the reference switch. 3. Run at crawling velocity in positive direction until reference switch becomes inactive. This position is taken as the homing point. 4. If this is parameterised: travel at positioning velocity to the axis zero point.



- 1) Only possible for motors with encoder/resolver with index pulse.
- 2) Limit switches are ignored during travel to the stop.
- 3) Since the axis is not to remain at the stop, the travel to the axis zero point must be parameterised and the axis zero point offset must be $\neq 0$.

Tab. 7.1 Overview of homing methods

7.2.4 Control of Reference Travel

Homing is controlled and monitored through the controlword / statusword. The start is made by setting bit 4 in the controlword. Successful completion of the travel is shown by a set bit 12 in the object statusword. A set bit 13 in the object statusword shows that an error has occurred during homing. The cause of the error can be determined via the objects error_register and pre_defined_error_field.

Bit 4	Significance
1	Reference travel is not active
0 → 1	Start Homing
1	Reference travel is active
1 → 0	Reference travel interrupted

Tab. 7.2 Description of the bits in the controlword

Bit 13	Bit 12	Significance
0	0	Reference travel is not completed yet
0	1	Reference travel performed successfully
1	0	Reference travel not performed successfully
1	1	Prohibited status

Tab. 7.3 Description of the bits in the status word

7.3 Positioning Operating Mode (Profile Position Mode)

7.3.1 Overview

The structure of this operating mode is evident in Fig. 7.3:

The target position (`target_position`) is passed on to the curve generator. This generates a setpoint position value (`position_demand_value`) for the position controller, which is described in the Position Controller chapter (Position Control Function, chapter 6). These two function blocks can be set independently of each other.

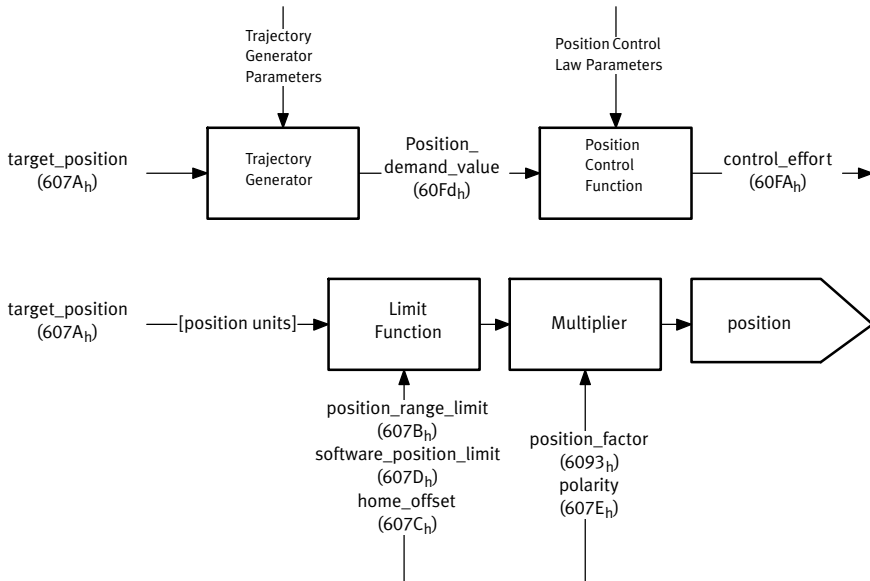


Fig. 7.3 Curve generator and position controller

All input variables of the curve generator are converted with the variables of the factor group (→ chap. 5.3) into the internal units of the controller. The internal variables are marked here with an asterisk and are normally not needed by the user.

7.3.2 Description of the objects

Objects treated in this chapter

Index	Object	Name	Type	Attr.
607A _h	VAR	target_position	INT32	rw
6081 _h	VAR	profile_velocity	UINT32	rw
6082 _h	VAR	end_velocity	UINT32	rw
6083 _h	VAR	profile_acceleration	UINT32	rw
6084 _h	VAR	profile_deceleration	UINT32	rw
6085 _h	VAR	quick_stop_deceleration	UINT32	rw
6086 _h	VAR	motion_profile_type	INT16	rw

Affected objects from other chapters

Index	Object	Name	Type	Chapter
6040 _h	VAR	controlword	INT16	6 Device control
6041 _h	VAR	statusword	UINT16	6 Device control
605A _h	VAR	quick_stop_option_code	INT16	6 Device control
607E _h	VAR	polarity	UINT8	5.3 Conversion factors
6093 _h	ARRAY	position_factor	UINT32	5.3 Conversion factors
6094 _h	ARRAY	velocity_encoder_factor	UINT32	5.3 Conversion factors
6097 _h	ARRAY	acceleration_factor	UINT32	5.3 Conversion factors

Object 607A_h: target_position

The object target_position (target position) determines which position of the motor controller should be traveled to. The current setting for speed, acceleration, brake delay and type of travel profile (motion_profile_type) etc. must be considered thereby. The target position (target_position) is interpreted either as an absolute or relative specification (controlword, bit 6).

Index	607A_h
Name	target_position
Object Code	VAR
Data Type	INT32

Access	rw
PDO mapping	yes
Units	position units
Value Range	–
Default Value	0

Object 6081_h: profile_velocity

The object profile_velocity specifies the speed that is normally reached at the end of the acceleration ramp during positioning. The object profile_velocity is specified in speed units.

Index	6081_h
Name	profile_velocity
Object Code	VAR
Data Type	UINT32

Access	rw
PDO mapping	yes
Units	speed units
Value Range	–
Default Value	1000

Object 6082_h: end_velocity

The object end_velocity (end speed) defines the speed the drive must have when it reaches the target position (target_position). Normally, this object must be set to zero so that the motor controller stops when it reaches the target position (target_position). For continuous positioning, a speed different from zero can be specified. The object end_velocity is specified in the same units as the object profile_velocity.

Index	6082_h
Name	end_velocity
Object Code	VAR
Data Type	UINT32

Access	rw
PDO mapping	yes
Units	speed units
Value Range	–
Default Value	0

Object 6083_h: profile_acceleration

The object profile_acceleration specifies the acceleration with which the motor accelerates to the nominal value. It is specified in user-defined acceleration units (→ chapter 5.3 Conversion factors (factor group)).

Index	6083_h
Name	profile_acceleration
Object Code	VAR
Data Type	UINT32

Access	rw
PDO mapping	yes
Units	acceleration units
Value Range	–
Default Value	10000 min ⁻¹ /s

Object 6084_h: profile_deceleration

The object profile_deceleration specifies the deceleration with which the motor is braked. It is specified in user-defined acceleration units (→ chapter 5.3 Conversion factors (factor group)).

Index	6084_h
Name	profile_deceleration
Object Code	VAR
Data Type	UINT32

Access	rw
PDO mapping	yes
Units	acceleration units
Value Range	–
Default Value	10000 min ⁻¹ /s

Object 6085_h: quick_stop_deceleration

The object quick_stop_deceleration specifies with which brake delay the motor stops when a quick stop is carried out (→ chapter 6). The object quick_stop_deceleration is specified in the same unit as the object profile_deceleration.

Index	6085_h
Name	quick_stop_deceleration
Object Code	VAR
Data Type	UINT32

Access	rw
PDO mapping	yes
Units	acceleration units
Value Range	–
Default Value	14100 min ⁻¹ /s

Object 6086_h: motion_profile_type

The object motion_profile_type is used to select the type of positioning profile.

Index	6086_h
Name	motion_profile_type
Object Code	VAR
Data Type	INT16

Access	rw
PDO mapping	yes
Units	–
Value Range	0, 2
Default Value	0

Value	Curve form
0	Linear ramp
2	Jerk-free ramp

7.3.3 Description of function

There are two possibilities for passing on a target position to the motor controller:

Simple positioning task

If the motor controller has reached a target position, it signals this to the host with the bit target_reached (bit 10 in the object statusword). In this operating mode, the motor controller stops when it has reached the goal.

Sequence of positioning tasks

After the motor controller has reached a target, it immediately begins travelling to the next target. This transition can occur smoothly, without the motor controller meanwhile coming to a standstill.

These two methods are controlled through the bits new_set_point and change_set_immediately in the object controlword and set_point_acknowledge in the object statusword. These bits are in a question-answer relationship to each other. This makes it possible to prepare a positioning task while another is still running.

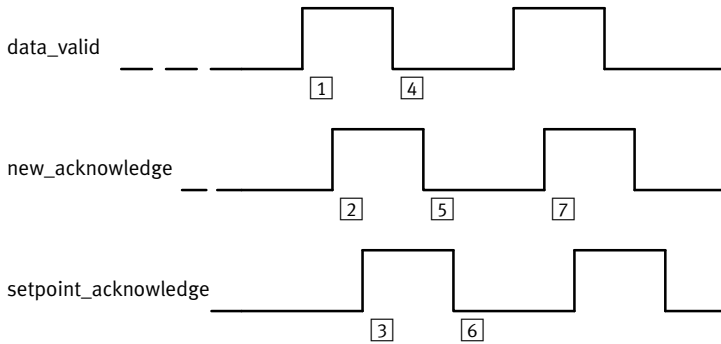


Fig. 7.4 Positioning job transmission from a host

In Fig. 7.4, you can see how the host and the motor controller communicate with each other via the CAN bus:

First, the positioning data (target position, travel speed, end speed and acceleration) are transmitted to the motor controller. When the positioning data set has been completely written [1], the host can start positioning by setting the bit new_set_point in the controlword to “1” [2]. After the motor controller recognises the new data and takes it over into its buffer, it reports this to the host by setting the bit set_point_acknowledge in the statusword [3].

Then the host can start to write a new positioning data set into the motor controller [4] and delete again the bit new_set_point [5]. Only when the motor controller can accept a new positioning job [6] does it signal this through a “0” in the set_point_acknowledge bit. Before this, no new positioning may be started by the host [7].

In Fig. 7.5, a new positioning task is only started after the previous one has been completely finished. To determine this, the host evaluates the bit target_reached in the object statusword.

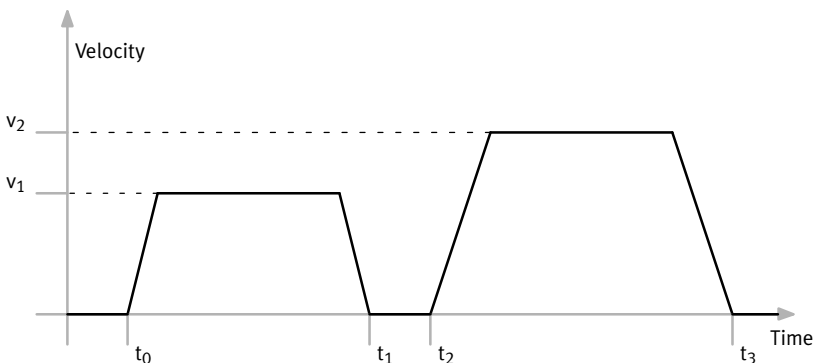


Fig. 7.5 Simple positioning task

In Fig. 7.6, a new positioning task is already started while the previous one is still in process. The host already passes the subsequent target on to the motor controller when the motor controller signals with deletion of the bit `set_point_acknowledge` that it has read the buffer and started the related positioning. In this way, positioning tasks follow each other seamlessly. For this operating mode, the object `end_velocity` should be written over with the same value as the object `profile_velocity` so that the motor controller does not briefly brake to zero each time between the individual positioning tasks.

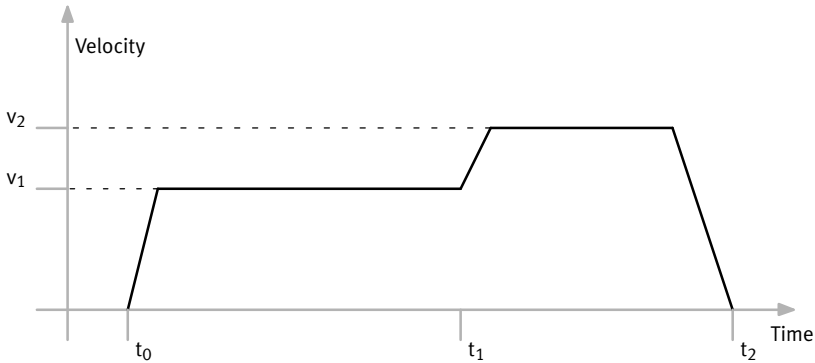


Fig. 7.6 Continuous sequence of positioning tasks

If besides the bit `new_set_point` the bit `change_set_immediately` is also set to “1” in the controlword, the host instructs the motor controller to start the new positioning task immediately. In this case, a positioning task already in process is interrupted.

7.4 Synchronous position specification (interpolated position mode)

7.4.1 Overview

The interpolated position mode (IP) permits specification of setpoint position values in a multi-axis application of the motor controller. For this, synchronisation telegrams (SYNC) and position setpoints are specified by a higher-order controller in a fixed time slot pattern (synchronisation interval). Since the interval is normally greater than one position controller cycle, the motor controller independently interpolates the data values between two specified position values, as shown in the following diagram.

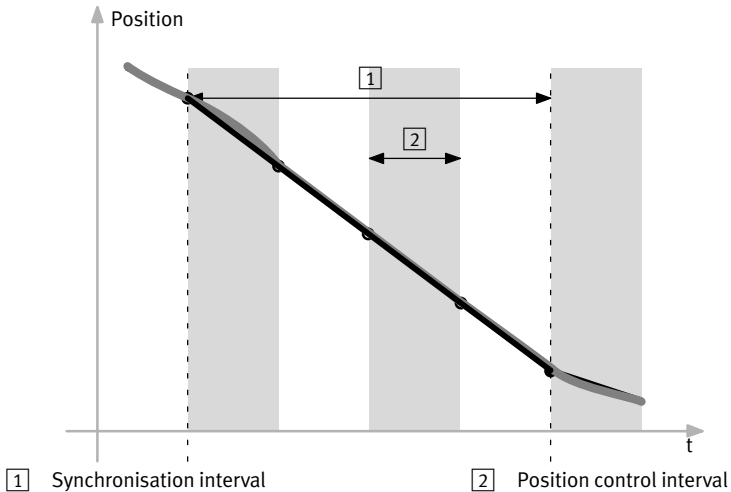


Fig. 7.7 Positioning task linear interpolation between two data values

In the following, the objects needed for the interpolated position mode are described first. A subsequent functional description comprehensively covers the activation and sequencing of parameter setting.

7.4.2 Description of the Objects

Objects treated in this chapter

Index	Object	Name	Type	Attr.
60C0 _h	VAR	interpolation_submode_select	INT16	rw
60C1 _h	REC	interpolation_data_record		rw
60C2 _h	REC	interpolation_time_period		rw
60C3 _h	ARRAY	interpolation_sync_definition	UINT8	rw
60C4 _h	REC	interpolation_data_configuration		rw

Affected objects from other chapters

Index	Object	Name	Type	Chapter
6040 _h	VAR	controlword	INT16	6 Device control
6041 _h	VAR	statusword	UINT16	6 Device control
6093 _h	ARRAY	position_factor	UINT32	5.3 Conversion factors
6094 _h	ARRAY	velocity_encoder_factor	UINT32	5.3 Conversion factors
6097 _h	ARRAY	acceleration_factor	UINT32	5.3 Conversion factors

Object 60C0_h: interpolation_submode_select

The type of interpolation is established via the object interpolation_submode_select. Currently, only the manufacturer-specific variant “Linear Interpolation without Buffer” is available.

Index	60C0_h
Name	interpolation_submode_select
Object Code	VAR
Data Type	INT16

Access	rw
PDO mapping	yes
Units	–
Value Range	-2
Default Value	-2

Value	Interpolation type
-2	Linear Interpolation without Buffer

Object 60C1_h: interpolation_data_record

The object record interpolation_data_record represents the actual data record. It consists of an entry for the position value (ip_data_position) and a control word (ip_data_controlword), which specifies whether the position value should be interpreted absolutely or relatively. Specification of the control word is optional. If it is not specified, the position value is interpreted absolutely. If the control word should also be specified, for reasons of data consistency, first subindex 2 (ip_data_controlword) and then sub-index 1 (ip_data_position) are written, since data transfer is triggered internally with write access to ip_data_position.

Index	60C1_h
Name	interpolation_data_record
Object Code	RECORD
No. of Elements	2

Sub-Index	01_h
Description	ip_data_position
Data Type	INT32
Access	rw
PDO mapping	yes
Units	position units
Value Range	–
Default Value	–

Sub-Index	02_h
Description	ip_data_controlword
Data Type	UINT8
Access	rw
PDO mapping	yes
Units	–
Value Range	0, 1
Default Value	0

Value	ip_data_controlword
0	Absolute position
1	Relative distance



The internal data transfer takes place with write access to sub-index 1. If sub-index 2 should also be used, it must be written before sub-index 1.

Object 60C2_h: interpolation_time_period

The synchronisation interval can be set via the object record interpolation_time_period. Via ip_time_index, the unit (ms or 1/10 ms) of the interval is established, which is parametrised via ip_time_units. To achieve synchronisation, the complete controller cascade (current, speed and position controller) is synchronised up to the external pulse. A change in the synchronisation interval is therefore effective only after a reset. Therefore, if the interpolation interval is to be revised via the CAN bus, the parameter set must be saved (➔ chapter 5.1) and a reset performed (➔ chapter 6), so that the new synchronisation interval becomes effective. The synchronisation interval must be maintained exactly.

Index	60C2_h
Name	interpolation_time_period
Object Code	RECORD
No. of Elements	2

Sub-Index	01_h
Description	ip_time_units
Data Type	UINT8
Access	rw
PDO mapping	yes
Units	according to ip_time_index
Value Range	ip_time_index = -3: 1, 2 ... 9, 10 ip_time_index = -4: 10, 20 ... 90, 100
Default Value	--

Sub-Index	02_h
Description	ip_time_index
Data Type	INT8
Access	rw
PDO mapping	yes
Units	–
Value Range	-3, -4
Default Value	-3

Value	ip_time_units is specified in
-3	10 ⁻³ seconds (ms)
-4	10 ⁻⁴ seconds (0.1 ms)



A change in the synchronisation interval is effective only after a reset. If the interpolation interval is to be revised via the CAN bus, the parameter set must be saved and a reset performed.

Object 60C3_h: interpolation_sync_definition

The object interpolation_sync_definition sets the type (synchronize_on_group) and number of (ip_sync_every_n_event) of synchronisation telegrams per synchronisation interval. For the CMMP series, only the standard SYNC telegram and 1 SYNC per interval can be set.

Index	60C3_h
Name	interpolation_sync_definition
Object Code	ARRAY
No. of Elements	2
Data Type	UINT8

Sub-Index	01_h
Description	synchronize_on_group
Access	rw
PDO mapping	yes
Units	–
Value Range	0
Default Value	0

Value	Significance
0	Use standard SYNC telegram

Sub-Index	02_h
Description	ip_sync_every_n_event
Access	rw
PDO mapping	yes
Units	–
Value Range	1
Default Value	1

Object 60C4_h: interpolation_data_configuration

Through the object record interpolation_data_configuration, the type (buffer_organisation) and size (max_buffer_size, actual_buffer_size) of a possibly available buffer as well as access to this (buffer_position, buffer_clear) can be configured. The size of a buffer element can be read out via the object size_of_data_record. Although no buffer is available for the interpolation type “Linear interpolation without buffer”, access via the object buffer_clear must still be enabled in this case as well.

Index	60C4_h
Name	interpolation_data_configuration
Object Code	RECORD
No. of Elements	6

Sub-Index	01_h
Description	max_buffer_size
Data Type	UINT32
Access	ro
PDO mapping	no
Units	–
Value Range	0
Default Value	0

Sub-Index	02_h
Description	actual_size
Data Type	UINT32
Access	rw
PDO mapping	yes
Units	–
Value Range	0... max_buffer_size
Default Value	0

Sub-Index	03_h
Description	buffer_organisation
Data Type	UINT8
Access	rw
PDO mapping	yes
Units	–
Value Range	0
Default Value	0

Value	Significance
0	FIFO

Sub-Index	04_h
Description	buffer_position
Data Type	UINT16
Access	rw
PDO mapping	yes
Units	–
Value Range	0
Default Value	0

Sub-Index	05_h
Description	size_of_data_record
Data Type	UINT8
Access	wo
PDO mapping	yes
Units	–
Value Range	2
Default Value	2

Sub-Index	06_h
Description	buffer_clear
Data Type	UINT8
Access	wo
PDO mapping	yes
Units	–
Value Range	0, 1
Default Value	0

Value	Significance
0	Delete buffer/access to 60C1 _h not permitted
1	Access to 60C1 _h enabled

7.4.3 Description of function

Preparatory parameter setting

Before the motor controller can be switched into the operating mode interpolated position mode, various settings must be made: These include setting of the interpolation interval (interpolation_time_period), that is, the time between two SYNC telegrams, the interpolation type (interpolation_submode_select) and the type of synchronisation (interpolation_sync_definition). In addition, access to the position buffer must be enabled via the object buffer_clear.

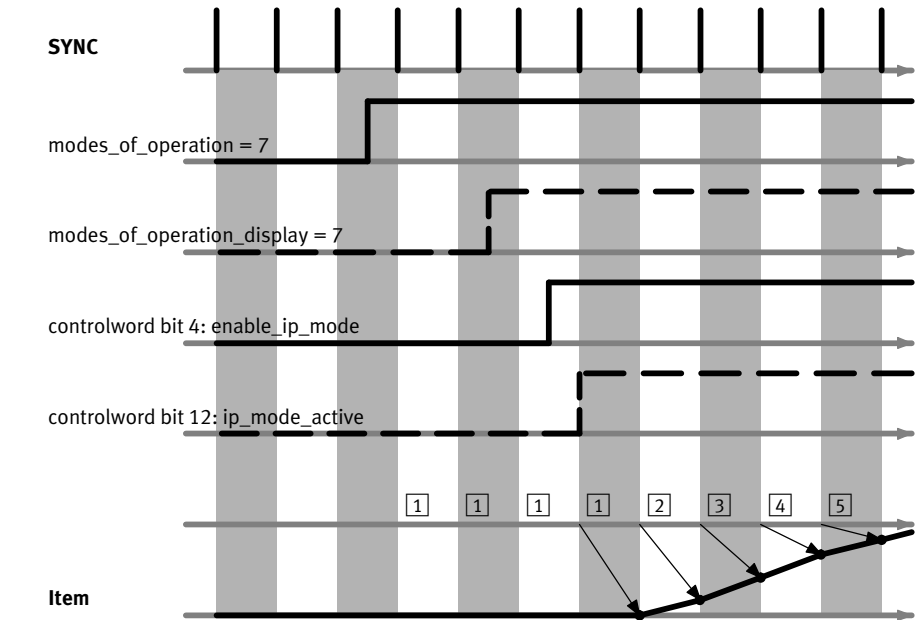
EXAMPLE				
Exercise		CAN object/COB		
Type of interpolation	-2	60C0 _h , interpolation_submode_select	=	-2
Time unit	0.1 ms	60C2 _{h_02} _h , interpolation_time_index	=	-4
Time interval	4 ms	60C2 _{h_01} _h , interpolation_time_units	=	40
Save parameters		1010 _{h_01} _h , save_all_parameters		
Perform reset		NMT reset node		
Waiting for bootup		Bootup message		
Buffer activation	1	60C4 _{h_06} _h , buffer_clear	=	1
Generate SYNC		SYNC (matrix 4 ms)		

Activation of the interpolated position mode and synchronisation

The IP is activated via the object modes_of_operation (6060_h). Starting with this time, the motor controller tries to synchronise itself to the external time grid, which is specified through the SYNC telegrams. If the motor controller was successfully able to synchronise itself, it reports the operating mode interpolated position mode in the object modes_of_operation_display (6061_h). During synchronisation, the motor controller reports back invalid mode of operation (-1). If the SYNC-telegrams are not sent in the right slot pattern after completed synchronisation, the motor controller switches back into the invalid mode of operation.

If the mode of operation is taken up, transfer of position data to the drive can begin. As is logical, the higher-order controller first reads the current actual position out of the controller and writes it cyclically into the motor controller as a new setpoint value (interpolation_data_record). Acceptance of data by the motor controller is activated via handshake bits of the controlword and statusword. By setting the bit enable_ip_mode in the controlword, the host shows that evaluation of the position data should begin. The data records are evaluated only when the motor controller acknowledges this via the status bit ip_mode_selected in the statusword.

In detail, therefore, the following assignment and procedure result:



[1]...[5]: Position specifications

Fig. 7.8 Synchronisation and data release

Event	CAN Object		
Generate SYNC message			
Request of the ip operating mode:	6060 _h , modes_of_operation	=	07
Wait until operating mode is taken	6061 _h , modes_of_operation_display	=	07
Reading out the current actual position	6064 _h , position_actual_value		
Writing back as current setpoint position	60C1 _{h_01h} , ip_data_position		
Start of interpolation	6040 _h , controlword, enable_ip_mode		
Acknowledgement by motor controller	6041 _h , statusword, ip_mode_active		
Changing the current setpoint position in accordance with trajectory	60C1 _{h_01h} , ip_data_position		

After the synchronous travel process is ended, deletion of the bit `enable_ip_mode` prevents further evaluation of position values.

Then the system can switch into another operating mode, if necessary.

Interruptions of interpolation in case of error

If an ongoing interpolation type (`ip_mode_active` set) is interrupted by occurrence of a controller error, the drive first behaves as specified for the respective error (e.g. removal of the controller enable and change into the status `SWITCH_ON_DISABLED`).

The interpolation can only be continued through a new synchronisation, since the motor controller must be brought back into the status `OPERATION_ENABLE`, through which the bit `ip_mode_active` is deleted.

7.5 Speed Adjustment Operating Mode (Profile Velocity Mode)

7.5.1 Overview

The speed-regulated operation (Profile Velocity Mode) contains the following subfunctions:

- Setpoint value generation through the ramp generator
- Speed recording through differentiation via the angle encoder
- Speed regulation with appropriate input and output signals
- Limitation of the torque setpoint value (`torque_demand_value`)
- Monitoring of the actual speed (`velocity_actual_value`) with the window function/threshold

The significance of the following parameters is described in the Positioning chapter (Profile Position Mode): `profile_acceleration`, `profile_deceleration`, `quick_stop`.

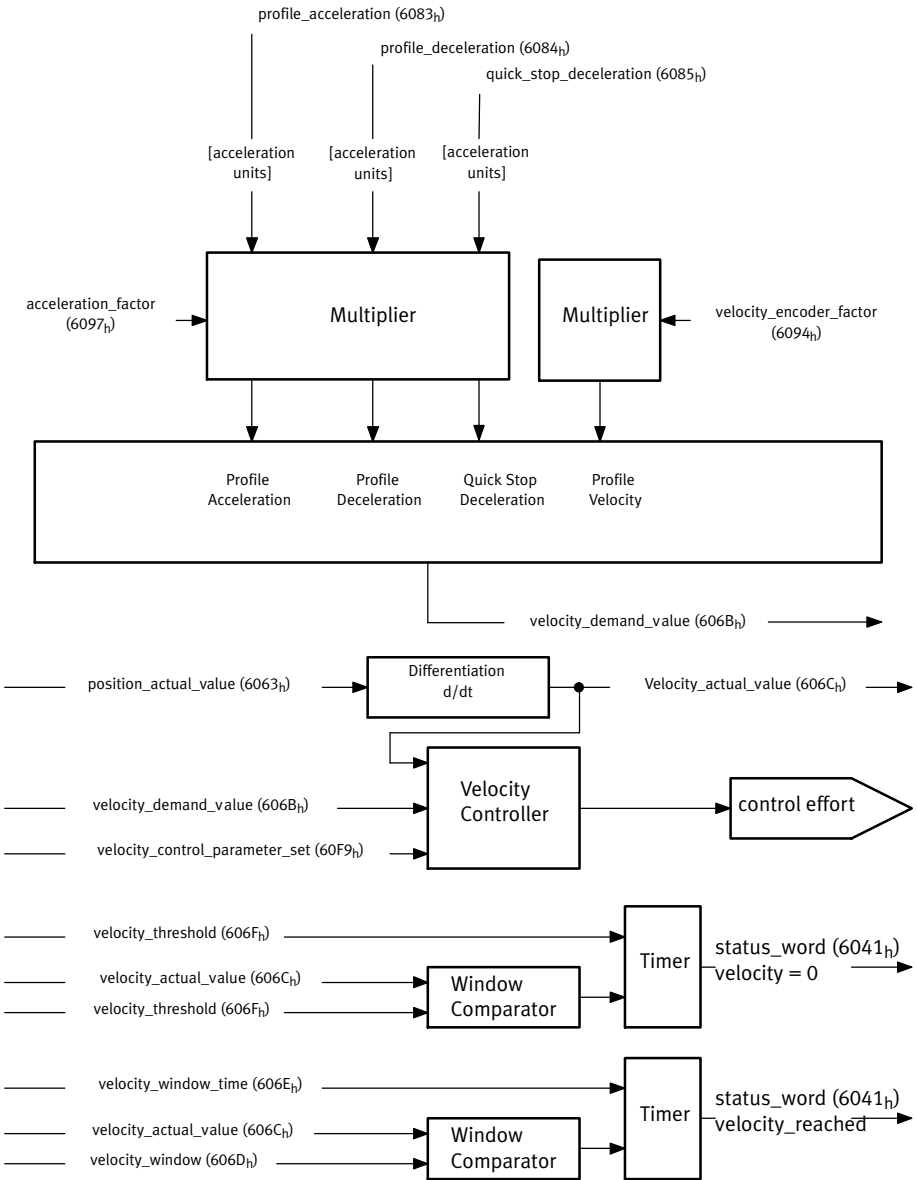


Fig. 7.9 Structure of the speed-regulated operation (profile velocity mode)

7.5.2 Description of the Objects

Objects treated in this chapter

Index	Object	Name	Type	Attr.
6069 _h	VAR	velocity_sensor_actual_value	INT32	ro
606A _h	VAR	sensor_selection_code	INT16	rw
606B _h	VAR	velocity_demand_value	INT32	ro
202E _h	VAR	velocity_demand_sync_value	INT32	ro
606C _h	VAR	velocity_actual_value	INT32	ro
606D _h	VAR	velocity_window	UINT16	rw
606E _h	VAR	velocity_window_time	UINT16	rw
606F _h	VAR	velocity_threshold	UINT16	rw
6080 _h	VAR	max_motor_speed	UINT32	rw
60FF _h	VAR	target_velocity	INT32	rw

Affected objects from other chapters

Index	Object	Name	Type	Chapter
6040 _h	VAR	controlword	INT16	6 Device control
6041 _h	VAR	statusword	UINT16	6 Device control
6063 _h	VAR	position_actual_value*	INT32	5.7 Position controller
6071 _h	VAR	target_torque	INT16	7.7 Torque controller
6072 _h	VAR	max_torque_value	UINT16	7.7 Torque controller
607E _h	VAR	polarity	UINT8	5.3 Conversion factors
6083 _h	VAR	profile_acceleration	UINT32	7.3 Positioning
6084 _h	VAR	profile_deceleration	UINT32	7.3 Positioning
6085 _h	VAR	quick_stop_deceleration	UINT32	7.3 Positioning
6086 _h	VAR	motion_profile_type	INT16	7.3 Positioning
6094 _h	ARRAY	velocity_encoder_factor	UINT32	5.3 Conversion factors

Object 6069_h: velocity_sensor_actual_value

With the object velocity_sensor_actual_value, the value of a possible speed encoder can be read out in internal units. A separate tachometer cannot be connected in the CMMP family. Therefore, to determine the actual speed value, the object 606C_h should be used.

Index	6069_h
Name	velocity_sensor_actual_value
Object Code	VAR
Data Type	INT32

Access	ro
PDO mapping	yes
Units	R/4096 min
Value Range	–
Default Value	–

Object 606A_h: sensor_selection_code

The speed sensor can be selected with this object. Currently, no separate speed sensor is planned. Therefore, only the standard angle encoder can be selected.

Index	606A_h
Name	sensor_selection_code
Object Code	VAR
Data Type	INT16

Access	rw
PDO mapping	yes
Units	–
Value Range	0
Default Value	0

Object 606B_h: velocity_demand_value

The current speed setpoint value of the speed regulator can be read with this object. It is acted upon by the nominal value of the ramp and curve generators. If the position controller is activated, its correction speed is also added.

Index	606B_h
Name	velocity_demand_value
Object Code	VAR
Data Type	INT32

Access	ro
PDO mapping	yes
Units	speed units
Value Range	–
Default Value	–

Object 202E_h: velocity_demand_sync_value

The target speed of the synchronisation encoder can be read out via this object. This is defined through the object 2022_h synchronization_encoder_select (chap. 5.11). This object is specified in user-defined increments.

Index	202E_h
Name	velocity_demand_sync_value
Object Code	VAR
Data Type	INT32

Access	ro
PDO mapping	no
Units	velocity units
Value Range	–
Default Value	–

Object 606C_h: velocity_actual_value

The actual speed value can be read via the object velocity_actual_value .

Index	606C_h
Name	velocity_actual_value
Object Code	VAR
Data Type	INT32

Access	ro
PDO mapping	yes
Units	speed units
Value Range	–
Default Value	–

Object 2074_h: velocity_actual_value_filtered

A filtered actual speed value can be read via the object velocity_actual_value_filtered, but it should only be used for display purposes.

In contrast to velocity_actual_value, velocity_actual_value_filtered is not used for control, but for spinning protection of the controller. The filter time constant can be set via the object 2073_h (velocity_display_filter_time). → Object 2073_h: velocity_display_filter_time

Index	2074_h
Name	velocity_actual_value_filtered
Object Code	VAR
Data Type	INT32

Access	ro
PDO mapping	yes
Units	speed units
Value Range	–
Default Value	–

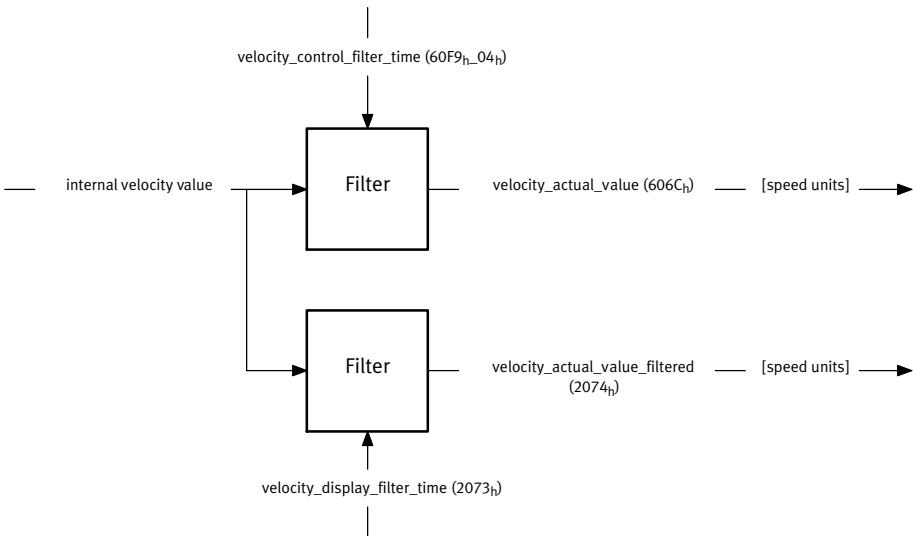


Fig. 7.10 Determination of velocity_actual_value and velocity_actual_value_filtered

Object 606D_h: velocity_window

The object velocity_window is used to set the window comparator. It compares the actual speed value with the prespecified final speed (object 60FF_h: target_velocity). If the difference is less than specified here for a certain period, the bit 10 target_reached is set in the object statusword. → also: object 606E_h (velocity_window_time).

Index	606D_h
Name	velocity_window
Object Code	VAR
Data Type	UINT16

Access	rw
PDO mapping	yes
Units	speed units
Value Range	0 ... 65536 min ⁻¹
Default Value	4 min ⁻¹

Object 606E_h: velocity_window_time

The object velocity_window_time is used to set the window comparator along with the object 606D_h: velocity_window. The speed must lie within the velocity_window for the time specified here so that the bit 10 target_reached is set in the object statusword.

Index	606E_h
Name	velocity_window_time
Object Code	VAR
Data Type	UINT16

Access	rw
PDO mapping	yes
Units	ms
Value Range	0 ... 4999
Default Value	0

Object 606F_h: velocity_threshold

The object velocity_threshold specifies from which actual speed value the drive is considered to be standing still. If the drive exceeds the speed value specified here for a specific time period, bit 12 (velocity = 0) is deleted in the statusword. The time period is determined through the object velocity_threshold_time.

Index	606F_h
Name	velocity_threshold
Object Code	VAR
Data Type	UINT16

Access	rw
PDO mapping	yes
Units	speed units
Value Range	0 ... 65536 min ⁻¹
Default Value	10

Object 6070_h: velocity_threshold_time

The object velocity_threshold_time specifies how long the drive may exceed the specified speed before bit 12 (velocity = 0) is deleted in the statusword.

Index	6070_h
Name	velocity_threshold_time
Object Code	VAR
Data Type	UINT16

Access	rw
PDO mapping	yes
Units	ms
Value Range	0 ... 4999
Default Value	0

Object 6080_h: max_motor_speed

The object max_motor_speed gives the highest allowed speed for the motor in min⁻¹. The object is used to protect the motor and can be taken from the motor technical data. The speed setpoint value is limited to this value.

Index	6080_h
Name	max_motor_speed
Object Code	VAR
Data Type	UINT16

Access	rw
PDO mapping	yes
Units	min ⁻¹
Value Range	0 ... 32768 min ⁻¹
Default Value	32768 min ⁻¹

Object 60FF_h: target_velocity

The object target_velocity is the setpoint specification for the ramp generator.

Index	60FF_h
Name	target_velocity
Object Code	VAR
Data Type	INT32

Access	rw
PDO mapping	yes
Units	speed units
Value Range	–
Default Value	–

7.6 Speed ramps

Selected as modes_of_operation - profile_velocity_mode; the setpoint value ramp is also activated. And so it is possible to limit a jump-like setpoint change to a specific speed change per time via the objects profile_acceleration and profile_deceleration. The controller not only permits specification of different values for braking deceleration and acceleration, but also differentiation between positive and negative speed. The following illustration depicts this behaviour:

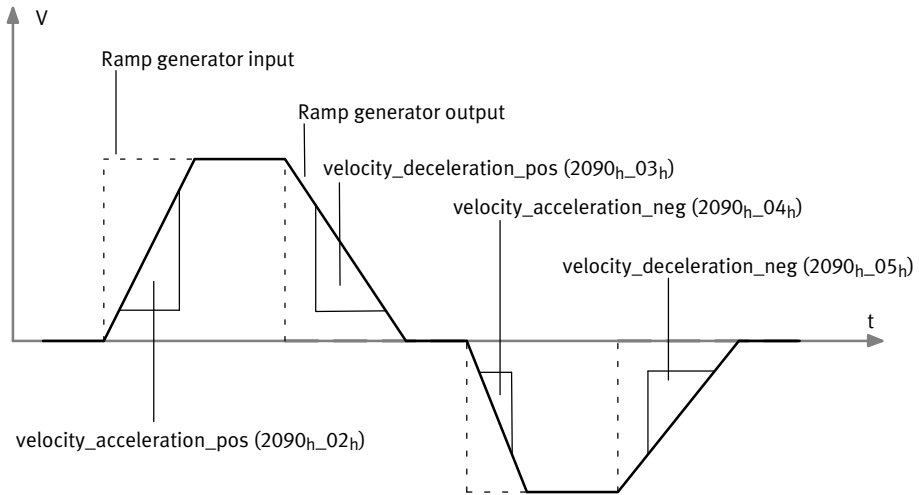


Fig. 7.11 Speed ramps

The object group `velocity_ramps` is available to parametrise these 4 accelerations. Observe that the objects `profile_acceleration` and `profile_deceleration` change the same internal accelerations as the `velocity_ramps`. If the `profile_acceleration` is written, `velocity_acceleration_pos` and `velocity_acceleration_neg` are revised together; if the `profile_deceleration` is written, `velocity_acceleration_pos` and `velocity_acceleration_neg` are revised together. The object `velocity_ramps_enable` determines whether or not the setpoint values are guided over the ramp generator.

Index	2090h
Name	velocity_ramps
Object Code	RECORD
No. of Elements	5

Sub-Index	01h
Description	velocity_ramps_enable
Data Type	UINT8
Access	rw
PDO mapping	no
Units	–
Value Range	0: Setpoint value NOT over the ramp generator 1: Setpoint value over the ramp generator
Default Value	1

Sub-Index	02_h
Description	velocity_acceleration_pos
Data Type	INT32
Access	rw
PDO mapping	no
Units	acceleration units
Value Range	–
Default Value	14 100 min ⁻¹ /s

Sub-Index	03_h
Description	velocity_deceleration_pos
Data Type	INT32
Access	rw
PDO mapping	no
Units	acceleration units
Value Range	–
Default Value	14 100 min ⁻¹ /s

Sub-Index	04_h
Description	velocity_acceleration_neg
Data Type	INT32
Access	rw
PDO mapping	no
Units	acceleration units
Value Range	–
Default Value	14 100 min ⁻¹ /s

Sub-Index	05_h
Description	velocity_deceleration_neg
Data Type	INT32
Access	rw
PDO mapping	no
Units	acceleration units
Value Range	–
Default Value	14 100 min ⁻¹ /s

7.7 Torque Regulation Operating Mode (Profile Torque Mode)

7.7.1 Overview

This chapter describes torque-regulated operation. This operating mode allows an external torque setpoint value `target_torque`, which can be smoothed using the integrated ramp generator, to be specified for the motor controller. It is thus possible for this motor controller to also be used for path control, with which both the position controller and the speed regulator are displaced to an external computer.

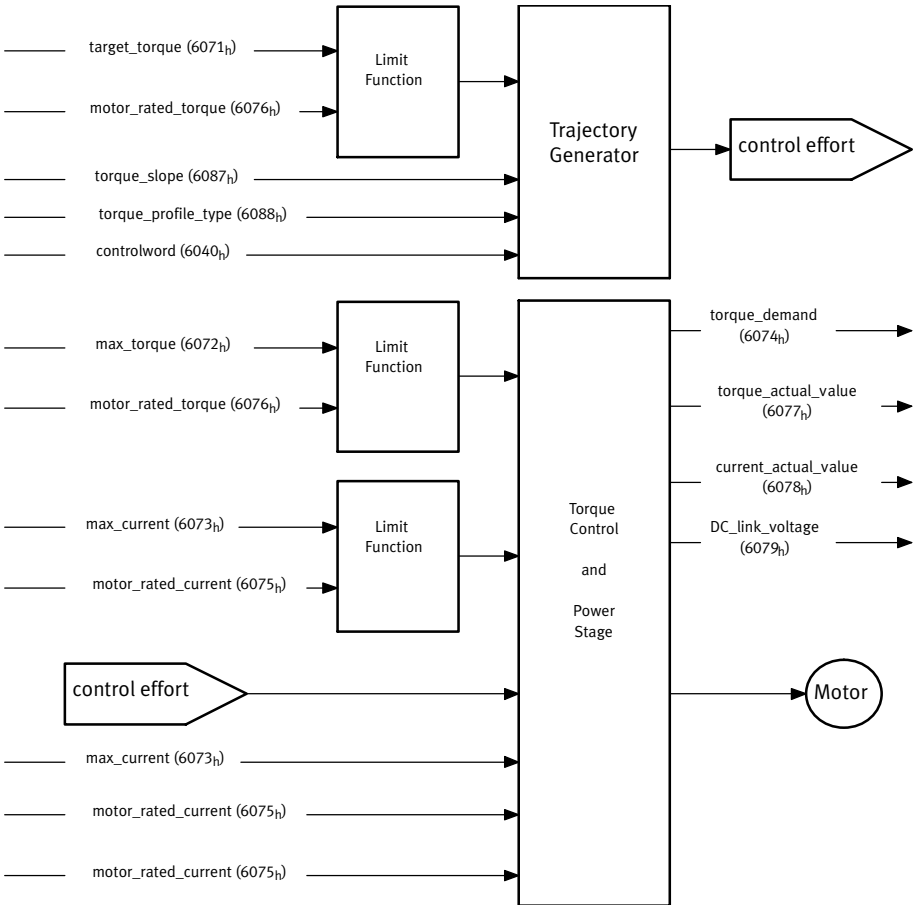


Fig. 7.12 Structure of torque-regulated operation

The parameters ramp steepness `torque_slope` and ramp shape `torque_profile_type` must be specified for the ramp generator.

If the bit 8 halt is set in the controlword, the ramp generator lowers the torque down to zero. It rises correspondingly again to the setpoint torque `target_torque` when bit 8 is deleted again. In both cases, the ramp generator takes into account the ramp steepness `torque_slope` and the ramp shape `torque_profile_type`.

All definitions within this document refer to rotatable motors. If linear motors have to be used, all “torque” objects must refer to a “force” instead. For simplicity, the objects do not appear twice and their names should not be changed.

The operating modes positioning mode (profile position mode) and speed regulator (profile velocity mode) need the torque controller to work. That is why it is always necessary to set its parameters.

7.7.2 Description of the Objects

Objects treated in this chapter

Index	Object	Name	Type	Attr.
6071 _h	VAR	<code>target_torque</code>	INT16	rw
6072 _h	VAR	<code>max_torque</code>	UINT16	rw
6074 _h	VAR	<code>torque_demand_value</code>	INT16	ro
6076 _h	VAR	<code>motor_rated_torque</code>	UINT32	rw
6077 _h	VAR	<code>torque_actual_value</code>	INT16	ro
6078 _h	VAR	<code>current_actual_value</code>	INT16	ro
6079 _h	VAR	<code>DC_link_circuit_voltage</code>	UINT32	ro
6087 _h	VAR	<code>torque_slope</code>	UINT32	rw
6088 _h	VAR	<code>torque_profile_type</code>	INT16	rw
60F7 _h	RECORD	<code>power_stage_parameters</code>		rw
60F6 _h	RECORD	<code>torque_control_parameters</code>		rw

Affected objects from other chapters

Index	Object	Name	Type	Chapter
6040 _h	VAR	<code>controlword</code>	INT16	6 Device Control
60F9 _h	RECORD	<code>motor_parameters</code>		5.5 Current Regulator and Motor Adjustment
6075 _h	VAR	<code>motor_rated_current</code>	UINT32	5.5 Current Regulator and Motor Adjustment
6073 _h	VAR	<code>max_current</code>	UINT16	5.5 Current Regulator and Motor Adjustment

Object 6071_h: target_torque

This parameter is the entry value for the torque regulator in torque-regulated mode (Profile Torque Mode). It is specified in thousandths of the nominal torque (object 6076_h).

Index	6071_h
Name	target_torque
Object Code	VAR
Data Type	INT16

Access	rw
PDO mapping	yes
Units	motor_rated_torque/1000
Value Range	-32768 ... 32768
Default Value	0

Object 6072_h: max_torque

This value represents the motor's maximum permissible torque. It is specified in thousandths of the nominal torque (object 6076_h). If, for example, a 2-fold overloading of the motor is briefly permissible, the value 2000 is entered here.



The object 6072_h: max_torque corresponds with the object 6073_h: max_current and may not be overwritten until the object 6075_h: motor_rated_current is overwritten with a valid value.

Index	6072_h
Name	max_torque
Object Code	VAR
Data Type	UINT16

Access	rw
PDO mapping	yes
Units	motor_rated_torque/1000
Value Range	-1000 ... 65536
Default Value	2023

Object 6074_h: torque_demand_value

By means of this object, the current nominal torque can be read out in thousands of the nominal torque (6076_h). The internal limitations of the controller (current limit values and I²t monitoring) are hereby taken into account.

Index	6074_h
Name	torque_demand_value
Object Code	VAR
Data Type	INT16

Access	ro
PDO mapping	yes
Units	motor_rated_torque/1000
Value Range	--
Default Value	--

Object 6076_h: motor_rated_torque

This object specifies the nominal torque of the motor. This can be taken from the motor's name plate. It is entered in the unit 0.001 Nm.

Index	6076_h
Name	motor_rated_torque
Object Code	VAR
Data Type	UINT32

Access	rw
PDO mapping	yes
Units	0.001 mNm
Value Range	–
Default Value	296

Object 6077_h: torque_actual_value

By means of this object, the motor's actual torque can be read out in thousandths of the nominal torque (object 6076_h).

Index	6077_h
Name	torque_actual_value
Object Code	VAR
Data Type	INT16

Access	ro
PDO mapping	yes
Units	motorRatedTorque/1000
Value Range	–
Default Value	–

Object 6078_h: current_actual_value

By means of this object, the motor's actual current can be read out in thousandths of the nominal current (object 6075_h).

Index	6078_h
Name	current_actual_value
Object Code	VAR
Data Type	INT16

Access	ro
PDO mapping	yes
Units	motorRatedCurrent/1000
Value Range	–
Default Value	–

Object 6079_h: dc_link_circuit_voltage

The intermediate circuit voltage of the controller can be read via this object. The voltage is specified in the unit millivolts.

Index	6079_h
Name	dc_link_circuit_voltage
Object Code	VAR
Data Type	UINT32

Access	ro
PDO mapping	yes
Units	mV
Value Range	–
Default Value	–

Object 6087_h: torque_slope

This parameter describes the modification speed of the setpoint value ramp value. This is specified in thousandths of the nominal torque per second. For example, the torque setpoint value target_torque is raised from 0 Nm to the value motor_rated_torque. If the initial value of the intermediately switched torque ramp should reach this value in one second, the value 1000 is written in this object.

Index	6087_h
Name	torque_slope
Object Code	VAR
Data Type	UINT32

Access	rw
PDO mapping	yes
Units	motor_rated_torque/1000 s
Value Range	–
Default Value	0E310F94 _h

Object 6088_h: torque_profile_type

The object torque_profile_type specifies with which curve shape a setpoint value jump should be executed. Currently, only the linear ramp is implemented in this controller, so this object can only be written with the value 0.

Index	6088_h
Name	torque_profile_type
Object Code	VAR
Data Type	INT16

Access	rw
PDO mapping	yes
Units	–
Value Range	0
Default Value	0

Value	Significance
0	Linear ramp

A Technical appendix

A.1 Technical Data Interface EtherCAT

M3

This section is only applicable for the motor controller CMMP-AS-...-M3.

A.1.1 General

Mechanical		
Length / width / height	[mm]	112.6 x 87.2 x 28.3
Weight	[g]	55
Slot		Slot Ext2
Note on materials		Conforms to RoHS

Tab. A.1 Technical data: mechanical

Electric		
Signal level	[V DC]	0 ... 2.5
Differential voltage	[V DC]	1.9 ... 2.1

Tab. A.2 Technical data: electrical

A.1.2 Operating and environmental conditions

Transport		
Temperature range	[°C]	0 ... +50
Air humidity, at max. 40 °C ambient temperature, non-condensing	[%]	0 ... 90

Tab. A.3 Technical data: transport

Storage		
Storage temperature	[°C]	-25 ... +75
Air humidity, at max. 40 °C ambient temperature, non-condensing	[%]	0 ... 90
Permissible altitude (above sea level)	[m]	< 1000

Tab. A.4 Technical data: storage

B Diagnostic messages

If an error occurs, the motor controller CMMP-AS-...-M3/-M0 shows a diagnostic message cyclically in the 7-segments display. An error message consists of an E (for Error), a main index and sub-index, e.g.: - **E 0 1 0** -.

Warnings have the same number as an error message. In contrast to error messages, however, warnings are preceded and followed by hyphens, e.g. - **1 7 0** -.

B.1 Explanations on the diagnostic messages

The following table summarises the significance of the diagnostic messages and the actions to be taken in response to them:

Terms	Significance
No.	Main index (error group) and sub-index of the diagnostic message. Shown in the display, in FCT or diagnostic memory via FHPP.
Code	The Code column includes the error code (Hex) via CiA 301.
Message	Message that is displayed in the FCT.
Cause	Possible causes for the message.
Action	Action by the user.
Reaction	The Reaction column includes the error response (default setting, partially configurable): <ul style="list-style-type: none"> - PS off (switch off output stage), - MCStop (fast stop with maximum current), - QStop (fast stop with parameterised ramp), - Warn (warning), - Ignore (No message, only entry in diagnostic memory), - NoLog (No message and no entry in diagnostic memory).

Tab. B.1 Explanations of the diagnostic messages

Under section B.2, you will find the error codes in accordance with CiA301/402 with assignment to the error numbers of the diagnostic messages.

A complete list of the diagnostic messages corresponding to the firmware statuses at the time of printing of this document can be found in section B.3.

B.2 Error codes via CiA 301/402

Diagnostic messages			
Code	No.	Message	Reaction
2311h	31-1	I²t-servo controller	Configurable
2312h	31-0	I²t-motor	Configurable
2313h	31-2	I²t-PFC	Configurable
2314h	31-3	I²t braking resistor	Configurable
2320h	06-0	Short circuit in output stage	PS off
	06-1	Overload current brake chopper	PS off
3210h	07-0	Overvoltage in intermediate circuit	PS off
3220h	02-0	Undervoltage in intermediate circuit	Configurable
3280h	32-0	Intermediate circuit charging time exceeded	Configurable
3281h	32-1	Undervoltage for active PFC	Configurable
3282h	32-5	Brake chopper overload. Intermediate circuit could not be discharged.	Configurable
3283h	32-6	Intermediate circuit discharge time exceeded	Configurable
3284h	32-7	Power supply missing for controller enable	Configurable
3285h	32-8	Power supply failure during controller enable	QStop
3286h	32-9	Phase failure	QStop
4210h	04-0	Power end stage over-temperature	Configurable
4280h	04-1	Intermediate circuit over-temperature	Configurable
4310h	03-0	Analogue motor over-temperature	QStop
	03-1	Digital motor over-temperature	Configurable
	03-2	Analogue motor over-temperature: broken cable	Configurable
	03-3	Analogue motor over-temperature: short circuit	Configurable
5080h	90-0	Missing hardware components (SRAM)	PS off
	90-2	Error when booting FPGA	PS off
	90-3	Error at SD-ADU start	PS off
	90-4	SD-ADU synchronisation error after start	PS off
	90-5	SD-ADU not synchronous	PS off
	90-6	IRQ0 (current regulator): trigger error	PS off
	90-9	DEBUG firmware loaded	PS off
5114h	05-0	Failure of internal voltage 1	PS off
5115h	05-1	Failure of internal voltage 2	PS off
5116h	05-2	Failure of driver supply	PS off
5280h	21-0	Error 1 current measurement U	PS off
5281h	21-1	Error 1 current measurement V	PS off
5282h	21-2	Error 2 current measurement U	PS off
5283h	21-3	Error 2 current measurement V	PS off
5410h	05-3	Undervoltage dig. I/O	PS off
	05-4	Overload current dig. I/O	PS off
5580h	26-0	Missing user parameter set	PS off

Diagnostic messages			
Code	No.	Message	Reaction
5581h	26-1	Checksum error	PS off
5582h	26-2	Flash: error when writing	PS off
5583h	26-3	Flash: error during deletion	PS off
5584h	26-4	Flash: error in internal flash	PS off
5585h	26-5	Missing calibration data	PS off
5586h	26-6	Missing user position data records	PS off
6000h	91-0	Internal initialisation error	PS off
6080h	25-0	Invalid device type	PS off
6081h	25-1	Device type not supported	PS off
6082h	25-2	Hardware revision not supported	PS off
6083h	25-3	Device function restricted!	PS off
6180h	01-0	Stack overflow	PS off
6181h	16-0	Program execution faulty	PS off
6182h	16-1	Illegal interrupt	PS off
6183h	16-3	Unexpected status	PS off
6185h	15-0	Division by 0	PS off
6186h	15-1	Range exceeded	PS off
6187h	16-2	Initialisation error	PS off
6320h	36-0	Parameter was limited	Configurable
	36-1	Parameter was not accepted	Configurable
6380h	30-0	Internal conversion error	PS off
7380h	08-0	Resolver angle encoder error	Configurable
7382h	08-2	Error in incremental encoder tracking signal Z0	Configurable
7383h	08-3	Error in incremental encoder tracking signals Z1	Configurable
7384h	08-4	Digital incremental encoder track signals error [X2B]	Configurable
7385h	08-5	Error in increment generator of Hall-effect encoder signals	Configurable
7386h	08-6	Angle encoder communication fault	Configurable
7387h	08-7	Signal amplitude of incremental tracks faulty [X10]	Configurable
7388h	08-8	Internal angle encoder error	Configurable
7389h	08-9	Angle encoder at [X2B] is not supported	Configurable
73A1h	09-0	Old angle encoder parameter set	Configurable
73A2h	09-1	Angle encoder parameter set cannot be decoded	Configurable
73A3h	09-2	Unknown version of angle encoder parameter set	Configurable
73A4h	09-3	Defective data structure in angle encoder parameter set	Configurable
73A5h	09-7	Write-protected EEPROM angle encoder	Configurable
73A6h	09-9	Angle encoder's EEPROM too small	Configurable
8081h	43-0	Limit switch: negative setpoint value blocked	Configurable
8082h	43-1	Limit switch: positive setpoint value blocked	Configurable
8083h	43-2	Limit switch: positioning suppressed	Configurable
8120h	12-1	CAN: Communication error, bus OFF	Configurable
8180h	12-0	CAN: double node number	Configurable

Diagnostic messages			
Code	No.	Message	Reaction
8181h	12-2	CAN: communication error during transmission	Configurable
8182h	12-3	CAN: communication error during reception	Configurable
8480h	35-0	Linear motor spinning protection	Configurable
8611h	17-0	Following error monitoring	Configurable
	17-1	Encoder difference monitoring	Configurable
	27-0	Following error warning threshold	Configurable
8612h	40-0	Negative software limit switch	Configurable
	40-1	Positive software limit switch reached	Configurable
	40-2	Target position behind the negative software limit switch	Configurable
	40-3	Target position behind the positive software limit switch	Configurable
8680h	42-0	Positioning: no subsequent positioning: stop	Configurable
8681h	42-1	Positioning: reversing direction of rotation not allowed: stop	Configurable
8682h	42-2	Positioning: reversing after halt not allowed	Configurable
8780h	34-0	No synchronisation via fieldbus	Configurable
8781h	34-1	Fieldbus synchronisation error	Configurable
8A80h	11-0	Error when homing is started	Configurable
8A81h	11-1	Error during homing	Configurable
8A82h	11-2	Homing: no valid zero impulse	Configurable
8A83h	11-3	Homing: timeout	Configurable
8A84h	11-4	Homing: wrong / invalid limit switch	Configurable
8A85h	11-5	Homing: I²t / following error	Configurable
8A86h	11-6	Homing: end of search path	Configurable
8A87h	33-0	Encoder emulation following error	Configurable
F080h	80-0	Current regulator IRQ overflow	PS off
F081h	80-1	Speed regulator IRQ overflow	PS off
F082h	80-2	Overflow position controller IRQ	PS off
F083h	80-3	Interpolator IRQ overflow	PS off
F084h	81-4	Low-level IRQ overflow	PS off
F085h	81-5	MDC IRQ overflow	PS off
FF01h	28-0	Missing hours-run meter	Configurable
FF02h	28-1	Hours-run meter: write error	Configurable

B.3 Diagnostic messages with instructions for fault clearance

Error group 0		Information		
No.	Code	Message	Reaction	
0-0	-	Invalid error		Ignore
		Cause	Information: An invalid error entry (corrupted) was found in the diagnostic memory marked with this error number. The system time entry is set to 0.	
		Measure	–	
0-1	-	Invalid error detected and corrected		Ignore
		Cause	Information: An invalid error entry (corrupted) was found in the diagnostic memory and corrected. The Additional informationrma-tion contains the original error number. The system time entry includes the address of the corrupted error number.	
		Measure	–	
0-2	-	Error cleared		Ignore
		Cause	Information: Active errors were acknowledged.	
		Measure	–	
0-4	-	Serial number / device type changed (change of modules)		Ignore
		Cause	Information: → Entry in the diagnostic memory.	
		Measure	–	
0-7	-	Consecutive Entry		Ignore
		Cause	Information: → Entry in the diagnostic memory.	
		Measure	–	
0-8	-	Controller switched on		Ignore
		Cause	Information: → Entry in the diagnostic memory.	
		Measure	–	
0-9	-	Controller safety parameters changed		Ignore
		Cause	Information: → Entry in the diagnostic memory.	
		Measure	–	
0-11	-	Module change: Previous module		Ignore
		Cause	Information: → Entry in the diagnostic memory.	
		Measure	–	
0-12	-	Module change: Current module		Ignore
		Cause	Information: → Entry in the diagnostic memory.	
		Measure	–	
0-21	-	Log entry of the Safety module		Ignore
		Cause	Information: → Entry in the diagnostic memory.	
		Measure	–	
0-22	-	Default parameter set loaded		Ignore
		Cause	Information: → Entry in the diagnostic memory.	
		Measure	–	

Error group 1		Stack overflow	
No.	Code	Message	Reaction
1-0	6180h	Stack overflow	
		Cause	<ul style="list-style-type: none"> – Incorrect firmware? – Sporadic high processor load due to cycle time being too short and specific processor-intensive processes (save parameter set, etc.).
		Measure	<ul style="list-style-type: none"> • Load an approved firmware. • Reduce processor load. • Contact Technical Support.
			PSoff

Error group 2		Intermediate circuit	
No.	Code	Message	Reaction
2-0	3220h	Intermediate circuit undervoltage	
			configurable
		Cause	Intermediate circuit voltage falls below the parameterised threshold (➔ Additional information). Error priority set too high?
		Measure	<ul style="list-style-type: none"> • Quick discharge due to switched-off mains supply. • Check the power supply. • Couple intermediate circuits if technically permissible. • Check intermediate circuit voltage (measure). • Check undervoltage monitoring (threshold value).
Additional information	Additional information in PNU 203/213: Top 16 bits: Status number of internal state machine Bottom 16 bits: Intermediate circuit voltage (internal scaling approx. 17.1 digit/V).		

Error group 3		Motor over-temperature	
No.	Code	Message	Reaction
3-0	4310h	Analogue motor overtemperature	
			QStop
		Cause	Motor overloaded, temperature too high. <ul style="list-style-type: none"> – Motor too hot? – Incorrect sensor? – Sensor faulty? – Broken cable?
Measure	<ul style="list-style-type: none"> • Check parameterisation (current regulator, current limits). • Check the parameterisation of the sensor or the sensor characteristics. <p>If the error persists when the sensor is bypassed: device faulty.</p>		

Error group 3		Motor over-temperature	
No.	Code	Message	Reaction
3-1	4310h	Digital motor overtemperature	
		Cause	<ul style="list-style-type: none"> – Motor overloaded, temperature too high. – Suitable sensor or sensor characteristics parameterised? – Sensor faulty?
		Measure	<ul style="list-style-type: none"> • Check parameterisation (current regulator, current limits). • Check the parameterisation of the sensor or the sensor characteristics. <p>If the error persists when the sensor is bypassed: device faulty.</p>
3-2	4310h	Analogue motor overtemperature: Broken wire	
		Cause	The measured resistance value is above the threshold for wire break detection.
		Measure	<ul style="list-style-type: none"> • Check the connecting cables of the temperature sensor for wire breaks. • Check the parameterisation (threshold value) for wire break detection.
3-3	4310h	Analogue motor overtemperature: Short circuit	
		Cause	The measured resistance value is below the threshold for short circuit detection.
		Measure	<ul style="list-style-type: none"> • Check the connecting cables of the temperature sensor for wire breaks. • Check the parameterisation (threshold value) for short circuit detection.

Error group 4		Power section/intermediate circuit over-temperature	
No.	Code	Message	Reaction
4-0	4210h	Power section overtemperature	
		Cause	<p>Device is overheated</p> <ul style="list-style-type: none"> – Is displayed temperature plausible? – Device fan faulty? – Device overloaded?
		Measure	<ul style="list-style-type: none"> • Check installation conditions; control cabinet fan filter dirty? • Check the cylinder sizing (due to possible overloading in continuous duty).

Error group 4		Power section/intermediate circuit over-temperature	
No.	Code	Message	Reaction
4-1	4280h	Intermediate circuit overtemperature	
		Cause	Device is overheated – Is displayed temperature plausible? – Device fan faulty? – Device overloaded?
		Measure	<ul style="list-style-type: none"> • Check installation conditions; control cabinet fan filter dirty? • Check the cylinder sizing (due to possible overloading in continuous duty).

Error group 5		Internal voltage supply	
No.	Code	Message	Reaction
5-0	5114h	Failure of internal voltage 1	
		Cause	Internal power supply monitor has detected undervoltage. This is either due to an internal defect or an overload/short circuit caused by connected peripherals.
		Measure	<ul style="list-style-type: none"> • Check digital outputs and brake output for short circuit or specified load. • Separate device from the entire peripheral equipment and check whether the error is still present after reset. If so, an internal defect is present → Repair by the manufacturer.
5-1	5115h	Failure of internal voltage 2	
		Cause	Internal power supply monitor has detected undervoltage. This is either due to an internal defect or an overload/short circuit caused by connected peripherals.
		Measure	<ul style="list-style-type: none"> • Check digital outputs and brake output for short circuit or specified load. • Separate device from the entire peripheral equipment and check whether the error is still present after reset. If so, an internal defect is present → Repair by the manufacturer.
5-2	5116h	Failure of driver supply	
		Cause	Internal power supply monitor has detected undervoltage. This is either due to an internal defect or an overload/short circuit caused by connected peripherals.
		Measure	<ul style="list-style-type: none"> • Check digital outputs and brake output for short circuit or specified load. • Separate device from the entire peripheral equipment and check whether the error is still present after reset. If so, an internal defect is present → Repair by the manufacturer.

Error group 5		Internal voltage supply	
No.	Code	Message	Reaction
5-3	5410h	Undervoltage of digital I/O	
		Cause	Overloading of the I/Os? Faulty peripheral device?
		Measure	<ul style="list-style-type: none"> • Check connected peripherals for short circuit / rated loads. • Check connection of the brake (connected incorrectly?).
5-4	5410h	Overcurrent of digital I/O	
		Cause	Overloading of the I/Os? Faulty peripheral device?
		Measure	<ul style="list-style-type: none"> • Check connected peripherals for short circuit / rated loads. • Check connection of the brake (connected incorrectly?).
5-5	-	Module supply voltage failure	
		Cause	Defect on the plugged-in interface.
		Measure	<ul style="list-style-type: none"> • Interface replacement → Repair by the manufacturer.
5-6	-	X10, [X11] and RS232 supply voltage failure	
		Cause	Overloading through connected peripherals.
		Measure	<ul style="list-style-type: none"> • Check pin allocation of the connected peripherals. • Short circuit?
5-7	-	Safety module internal voltage failure	
		Cause	Defect on the safety module.
		Measure	<ul style="list-style-type: none"> • Internal defect → Repair by the manufacturer.
5-8	-	Failure of Internal voltage 3 (15V)	
		Cause	Defect in the motor controller.
		Measure	<ul style="list-style-type: none"> • Internal defect → Repair by the manufacturer.
5-9	-	Encoder supply defective	
		Cause	Back measurement of the encoder voltage not OK.
		Measure	<ul style="list-style-type: none"> • Internal defect → Repair by the manufacturer.

Error group 6		Overload current	
No.	Code	Message	Reaction
6-0	2320h	Output stage short-circuit	
			PSoff
		Cause	<ul style="list-style-type: none"> – Faulty motor, e.g. winding short circuit due to motor overheating or short to PE inside motor. – Short circuit in the cable or the connecting plugs, i.e. short circuit between motor phases or to the screening/PE. – Output stage faulty (short circuit). – Incorrect parameterisation of the current regulator.
	Measure	Dependent on the status of the system → Additional information, cases a) to f).	
	Additional information	<p>Actions:</p> <p>a) Error only with active brake chopper: Check external braking resistor for short circuit or insufficient resistance value. Check circuitry of the brake chopper output at the motor controller (jumper, etc.).</p> <p>b) Error message immediately when the power supply is connected: internal short circuit in the output stage (short circuit of a complete half-jumper). The motor controller can no longer be connected to the power supply; the internal (and possibly external) fuses are tripped. Repair by the manufacturer required.</p> <p>c) Short circuit error message not until the output stage or controller is enabled.</p> <p>d) Disconnection of motor plug [X6] directly at the motor controller. If the error still occurs, there is a fault in the motor controller. Repair by the manufacturer required.</p> <p>e) If the error only occurs when the motor cable is connected: Check the motor and cable for short circuits, e.g. with a multimeter.</p> <p>f) Check parameterisation of the current regulator. Oscillations in an incorrectly parameterised current regulator can generate currents up to the short circuit threshold, usually clearly audible as a high-frequency whistling. Verification, if necessary, with the trace in the FCT (actual active current value).</p>	
6-1	2320h	Brake chopper overcurrent	
			PSoff
		Cause	Overload current at the brake chopper output.
	Measure	<ul style="list-style-type: none"> • Check external braking resistor for short circuit or insufficient resistance value. • Check circuitry of the brake chopper output at the motor controller (jumpers, etc.). 	

Error group 7		Overvoltage in intermediate circuit	
No.	Code	Message	Reaction
7-0	3210h	Intermediate circuit overvoltage	
		PSoff	
		Cause	Braking resistor is overloaded; too much braking energy, which cannot be dissipated quickly enough. <ul style="list-style-type: none"> – Incorrect level of resistance? – Resistor not connected correctly? – Check design (application).
Measure	<ul style="list-style-type: none"> • Check the design of the braking resistor; resistance value may be too great. • Check the connection to the braking resistor (internal/external). 		

Error group 8		Angle encoder error	
No.	Code	Message	Reaction
8-0	7380h	Resolver angle encoder error	
		configurable	
		Cause	Resolver signal amplitude is faulty.
		Measure	Step-by-step procedure → Additional information, cases a) to c).
Additional information	<p>a) If possible, test with a different (error-free) resolver (replace the connecting cable, too). If the error still occurs, there is a fault in the motor controller. Repair by the manufacturer required.</p> <p>b) If the error occurs only with a special resolver and its connecting cable: Check resolver signals (carrier and SIN/COS signal), see specification. If the signals do not comply with the signal specifications, replace the resolver.</p> <p>c) If the error recurs sporadically, check the screen bonding or check whether the resolver simply has an insufficient transmission ratio (standard resolver: A = 0.5).</p>		

Error group 8		Angle encoder error	
No.	Code	Message	Reaction
8-1	-	Direction of rotation of the serial and incremental position evaluation is not identical	
		configurable	
		Cause	Only encoders with serial position transmission combined with an analogue SIN/COS signal track: The directions of rotation for position determination in the encoder and for incremental evaluation of the analogue track system in the motor controller are the wrong way round → Additional information.
		Measure	Swap the following signals on the [X2B] angle encoder interface (the wires in the connecting plug must be changed around), observing the technical data for the angle encoder where applicable: <ul style="list-style-type: none"> – Swap SIN / COS track. – Swap the SIN+/SIN- or COS+/COS- signals, as applicable.
Additional information	The encoder counts internally, for example positively in clockwise rotation, while the incremental evaluation counts in negative direction with the same mechanical rotation. The interchange of the direction of rotation is detected mechanically at the first movement of over 30°, and the error is triggered.		
8-2	7382h	Incremental encoder Z0 track signals error	
		configurable	
		Cause	Signal amplitude of the Z0 track at [X2B] is faulty. <ul style="list-style-type: none"> – Angle encoder connected? – Angle encoder cable defective? – Angle encoder faulty?
		Measure	Check configuration of the angle encoder interface: <ol style="list-style-type: none"> a) Z0 evaluation activated, but no tracking signals connected or on hand → Additional information. b) Encoder signals faulty? c) Test with another encoder. → Tab. B.2, page 281.
Additional information	For example, EnDat 2.2 or EnDat 2.1 without analogue track. Heidenhain encoder: order codes EnDat 22 and EnDat 21. With these encoders there are no incremental signals, even when the cables are connected.		

Error group 8		Angle encoder error	
No.	Code	Message	Reaction
8-3	7383h	Incremental encoder Z1 track signals error	
		Cause	Signal amplitude of the Z1 track at X2B is faulty. – Angle encoder connected? – Angle encoder cable defective? – Angle encoder faulty?
		Measure	Check configuration of the angle encoder interface: a) Z1 evaluation activated but not connected. b) Encoder signals faulty? c) Test with another encoder. ➔ Tab. B.2, page 281.
8-4	7384h	Digital incremental encoder track signals error [X2B]	
		Cause	Faulty A, B or N tracking signals at [X2B]. – Angle encoder connected? – Angle encoder cable defective? – Angle encoder faulty?
		Measure	Check the configuration of the angle encoder interface. a) Encoder signals faulty? b) Test with another encoder. ➔ Tab. B.2, page 281.
8-5	7385h	Incremental encoder Hall generator signals error	
		Cause	Hall encoder signals of a dig. inc. at [X2B] faulty. – Angle encoder connected? – Angle encoder cable defective? – Angle encoder faulty?
		Measure	Check the configuration of the angle encoder interface. a) Encoder signals faulty? b) Test with another encoder. ➔ Tab. B.2, page 281.

Error group 8		Angle encoder error	
No.	Code	Message	Reaction
8-6	7386h	Faulty angle encoder communication	
			configurable
		Cause	<p>Communication to serial angle encoders is disrupted (EnDat encoders, HIPERFACE encoders, BiSS encoders).</p> <ul style="list-style-type: none"> – Angle encoder connected? – Angle encoder cable defective? – Angle encoder faulty?
		Measure	<p>Check configuration of the angle encoder interface, procedure corresponding to a) to c):</p> <ul style="list-style-type: none"> a) Serial encoder parameterised but not connected? Incorrect serial protocol selected? b) Encoder signals faulty? c) Test with another encoder. <p>➔ Tab. B.2, page 281.</p>
8-7	7387h	Signal amplitude of encoder erroneous [X10]	
			configurable
		Cause	<p>Faulty A, B, or N tracking signals at [X10].</p> <ul style="list-style-type: none"> – Angle encoder connected? – Angle encoder cable defective? – Angle encoder faulty?
		Measure	<p>Check the configuration of the angle encoder interface.</p> <ul style="list-style-type: none"> a) Encoder signals faulty? b) Test with another encoder. <p>➔ Tab. B.2, page 281.</p>
8-8	7388h	Internal angle encoder error	
			configurable
		Cause	<p>Internal monitoring of the angle encoder [X2B] has detected an error and forwarded it via serial communication to the controller.</p> <ul style="list-style-type: none"> – Diminishing illumination intensity with visual encoders? – Excess rotational speed? – Angle encoder faulty?
		Measure	<p>If the error occurs repeatedly, the encoder is faulty. ➔ Replace encoder.</p>

Error group 8		Angle encoder error	
No.	Code	Message	Reaction
8-9	7389h	Angle encoder at [X2B] not supported	
		Cause	<p>Angle encoder type read at [X2B], which is not supported or cannot be used in the desired operating mode.</p> <ul style="list-style-type: none"> – Incorrect or inappropriate protocol type selected? – Firmware does not support the connected encoder variant?
		Measure	<p>Depending on the Additional information of the error message → Additional information:</p> <ul style="list-style-type: none"> • Load appropriate firmware. • Check/correct the configuration for encoder analysis. • Connect an appropriate encoder type.
Additional information	<p>Additional information (PNU 203/213):</p> <p>0001: HIPERFACE: Encoder type is not supported by the firmware → connect another encoder type or load more recent firmware, if applicable.</p> <p>0002: EnDat: The address space in which the encoder parameters would have to lie does not exist with the connected EnDat encoder → check the encoder type.</p> <p>0003: EnDat: Encoder type is not supported by the firmware → connect another encoder type or load more recent firmware, if applicable.</p> <p>0004: EnDat: Encoder rating plate cannot be read from the connected encoder. → Change encoder or load more recent firmware, if applicable.</p> <p>0005: EnDat: EnDat 2.2 interface parameterised, but connected encoder supports only EnDat2.1. → Replace encoder type or reparameterise to EnDat 2.1.</p> <p>0006: EnDat: EnDat2.1 interface with analogue track evaluation parameterised, but according to rating plate the connected encoder does not support tracking signals. → Replace encoder or switch off Z0 tracking signal evaluation.</p> <p>0007: Code length measuring system with EnDat2.1 connected, but parameterised as a purely serial encoder. Purely serial evaluation is not possible due to the long response times of this encoder system. Encoder must be operated with analogue tracking signal evaluation → connect to analogue Z0 tracking signal evaluation.</p>		
			configurable

Error group 9		Error in the angle encoder parameter set	
No.	Code	Message	Reaction
9-0	73A1h	Old encoder parameter set	
		Cause	Warning: An encoder parameter set in an old format was found in the EEPROM of the connected encoder. This has been converted and saved in the new format.
		Measure	No action necessary at this point. The warning should not re-appear when the 24 V supply is switched back on.
9-1	73A2h	Encoder parameter set cannot be decoded	
		Cause	Data in the EEPROM of the angle encoder could not be read completely, or access to it was partly refused.
		Measure	The EEPROM of the encoder contains data (communication objects) which is not supported by the loaded firmware. The data in question is then discarded. <ul style="list-style-type: none"> • The parameter set can be adapted to the current firmware by writing the encoder data to the encoder. • Alternatively, load appropriate (more recent) firmware.
9-2	73A3h	Unknown encoder parameter set version	
		Cause	The data saved in the EEPROM is not compatible with the current version. A data structure was found which the loaded firmware is unable to decode.
		Measure	<ul style="list-style-type: none"> • Save the encoder parameters again in order to delete the parameter record in the encoder and replace it with a readable record (this will, however, delete the data in the encoder irreversibly). • Alternatively, load appropriate (more recent) firmware.
9-3	73A4h	Defective data structure angle encoder parameter set	
		Cause	Data in EEPROM does not match the stored data structure. The data structure was identified as valid but may be corrupted.
		Measure	<ul style="list-style-type: none"> • Save the encoder parameters again in order to delete the parameter record in the encoder and replace it with a readable record. If the error still occurs after that, the encoder may be faulty. • Replace the encoder as a test.

Error group 9		Error in the angle encoder parameter set	
No.	Code	Message	Reaction
9-4	-	EEPROM data: User-specific configuration faulty	
		Cause	Only for special motors: The plausibility check returns an error, e.g. because the motor was repaired or replaced.
		Measure	<ul style="list-style-type: none"> • If motor repaired: Carry out homing again and save in the angle encoder, after that (!) save in the motor controller. • If motor replaced: Parameterise the controller again, then carry out homing again and save in the angle encoder, after that (!) save in the motor controller.
9-5	-	Read/Write Error EEPROM parameter data	
		Cause	Error occurred during reading or writing data to the internal encoder parameter set.
		Measure	Occurs with Hiperface encoders: A data field of the encoder is not suitable to be read from the firmware or data can not be written for unknown reasons. <ul style="list-style-type: none"> • Send motor to the manufacturer for inspection.
9-7	73A5h	Encoder EEPROM is write protected	
		Cause	Data cannot be saved in the EEPROM of the angle encoder. Occurs with Hiperface encoders.
		Measure	A data field in the encoder EEPROM is write-protected (e.g. after operation on a motor controller of another manufacturer). No solution possible, encoder memory must be unlocked with a corresponding parameterisation tool (from manufacturer).
9-9	73A6h	Memory size of encoder EEPROM too small	
		Cause	It is not possible to save all the data in the EEPROM of the angle encoder.
		Measure	<ul style="list-style-type: none"> • Reduce the number of data records to be saved. Please read the documentation or contact Technical Support.

Error group 10		Exceeding max. speed	
No.	Code	Message	Reaction
10-0	-	Overspeed	
		Cause	<ul style="list-style-type: none"> – Motor racing ("spinning") because the commutation angle offset is incorrect. – Motor is parameterised correctly, but the limit for spinning protection is set too low.
		Measure	<ul style="list-style-type: none"> • Check the commutation angle offset. • Check the parameterisation of the limit value.

Error group 11		Homing	
No.	Code	Message	Reaction
11-0	8A80h	Error when homing is started	
		Cause	Controller enable missing.
		Measure	Homing can only be started when closed-loop controller enable is active. <ul style="list-style-type: none"> • Check the condition or sequence.
11-1	8A81h	Error during homing	
		Cause	Homing was interrupted, e.g. by: <ul style="list-style-type: none"> – Withdrawal of controller release. – Reference switch is beyond the limit switch. – External stop signal (a phase was aborted during homing).
		Measure	<ul style="list-style-type: none"> • Check homing sequence. • Check arrangement of the switches. • If applicable, lock the stop input during homing if it is not desired.
11-2	8A82h	Homing: No valid zero pulse	
		Cause	Required zero pulse during homing missing.
		Measure	<ul style="list-style-type: none"> • Check the zero pulse signal. • Check the angle encoder settings.
11-3	8A83h	Homing: Timeout	
		Cause	The parameterised maximum time for the homing run was exceeded before homing was completed.
		Measure	<ul style="list-style-type: none"> • Check the time setting in the parameters.
11-4	8A84h	Homing: Incorrect limit switch	
		Cause	<ul style="list-style-type: none"> – Associated limit switch not connected. – Limit switches swapped? – No reference switch found between the two limit switches. – Reference switch is on the limit switch. – Current position with zero pulse method: Limit switch active in the area of the zero pulse (not permissible). – Both limit switches active at the same time.
		Measure	<ul style="list-style-type: none"> • Check whether the limit switches are connected in the correct direction of travel or whether the limit switches have an effect on the intended inputs. • Reference switch connected? • Check configuration of the reference switches. • Move limit switch so that it is not in the zero pulse area. • Check limit switch parameterisation (N/C contact/N/O contact).

Error group 11		Homing	
No.	Code	Message	Reaction
11-5	8A85h	Homing: I²t / following error	
		Cause	<ul style="list-style-type: none"> – Unsuitable acceleration ramp parameters. – Change of direction due to premature triggering of following error; check parameterisation of following error. – No reference switch reached between the end stops. – Zero pulse method: End stop reached (not permissible here).
		Measure	<ul style="list-style-type: none"> • Parameterise the acceleration ramps so they are flatter. • Check connection of a reference switch. • Method appropriate for the application?
11-6	8A86h	Homing: End of search path	
		Cause	The maximum permissible path for the homing run has been travelled without reaching the point of reference or the homing run destination.
		Measure	Fault in switch detection. <ul style="list-style-type: none"> • Switch for homing faulty?
11-7	-	Homing: Error encoder difference monitoring	
		Cause	Deviation between the actual position value and commutation position is too great. External angle encoder not connected or faulty?
		Measure	<ul style="list-style-type: none"> • Deviation fluctuating, e.g. due to gear backlash; increase cut-off threshold if necessary. • Check connection of the actual value encoder.

Error group 12		CAN communication	
No.	Code	Message	Reaction
12-0	8180h	CAN: Double node number	
		Cause	Node number assigned twice.
		Measure	<ul style="list-style-type: none"> • Check the configuration of the participants on the CAN bus.
12-1	8120h	CAN: Communication error, bus OFF	
		Cause	The CAN chip has switched off communication due to communication errors (BUS OFF).
		Measure	<ul style="list-style-type: none"> • Check cabling: cable specification adhered to, broken cable, maximum cable length exceeded, correct terminating resistors, cable screening earthed, all signals terminated? • If necessary, replace device as a test. If a different device works without errors with the same cabling, send the device to the manufacturer for inspection.

Error group 12		CAN communication	
No.	Code	Message	Reaction
12-2	8181h	CAN: Communication error during transmission	
		Cause	The signals are corrupted when transmitting messages. Device boot-up is so fast that no other nodes on the bus have yet been detected when the boot-up message is sent.
		Measure	<ul style="list-style-type: none"> • Check cabling: cable specification adhered to, broken cable, maximum cable length exceeded, correct terminating resistors, cable screening earthed, all signals terminated? • If necessary, replace device as a test. If a different device works without errors with the same cabling, send the device to the manufacturer for inspection.
12-3	8182h	CAN: Communication error during reception	
		Cause	The signals are corrupted when receiving messages.
		Measure	<ul style="list-style-type: none"> • Check cabling: cable specification adhered to, broken cable, maximum cable length exceeded, correct terminating resistors, cable screening earthed, all signals terminated? • If necessary, replace device as a test. If a different device works without errors with the same cabling, send the device to the manufacturer for inspection.
12-4	-	No Node Guarding-telegram received	
		Cause	Node guarding telegram not received within the parameterised time. Signals corrupted?
		Measure	<ul style="list-style-type: none"> • Compare the cycle time of the remote frames with that of the controller. • Check: failure of the controller?
12-5	-	CAN: RPDO too short	
		Cause	A received RPDO does not contain the parameterised number of bytes.
		Measure	<p>The number of parameterised bytes does not match the number of bytes received.</p> <ul style="list-style-type: none"> • Check and correct parameterisation.
12-9	-	CAN: Protocol error	
		Cause	Faulty bus protocol.
		Measure	<ul style="list-style-type: none"> • Check the parameterisation of the selected CAN bus protocol.

Error group 13		CAN- bus timeout	
No.	Code	Message	Reaction
13-0	-	CAN: Timeout	
		Cause	Error message from manufacturer-specific protocol.
		Measure	<ul style="list-style-type: none"> • Check the CAN parameters.

Error group 14		Identification		
No.	Code	Message		Reaction
14-0	-	Automatic current controller identification: Insufficient intermediate circuit voltage		PSoff
		Cause	Current regulator parameters cannot be determined (insufficient supply).	
		Measure	The available intermediate circuit voltage is too low to carry out the measurement.	
14-1	-	Automatic current controller identification: Measurement cycle insufficient		PSoff
		Cause	Too few or too many measurement cycles required for the connected motor.	
		Measure	Automatic parameter definition providing a time constant that is outside the parameterisable value range. <ul style="list-style-type: none"> • The parameters must be manually optimised. 	
14-2	-	Automatic current controller identification: Power stage could not be enabled		PSoff
		Cause	The output stage has not been enabled.	
		Measure	<ul style="list-style-type: none"> • Check the connection of DIN4. 	
14-3	-	Automatic current controller identification: Output stage was switched off prematurely		PSoff
		Cause	Output stage enable was switched off while identification was in progress.	
		Measure	<ul style="list-style-type: none"> • Check the sequence control. 	
14-5	-	Automatic angle encoder identification: Zero pulse could not be found		PSoff
		Cause	The zero pulse could not be found following execution of the maximum permissible number of electrical revolutions.	
		Measure	<ul style="list-style-type: none"> • Check the zero pulse signal. • Angle encoder parameterised correctly? 	
14-6	-	Automatic angle encoder identification: Faulty Hall signals		PSoff
		Cause	Hall signals faulty or invalid. The pulse train or segmenting of the Hall signals is inappropriate.	
		Measure	<ul style="list-style-type: none"> • Check connection. • Refer to the technical data to check whether the encoder shows three Hall signals with 1205 or 605 segments; if necessary, contact Technical Support. 	

Error group 14		Identification	
No.	Code	Message	Reaction
14-7	-	Automatic angle encode identification: Identification not possible	
		Cause	Angle encoder at a standstill.
		Measure	<ul style="list-style-type: none"> • Ensure sufficient intermediate circuit voltage. • Encoder cable connected to the right motor? • Motor blocked, e.g. holding brake does not release?
14-8	-	Automatic angle encoder identification: Invalid number of pairs of poles	
		Cause	The calculated number of pole pairs lies outside the parameterisable range.
		Measure	<ul style="list-style-type: none"> • Compare result with the technical data specifications for the motor. • Check the parameterised number of lines.

Error group 15		Invalid operation	
No.	Code	Message	Reaction
15-0	6185h	Division by zero	
		Cause	Internal firmware error. Division by 0 when using the math library.
		Measure	<ul style="list-style-type: none"> • Load factory settings. • Check the firmware to make sure that approved firmware has been loaded.
15-1	6186h	Mathematical overflow during division	
		Cause	Internal firmware error. Overflow when using the math library.
		Measure	<ul style="list-style-type: none"> • Load factory settings. • Check the firmware to make sure that approved firmware has been loaded.
15-2	-	Mathematical underflow	
		Cause	Internal firmware error. Internal correction factors could not be calculated.
		Measure	<ul style="list-style-type: none"> • Check the setting of the factor group for extreme values and change, if necessary.

Error group 16		Internal error	
No.	Code	Message	Reaction
16-0	6181h	Error during program execution	
		Cause	Internal firmware error. Error during program execution. Illegal CPU command found in the program sequence.
		Measure	<ul style="list-style-type: none"> • In case of repetition, load firmware again. If the error occurs repeatedly, the hardware is defective.

Error group 16		Internal error	
No.	Code	Message	Reaction
16-1	6182h	Illegal interrupt	
		Cause	Error during program execution. An unused IRQ vector was used by the CPU.
		Measure	<ul style="list-style-type: none"> In case of repetition, load firmware again. If the error occurs repeatedly, the hardware is defective.
16-2	6187h	Initialisation error	
		Cause	Error in initialising the default parameters.
		Measure	<ul style="list-style-type: none"> In case of repetition, load firmware again. If the error occurs repeatedly, the hardware is defective.
16-3	6183h	Unexpected state	
		Cause	Error during periphery access within the CPU or error in the program sequence (illegal branching in case structures).
		Measure	<ul style="list-style-type: none"> In case of repetition, load firmware again. If the error occurs repeatedly, the hardware is defective.

Error group 17		Following error exceeded	
No.	Code	Message	Reaction
17-0	8611h	Following error limit exceeded	
		Cause	Comparison threshold for the limit value of the following error exceeded.
		Measure	<ul style="list-style-type: none"> Enlarge error window. Parameterise acceleration to be less. Motor overloaded (current limiter from I²t monitoring active?).
17-1	8611h	Encoder difference monitoring	
		Cause	Deviation between the actual position value and commutation position is too great. External angle encoder not connected or faulty?
		Measure	<ul style="list-style-type: none"> Deviation fluctuating, e.g. due to gear backlash; increase cut-off threshold if necessary. Check connection of the actual value encoder.

Error group 18		Temperature warning thresholds	
No.	Code	Message	Reaction
18-0	-	Analogue motor temperature	
		Cause	Motor temperature (analogue) more than 5° below T _{max} .
		Measure	<ul style="list-style-type: none"> Check parameterisation of current regulator and/or speed regulator. Motor permanently overloaded?

Error group 21		Current measurement	
No.	Code	Message	Reaction
21-0	5280h	Error 1 current measurement U	
		Cause	Offset for current measurement 1 phase U is too great. The controller carries out offset compensation of the current measurement every time its controller enable is issued. Tolerances that are too large result in an error.
		Measure	If the error occurs repeatedly, the hardware is defective.
21-1	5281h	Error 1 current measurement V	
		Cause	Offset for current measurement 1 phase V is too great.
		Measure	If the error occurs repeatedly, the hardware is defective.
21-2	5282h	Error 2 current measurement U	
		Cause	Offset for current measurement 2 phase U is too great.
		Measure	If the error occurs repeatedly, the hardware is defective.
21-3	5283h	Error 2 current measurement V	
		Cause	Offset for current measurement 2 phase V is too great.
		Measure	If the error occurs repeatedly, the hardware is defective.

Error group 22		PROFIBUS (only CMMP-AS-...-M3)	
No.	Code	Message	Reaction
22-0	-	PROFIBUS: Initialisation error	
		Cause	Faulty initialisation of the PROFIBUS interface. Interface faulty?
		Measure	<ul style="list-style-type: none"> Replace interface. Repair by the manufacturer may be an option.
22-2	-	PROFIBUS: Faulty communication	
		Cause	Malfunctions in communication.
		Measure	<ul style="list-style-type: none"> Check the configured slave address. Check the bus termination. Check the wiring.
22-3	-	PROFIBUS: Invalid slave address	
		Cause	Communication was started with slave address 126.
		Measure	<ul style="list-style-type: none"> Select a different slave address.
22-4	-	PROFIBUS: Conversion error	
		Cause	During conversion with the factor group, the range of values was exceeded. Mathematical error in the conversion of the physical units.
		Measure	The value ranges of the data and the physical units do not match. <ul style="list-style-type: none"> Check and correct.

Error group 23		Store/Restore actual position	
No.	Code	Message	Reaction
23-0	-	Actual position: No valid record available	
		Cause	<ul style="list-style-type: none"> – No entry stored after activation. – No position stored, because drive is not referenced. – Hardware reset occurred too early.
		Measure	Observe activation sequence: <ol style="list-style-type: none"> 1. Activate function. 2. Save and restart. 3. Execute homing.
23-1	-	Actual position: invalid checksum	
		Cause	Save operation can't be attained.
		Measure	Repeat activation. Observe activation sequence: <ol style="list-style-type: none"> 1. Activate function. 2. Save and restart. 3. Execute homing.
23-2	-	Actual position: Flash content inconsistent	
		Cause	Internal error during saving operation.
		Measure	Repeat activation. Observe activation sequence: <ol style="list-style-type: none"> 1. Activate function. 2. Save and restart. 3. Execute homing.

Error group 25		Device type/function	
No.	Code	Message	Reaction
25-0	6080h	Invalid device type	
		Cause	Device coding not recognised or invalid.
		Measure	This fault cannot be fixed by the user. <ul style="list-style-type: none"> • Send motor controller to the manufacturer.
25-1	6081h	Device type not supported	
		Cause	Device coding invalid, is not supported by the loaded firmware.
		Measure	<ul style="list-style-type: none"> • Load up-to-date firmware. • If newer firmware is not available, the problem may be a hardware defect. Send motor controller to the manufacturer.
25-2	6082h	Invalid hardware revision	
		Cause	The controller's hardware version is not supported by the loaded firmware.
		Measure	<ul style="list-style-type: none"> • Check the firmware version; update the firmware to a more recent version if necessary.

Error group 25		Device type/function	
No.	Code	Message	Reaction
25-3	6083h	Device with restricted functionality: Firmware cannot be executed	
		Cause	Device is not enabled for this function.
		Measure	Device is not enabled for the desired functionality and may need to be enabled by the manufacturer. The device must be sent in for this purpose.
25-4	-	Invalid power stage type	
		Cause	<ul style="list-style-type: none"> – Power section area in the EEPROM is unprogrammed. – Power section is not supported by the firmware.
		Measure	<ul style="list-style-type: none"> • Load appropriate firmware.

Error group 26		Internal data error	
No.	Code	Message	Reaction
26-0	5580h	Missing user parameter set	
		Cause	No valid user parameter set in the flash memory.
		Measure	<ul style="list-style-type: none"> • Load factory settings. If the error remains, the hardware may be defective.
26-1	5581h	Checksum error	
		Cause	Checksum error of a parameter set.
		Measure	<ul style="list-style-type: none"> • Load factory settings. If the error remains, the hardware may be defective.
26-2	5582h	Flash: Error when writing	
		Cause	Error when writing the internal flash memory.
		Measure	<ul style="list-style-type: none"> • Execute the last operation again. If the error appears again, the hardware may be faulty.
26-3	5583h	Flash: Error during deletion	
		Cause	Error during deletion of the internal flash memory.
		Measure	<ul style="list-style-type: none"> • Execute the last operation again. If the error appears again, the hardware may be faulty.
26-4	5584h	Flash: Internal flash error	
		Cause	The default parameter set is corrupted / data error in the FLASH area where the default parameter set is located.
		Measure	<ul style="list-style-type: none"> • Load firmware again. If the error appears again, the hardware may be faulty.
26-5	5585h	Missing calibration data	
		Cause	Factory-set calibration parameters incomplete / corrupted.
		Measure	This fault cannot be fixed by the user.

Error group 26		Internal data error	
No.	Code	Message	Reaction
26-6	5586h	Missing position data sets	
		Cause	Position data sets incomplete or corrupted.
		Measure	<ul style="list-style-type: none"> • Load the factory settings or • save the current parameters again so that the position data is written again.
26-7	-	Faulty data tables (CAM)	
		Cause	Data for the cam disc is corrupted.
		Measure	<ul style="list-style-type: none"> • Load factory settings. • Reload the parameter set if necessary. If the error persists, contact Technical Support.

Error group 27		Following error monitoring	
No.	Code	Message	Reaction
27-0	8611h	Following error warning threshold	
		Cause	<ul style="list-style-type: none"> – Motor overloaded? Check motor capacity. – Acceleration or braking ramps are set too steep. – Motor blocked? Commutation angle correct?
		Measure	<ul style="list-style-type: none"> • Check the parameterisation of the motor data. • Check the parameterisation of the following error.

Error group 28		Operating hour counter	
No.	Code	Message	Reaction
28-0	FF01h	Missing operating hour counter	
		Cause	No record for an operating hour counter could be found in the parameter block. A new operating hour counter was created. Occurs during initial start-up or a processor change.
		Measure	Warning only, no further action required.
28-1	FF02h	Operating hour counter: Write error	
		Cause	The data block in which the operating hour counter is stored could not be written to. Cause unknown; possibly problems with the hardware.
		Measure	Warning only, no further action required. If the error occurs again, the hardware may be faulty.

Error group 28		Operating hour counter	
No.	Code	Message	Reaction
28-2	FF03h	Operating hour counter corrected	
			configurable
		Cause	The operating hour counter has a backup copy. If the controller's 24 V power supply fails precisely when the operating hour counter is being updated, the written record may be corrupted. In such cases, the controller restores the operating hour counter from the intact backup copy when it switches back on.
	Measure	Warning only, no further action required.	
28-3	FF04h	Operating hour counter converted	
			configurable
		Cause	Firmware was loaded in which the operating hour counter has a different data format. The next time the controller is switched on, the old operating hour counter record is converted to the new format.
	Measure	Warning only, no further action required.	

Error group 29		Memory card	
No.	Code	Message	Reaction
29-0	-	Memory card not available	
			configurable
		Cause	This error is triggered in the following cases: <ul style="list-style-type: none"> – If an action should be carried out on the memory card (load or create DCO file, firmware download), but no memory card is plugged in. – The DIP switch S3 is set to ON, but no card is plugged in after the reset/restart.
	Measure	Insert appropriate memory card in the slot. Only if expressly desired!	
29-1	-	Memory card: Initialisation error	
			configurable
		Cause	This error is triggered in the following cases: <ul style="list-style-type: none"> – Memory card could not be initialised. Card type may not be supported! – File system not supported. – Error in connection with the shared memory.
	Measure	<ul style="list-style-type: none"> • Check card type used. • Connect memory card to a PC and format again. 	

Error group 29		Memory card	
No.	Code	Message	Reaction
29-2	-	Memory card: Data error	
			configurable
		Cause	<p>This error is triggered in the following cases:</p> <ul style="list-style-type: none"> – A load or storage process is already running, but a new load or storage process is requested. DCO file » Servo – The DCO file to be loaded has not been found. – The DCO file to be loaded is not appropriate for the device. – The DCO file to be loaded is defective. – Servo » DCO file – The memory card is write-protected. – Other error while saving the parameter set as a DCO file. – Error in creating the file INFO.TXT.
Measure	<ul style="list-style-type: none"> • Execute load or storage procedure again after waiting 5 seconds. • Connect memory card to a PC and check the files included. • Remove write protection from the memory card. 		
29-3	-	Memory card: Write error	
			configurable
		Cause	<ul style="list-style-type: none"> – This error is triggered while saving the DCO file or INFO.TXT file if the memory card is discovered to be already full. – The maximum file index (99) already exists. That is, all file indexes are assigned. No file name can be issued!
Measure	<ul style="list-style-type: none"> • Insert another memory card. • Change file names. 		
29-4	-	Memory card: Firmware download error	
			configurable
		Cause	<p>This error is triggered in the following cases:</p> <ul style="list-style-type: none"> – No firmware file on the memory card. – The firmware file is not appropriate for the device. – Other error during firmware download.
Measure	<ul style="list-style-type: none"> • Connect memory card to PC and transfer firmware file. 		

Error group 30		Internal conversion error	
No.	Code	Message	Reaction
30-0	6380h	Internal conversion error	
			PSoff
		Cause	Range exceeded for internal scaling factors, which are dependent on the parameterised controller cycle times.
Measure	<ul style="list-style-type: none"> • Check whether extremely short or extremely long cycle times were set in the parameters. 		

Error group 31		I²t monitoring	
No.	Code	Message	Reaction
31-0	2312h	Motor I²t	
		Cause	I ² t monitoring of the controller has been triggered. <ul style="list-style-type: none"> – Motor/mechanical system blocked or sluggish. – Motor under-sized?
		Measure	<ul style="list-style-type: none"> • Check the performance rating of the drive package.
31-1	2311h	Power stage I²t	
		Cause	The I ² t monitoring is being triggered frequently. <ul style="list-style-type: none"> – Motor controller does not have the required capacity? – Mechanical system sluggish?
		Measure	<ul style="list-style-type: none"> • Check design of the motor controller, • if necessary use a more powerful type. • Check the mechanical system.
31-2	2313h	PFC I²t	
		Cause	PFC power rating exceeded.
		Measure	<ul style="list-style-type: none"> • Parameterise operation without PFC (FCT).
31-3	2314h	Braking resistor I²t	
		Cause	– Overloading of the internal braking resistor.
		Measure	<ul style="list-style-type: none"> • Use external braking resistor. • Reduce resistance value or use resistor with higher pulse load.

Error group 32		Intermediate circuit fault	
No.	Code	Message	Reaction
32-0	3280h	Intermediate circuit charging time exceeded	
		Cause	The intermediate circuit could not be charged after the mains voltage was applied. <ul style="list-style-type: none"> – A fuse may be faulty, or – an internal braking resistor may be faulty, or, – in the case of operation with an external resistor, that resistor is not connected.
		Measure	<ul style="list-style-type: none"> • Check interface to the external braking resistor. • Alternatively, check whether the jumper for the internal braking resistor is in place. <p>If the interface is correct, the internal braking resistor or the built-in fuse is probably faulty. On-site repair is not possible.</p>
32-1	3281h	Undervoltage for active PFC	
		Cause	The PFC cannot be activated at all until an intermediate circuit voltage of about 130 V DC is reached.
		Measure	<ul style="list-style-type: none"> • Check the power supply.

Error group 32		Intermediate circuit fault	
No.	Code	Message	Reaction
32-5	3282h	Brake chopper overload	
			configurable
		Cause	The extent of utilisation of the brake chopper when quick discharge began was already in the range above 100%. Quick discharge took the brake chopper to the maximum load limit and was prevented/aborted.
	Measure	No action required.	
32-6	3283h	Intermediate circuit discharge time exceeded	
			configurable
		Cause	Intermediate circuit could not be quickly discharged. The internal braking resistor may be faulty or, in the case of operation with an external resistor, that resistor is not connected.
	Measure	<ul style="list-style-type: none"> • Check interface to the external braking resistor. • Alternatively, check whether the jumper for the internal braking resistor is in place. If the internal resistor has been activated and the jumper has been set correctly, the internal braking resistor is probably faulty.	
32-7	3284h	Power supply missing for controller enable	
			configurable
		Cause	Controller enable was issued when the intermediate circuit was still in its charging phase after mains voltage was applied and the mains relay was not yet activated. The drive cannot be enabled in this phase, because the drive is not yet firmly connected to the mains (through the mains relay).
	Measure	<ul style="list-style-type: none"> • In the application, check whether the mains supply and controller enable signals were sent quickly one after the other. 	
32-8	3285h	Power supply failure during controller enable	
			QStop
		Cause	Interruptions / failure in the power supply while the controller enable was activated.
	Measure	<ul style="list-style-type: none"> • Check the power supply. 	
32-9	3286h	Phase failure	
			QStop
		Cause	Failure of one or more phases (only in the case of three-phase supply).
	Measure	<ul style="list-style-type: none"> • Check the power supply. 	

Error group 33		Encoder emulation following error	
No.	Code	Message	Reaction
33-0	8A87h	Encoder emulation following error	
			configurable
		Cause	The critical frequency for encoder emulation was exceeded (see manual) and the emulated angle at [X11] was no longer able to follow. Can occur if very high numbers of lines are programmed for [X11] and the drive reaches high speeds.
	Measure	<ul style="list-style-type: none"> • Check whether the parameterised number of lines may be too high for the speed being represented. • Reduce the number of lines if necessary. 	

Error group 34		Fieldbus synchronisation	
No.	Code	Message	Reaction
34-0	8780h	No synchronisation via field bus	
			configurable
		Cause	When activating the interpolated position mode, the controller could not be synchronised to the fieldbus. <ul style="list-style-type: none"> – The synchronisation messages from the master may have failed or – the IPO interval is not correctly set to the synchronisation interval of the fieldbus.
	Measure	<ul style="list-style-type: none"> • Check the settings for the controller cycle times. 	
34-1	8781h	Field bus synchronisation error	
			configurable
		Cause	<ul style="list-style-type: none"> – Synchronisation via fieldbus messages during ongoing operation (interpolated position mode) has failed. – Synchronisation messages from master failed? – Synchronisation interval (IPO interval) parameterised too small/too large?
	Measure	<ul style="list-style-type: none"> • Check the settings for the controller cycle times. 	

Error group 35		Linear motor	
No.	Code	Message	Reaction
35-0	8480h	Linear motor spinning protection	
		Cause	Encoder signals are faulty. The motor may be racing ("spinning") because the commutation position has been shifted by the faulty encoder signals.
		Measure	<ul style="list-style-type: none"> • Check that the installation conforms to the EMC recommendations. • In the case of linear motors with inductive/optical encoders with separately mounted measuring tape and measuring head, check the mechanical clearance. • In the case of linear motors with inductive encoders, make sure that the magnetic field of the magnets or the motor winding does not leak into the measuring head (this effect usually occurs when high accelerations = high motor current).
35-5	-	Error during the determination of the commutation position	
		Cause	The rotor position could not be clearly identified. <ul style="list-style-type: none"> – The selected method may be inappropriate. – The selected motor current for the identification may not be set appropriately.
		Measure	<ul style="list-style-type: none"> • Check the method for determining the commutation position ➔ Additional information.
		Additional information	<p>Information about determining commutation position:</p> <ul style="list-style-type: none"> a) The alignment method is inappropriate for locked or sluggish drives or drives capable of low-frequency oscillation. b) The microstep method is appropriate for air-core and iron-core motors. As only very small movements are carried out, it works even when the drive is on elastic stops or is locked but can still be moved elastically to some extent. Due to the high excitation frequency, however, the method is very susceptible to oscillations in the case of poorly damped drives. In such cases, you can attempt to reduce the excitation current (%). c) The saturation method uses local occurrences of saturation in the iron of the motor. Recommended for locked drives. Air-core drives are by definition not suitable for this method. If the (iron-core) drive moves too much when locating the commutation position, the measurement result may be adulterated. If this is the case, reduce the excitation current. In the opposite case, if the drive does not move, the excitation current may not be strong enough, causing the saturation to be insufficient.

Error group 36		Parameter	
No.	Code	Message	Reaction
36-0	6320h	Parameter was limited	
		Cause	An attempt was made to write a value which was outside the permitted limits, so the value was limited.
		Measure	<ul style="list-style-type: none"> • Check the user parameter set.
36-1	6320h	Parameter was not accepted	
		Cause	An attempt was made to write to an object which is "read only" or is not write-capable in the current status (e.g. with controller enable active).
		Measure	<ul style="list-style-type: none"> • Check the user parameter set.

Error group 40		Software limits	
No.	Code	Message	Reaction
40-0	8612h	Negative software limit reached	
		Cause	The position setpoint has reached or exceeded the negative software limit switch.
		Measure	<ul style="list-style-type: none"> • Check target data. • Check the positioning range.
40-1	8612h	Positive software limit reached	
		Cause	The position setpoint has reached or exceeded the positive software limit switch.
		Measure	<ul style="list-style-type: none"> • Check target data. • Check the positioning range.
40-2	8612h	Positioning beyond negative software limit suppressed	
		Cause	Start of a positioning task was suppressed because the target lies behind the negative software limit switch.
		Measure	<ul style="list-style-type: none"> • Check target data. • Check the positioning range.
40-3	8612h	Positioning beyond positive software limit suppressed	
		Cause	The start of a positioning task was suppressed because the target lies behind the positive software limit switch.
		Measure	<ul style="list-style-type: none"> • Check target data. • Check the positioning range.

Error group 41		Record sequence	
No.	Code	Message	Reaction
41-0	-	Record sequence: Synchronisation error	
		Cause	Start of synchronisation without prior sampling pulse.
		Measure	<ul style="list-style-type: none"> • Check parameterisation of the lead section.

Error group 42		Positioning	
No.	Code	Message	Reaction
42-0	8680h	Positioning: Drive stops automatically because there is no follow-up positioning	
		Cause	The positioning target cannot be reached through the positioning or edge conditions options.
		Measure	<ul style="list-style-type: none"> • Check the parameterisation of the relevant position sets.
42-1	8681h	Positioning: Drive stops as rotation reversal is not allowed	
		Cause	The positioning target cannot be reached through the positioning or edge conditions options.
		Measure	<ul style="list-style-type: none"> • Check the parameterisation of the relevant position sets.
42-2	8682h	Positioning: Illegal rotation reversal after "stop"	
		Cause	The positioning target cannot be reached through the positioning or edge conditions options.
		Measure	<ul style="list-style-type: none"> • Check the parameterisation of the relevant position sets.
42-3	-	Start positioning rejected: Wrong mode of operation	
		Cause	Switching of the operating mode by means of the position record was not possible.
		Measure	<ul style="list-style-type: none"> • Check the parameterisation of the relevant position sets.
42-4	-	Please enforce homing run!	
		Cause	A normal position record was started, but the drive needs a valid reference position before starting.
		Measure	<ul style="list-style-type: none"> • Execute new homing.
42-5	-	Rotary axis: Direction of rotation is not allowed	
		Cause	<ul style="list-style-type: none"> – The positioning target cannot be reached through the positioning or edge conditions options. – The calculated direction of rotation is not permitted for the modulo positioning in the set mode.
		Measure	<ul style="list-style-type: none"> • Check the chosen mode.
42-9	-	Error at starting the positioning	
		Cause	<ul style="list-style-type: none"> – Acceleration limit value exceeded. – Position record blocked.
		Measure	<ul style="list-style-type: none"> • Check parameterisation and sequence control, correct if necessary.

Error group 43		Hardware limit switch	
No.	Code	Message	Reaction
43-0	8081h	Limit switch: Negative setpoint value blocked	
		Cause	Negative hardware limit switch reached.
		Measure	<ul style="list-style-type: none"> • Check parameterisation, wiring and limit switches.

Error group 43		Hardware limit switch	
No.	Code	Message	Reaction
43-1	8082h	Limit switch: Positive setpoint value blocked	
		Cause	Positive hardware limit switch reached.
		Measure	<ul style="list-style-type: none"> • Check parameterisation, wiring and limit switches.
43-2	8083h	Limit switch: Positioning suppressed	
		Cause	<ul style="list-style-type: none"> – The drive has left the designated range of motion. – Technical defect in the system?
		Measure	<ul style="list-style-type: none"> • Check the designated range of motion.

Error group 44		Cam disc error	
No.	Code	Message	Reaction
44-0	-	Error in Cam data tables	
		Cause	The cam disc to be started is not available.
		Measure	<ul style="list-style-type: none"> • Check transferred cam disc no. • Correct parameterisation. • Correct programming.
44-1	-	Cam Disc: General error homing	
		Cause	– Start of a cam disc, but the drive is not yet referenced.
		Measure	<ul style="list-style-type: none"> • Carry out homing.
		Cause	– Start homing with active cam disk.
		Measure	<ul style="list-style-type: none"> • Deactivate cam disc. Then restart cam disc, if necessary.

Error group 47		Setting-up	
No.	Code	Message	Reaction
47-0	-	Timeout setup mode	
		Cause	Failed to fall below the speed required for setting-up within time allowed.
		Measure	Check processing of the request on the control side.

Error group 48		Homing required	
No.	Code	Message	Reaction
48-0	-	Please enforce homing run!	
		Cause	An attempt is being made to switch to the speed control or torque control operating mode or to issue the controller enable in one of these operating modes, although the drive requires a valid reference position for this.
		Measure	<ul style="list-style-type: none"> • Carry out homing.

Error group 49		DCO file	
No.	Code	Message	Reaction
49-1	-	DCO file: wrong password	
		Cause	<ul style="list-style-type: none"> – Parameter file with wrong password shall be loaded. – Old parameter file (no password defined) should be loaded in protected motor controller.
		Measure	Loading only possible with valid password.

Error group 50		CAN communication	
No.	Code	Message	Reaction
50-0	-	Too many synchronous PDOs	
		Cause	<p>More PDOs have been activated than can be processed in the underlying SYNC interval.</p> <p>This message also appears if only one PDO is to be transmitted synchronously, but a high number of other PDOs with a different transmission type have been activated.</p>
		Measure	<ul style="list-style-type: none"> • Check the activation of PDOs. <p>If the configuration is appropriate, the warning can be suppressed using error management.</p> <ul style="list-style-type: none"> • Extend the synchronisation interval.
50-1	-	SDO error has occurred	
		Cause	<p>An SDO transfer has caused an SDO abort.</p> <ul style="list-style-type: none"> – Data exceed the range of values. – Access to non-existent object.
		Measure	<ul style="list-style-type: none"> • Check the command sent.

Error group 51		Safety module/function	
No.	Code	Message	Reaction
51-0	8091h	Unknown Safety module or driver supply defective	
			PSoff
		Cause	CMMP-AS-...-M0: Internal voltage error of the STO circuit.
		Measure	<ul style="list-style-type: none"> Protection circuit defective. No action possible, please contact Festo. If possible, replace with another motor controller.
		Cause	CMMP-AS-...-M3: Internal voltage error of the safety module or micro switch module.
		Measure	<ul style="list-style-type: none"> Module presumably defective. If possible, replace with another basic unit.
51-2	8093h	Safety module: Dissimilar module type	
			PSoff
		Cause	Type or version of the module does not fit the design.
		Measure	<ul style="list-style-type: none"> Check whether correct module type and correct version are being used. With module replacement: module type not yet designed. Accept currently integrated safety or micro switch module.
		Cause	
		Measure	
51-3	8094h	Safety module: Dissimilar module version	
			PSoff
		Cause	Module type or revision are not supported.
		Measure	<ul style="list-style-type: none"> Mount a module that is compatible to the given hardware and firmware. Load firmware that is appropriate for the module, see type designation on the module.
		Cause	The module type is correct but the module version is not supported by the basic unit.
		Measure	<ul style="list-style-type: none"> Check module version; if possible use module of same version after replacement. Install suitable safety or micro switch module for the firmware and hardware. If only a module with a more recent version is available: Load firmware that is appropriate for the module, see type designation on the module.

Error group 51		Safety module/function	
No.	Code	Message	Reaction
51-4	8095h	Safety module: SSIO communication error	
			PSoff
		Cause	Fault in the internal communication connection between the basic unit and the safety module.
	Measure	<ul style="list-style-type: none"> • This error may occur if a CAMC-G-S3 was designed into the basic unit but a different module type was plugged in. • Load a firmware suitable for the safety or micro switch module, see type designation on the module. 	
51-5	8096h	Safety module: Brake control error	
			PSoff
		Cause	Internal hardware error (brake actuation control signals) of the safety module or micro switch module.
		Measure	<ul style="list-style-type: none"> • Module presumably defective. If possible, replace with another module.
		Cause	Error in brake driver circuit section in the basic unit.
	Measure	<ul style="list-style-type: none"> • Module presumably defective. If possible, replace with another basic unit. 	
51-6	8097h	Safety module: Dissimilar serial number	
			PSoff
		Cause	Serial number of currently connected safety module is different from the stored serial number.
	Measure	Error only occurs after replacement of the CAMC-G-S3. <ul style="list-style-type: none"> • With module replacement: module type not yet designed. Accept currently integrated CAMC-G-S3. 	

Error group 52		Safety function	
No.	Code	Message	Reaction
52-1	8099h	Safety function: Discrepancy time expired	
			PSoff
		Cause	– Control ports STO-A and STO-B are not actuated simultaneously.
		Measure	<ul style="list-style-type: none"> • Check discrepancy time.
		Cause	– Control ports STO-A and STO-B are not wired in the same way.
		Measure	<ul style="list-style-type: none"> • Check discrepancy time.
		Cause	Upper and lower switch supply not simultaneously activated (discrepancy time exceeded) <ul style="list-style-type: none"> – Error in control / external circuitry of safety module. – Error in safety module.
	Measure	<ul style="list-style-type: none"> • Check circuitry of the safety module – are the inputs STO-A and STO-B switched off on two channels and simultaneously? • Replace safety module if you suspect it is faulty. 	

Error group 52		Safety function	
No.	Code	Message	Reaction
52-2	809Ah	Safety function: Failure of driver supply with active PWM control	
		Cause	This error message does not occur with devices delivered from the factory. It can occur with use of a user-specific device firmware.
		Measure	<ul style="list-style-type: none"> The safe status was requested with enabled power output stage. Check inclusion in the safety-oriented interface.
52-3	809Bh	Safety module: Overlapping velocity limits in basic unit	
		Cause	– Basic unit reports error if the currently requested direction of movement is not possible because the safety module has blocked the setpoint value in this direction.
		Measure	<p>Error may occur in connection with the SSF if an asymmetrical speed window is used where one limit is set to zero.</p> <p>In this case, the error occurs when the basic unit moves in the "blocked" direction in the Positioning mode.</p> <ul style="list-style-type: none"> Check application and change if necessary.

Error group 53		Violation of Safety conditions (only CMMP-AS-...-M3)	
No.	Code	Message	Reaction
53-0	80A1h	USF0: Safety condition violated	
		Cause	– Violation of monitored speed limits of the SSF0 in operation / when USF0 / SSF0 requested.
		Measure	<p>Check when the violation of the safety condition occurs:</p> <ol style="list-style-type: none"> During dynamic braking to the safe speed After the drive has reached the safe speed. <ul style="list-style-type: none"> With a) Critical check of braking ramp – record trace - can the drive follow the ramp? Change parameters for the braking ramp or start time / delay times for monitoring. With b) Check how far the current speed is from the monitored limit speed; increase distance if necessary (parameter in safety module) or correct speed specified by controller.
53-1	80A2h	USF1: Safety condition violated	
		Cause	– Violation of monitored speed limits of the SSF1 in operation / when USF1 / SSF1 requested.
		Measure	<ul style="list-style-type: none"> See USF0, error 53-0.
53-2	80A3h	USF2: Safety condition violated	
		Cause	– Violation of monitored speed limits of the SSF2 in operation / when USF2 / SSF2 requested.
		Measure	<ul style="list-style-type: none"> See USF0, error 53-0.

Error group 53		Violation of Safety conditions (only CMMP-AS-...-M3)	
No.	Code	Message	Reaction
53-3	80A4h	USF3: Safety condition violated	
		Cause	– Violation of monitored speed limits of the SSF3 in operation / when USF3 / SSF3 requested.
		Measure	• See USF0, error 53-0.
			configurable

Error group 54		Violation of Safety conditions (only CMMP-AS-...-M3)	
No.	Code	Message	Reaction
54-0	80AAh	SBC: Safety condition violated	
		Cause	– Brake should engage; no feedback received within the expected time.
		Measure	<ul style="list-style-type: none"> • Check how the feedback signal is configured – was the correct input selected for the feedback signal? • Does the feedback signal have the correct polarity? • Check whether the feedback signal is actually switching. • Is the parameterised delay time for the evaluation of the feedback signal appropriate to the brake used (measure switching time if necessary)?
			configurable
54-2	80ACh	SS2: Safety condition violated	
		Cause	– Actual speed outside permitted limits for too long.
		Measure	<p>Check when the violation of the safety condition occurs:</p> <p>a) During dynamic braking to zero.</p> <p>b) After the drive has reached zero speed.</p> <ul style="list-style-type: none"> • With a) Critical check of braking ramp – record trace - can the drive follow the ramp? Change parameters for the braking ramp or start time / delay times for monitoring. • With a) If the option "Trigger basic unit quick stop" is activated: Critical check of the basic unit's quick stop ramp. • With b) Check whether the drive continues to oscillate after reaching the zero speed or remains still and stable – increase monitoring tolerance time if necessary. • With b) If the actual speed value is very noisy at rest. Check and if necessary adjust expert parameters for speed recording and detection of standstill.
			configurable

Error group 54		Violation of Safety conditions (only CMMP-AS-...-M3)	
No.	Code	Message	Reaction
54-3	80ADh	SOS: Safety condition violated	
			configurable
		Cause	<ul style="list-style-type: none"> – Angle encoder evaluation reports "Motor running" (actual speed exceeds limit). – Drive has rotated out of its position since reaching the safe state.
		Measure	<ul style="list-style-type: none"> • Check position tolerance for the SOS monitoring and increase if necessary, if this is permissible. • If the actual speed value is very noisy when at rest: Check and if necessary adjust expert parameters for speed recording and detection of standstill.
54-4	80AEh	SS1: Safety condition violated	
			configurable
		Cause	– Actual speed outside permitted limits for too long.
		Measure	<p>Check when the violation of the safety condition occurs:</p> <ol style="list-style-type: none"> a) During dynamic braking to zero. b) After the drive has reached zero speed. <ul style="list-style-type: none"> • With a) Critical check of braking ramp – record trace - can the drive follow the ramp? Change parameters for the braking ramp or start time / delay times for monitoring. • With a) If the option "Trigger basic unit quick stop" is activated: Critical check of the basic unit's quick stop ramp. • With b) Check whether the drive continues to oscillate after reaching the zero speed or remains still and stable – increase monitoring tolerance time if necessary. • With b) If the actual speed value is very noisy when at rest: Check and if necessary adjust expert parameters for speed recording and detection of standstill.
54-5	80AFh	STO: Safety condition violated	
			configurable
		Cause	– Internal hardware error (voltage error) of the safety module.
		Measure	• Module presumably defective. If possible, replace with another module.
		Cause	– Error in driver circuit section in the basic unit.
		Measure	• Module presumably defective. If possible, replace with another basic unit.
		Cause	– No feedback received from basic unit to indicate that output stage was switched off.
		Measure	• Check whether the error can be acknowledged and whether it occurs again upon a new STO request – if yes: basic unit is presumably faulty. If possible, replace with another basic unit.

Error group 54		Violation of Safety conditions (only CMMP-AS-...-M3)	
No.	Code	Message	Reaction
54-6	80B0h	SBC: Brake not released for > 24h	
		Cause	– Error occurs when SBC is requested and the brake has not been opened by the basic unit in the last 24 hours.
		Measure	<ul style="list-style-type: none"> • If the brake is actuated via the brake driver in the basic unit [X6]: The brake must be energised at least once within 24 V before the SBC request because the circuit breaker check can only be performed when the brake is switched on (energised). • Only if brake control takes place via DOUT4x and an external brake controller: Deactivate 24h monitoring in the SBC parameters if the external brake controller allows this.
54-7	80B1h	SOS: SOS requested for > 24 h	
		Cause	– If SOS is requested for more than 24 hours, the error is triggered.
		Measure	• Terminate SOS occasionally; move axis once occasionally.

Error group 55		Measuring of actual value 1 (only CMMP-AS-...-M3)	
No.	Code	Message	Reaction
55-0	80C1h	No actual speed / position value available or standstill for > 24 h	
		Cause	<ul style="list-style-type: none"> – Subsequent error when a position encoder fails. – Safety function SSF, SS1, SS2 or SOS requested and actual speed value is not valid.
		Measure	• Check the function of the position encoder(s) (see following error).
55-1	80C2h	SINCOS encoder [X2B] - signal error	
		Cause	<ul style="list-style-type: none"> – Vector length $\sin^2 + \cos^2$ is outside the permissible range. – The amplitude of one of the two signals is outside the permissible range. – Offset between analogue and digital signal is greater than 1 quadrant.
		Measure	<p>Error may occur with SIN/COS and Hiperface encoders.</p> <ul style="list-style-type: none"> • Check the position encoder. • Check the connection wiring (broken wire, short between two signals or signal / screening). • Check the supply voltage for the position encoder. • Check the motor cable / screening on motor and drive side – EMC problems may trigger the error.

Error group 55		Measuring of actual value 1 (only CMMP-AS-...-M3)	
No.	Code	Message	Reaction
55-2	80C3h	SINCOS encoder [X2B] - standstill > 24 h	
			configurable
		Cause	<ul style="list-style-type: none"> – Input signals of the SinCos encoder have not changed by a minimum amount for 24 hours (when safety function is requested).
		Measure	<ul style="list-style-type: none"> • Terminate SS1, SS2 or SOS occasionally; move axis once occasionally.
55-3	80C4h	Resolver [X2A] - signal error	
			configurable
		Cause	<ul style="list-style-type: none"> – Vector length $\sin^2 + \cos^2$ is outside the permissible range. – The amplitude of one of the two signals is outside the permissible range. – Input signal is static (same values to right and left of maximum).
		Measure	<ul style="list-style-type: none"> • Check the resolver. • Check the connection wiring (broken wire, short between two signals or signal / screening). • Check for failure of the exciter signal • Check the motor cable / screening on motor and drive side – EMC problems may trigger the error.
55-4	-	EnDat encoder [X2B] - sensor error	
			configurable
		Cause	<ul style="list-style-type: none"> – Communication error between safety module and the ENDAT encoder. – Error message of the ENDAT encoder present.
		Measure	<ul style="list-style-type: none"> • Check the ENDAT encoder. • Check the connection wiring (broken wire, short between two signals or signal / screening). • Check the supply voltage for the ENDAT encoder. • Check of the motor cable / screening on motor and drive side – EMC problems may trigger the error.
55-5	-	EnDat encoder [X2B] - wrong sensor / type	
			configurable
		Cause	<ul style="list-style-type: none"> – Number of lines does not correspond to parameterisation. – Serial no. Does not correspond to parameterisation. – Sensor type does not correspond to parameterisation.
		Measure	<ul style="list-style-type: none"> • Check the parameterisation. • Use only approved encoders.
55-6	80C5h	Incremental encoder X10 - signal error	
			configurable
		Cause	<ul style="list-style-type: none"> – Signal error at incremental encoder.
		Measure	<ul style="list-style-type: none"> • Check the connection wiring (broken wire, short between two signals or signal / screening). • Check the motor cable / screening on motor and drive side – EMC problems may trigger the error.

Error group 55		Measuring of actual value 1 (only CMMP-AS-...-M3)	
No.	Code	Message	Reaction
55-7	80C6h	Other encoder [X2B] - Faulty angle information	
			configurable
		Cause	<ul style="list-style-type: none"> - "Angle faulty" message is sent from basic unit when status lasts for longer than the allowed time. - Encoder at X2B is evaluated by the basic unit, - encoder is faulty.
		Measure	<ul style="list-style-type: none"> • Check the position encoder at X2B. • Check the connection wiring (broken wire, short between two signals or signal / screening). • Check the supply voltage for the ENDAT encoder. • Check the motor cable / screening on motor and drive side – EMC problems may trigger the error?
55-8	-	Impermissible acceleration detected	
			configurable
		Cause	<ul style="list-style-type: none"> - Encoder error. - EMC problems may trigger the error. - Too high acceleration values. - Max. acceleration is parameterised too low. - Snap angle after homing in the transmitted data from the base unit to the safety module.
		Measure	<ul style="list-style-type: none"> • Check the connection wiring (broken wire, short between two signals or signal / screening). • Check the target values given by PLC for invalid acceleration values (P06.07)? • Check the parameterised max. values for correctness. The upper limit (P06.07) should be at least 30...50% above the max. process values. • With snap angle in the data from the base device: Acknowledge it one times.

Error group 56		Measuring of actual value 2 (only CMMP-AS-...-M3)	
No.	Code	Message	Reaction
56-8	80D1h	Speed / angle difference encoder 1 - 2	
			configurable
		Cause	<ul style="list-style-type: none"> - Speed difference between encoders 1 and 2 of one μC for longer than allowed time outside the permissible range. - Angle difference between encoders 1 and 2 of one μC for longer than allowed time outside the permissible range.
		Measure	<ul style="list-style-type: none"> • Problem may occur if two position encoders are used in the system and they are not "rigidly coupled". • Check for elasticity or looseness, improve mechanical system. • Adjust the expert parameters for the position comparison if this is acceptable from an application point of view.

Error group 56		Measuring of actual value 2 (only CMMP-AS-...-M3)	
No.	Code	Message	Reaction
56-9	-	Error Cross comparison encoder evaluation	
		Cause	Cross-comparison between μ C1 and μ C2 has detected an angle difference or speed difference or difference in capture times for the position encoders.
		Measure	<ul style="list-style-type: none"> Timing disrupted. If the error occurs against after a reset, the safety module is presumably faulty.

Error group 57		Input/output error (only CMMP-AS-...-M3)	
No.	Code	Message	Reaction
57-0	80E1h	Self test I/O error (internal/external)	
		Cause	<ul style="list-style-type: none"> – Error at outputs DOUT40 ... DOUT42 (detection by test pulses). – Internal error of digital inputs DIN40 ... DIN49 (via internal test signals). – Error at brake output at X6 (signalling, detection by test pulses). – Internal error of brake output (via internal test signals). – Internal error of digital outputs DOUT40 – DOUT42 (via internal test signals).
		Measure	<ul style="list-style-type: none"> • Check the connection wiring for the digital outputs DOUT40 ... DOUT42 (short circuit, cross circuit, etc.). • Check the connection wiring for the brake (short circuit, cross circuit, etc.). • Brake connection: The error may occur with longer motor cables if: <ol style="list-style-type: none"> 1. The brake output X6 was configured for the brake (this is the case with factory settings!) and 2. A motor without a holding brake is used and the brake connection lines in the motor cable are terminated at X6. In this case: Disconnect the brake connection lines at X6. • If there is not error in the connection wiring, there may be an internal error in the module (check by swapping the module).

Error group 57		Input/output error (only CMMP-AS-...-M3)	
No.	Code	Message	Reaction
57-1	80E2h	Digital inputs - wrong signal level	
		Cause	Exceeding / violation of discrepancy time with multi-channel inputs (DIN40 ... DIN43, two-handed control device, mode selector switch).
		Measure	<ul style="list-style-type: none"> • Check the external active and passive sensors – do they switch on two channels and simultaneously (within the parameterised discrepancy time). • Two-handed control device: Check how the device is operated by the user – are both pushbuttons pressed within the discrepancy time? Give training if necessary. • Check the set discrepancy times – are they sufficient?
57-2	-	Digital inputs - missing test pulse	
		Cause	– One or more inputs (DIN40 ... DIN49) were configured for the evaluation of test pulses from the outputs (DOUT40 ... DOUT 42). The test pulses from DOUTx do not arrive at DIN4x.
		Measure	<ul style="list-style-type: none"> • Check the wiring (shorts after 0 V, 24 V, cross circuits). • Check the assignment – correct output selected / configured for test pulse?
57-6	-	Electronic temperature too high	
		Cause	– The safety module's temperature monitor has been triggered; the temperature of μ C1 or μ C2 was below -20° or above $+75^{\circ}$ C.
		Measure	<ul style="list-style-type: none"> • Check the operating conditions (ambient temperature, control cabinet temperature, installation situation in the control cabinet). • If the motor controller is experiencing high thermal load (high control cabinet temperature, high power consumption / output to motor, large number of occupied slots), a motor controller of the next highest output level should be used.

Error group 58		Error during communication / parameterisation (only CMMP-AS-...-M3)	
No.	Code	Message	Reaction
58-0	80E9h	Plausibility check parameters	
		Cause	The plausibility check in the safety module produced errors, e.g. an invalid angle encoder configuration; the error is triggered when a validation code is requested by the SafetyTool and when parameters are backed up in the safety module.
		Measure	<ul style="list-style-type: none"> • Note instructions for SafetyTool for complete validation; critically check parameterisation.

Error group 58		Error during communication / parameterisation (only CMMP-AS-...-M3)	
No.	Code	Message	Reaction
58-1	-	General error parameterisation	
			configurable
		Cause	Parameterisation session for more than 8 h active. The safety module aborted the parameterisation session. The error message is stored in the diagnostic memory.
		Measure	<ul style="list-style-type: none"> Finish the parameterisation session before the 8 h limit or break and restart the session.
58-4	80E9h	Buffer internal communication	
			configurable
		Cause	<ul style="list-style-type: none"> Communication connection faulty. Timeout / data error / incorrect sequence (packet counter) in data transmission between the basic unit and safety module. Too much data traffic, new requests are being sent to safety module before old ones have been responded to.
		Measure	<ul style="list-style-type: none"> Check communication interfaces, wiring, screening, etc. Check whether other devices have read access to the motor controller and safety module during a parameterisation session - this may overload the communication connection. Check whether the firmware versions of the safety module and basic unit and the versions of the FCT plugin and SafetyTool are compatible.
58-5	80EAh	Communication safety module - base unit	
			configurable
		Cause	<ul style="list-style-type: none"> Packet counter error during transmission $\mu\text{C1} \leftrightarrow \mu\text{C2}$. Checksum error during transmission $\mu\text{C1} \leftrightarrow \mu\text{C2}$.
		Measure	<ul style="list-style-type: none"> Internal malfunction in the motor controller. Check whether the firmware versions of the safety module and basic unit and the versions of the FCT plugin and SafetyTool are compatible.

Error group 58		Error during communication / parameterisation (only CMMP-AS-...-M3)	
No.	Code	Message	Reaction
58-6	80EBh	Cross comparison error processor 1 - 2	
			configurable
		Cause	<p>Timeout during cross-comparison (no data) or cross-comparison faulty (data for μC1 and μC2 are different).</p> <ul style="list-style-type: none"> – Error in cross-comparison for digital IO. – Error in cross-comparison for analogue input. – Error in cross-comparison for internal operating voltage measurement (5 V, 3.3 V, 24 V) and reference voltage (2.5 V). – Error in cross-comparison for SIN/COS angle encoder analogue values. – Error in cross-comparison for programme sequence monitoring. – Error in cross-comparison for interrupt counter. – Error in cross-comparison for input map. – Error in cross-comparison for violation of safety conditions. – Error in cross-comparison for temperature measurement.
		Measure	<p>This is an internal error in the module that should not occur during operation.</p> <ul style="list-style-type: none"> • Check the operating conditions (temperature, air humidity, condensation). • Check the EMC – wiring as specified, screening concept, are there any external interference sources? • Safety module may be faulty – is error resolved after replacing the module? • Check whether a new firmware for the motor controller or a new version of the safety module is available from the manufacturer.

Error group 59		Internal safety module error (only CMMP-AS-...-M3)	
No.	Code	Message	Reaction
59-1	80F1h	Failsafe supply/safe pulse inhibitor	
			configurable
		Cause	– Internal error in module in failsafe supply circuit section or in the driver supply for the upper and lower switches.
		Measure	• Module faulty, replace.
59-2	80F2h	External voltage supply error	
			configurable
		Cause	– Reference voltage 2.5V outside tolerance. – Logic supply overvoltage +24 V detected.
		Measure	• Module faulty, replace.
59-3	80F3h	Internal voltage supply error	
			configurable
		Cause	– Voltage (internal 3.3 V, 5 V, ADU reference) outside the permissible range.
		Measure	• Module faulty, replace.

Error group 59		Internal safety module error (only CMMP-AS-...-M3)	
No.	Code	Message	Reaction
59-4	80F4h	Error management: Too many errors	
		Cause	– Too many errors have occurred simultaneously.
		Measure	<ul style="list-style-type: none"> • Clarify: What is the status of the installed safety module - does it contain a valid parameter set? • Read out and analyse the log file of the basic unit via FCT. • Remedy causes of error step by step. • Install safety module with "delivery status" and perform commissioning of basic unit. • If this is not available: Set factory settings in the safety module, then copy data from the basic unit and perform complete validation. Check whether the error occurs again.
59-5	80F5h	Diagnosis Memory writing error	
		Cause	Subsequent error if internal communication is disrupted. – Basic unit not ready for operation, faulty or memory error.
		Measure	<ul style="list-style-type: none"> • Check the function of the basic unit • Generate an error in the basic unit, e.g. by unplugging the position encoder, and check whether the basic unit writes an entry to the log file. • Module or basic unit faulty; replace.
59-6	80F6h	Error on saving parameter set	
		Cause	– Voltage interruption / power off while parameters were being saved.
		Measure	<ul style="list-style-type: none"> • Maintain a voltage supply of 24 V throughout the parameterisation session. • Once the error has occurred, parameterise the module again and validate the parameter set again.
59-7	80F7h	FLASH checksum error	
		Cause	<ul style="list-style-type: none"> – Voltage interruption / power off while parameters were being saved. – Flash memory in safety module corrupted (e.g. by extreme malfunctions).
		Measure	Check whether the error recurs after a reset. If it does: <ul style="list-style-type: none"> • Parameterise the module again and validate the parameter set again. If the error remains: • Module is faulty; replace.

Error group 59		Internal safety module error (only CMMP-AS-...-M3)	
No.	Code	Message	Reaction
59-8	80F8h	Internal monitoring processor 1 - 2	
		Cause	<ul style="list-style-type: none"> – Serious internal error in the safety module: Error detected while dynamising internal signals – Disrupted programme sequence, stack error or OP code test failed, processor exception / interrupt.
		Measure	Check whether the error recurs after a reset. If it does: <ul style="list-style-type: none"> • Module is faulty; replace.
59-9	80F9h	Other unexpected error	
		Cause	Triggering of internal programme sequence monitoring.
		Measure	<ul style="list-style-type: none"> • Check the firmware version of the basic unit and the version of the safety module – update available? • Safety module faulty; replace.

Error group 62		EtherCAT (only CMMP-AS-...-M3)	
No.	Code	Message	Reaction
62-0	-	EtherCAT: Initialisation error	
		Cause	No EtherCAT bus present.
		Measure	<ul style="list-style-type: none"> • Switch on the EtherCAT master. • Check the wiring.
62-1	-	EtherCAT: Initialisation error	
		Cause	Error in the hardware.
		Measure	<ul style="list-style-type: none"> • Replace the interface and send it to the manufacturer for inspection.
62-2	-	EtherCAT: Protocol error	
		Cause	CAN over EtherCAT is not in use.
		Measure	<ul style="list-style-type: none"> • Incorrect protocol. • EtherCAT bus wiring fault.
62-3	-	EtherCAT: Invalid RPDO length	
		Cause	Sync manager 2 buffer size is too large.
		Measure	<ul style="list-style-type: none"> • Check the RPDO configuration of the motor controller and the higher-level control system.
62-4	-	EtherCAT: Invalid TPDO length	
		Cause	Sync manager 3 buffer size is too large.
		Measure	<ul style="list-style-type: none"> • Check the TPDO configuration of the motor controller and the higher-level control system.
62-5	-	EtherCAT: Erroneous cyclic communication	
		Cause	Emergency shut-down due to failure of cyclic data transmission.
		Measure	<ul style="list-style-type: none"> • Check the configuration of the master. Synchronous transmission is unstable.

Error group 63		EtherCAT (only CMMP-AS-...-M3)	
No.	Code	Message	Reaction
63-0	-	EtherCAT: Defective module	
		Cause	Error in the hardware.
		Measure	<ul style="list-style-type: none"> Replace the interface and send it to the manufacturer for inspection.
63-1	-	EtherCAT: Invalid data	
		Cause	Faulty telegram type.
		Measure	<ul style="list-style-type: none"> Check the wiring.
63-2	-	EtherCAT: TPDO data has not been read	
		Cause	The buffer for sending the data is full.
		Measure	The data was sent faster than the motor controller could process it. <ul style="list-style-type: none"> Reduce the cycle time on the EtherCAT bus.
63-3	-	EtherCAT: No distributed clocks active	
		Cause	Warning: Firmware is synchronising with the telegram, not with the distributed clocks system. When the EtherCAT was started, no hardware SYNC (distributed clocks) was found. The firmware now synchronises with the EtherCAT frame.
		Measure	<ul style="list-style-type: none"> If necessary, check whether the master supports the distributed clocks feature. Otherwise: Ensure that the EtherCAT frames are not interrupted by other frames if the Interpolated Position Mode is to be used.
63-4	-	EtherCAT: Missing SYNC message in IPO cycle	
		Cause	Telegrams are not being sent in the time slot pattern of the IPO.
		Measure	<ul style="list-style-type: none"> Check responsible participant for distributed clocks.

Error group 64		DeviceNet (only CMMP-AS-...-M3)	
No.	Code	Message	Reaction
64-0	-	DeviceNet: Duplicate MAC ID	
		Cause	The duplicate MAC-ID check has found two nodes with the same MAC-ID.
		Measure	<ul style="list-style-type: none"> Change the MAC-ID of one node to an unused value.
64-1	-	DeviceNet: Bus power lost	
		Cause	The DeviceNet interface is not supplied with 24 V DC.
		Measure	<ul style="list-style-type: none"> In addition to the motor controller, the DeviceNet interface must also be connected to 24 V DC.
64-2	-	DeviceNet: RX queue overflow	
		Cause	Too many messages received within a short period.
		Measure	<ul style="list-style-type: none"> Reduce the scan rate.

Error group 64		DeviceNet (only CMMP-AS-...-M3)	
No.	Code	Message	Reaction
64-3	-	DeviceNet: TX queue overflow	
		Cause	insufficient free space on the CAN bus for sending messages.
		Measure	<ul style="list-style-type: none"> • Increase the baud rate. • Reduce the number of nodes. • Reduce the scan rate.
64-4	-	DeviceNet: IO message not sent	
		Cause	Error sending I/O data.
		Measure	<ul style="list-style-type: none"> • Check that the network is connected correctly and has no faults.
64-5	-	DeviceNet: Bus OFF	
		Cause	The CAN controller is BUS OFF.
		Measure	<ul style="list-style-type: none"> • Check that the network is connected correctly and has no faults.
64-6	-	DeviceNet: CAN controller overflow	
		Cause	The CAN controller has an overflow.
		Measure	<ul style="list-style-type: none"> • Increase the baud rate. • Reduce the number of nodes. • Reduce the scan rate.

Error group 65		DeviceNet (only CMMP-AS-...-M3)	
No.	Code	Message	Reaction
65-0	-	DeviceNet active, but no module	
		Cause	The DeviceNet communication is activated in the parameter set of the motor controller, but no interface is available.
		Measure	<ul style="list-style-type: none"> • Deactivate DeviceNet communication. • Connect an interface.
65-1	-	Timeout IO connection	
		Cause	Interruption of an I/O connection.
		Measure	<ul style="list-style-type: none"> • No I/O message was received within the expected time.

Error group 66		Modbus/TCP	
No.	Code	Message	Reaction
66-0	-	Modbus/TCP: No free TCP/IP instances	
		Cause	Ethernet stack can download the requested TCP connection does not provide. Internal device error.
		Measure	<ul style="list-style-type: none"> • Restart device or restore factory settings. • If the error occurs lasting effect on the HW is defective. Can not be repaired on site.

Error group 67		Modbus/TCP	
No.	Code	Message	Reaction
67-0	-	Modbus/TCP: Timeout TCP/IP	
			configurable
		Cause	Existing TCP connection between the host and the controller has been disconnected.
	Measure	<ul style="list-style-type: none"> Ethernet cable connected correctly? Host switched off or not reachable? 	
67-1	-	Modbus/TCP: Timeout Modbus TCP/IP	
			configurable
		Cause	TCP connection between host and controller still exists, but the host does not send any more data.
	Measure	<ul style="list-style-type: none"> Crashed host? 	
67-2	-	Modbus/TCP: Buffer overflow	
			configurable
		Cause	Internal buffer for editing the data is full. Data sent from the host faster than the controller can process it.
	Measure	<ul style="list-style-type: none"> Reduce update time of the host. 	
67-3	-	Modbus/TCP: Telegram length too short	
			configurable
		Cause	The data transmitted from the host data is too long. Host sends less data than expected by the controller.
	Measure	<ul style="list-style-type: none"> Correct data length in the host. 	
67-4	-	Modbus/TCP: Telegram length too long	
			configurable
		Cause	The data transmitted from the host data is too long. Host sends more data than expected by the controller.
	Measure	<ul style="list-style-type: none"> Correct data length in the host. 	

Error group 68		EtherNet/IP (only CMMP-AS-...-M3)	
No.	Code	Message	Reaction
68-0	-	EtherNet/IP: Serious fault	
			configurable
		Cause	A serious internal error has occurred. It can be triggered by a defective interface, for example.
	Measure	<ul style="list-style-type: none"> Try to acknowledge the error. Carry out a reset. Replace the interface. If the error continues, contact Technical Support. 	
68-1	-	EtherNet/IP: General communication fault	
			configurable
		Cause	A serious error was detected in the EtherNet/IP interface.
	Measure	<ul style="list-style-type: none"> Try to acknowledge the error. Carry out a reset. Replace the interface. If the error continues, contact Technical Support. 	
68-2	-	EtherNet/IP: Connection closed	
			configurable
		Cause	The connection was closed via the controller.
	Measure	A new connection to the controller must be established.	

Error group 68		EtherNet/IP (only CMMP-AS-...-M3)	
No.	Code	Message	Reaction
68-3	-	EtherNet/IP: Connection aborted	
		Cause	A connection interruption occurred during operation.
		Measure	<ul style="list-style-type: none"> • Check the cabling between the motor controller and the higher-level control system. • Establish a new connection to the control system.
68-4	-	EtherNet/IP: Duplicate network address	
		Cause	At least one device with the same IP address exists in the network.
		Measure	<ul style="list-style-type: none"> • Use unique IP addresses for all devices in the network.

Error group 69		EtherNet/IP (only CMMP-AS-...-M3)	
No.	Code	Message	Reaction
69-0	-	EtherNet/IP: Minor fault	
		Cause	A minor error was detected in the EtherNet/IP interface.
		Measure	<ul style="list-style-type: none"> • Try to acknowledge the error. • Carry out a reset.
69-1	-	EtherNet/IP: Incorrect IP configuration	
		Cause	An incorrect IP configuration has been detected.
		Measure	<ul style="list-style-type: none"> • Correct the IP configuration.
69-2	-	EtherNet/IP: Field bus module not found	
		Cause	There is no EtherNet/IP interface in the slot.
		Measure	<ul style="list-style-type: none"> • Please check whether an EtherNet/IP interface is in slot Ext2.
69-3	-	EtherNet/IP: Module version not supported	
		Cause	There is an EtherNet/IP interface with incompatible version in the slot.
		Measure	<ul style="list-style-type: none"> • Carry out a firmware update to the most up-to-date motor controller firmware.

Error group 70		FHPP protocol	
No.	Code	Message	Reaction
70-1	-	FHPP: Mathematical error	
		Cause	Overrun/underrun or division by zero during calculation of cyclic data.
		Measure	<ul style="list-style-type: none"> • Check the cyclic data. • Check the factor group.
70-2	-	FHPP: Factor group invalid	
		Cause	Calculation of the factor group leads to invalid values.
		Measure	<ul style="list-style-type: none"> • Check the factor group.

Error group 70		FHPP protocol	
No.	Code	Message	Reaction
70-3	-	FHPP: Invalid operating mode change	
		Cause	Changing from the current to the desired operating mode is not permitted. <ul style="list-style-type: none"> – Error occurs when the OPM bits in the status S5 'Reaction to fault' or S4 'Operation enabled' are changed. – Exception: In the status SA1 'Ready', the change between 'Record select' and 'Direct Mode' is permissible.
		Measure	<ul style="list-style-type: none"> • Check your application. It may be that not every change is permissible.

Error group 71		FHPP protocol	
No.	Code	Message	Reaction
71-1	-	FHPP: Wrong receive telegram length	
		Cause	Too little data is being transmitted by the control system (data length too small).
		Measure	<ul style="list-style-type: none"> • Check the data length parameterised in the control system for the controller's receive telegram. • Check the configured data length in the FHPP+ Editor of the FCT.
71-2	-	FHPP: Wrong response telegram length	
		Cause	Too much data is to be transmitted from the motor controller to the control system (data length too large).
		Measure	<ul style="list-style-type: none"> • Check the data length parameterised in the control system for the controller's receive telegram. • Check the configured data length in the FHPP+ Editor of the FCT.

Error group 72		PROFINET (only CMMP-AS-...-M3)	
No.	Code	Message	Reaction
72-0	-	PROFINET: Initialising error	
		Cause	Interface presumably includes an incompatible stack version or is faulty.
		Measure	<ul style="list-style-type: none"> • Replace interface.
72-1	-	PROFINET: Bus error	
		Cause	No communication possible (e.g. line removed).
		Measure	<ul style="list-style-type: none"> • Check the wiring • Restart PROFINET communication.

Error group 72		PROFINET (only CMMP-AS-...-M3)	
No.	Code	Message	Reaction
72-3	-	PROFINET: Invalid IP configuration	
		Cause	An invalid IP configuration was entered in the interface. The interface cannot start with this configuration.
		Measure	<ul style="list-style-type: none"> Parameterise a permissible IP configuration via FCT.
72-4	-	PROFINET: Invalid Device name	
		Cause	A PROFINET device name was assigned with which the controller cannot communicate with the PROFINET (character specification from PROFINET standard).
		Measure	<ul style="list-style-type: none"> Parameterise a permissible PROFINET device name via FCT.
72-5	-	PROFINET: Module faulty	
		Cause	Interface CAMC-F-PN faulty.
		Measure	<ul style="list-style-type: none"> Replace interface.
72-6	-	PROFINET: Indication invalid/not supported	
		Cause	A message was issued by the PROFINET interface that is not supported by the motor controller.
		Measure	<ul style="list-style-type: none"> Please contact Technical Support.

Error group 73		PROFINET (only CMMP-AS-...-M3)	
No.	Code	Message	Reaction
73-0	-	PROFInergy: State not possible	
		Cause	An attempt was made in a positioning motion to place the controller in the energy-saving status. This is only possible at rest. The drive does not take on the status and continues to travel.
		Measure	–

Error group 78		NRT communication (only CMMP-AS-...-M3)	
No.	Code	Message	Reaction
78-0	-	NRT frame can't be send	
		Cause	NRT Frame can't be send because of too much bus load.
		Measure	<ul style="list-style-type: none"> Switch off or disconnect other bus devices during parametrisation.

Error group 80		IRQ overflow	
No.	Code	Message	Reaction
80-0	F080h	Overflow current controller IRQ	
		Cause	The process data could not be calculated in the set current/speed/position interpolator cycle.
		Measure	<ul style="list-style-type: none"> Please contact Technical Support.

Error group 80		IRQ overflow	
No.	Code	Message	Reaction
80-1	F081h	Overflow speed controller IRQ	
			PSoff
		Cause	The process data could not be calculated in the set current/speed/position interpolator cycle.
	Measure	<ul style="list-style-type: none"> • Please contact Technical Support. 	
80-2	F082h	Overflow position controller IRQ	
			PSoff
		Cause	The process data could not be calculated in the set current/speed/position interpolator cycle.
	Measure	<ul style="list-style-type: none"> • Please contact Technical Support. 	
80-3	F083h	Overflow interpolator IRQ	
			PSoff
		Cause	The process data could not be calculated in the set current/speed/position interpolator cycle.
	Measure	<ul style="list-style-type: none"> • Please contact Technical Support. 	

Error group 81		IRQ overflow	
No.	Code	Message	Reaction
81-4	F084h	Overflow low-level IRQ	
			PSoff
		Cause	The process data could not be calculated in the set current/speed/position interpolator cycle.
	Measure	<ul style="list-style-type: none"> • Please contact Technical Support. 	
81-5	F085h	Overflow MDC IRQ	
			PSoff
		Cause	The process data could not be calculated in the set current/speed/position interpolator cycle.
	Measure	<ul style="list-style-type: none"> • Please contact Technical Support. 	

Error group 82		Internal sequence control	
No.	Code	Message	Reaction
82-0	-	Internal sequencing control: Event	
			configurable
		Cause	IRQ4 overflow (10 ms low-level IRQ).
	Measure	<ul style="list-style-type: none"> • Internal sequence control: Process was interrupted. • For information only - no action required. 	
82-1	-	Multiple-started KO write access	
			configurable
		Cause	Parameters in cyclical and acyclical operation are used concurrently.
	Measure	<ul style="list-style-type: none"> • Only one parameterisation interface can be used (USB or Ethernet). 	

Error group 83		Modules in Ext1/Ext2 (only CMMP-AS-...-M3)	
No.	Code	Message	Reaction
83-0	-	Invalid module	
		Cause	<ul style="list-style-type: none"> – The plugged-in interface could not be detected. – The loaded firmware is not known. – A supported interface might be plugged into the wrong slot (e.g. SERCOS 2, EtherCAT).
		Measure	<ul style="list-style-type: none"> • Check firmware whether interface is supported. If yes: • Check that the interface is in the right place and is plugged in correctly. • Replace interface and/or firmware.
83-1	-	Module not supported	
		Cause	The plugged-in interface could be detected but is not supported by the loaded firmware.
		Measure	<ul style="list-style-type: none"> • Check firmware whether interface is supported. • If necessary, replace the firmware.
83-2	-	Module: Hardware revision not supported	
		Cause	The plugged-in interface could be detected and is basically also supported. In this case, however, the current hardware version is not supported (because it is too old).
		Measure	<ul style="list-style-type: none"> • The interface must be exchanged. If necessary, contact Technical Support.

Error group 84		Conditions for controller enabled	
No.	Code	Message	Reaction
84-0	-	Conditions for controller enable not fulfilled	
		Cause	One or more conditions for controller enable are not fulfilled. This includes: <ul style="list-style-type: none"> – DIN4 (output stage enable) is off. – DIN5 (controller enable) is off. – Intermediate circuit not yet loaded. – Encoder is not yet ready for operation. – Angle encoder identification is still active. – Automatic current regulator identification is still active. – Encoder data are invalid. – Status change of the safety function not yet completed. – Firmware or DCO download via Ethernet (TFTP) active. – DCO download onto memory card still active. – Firmware download via Ethernet active.
		Measure	<ul style="list-style-type: none"> • Check status of digital inputs. • Check encoder cables. • Wait for automatic identification. • Wait for completion of the firmware or DCO download.
			Warn

Error group 90		Internal error	
No.	Code	Message	Reaction
90-0	5080h	External RAM not recognized	
		Cause	External SRAM not detected / not sufficient. Hardware error (SRAM component or board is faulty).
		Measure	<ul style="list-style-type: none"> • Please contact Technical Support.
90-2	5080h	Error at FPGA boot-up	
		Cause	The FPGA (hardware) cannot be booted. The FPGA is booted serially when the device is started, but in this case it could not be loaded with data or it reported a checksum error.
		Measure	<ul style="list-style-type: none"> • Switch on the device again (24 V). If the error occurs again, the hardware is faulty.
90-3	5080h	Error at SD-ADU start	
		Cause	SD-ADUs (hardware) cannot be started. One or more SD-ADUs are not supplying any serial data.
		Measure	<ul style="list-style-type: none"> • Switch on the device again (24 V). If the error occurs again, the hardware is faulty.
			PSoff

Error group 90		Internal error	
No.	Code	Message	Reaction
90-4	5080h	SD-ADU synchronisation error after start	
		Cause	SD-ADU (hardware) not synchronous after starting. During operation, the SD-ADUs for the resolver signals continue running with strict synchronisation once they have been initially started synchronously. The SD-ADUs could not be started at the same time during that initial start phase.
		Measure	<ul style="list-style-type: none"> Switch on the device again (24 V). If the error occurs again, the hardware is faulty.
90-5	5080h	SD-ADU not synchronous	
		Cause	SD-ADU (hardware) not synchronous after starting. During operation, the SD-ADUs for the resolver signals continue running with strict synchronisation once they have been initially started synchronously. This is checked continually during operation and an error is triggered if appropriate.
		Measure	<ul style="list-style-type: none"> Possibly massive EMC coupling. Switch on the device again (24 V). If the error occurs again, the hardware is faulty.
90-6	5080h	IRQ0 (current controller): Trigger error	
		Cause	The output stage is not triggering the software IRQ, which then operates the current regulator. Very likely to be a hardware error on the board or in the processor.
		Measure	<ul style="list-style-type: none"> Switch on the device again (24 V). If the error occurs again, the hardware is faulty.
90-9	5080h	Illegal firmware version	
		Cause	A beta version compiled for the debugger was loaded regularly.
		Measure	<ul style="list-style-type: none"> Check the firmware version, and update the firmware if necessary.

Error group 91		Initialisation error	
No.	Code	Message	Reaction
91-0	6000h	Internal initialising error	
		Cause	Internal SRAM too small for the compiled firmware. Can only occur with beta versions.
		Measure	<ul style="list-style-type: none"> Check the firmware version, and update the firmware if necessary.

Error group 91		Initialisation error	
No.	Code	Message	Reaction
91-1	-	Memory error when copying	
		Cause	Firmware parts were not copied correctly from the external FLASH into the internal RAM upon starting.
		Measure	<ul style="list-style-type: none"> Switch on the device again (24 V). If the error occurs repeatedly, check the firmware version and update the firmware if necessary.
91-2	-	Error when reading the controller/power section coding	
		Cause	The ID-EEPROM in the controller or power section could either not be addressed at all or does not have consistent data.
		Measure	<ul style="list-style-type: none"> Switch on the device again (24 V). If the error occurs repeatedly, the hardware is faulty. No repair possible.
91-3	-	Software initialisation error	
		Cause	One of the following components is missing or could not be initialised: <ol style="list-style-type: none"> Shared memory not available or faulty. Driver library not available or faulty.
		Measure	<ul style="list-style-type: none"> Check firmware version, update if necessary.

Error group 92		Boot loader/firmware update	
No.	Code	Message	Reaction
92-0	-	Error during firmware download	
		Cause	Error during requested firmware download.
		Measure	<ul style="list-style-type: none"> Check the firmware file. Restart firmware download.
92-1	-	Error during bootloader update	
		Cause	Error during requested bootloader download.
		Measure	<ul style="list-style-type: none"> Restart bootloader download. Send the device to the manufacturer for inspection.

Instructions on actions with the error messages 08-2 ... 08-7	
Action	Notes
<ul style="list-style-type: none"> • Check whether encoder signals are faulty. 	<ul style="list-style-type: none"> – Check the wiring, e.g. are one or more phases of the track signals interrupted or short-circuited? – Check that installation complies with EMC recommendations (cable screening on both sides?). – Only with incremental encoders: With TTL single-ended signals (HALL signals are always TTL single-ended signals): Check whether there might be an excessive voltage drop on the GND line; in this case = signal reference. Check whether there might be an excessive voltage drop on the GND line; in this case = signal reference. – Check the level of supply voltage on the encoder. Sufficient? If not, change the cable diameter (connect unused lines in parallel) or use voltage feedback (SENSE+ and SENSE-).
<ul style="list-style-type: none"> • Test with other encoders. 	<ul style="list-style-type: none"> – If the error still occurs when the configuration is correct, test with a different (error-free) encoder (replace the connecting cable as well). If the error still occurs, there is a fault in the motor controller. Repair by the manufacturer required.

Tab. B.2 Instructions on error messages 08-2 ... 08-7

Index

A	
Acceleration	
– Brake (position)	189
– Quick stop (position)	189
acceleration_factor	86
Activate undervoltage monitor	94
Actual position value (increments)	110
Actual speed value	205
Actual value	
– Position in increments	
(position_actual_value_s)	110
– Position in position units	
(position_actual_value)	110
– Torque (torque_actual_value)	215
actual_dc_link_circuit_voltage	93
actual_size	198
analog_input_offset	128
analog_input_offset_ch_0	128
analog_input_offset_ch_1	128
analog_input_offset_ch_2	128
analog_input_voltage	127
analog_input_voltage_ch_0	127
analog_input_voltage_ch_1	127
analog_input_voltage_ch_2	128
Analogue inputs	
– Input voltage	127
– Input voltage channel 0	127
– Input voltage channel 1	127
– Input voltage channel 2	128
– Offset voltage	128
– Offset voltage channel 0	128
– Offset voltage channel 1	128
– Offset voltage channel 2	128
Angle encoder offset	100
Approach new position	191
B	
Behaviour with command	
– disable operation	171
– quick stop	171
– shutdown	170
Brake delay time	141
brake_delay_time	141
buffer_clear	199
buffer_organisation	198
buffer_position	198
C	
cob_id_sync	35
cob_id_used_by_pdo	29
commissioning_state	146
Contouring error	105
– Error window	111
– Limit value overrun	113
– Time-out time	111
Contouring error limit value	113
Contouring error limit value exceeded	113
Control word for interpolation data	195
control_effort	112
Controller enable logic	91
Controller error	37
controlword	156
– Bit assignment	152, 155, 157
– Commands	157
– Object description	156
Conversion factors	80
– Choice of prefix	89
– Position factor	82
Correction velocity	108
Current following error	111
Current intermediate circuit voltage	93
Current limiting	116
Current regulator	
– Gain	102
– Parameters	102
– Time constant	102
Current setpoint value	215
current_actual_value	216
current_limitation	116
curve generator	186
Cycle time	
– Current regulator	145
– Position control	145, 146
– Speed regulator	145

Cycle time PDOs	29	enable_logic	91
cycletime_current_controller	145	encoder_emulation_data	123
cycletime_position_controller	145	encoder_emulation_offset	123
cycletime_trajectory_generator	146	encoder_emulation_resolution	123
cycletime_velocity_controller	145	encoder_offset_angle	100
D		encoder_x10_counter	122
dc_link_circuit_voltage	216	encoder_x10_data_field	121
Deactivate undervoltage monitor	94	encoder_x10_divisor	122
Device Control	150	encoder_x10_numerator	122
Device nominal current	95	encoder_x10_resolution	121
Device nominal voltage	93	encoder_x2a_data_field	119
dig_out_state_mapp_dout_1	131	encoder_x2a_divisor	119
dig_out_state_mapp_dout_2	131	encoder_x2a_numerator	119
dig_out_state_mapp_dout_3	131	encoder_x2a_resolution	119
dig_out_state_mapp_ea88_0_high	133	encoder_x2b_counter	121
dig_out_state_mapp_ea88_0_low	133	encoder_x2b_data_field	120
Digital inputs	129	encoder_x2b_divisor	120
Digital outputs	130	encoder_x2b_numerator	120
– Mapping	131	encoder_x2b_resolution	120
– Mapping of CAMC-EA	133	end_velocity	188
– Mapping of DOUT1	131	Error management	148
– Mapping of DOUT2	131	error_management	148
– Mapping of DOUT3	131	error_register	37
– Mask	130	Extended sine modulation	92
– Statuses	130	F	
digital_inputs	129	Factor group	80
digital_outputs	130	– acceleration_factor	86
digital_outputs_data	130	– polarity	89
digital_outputs_mask	130	– position_factor	81
digital_outputs_state_mapping	131	– velocity_encoder_factor	84
disable_operation_option_code	171	fault_reaction_option_code	172
Divisor		Filter time constant synchronous speed	126
– acceleration_factor	87	firmware_custom_version	144
– position_factor	82	firmware_main_version	143
– velocity_encoder_factor	84	first_mapped_object	30
drive_data	91, 99, 113, 134, 140	Following error time-out time	111
E		Following error window	111
EMERGENCY Message	37	Following_Error	105
Enable Logic	91	following_error_current_value	111
enable_dc_link_undervoltage_error	94	following_error_time_out	111
enable_enhanced_modulation	92	following_error_window	111
		fourth_mapped_object	31

G			
Gain of the current regulator	102	– Current	93
		– Maximum	94
		– Minimal	94
H		Interpolation data	194
home_offset	177	Interpolation type	194
Homing		interpolation_data_configuration	197
– Creep speed	179	interpolation_data_record	194
– Method	178	interpolation_submode_select	194
– Search speed	178	interpolation_sync_definition	196
– speeds	178	interpolation_time_period	195
– Zero point offset	177	ip_data_controlword	195
Homing mode	175	ip_data_position	195
– home_offset	177	ip_sync_every_n_event	197
– homing_acceleration	179	ip_time_index	196
– homing_method	177	ip_time_units	196
– homing_speeds	178		
Homing run	175	L	
– Control of the	184	Limit switch	134
– Timeout	179	– Emergency stop ramp	136
homing_acceleration	179	– Polarity	134
homing_method	177	limit_current	116, 117
homing_speeds	178	limit_current_input_channel	116
homing_switch_polarity	135	limit_speed_input_channel	117
homing_switch_selector	136	limit_switch_deceleration	136
homing_timeout	179	limit_switch_polarity	134
		Load default parameters	77
I			
I2t extent of utilisation	99	M	
I2t time	98	Manufacturer code	141
Identification of the device	141	manufacturer_statusword_1	164
Identifier for PDO	29	manufacturer_statusword	164
identity_object	141	manufacturer_statuswords	163
iit_error_enable	99	Manufacturer-specific status word	163, 164
iit_ratio_motor	99	Manufacturer-specific status word 1	164
iit_time_motor	98	Mapping parameter for PDOs	30
Incremental encoder emulation		max_buffer_size	197
– Offset	123	max_current	97
– Resolution	123	max_dc_link_circuit_voltage	94
inhibit_time	29	max_motor_speed	209
Instructions on this documentation	7	max_position_range_limit	114
Intermediate circuit monitoring	94	max_power_stage_temperature	93
Intermediate circuit voltage		max_torque	214

Maximum current	95	– Object 1001h	37
Maximum intermediate circuit voltage	94	– Object 1003h	38
Maximum motor speed	209	– Object 1003h_01h	38
Maximum output stage temperature	93	– Object 1003h_02h	38
Maximum torque	214	– Object 1003h_03h	38
Methods of homing	180	– Object 1003h_04h	38
min_dc_link_circuit_voltage	94	– Object 1005h	35
min_position_range_limit	114	– Object 1010h	77
Minimum intermediate circuit voltage	94	– Object 1010h_01h	78
modes_of_operation	173	– Object 1011h	77
modes_of_operation_display	174	– Object 1011h_01h	77
motion_profile_type	190	– Object 1018h	141
Motor parameter		– Object 1018h_01h	141
– I2t time	98	– Object 1018h_02h	142
– Nominal current	97	– Object 1018h_03h	142
– Pole (pair) number	98	– Object 1018h_04h	142
– Resolver offset angle	100	– Object 1100h	55
Motor peak current	97	– Object 1402h	34
Motor rated current	97	– Object 1403h	34
Motor Rated Torque	215	– Object 1602h	34
motor_data	98, 100	– Object 1603h	34
motor_rated_current	97	– Object 1800h	29, 31
motor_rated_torque	215	– Object 1800h_01h	29
motor_temperatur_sensor_polarity	101	– Object 1800h_02h	29
		– Object 1800h_03h	29
		– Object 1801h	31
N		– Object 1802h	32
Nominal speed for speed adjustment	209	– Object 1803h	32
nominal_current	95	– Object 1A00h	30, 31
nominal_dc_link_circuit_voltage	93	– Object 1A00h_00h	30
Not Ready to Switch On	154	– Object 1A00h_01h	30
Number of mapped objects	30	– Object 1A00h_02h	30
Number of pole pairs	98	– Object 1A00h_03h	30
Number of poles	98	– Object 1A00h_04h	31
number_of_mapped_objects	30	– Object 1A01h	31
numerator	89	– Object 1A02h	32
– acceleration_factor	87	– Object 1A03h	32
Numerator		– Object 1C00h	55
– position_factor	82	– Object 1C00h_00h	55
– velocity_encoder_factor	84	– Object 1C00h_01h	55
		– Object 1C00h_02h	56
O		– Object 1C00h_03h	56
Objects			

- Object 1C00h_04h	56	- Object 202Dh	109
- Object 1C10h	56	- Object 202Eh	205
- Object 1C11h	57	- Object 202Fh	126
- Object 1C12h	57	- Object 202Fh_07h	126
- Object 1C12h_00h	58	- Object 2045h	179
- Object 1C12h_01h	58	- Object 204Ah	137
- Object 1C12h_02h	58	- Object 204Ah_01h	138
- Object 1C12h_03h	58	- Object 204Ah_02h	138
- Object 1C12h_04h	58	- Object 204Ah_03h	138
- Object 1C13h	59	- Object 204Ah_04h	139
- Object 1C13h_00h	59	- Object 204Ah_05h	139
- Object 1C13h_01h	59	- Object 204Ah_06h	139
- Object 1C13h_02h	59	- Object 2090h	210
- Object 1C13h_03h	60	- Object 2090h_01h	210
- Object 1C13h_04h	60	- Object 2090h_02h	211
- Object 2000h	163	- Object 2090h_03h	211
- Object 2000h_00h	164	- Object 2090h_04h	211
- Object 2000h_01h	164	- Object 2090h_05h	211
- Object 2014h	32	- Object 2100h	148
- Object 2015h	32	- Object 2400h	127
- Object 2016h	33	- Object 2400h_01h	127
- Object 2017h	33	- Object 2400h_02h	127
- Object 201Ah	123	- Object 2400h_03h	128
- Object 201Ah_01h	123	- Object 2401h	128
- Object 201Ah_02h	123	- Object 2401h_01h	128
- Object 2021h	124	- Object 2401h_02h	128
- Object 2022h	125	- Object 2401h_03h	128
- Object 2023h	126	- Object 2415h	116
- Object 2024h	119	- Object 2415h_01h	116
- Object 2024h_01h	119	- Object 2415h_02h	116
- Object 2024h_02h	119	- Object 2416h	117
- Object 2024h_03h	119	- Object 2416h_01h	117
- Object 2025h	121	- Object 2416h_02h	117
- Object 2025h_01h	121	- Object 2420h	131
- Object 2025h_02h	122	- Object 2420h_01h	131
- Object 2025h_03h	122	- Object 2420h_02h	131
- Object 2025h_04h	122	- Object 2420h_03h	131
- Object 2026h	120	- Object 2420h_11h	133
- Object 2026h_01h	120	- Object 2420h_12h	133
- Object 2026h_02h	120	- Object 2600h	168
- Object 2026h_03h	120	- Object 2600h_01h	168
- Object 2026h_04h	121	- Object 2600h_02h	169
- Object 2028h	123	- Object 2602h	169

– Object 2602h_01h	169	– Object 6083h	188
– Object 6040h	156	– Object 6084h	189
– Object 6041h	160	– Object 6085h	189
– Object 604Dh	98	– Object 6086h	190
– Object 605Ah	171	– Object 6087h	217
– Object 605Bh	170	– Object 6088h	217
– Object 605Ch	171	– Object 608Ah	61
– Object 605Eh	172	– Object 608Bh	61
– Object 6060h	173	– Object 608Ch	61
– Object 6061h	174	– Object 608Dh	61
– Object 6062h	109	– Object 608Eh	61
– Object 6063h	110	– Object 6093h	81
– Object 6064h	110	– Object 6093h_01h	82
– Object 6065h	111	– Object 6093h_02h	82
– Object 6066h	111	– Object 6094h	84
– Object 6067h	112	– Object 6094h_01h	84
– Object 6068h	113	– Object 6094h_02h	84
– Object 6069h	203	– Object 6097h	86
– Object 606Ah	204	– Object 6097h_01h	87
– Object 606Bh	204	– Object 6097h_02h	87
– Object 606Ch	205	– Object 6098h	177
– Object 606Dh	207	– Object 6099h	178
– Object 606Eh	207	– Object 6099h_01h	178
– Object 606Fh	208	– Object 6099h_02h	179
– Object 6070h	208	– Object 609Ah	179
– Object 6071h	214	– Object 60C0h	194
– Object 6072h	214	– Object 60C1h	194
– Object 6073h	97	– Object 60C1h_01h	195
– Object 6074h	215	– Object 60C1h_02h	195
– Object 6075h	97	– Object 60C2h	195
– Object 6076h	215	– Object 60C2h_01h	196
– Object 6077h	215	– Object 60C2h_02h	196
– Object 6078h	216	– Object 60C3h	196
– Object 6079h	216	– Object 60C3h_01h	197
– Object 607Ah	187	– Object 60C3h_02h	197
– Object 607Bh	114	– Object 60C4h	197
– Object 607Bh_01h	114	– Object 60C4h_01h	197
– Object 607Bh_02h	114	– Object 60C4h_02h	198
– Object 607Ch	177	– Object 60C4h_03h	198
– Object 607Eh	89	– Object 60C4h_04h	198
– Object 6080h	209	– Object 60C4h_05h	198
– Object 6081h	188	– Object 60C4h_06h	199
– Object 6082h	188	– Object 60F4h	111

– Object 60F6h	102	– Object 6510h_40h	95
– Object 60F6h_01h	102	– Object 6510h_41h	95
– Object 60F6h_02h	102	– Object 6510h_A9h	143
– Object 60F9h	103	– Object 6510h_AAh	144
– Object 60F9h_01h	103	– Object 6510h_B0h	145
– Object 60F9h_02h	104	– Object 6510h_B1h	145
– Object 60F9h_04h	104	– Object 6510h_B2h	145
– Object 60FAh	112	– Object 6510h_B3h	146
– Object 60FBh	107	– Object 6510h_C0h	146
– Object 60FBh_01h	108	Offset of the angle encoder	100
– Object 60FBh_02h	108	Operating mode	173, 174
– Object 60FBh_04h	108	– Homing run	175
– Object 60FBh_05h	108	– Modifying of the	173
– Object 60FDh	129	– Reading of the	174
– Object 60FEh	130	– Setting of the	173
– Object 60FEh_01h	130	Output stage parameter	90
– Object 60FEh_02h	130	– Device nominal current	95
– Object 60FFh	209	– Device nominal voltage	93
– Object 6410h	98	– Enable Logic	91
– Object 6410h_03h	98	– Intermediate circuit voltage	93
– Object 6410h_04h	99	– Max. intermediate circuit voltage	94
– Object 6410h_10h	100	– Maximum current	95
– Object 6410h_11h	100	– Maximum temperature	93
– Object 6410h_14h	101	– Min. intermediate circuit voltage	94
– Object 6510h	91	– PWM frequency	91
– Object 6510h_10h	91		
– Object 6510h_11h	134	P	
– Object 6510h_13h	136	Parameter sets	
– Object 6510h_14h	135	– Load default values	77
– Object 6510h_15h	136	– Loading and saving	75
– Object 6510h_18h	141	– Save parameter set	77
– Object 6510h_20h	115	Parametrisation status	146
– Object 6510h_22h	113	PDO	25
– Object 6510h_30h	91	– 1st mapped object	30
– Object 6510h_31h	92	– 2nd mapped object	30
– Object 6510h_32h	93	– 3rd mapped object	30
– Object 6510h_33h	93	– 4th mapped object	31
– Object 6510h_34h	93	– RPDO3	
– Object 6510h_35h	94	1st mapped object	34
– Object 6510h_36h	94	2nd mapped object	34
– Object 6510h_37h	94	3rd mapped object	34
– Object 6510h_38h	99	4th mapped object	34
– Object 6510h_3Ah	92	COB-ID used by PDO	34

first mapped object	34	Identifier	31
fourth mapped object	34	Inhibit time	31
Identifier	34	Number of mapped objects	31
Number of mapped objects	34	second mapped object	31
second mapped object	34	third mapped object	31
third mapped object	34	Transmission type	31
Transmission type	34	Transmit mask	32
– RPDO4		– TPDO3	
1st mapped object	34	1st mapped object	32
2nd mapped object	34	2nd mapped object	32
3rd mapped object	34	3rd mapped object	32
4th mapped object	34	4th mapped object	32
COB-ID used by PDO	34	COB-ID used by PDO	32
first mapped object	34	first mapped object	32
fourth mapped object	34	fourth mapped object	32
Identifier	34	Identifier	32
Number of mapped objects	34	Inhibit time	32
second mapped object	34	Number of mapped objects	32
third mapped object	34	second mapped object	32
Transmission type	34	third mapped object	32
– TPDO1		Transmission type	32
1st mapped object	31	Transmit mask	33
2nd mapped object	31	– TPDO4	
3rd mapped object	31	1st mapped object	32
4th mapped object	31	2nd mapped object	32
COB-ID used by PDO	31	3rd mapped object	32
first mapped object	31	4th mapped object	32
fourth mapped object	31	COB-ID used by PDO	32
Identifier	31	first mapped object	32
Inhibit time	31	fourth mapped object	32
Number of mapped objects	31	Identifier	32
second mapped object	31	Inhibit time	32
third mapped object	31	Number of mapped objects	32
Transmission type	31	second mapped object	32
Transmit mask	32	third mapped object	32
– TPDO2		Transmission type	32
1st mapped object	31	Transmit mask	33
2nd mapped object	31	PDO Message	25
3rd mapped object	31	Peak current	
4th mapped object	31	– Motor	97
COB-ID used by PDO	31	– Motor controller	95
first mapped object	31	peak_current	95
fourth mapped object	31	Permissible torque	214

phase_order	100	– Jerk-free	190
Pin allocation CAN	11	– Linear	190
Polarity of motor temperature sensor	101	– Sine2	190
pole_number	98	Positioning speed	188
Position actual value (position units)	110	Positioning Window	
Position control	105	– Position window	112
– Dead range	108	– Time	113
– Gain	108	power_stage_temperature	92
– Output of the	112	pre_defined_error_field	38
– Parameters	108	Product code	142
– Time constant	108	product_code	142
position control function	105	Profile Position Mode	
Position control parameters	108	– end_velocity	188
Position controller gain	108	– motion_profile_type	190
Position controller output	112	– profile_acceleration	188
Position controller time constant	108	– profile_deceleration	189
Position value interpolation	195	– profile_velocity	188
position_actual_value	110	– quick_stop_deceleration	189
position_actual_value_s	110	– target_position	187
position_control_gain	108	Profile Torque Mode	212
position_control_parameter_set	108	– current_actual_value	216
position_control_time	108	– dc_link_circuit_voltage	216
position_control_v_max	108	– max_torque	214
position_demand_sync_value	109	– motor_rated_torque	215
position_demand_value	109	– target_torque	214
position_encoder_selection	124	– torque_actual_value	215
position_error_switch_off_limit	113	– torque_demand_value	215
position_error_tolerance_window	108	– torque_profile_type	217
position_factor	81	– torque_slope	217
position_range_limit	114	Profile Velocity Mode	201
position_range_limit_enable	115	– max_motor_speed	209
position_reached	106	– sensor_selection_code	204
position_window	112	– target_velocity	209
position_window_time	113	– velocity_actual_value	205
Positioning	191	– velocity_demand_value	204
– Braking deceleration	189	– velocity_sensor	203
– Handshake	191	– velocity_threshold	208
– Quick stop deceleration	189	– velocity_threshold_time	208
– Speed at	188	– velocity_window	207
– Target position	187	– velocity_window_time	207
Positioning braking deceleration	189	profile_acceleration	188
Positioning profile		profile_deceleration	189

profile_velocity	188	Save parameter set	78
PWM frequency	91	save_all_parameters	78
pwm_frequency	91	Scaling factors	80
		– Choice of prefix	89
		– Position factor	82
Q		SDO	21
Quick stop deceleration	189	SDO Error Messages	23
quick_stop_deceleration	189	SDO message	20
quick_stop_option_code	171	second_mapped_object	30
		Selection of the position actual value	124
		Selection of the synchronisation source	125
R		sensor_selection_code	204
R-PDO 3	34	serial_number	142
R-PDO4	34	Service	7
Rated motor current	97	Setpoint	
Ready to Switch On	154	– Current	215
Receive_PDO_3	34	– Synchronous speed (velocity units)	205
Receive_PDO_4	34	– Torque	214
Reference switch	134, 136	Setpoint torque (torque regulation)	214
– Polarity	135	Setting parameters	75
Referencing method	178	Setting the operating mode	173
Resolver offset angle	100	shutdown_option_code	170
resolver_offset_angle	100	size_of_data_record	198
restore_all_default_parameters	77	Speed	
restore_parameters	77	– at positioning	188
Revision number CANopen	142	– during homing	178
revision_number	142	Speed adjustment	201
		– Max. motor speed	209
S		– Nominal speed	209
Sample		– Positioning Window	207
– Control systems	139	– Standstill threshold	208
– Mode	138	– Standstill threshold time	208
– State	138	– Target speed	209
– Status mask	138	– Target window time	207
SAMPLE input as reference switch	136	Speed adjustment operating mode	201
sample_control	139	Speed regulator	
sample_data	137	– Filter time constant	104
sample_mode	138	– Gain	103
sample_position_falling_edge	139	– Parameters	103
sample_position_rising_edge	139	– Time constant	104
sample_status	138	speed_during_search_for_switch	178
sample_status_mask	138	speed_during_search_for_zero	179
Sampling position		speed_limitation	117
– Falling edge	139		
– Rising edge	139		

Speed-limited torque operation	117	Target torque (torque regulation)	214
standard_error_field_0	38	Target window time	113
standard_error_field_1	38	Target window time with speed adjustment	207
standard_error_field_2	38	Target window with speed adjustment	207
standard_error_field_3	38	target_position	187
Standstill threshold during speed adjustment	208	target_torque	214
Standstill threshold time with speed adjustment	208	target_velocity	209
START input as reference switch	136	Technical data interface CANopen	218
Start positioning	191	third_mapped_object	30
State		Time constant of the current regulator	102
– Not Ready to Switch On	154	Torque actual value	215
– Ready to Switch On	154	Torque limitation	116
– Switch On Disabled	154	– Scaling	117
– Switched On	154	– Setpoint	116
Status		– Source	116
– Not Ready to Switch On	154	Torque regulation	
– Ready to Switch On	154	– Current setpoint value	215
– Switch On Disabled	154	– Max. torque	214
– Switched On	154	– Rated torque	215
statusword		– Setpoint torque	214
– Bit assignment	161	– Setpoint value profile	217
– Object description	160	– Target torque	214
store_parameters	77	– Torque actual value	215
Switch On Disabled	154	Torque regulation operating mode	212
SYNC	35	Torque regulations	212
SYNC message	35	torque_actual_value	215
synchronisation_encoder_selection	125	torque_control_gain	102
synchronisation_filter_time	126	torque_control_parameters	102
synchronisation_main	126	torque_control_time	102
synchronisation_selector_data	126	torque_demand_value	215
Synchronous speed (velocity units)	205	torque_profile_type	217
synchronize_on_group	197	torque_slope	217
T		Torque-limited speed operation	116
T-PDO 1	31	tpdo_1_transmit_mask	32
T-PDO 2	31	tpdo_2_transmit_mask	32
T-PDO 3	32	tpdo_3_transmit_mask	33
T-PDO 4	32	tpdo_4_transmit_mask	33
Target group	7	Transfer parameters for PDOs	29
Target position	187	transfer_PDO_1	31
Target position window	112	transfer_PDO_2	31
Target speed for speed adjustment	209	transfer_PDO_3	32
		transfer_PDO_4	32
		transmission_type	29

transmit_pdo_mapping	30	velocity_window	207
transmit_pdo_parameter	29	velocity_window_time	207
Trigger iit error	99	vendor_id	141
Type of transmission	29	Version	7
		Version number of the customer-specific variant	144
V		Version number of the firmware	143
Velocity limitation	117	X	
– Scaling	118	X10	
– Setpoint	117	– Counter	122
– Source	117	– Drive	122
velocity_acceleration_neg	211	– Drive-out	122
velocity_acceleration_pos	211	– Resolution	121
velocity_actual_value	205	X2A	
velocity_control_filter_time	104	– Drive	119
velocity_control_gain	103	– Drive-out	119
velocity_control_parameter_set	103	– Resolution	119
velocity_control_time	104	X2B	
velocity_deceleration_neg	211	– Counter	121
velocity_deceleration_pos	211	– Drive	120
velocity_demand_sync_value	205	– Drive-out	120
velocity_demand_value	204	– Resolution	120
velocity_encoder_factor	84	Z	
velocity_ramps	210	Zero point offset	177
velocity_ramps_enable	210		
velocity_sensor_actual_value	203		
velocity_threshold	208		
velocity_threshold_time	208		

Copyright:
Festo SE & Co. KG
Postfach
73726 Esslingen
Germany

Phone:
+49 711 347-0

Fax:
+49 711 347-2144

e-mail:
service_international@festo.com

Reproduction, distribution or sale of this document or communication of its contents to others without express authorization is prohibited. Offenders will be liable for damages. All rights reserved in the event that a patent, utility model or design patent is registered.

Internet:
www.festo.com

Original: de
Version: 1510b