Stepper motor Driver with CANopen interface R272-42-CAN

2018

R272-42-CAN

Given manual is an authentic guide for users

CANopen®, CiA® 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

Other symbols:

Caution

Hazards that can cause minor injuries or serious material damage.

Recommendations, tips, references to other documentation.

Essential or useful accessories

Text designations:

● Activities that may be carried out in any order
1. Activities that should be carried out in the order stated
– General lists

Table of Contents

Instructions on this documentation

This documentation describes the device profile CiA 402 (DS 402) for the stepper motor controller R272-42-CAN conforming to the section “Information on the version. This provides you with supplementary information about control, diagnostics and parameterisation of the motor controllers via the fieldbus.

Unconditionally observe the general safety regulations for the R272-42-CAN. **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 if you have any technical problems.

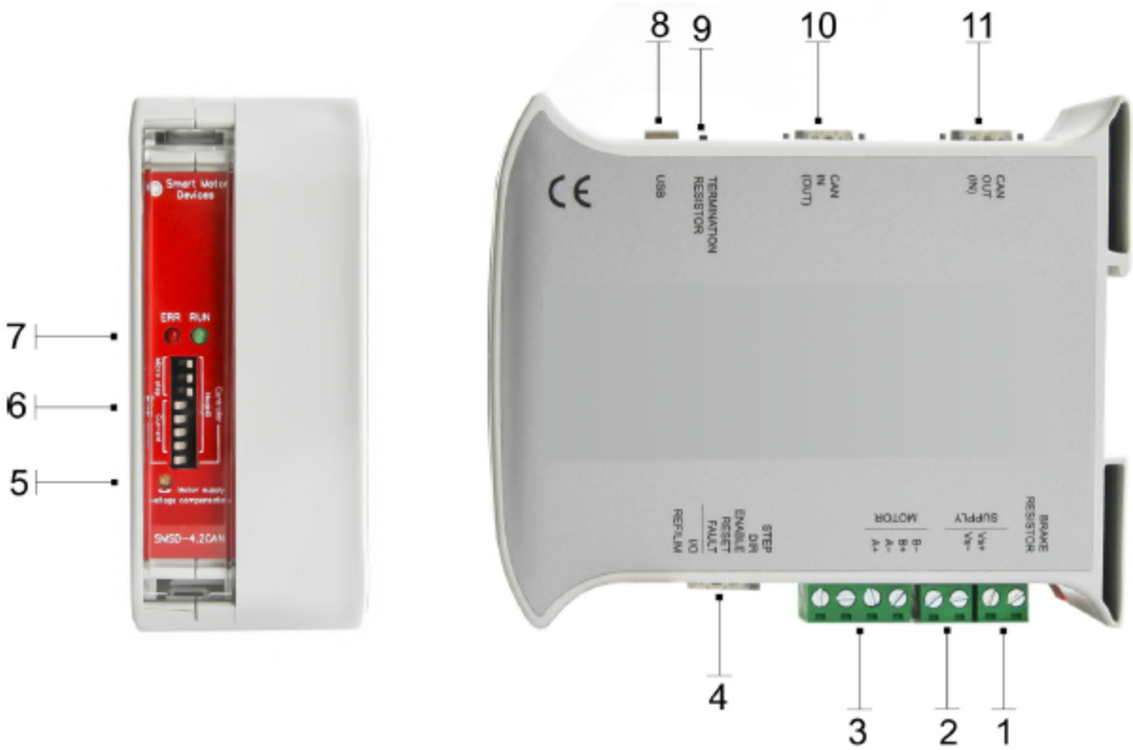
Information on the version

This description refers to the following versions::
- Motor controller from Rev 1.0;
- Built in software, from Rev 1.0;
- CANopen Builder Architect/Limited, from Rev 1.0.

Note
With newer firmware versions, check whether there is a newer version of this description
available: → ***www.smd.ee***

1. General information

Control and parameterisation via CiA 402 for R272-42-CAN is supported correspondingly through the fieldbus interfaces CANopen and USB.



Pic 1.1 — Motor Controller R272-42-CAN

[1] - Connector for additional braking resistor in parallel with internal (built-in resistor 33Ω)
[2] - Power supply input
[3] - Motor Connection input
[4] - Input/Output port connection
[5] - Voltage adjustment potentiometer
[6] - DIP-switch NodeID and operation mode (controller/driver)
[7] - CAN led indication
[8] - USB- a configuration and programming connector with communication indication and a button to enter the software downloading mode
[9] - DIP-switch for CAN terminal resistor
[10] - CAN bus (in/out)
[11] - CAN bus (out/in)

2. General information on CANopen

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:

CAN in Automation (CiA) International Headquarter

D-91058 Erlangen

Phone.: 09131-601091

Fax: 09131-601092

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

2.1. Cabling and pin assignment

2.1.1. Pin allocations

The CAN bus connection is designed as a 9-pole DSUB plug in.

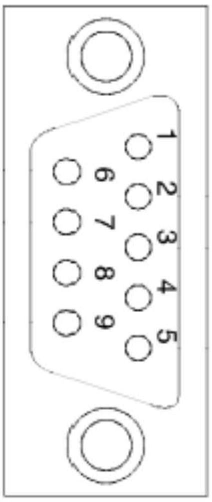
DSUB plug	Pin no.		Designation	Description
		1		Unused
	6			Unused
		2	CAN-L	Negative CAN signal CAN (Dominant Low)
	7		CAN-H	Positive CAN signal (Dominant High)
		3	CAN-GND	Ground
	8			Unused
		4		Unused
	9			Unused
		5	CAN-Shield	Screening

Table 2.1 — Pin assignment for CAN-interface.

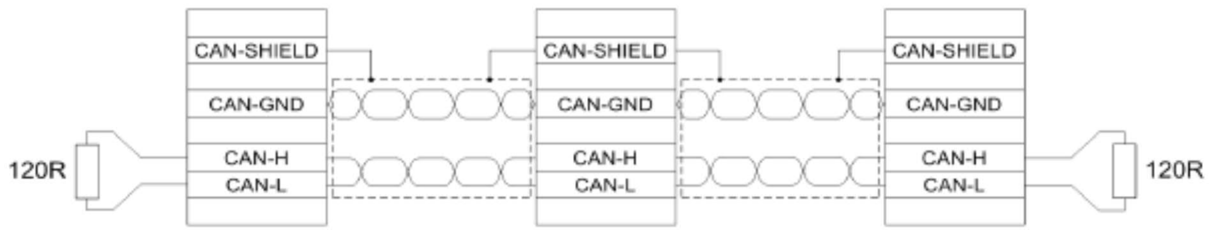
CAN bus cabling
When cabling the motor controller via the CAN bus, you should unconditionally observe the following information and instructions to obtain a stable, trouble-free system. If cabling is improperly done, malfunctions can occur on the CAN bus during operation. These can cause the motor controller 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 [9] (Refer to Pic. 1.1).

2.1.2. 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.



Pic. 2.1 — Cabling example.

- A terminating resistor of exactly 120 Ω +/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. Terminating resistor in R272-42-CAN is activated via DIP-switch [9] (Pic. 1.1).
- 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.

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.

2.2. Configuration of CANopen

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 CANopen Builder without connection to the CANopen bus should be carried out first.

Instructions on commissioning R272-42-CAN with the SMD CANopen Builder can be found in **Appendix B** of this manual.

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:.

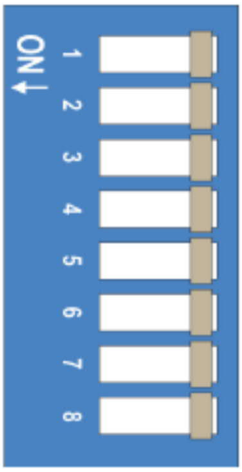
1. Setting of the offset of the node number via DIP-switch [6] (Pic. 1.1).

2. Bit rate settings are changed via CANopen Builder as well as parametrisation and physical units configurations (Factor Group)

3. Configuration of the CANopen master.

2.2.1. Setting of the node ID with DIP switches

Setting of the node number (NodeID) can be made with DIP switch [6] (Pic. 1.1). Node Number is set in binary code with value from 1 to 127, using switches from 1 to 7 (0 — 6 bit NodeID). Position «ON» stands for an active bit, «OFF» - inactive. Switch number 8 stands for choosing the mode: «OFF» - active controller mode via CANopen, «ON» - active Driver mode (DIP-switch is used differently please refer to chapter 7).

DIP- switches		**Value**		**Example**	
		«ON»	«OFF»		Value
	1	1	0	«ON»	1
	2	2	0	«OFF»	0
	3	4	0	«OFF»	0
	4	8	0	«ON»	8
	5	16	0	«OFF»	0
	6	32	0	«ON»	32
	7	64	0	«ON»	64
	8	Driver	Controller	«OFF»	Controller
Total 1...7 =Node number		1...127			105

Tab. 2.3 — Node ID.

It is not allowed to change the NodeID in Operational mode.

2.3. Setting of the transmission rate

Setting of the transmission rate is performed via CANopen Builder software. In the object dictionary of the device, this parameter corresponds to an object with the index 2045h, the possible values are given below, and each of them corresponds to a certain data rate. More detailed information on parameterization can be found in Appendix B of this technical manual.

Value range of Index 2045h	**Transmission rate**
5	50kbit/s
12	125kbit/s
25	250kbit/s
50	500kbit/s
100	1Mbps

Tab 2.4 — Setting of the transmission rate.

The default data transfer rate is **500kbit/s**. After changing the speed, it is necessary to save the network parameters, otherwise the controller will load the last saved configuration after it is rebooted (if there are no previous setting saved, factory settings will be applied).

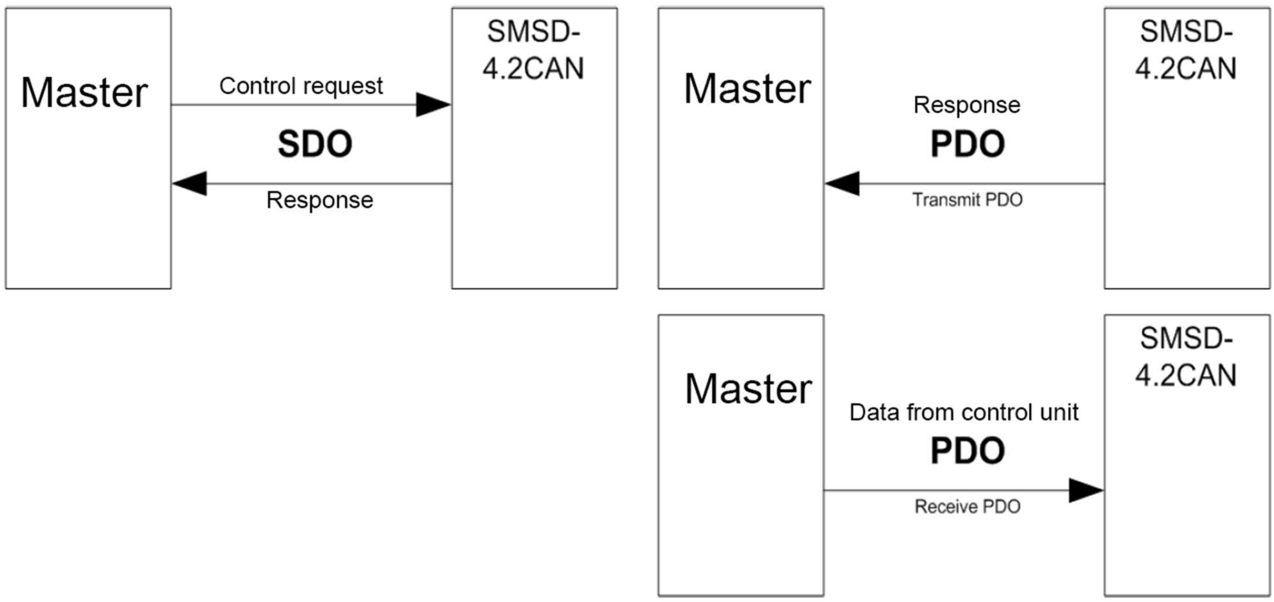
When the node number and transmission rate have been set, CANopen communication can be activated.

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 socalled SDOs), and an unconfirmed access type, in which no acknowledgement is made (via so-called PDOs).



Pic. 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
SYNC	Synchronisation Message	Synchronisation of multiple CAN nodes
EMCY	Emergency Message	Transmission of error messages
NMT	Network Management	Network service.
HEARTBEAT	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:

↓ Number of data bytes (here 8))

↓ Data bytes

601h	Len	D0	D1	D2	D3	D4	D5	D6	D7

↑ Identifier

Pic. 3.2 — 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 600h + node number of the applicable motor controller. The motor controller answers correspondingly with the identifier 580h + 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
INT8	8 bit value with algebraic sign	-128...127
UINT8	8 bit value without algebraic sign	0...255
INT16	16 bit value with algebraic sign	-32768 … 32767
UINT16	16 bit value without algebraic sign	0 … 65535
INT32	32-bit value with algebraic sign	$-(2^{31})\ldots(2^{32}-1)$
UINT32	32 bit value without algebraic sign	$0\ldots(2^{32}-1)$
REAL32	32-bit value with floating point	$-3.4\times10^{38}\ldots+3.4\times10^{38}$

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 bits	32 bits
Task identifier	2Fh	2Bh	23h
Response identifier	4Fh	4Bh	43h
Response identifier in case of error	-	-	80h

Tab 3.3 — SDO – response/task identifier.

EXAMPLE

UINT8/INT8	Reading of Obj. 6061_00h Return data: 01h	Writing of Obj. 1401_02h Data: EFh
Command	40h 61h 60h 00h	2Fh 01h 14h 02h EFh
Response:	4Fh 61h 60h 00h 01h	60h 01h 14h 02h
UINT16/INT16	Reading of Obj. 6041_00h Return data: 1234h	Writing of Obj. 6040_00h Data 03E8h
Command	40h 41h 60h 00h	2Bh 40h 60h 00h E8h 03h
Response:	4Bh 41h 60h 00h 34h 12h	60h 40h 60h 00h
UINT32/INT32	Reading of Obj. 6093_01h Return data: 12345678h	Writing of Obj. 6093_01h Data 12345678h
Command	40h 93h 60h 01h	23h 93h 60h 01h 78h 56h 34h 12h
Response:	43h 93h 60h 01h 78h 56h 34h 12h	60h 93h 60h 01h

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:

Command	23h	41h	60h	00h				
Response:	80h	41h	60h	00h	02h	00h	01h	06h
	↑				↑	↑	↑	↑
	Error identifier				Error code (4 byte)			

Pic 3.3 — Error message design.

Error code	Significance
05030000h	Protocol error: Toggle bit was not revised
05040005h	RAM Error
06010000h	Access type is not supported.
06010001h	Read access to an object that can only be written.
06010002h	Write access to an object that can only be read.
06020000h	The addressed object does not exist in the object directory.
06040041h	The object must not be entered into a PDO (e.g. ro-object in RPDO).
06040042h	The length of the objects entered in the PDO exceeds the PDO length.
06040043h	General parameter error.
06040047h	Overflow of an internal variable / general error.
06070010h	Protocol error: Length of the service parameter does not agree.
06070012h	Protocol error: Length of the service parameter is too large.
06070013h	Protocol error: Length of the service parameter is too small.
06090030h	The data exceed the range of values of the object.
06090031h	The data are too large for the object.
06090032h	The data are too small for the object.
08000000h	General error.

Tab 3.4 — List of SDO Error Messages.

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. 3.5 — PDO types.

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.

Number of data bytes (here 8)
↓

181h	Len	D0	D1	D2	D3	D4	D5	D6	D7

↑ Identifier

↑ Start actual position (D0...D3)

↑ Start actual speed (D4...D7)

Pic 3.4 – PDO Message Design.

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	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.
Number of entries	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).
PDO Mapping Entry... PDO Mapping Entry_8	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. The mapping information is gathered in table 3.7.
Transmission Type, Inhibit Time	Which event results in sending (transmit PDO) or evaluation (receive PDO) of a message can be determined for each PDO. (Table 3.8).
TPDOx Transmit mask low, TPDOx Transmit mask high	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. 3.6 — Object Description.

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

Tab. 3.7 — 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
00h	SYNC Message. Not cyclical. PDO will be analyzed immediately after the SYNC message.	RPDO
01h–F0h	SYNC Message. Cyclical. The numerical value specifies how many SYNC messages have to be received before the PDO – is sent (T-PDO) or – evaluated (R-PDO).	TPDO RPDO
FCh	Inquiry. Sending (TPDO) for SYNC after RTR request.	TPDO
FDh	Inquiry. Sending (TPDO) immediately after the RTR request.	TPDO
FEh	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.	TPDO (RPDO)
FFh	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.	TPDO

Tab 3.8 — 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	6041h_00h	Controller regulation
Modes_of_operation_display	6061h_00h	Operating mode
Digital_inputs	60FDh_00h	Digital inputs

CANopen access procedure

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, 187h should be used.

1	Deactivating PDO If the PDO is active, it must first be deactivated. Writing the identifier with set bit 31 (PDO is deactivated):	COB ID = C0000187h
2	Deleting number of objects Set the number of objects to zero in order to be able to change the object mapping.	Number of entries = 0
3	Parametrisation of objects that are to be mapped The above-listed objects must be combined into a 32 bit value:	
	Index= 6041h Sub-index=00h Length= 10h	Mapping Entry = 60410010h
	Index= 6061h Sub-index=00h Length= 10h	Mapping Entry_2 = 60610010h
	Index= 60FDh Sub-index=00h Length= 20h	Mapping Entry_3 = 60FD0020h
4	Parametrisation of number of objects The PDO should contain 3 objects	Number of entries = 3h
5	Parametrisation of transmission type The PDO should be sent when changes (to the digital inputs) are sent.	Transmission Type = FFh
	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 60FDh “come through”.	Transmit mask high = 00FFFF00h Transmit mask low = 00000000h
	The PDO should be sent no more than every 10 ms (100d).	Inhibit Time = 64h
6	Parametrisation of identifiers The PDO should be sent with identifier 187h. Write the new identifier and activate the PDO through deletion of bit 31.	COB ID = 187h

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

3.3.2. Objects for PDO Par ametrisation

The motor controllers of theR272-42-CAN have available a total of 4 out of 8 transmit and 4 out of 8receive 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	1800h
Name	Transmit PDO Communication Parameter 0
Object Type	RECORD
Number of entries	3

Sub-Index	01h
Description	COB ID
Data Type	UINT32
Access	RW
PDO Mapping	NO
Units	-
Value Range	181h … 1FFh, bit 30 and 31 may be set
Default Value	C0000180h

Sub-Index	02h
Description	Transmission Type
Data Type	UINT8
Access	RW
PDO Mapping	NO
Units	-
Value Range	01h…F0h, FEh, FFh
Default Value	FFh

Sub-Index	03h
Description	Inhibit Time
Data Type	UINT16
Access	RW
PDO Mapping	NO
Units	100 µs (i.e. 10 = 1ms)
Value Range	-
Default Value	0

Index	1A00h
Name	Transmit PDO Mapping Parameter 0
Object Type	RECORD
Number of entries	8

Sub-Index	00h
Description	Number of entries
Data Type	UINT32
Access	RW
PDO Mapping	NO
Units	-
Value Range	0...8
Default Value	-

Sub-Index	01h
Description	PDO Mapping Entry
Data Type	UINT32
Access	RW
PDO Mapping	NO
Units	-
Value Range	-
Default Value	-

Sub-Index	02h
Description	PDO Mapping Entry_2
Data Type	UINT32
Access	RW
PDO Mapping	NO
Units	-
Value Range	-
Default Value	-

Sub-Index	03h
Description	PDO Mapping Entry_3
Data Type	UINT32
Access	RW
PDO Mapping	NO
Units	-
Value Range	-
Default Value	-

Sub-Index	04h
Description	PDO Mapping Entry_4
Data Type	UINT32
Access	RW
PDO Mapping	NO
Units	-
Value Range	-
Default Value	-

Sub-Index	05h
Description	PDO Mapping Entry_5
Data Type	UINT32
Access	RW
PDO Mapping	NO
Units	-
Value Range	-
Default Value	-

Sub-Index	08h
Description	PDO Mapping Entry_8
Data Type	UINT32
Access	RW
PDO Mapping	NO
Units	-
Value Range	-
Default Value	-

Objects parameterization RPDO have a similar structure with objects parameterization TPDO. Communication Parameter RPDO - 1400h ... 1407h. In the RPDO object Inhibit Time is not used. Mapping Parameter - 1600h ... 1607h.

Observe that the object groups transmit_pdo_parameter_N and transmit_pdo_mapping_*N (N —PDO value)* can only be written when the PDO is deactivated (bit 31 in cob id is set)

Groups of objects 2014h ... 2017h - TPDO Transmit mask.

TPDO Transmit mask and Transmit PDO are not related to each other by values, but are connected in order. For example, active TPDO3, TPDO4 and TPDO7, but TPDO4 does not use the transfer mask, then TPDO3 uses TPDO1 Transmit mask, and TPDO7 uses TPDO3 Transmit mask.

CANopen access procedure

Index	Comment	Type	Access	Default value
2014h_00h	Number of entries	UINT8	RO	2
2014h_01h	TPDO1 Transmit mask low	UINT32	RW	0xFFFFFFFF
2014h_02h	TPDO1 Transmit mask high	UINT32	RW	0xFFFFFFFF

2015h ... 2017h Object groups have a similar structure with a group of objects 2014h.

3.3.3. Activation of PDOs

To receive or send PDO, the following points must be met:

– The object Number of entries of PDO Mapping Parameters must not equal zero.

– In the object cob_id, bit 31 must be deleted.

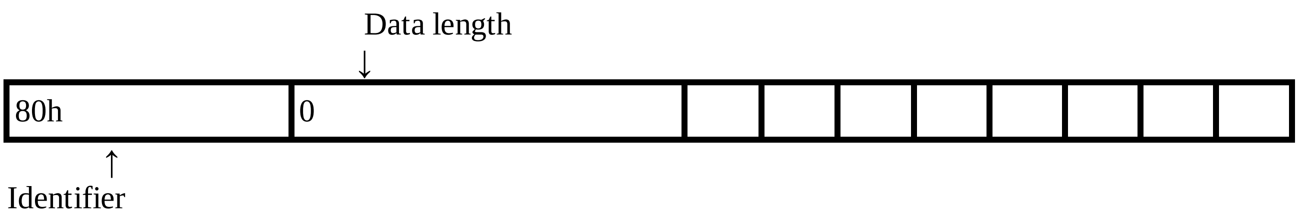
– The communication status of the motor controller must be operational) (chapter 3.6).

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).



Pic. 3.5 — SYNC Message design.

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 Type	VAR
Data Type	UINT32
Access	RW
PDO Mapping	NO
Units	-
Value Range	00000080h
Default Value	00000080h

Writing a value in object 1006h resets the counter synchronization messages.

3.5. EMERGENCY Message

The motor controller monitors the operation of its most important components. These include the power supply system, output stage, limit switches, temperature sensor. The cause of error messages can also be incorrect parameterization (division by zero, etc.).

3.5.1. EMERGENCY Message Design

When an error occurs, the motor controller displays an EMERGENCY Message. The identifier of this message is formed from identifier 80h and the node number of the corresponding engine controller.

EMERGENCY Message consists of eight bytes of data (the first two bytes are error_code), which are presented in Section 3.5.3. The third byte contains an additional error code (object1001h). The remaining five bytes contain zeros.

Data length ↓ Error Register (object 1001h) ↓

81h	8	E0	E1	R0	0	0	0	0	0

↑ Identifier: 80h + node number ↑ ↑ Error Code

Pic 3.6 —SYNC message design.

Error Register (R0)		
Bit	M/O[1)]	Significance
0	M	**Generic error**. Error is present
1	O	**Current**. Current error. For example, excess current in the motor winding.
2	O	**Voltage**. Voltage monitoring error.
3	O	**Temperature**: motor overtemperature
4	O	**Communication error**.: (overrun, error state)
5	O	-
6	O	-
7	O	**Software error**. Software error. For example, an error when dividing by zero.
Values: 0 = no error; 1 = error present 1) M = mandatory / O = optional.		

Tab. 3.9 —Error Register design.

3.5.2. Description of the Objects

Object 1003h: Predefined Error Field

The respective error_code of the error messages is also stored in a hex-stage error memory. This is structured like a shift register, so that the last occurring error is always stored in the object1003h_01h (Standard Error Field). Through read access on the object 1003h_00h (Number of Errors) it can be determined how many error messages are currently stored in the error memory. The error memory is cleared by writing the value 00h into the object 1003h_00h (pre_defined_error_field_0). Since objects having a sub-index from the first to the sixteenth are of the same type, only the object with the first subindex will be described below.

CANopen access procedure

Index	1003h
Name	Predefined Error Field
Object Type	ARRAY
Number of Errors	16

Sub-Index	01h
Description	Standard Error Field
Data Type	UINT32
Access	RO
PDO Mapping	NO

3.5.3. Error Codes

Error Code	**Transcript**
8220h	Communication error. Incorrect size of the PDO.
3101h	The supply voltage outside the specified value (min / max).
2301h	Overcurrent output stage.
3201h	Power supply gate output stage below nominal.
3103h	Supply voltage drop below normalized.
4201h	final-stage temperature exceeded 125 ° C.
4202h	Final-stage temperature exceeded 150C, shutdown stage.
4203h	Final-stage temperature exceeded 150C, disabling stage and controls it.
4204h	Predetermined temperature threshold is exceeded
FF01h	Wedging rotor (only for the voltage control mode).
FF02h	The final stage is unlocked. The shaft rotates freely.
2302h	Exceeding the rated current in the circuit 5B.
3100h	Supply voltage outside the specified value at the time of initialization.
3102h	Failure final-stage control system at the time of initialization.
6101h	Firmware error, the error when setting the parameters.
6102h	Error firmware in ROM write error
6201h	IN user error. Error generator curve movements.
FF03h	Error end sensors. Actuation of the opposite sensor or sensors simultaneous operation.
6103h	Buffer overflow.
6104h	Error data bus.
6202h	Error user data. Division by zero.
6203h	Starting the search starting point moving method when the actuator 35.
8111h	Communication error. Loss of CAN-frame.
6105h	Error firmware. Critical error.

Error Code	**Transcript**
6106h	Error firmware. Critical error internal bus.
8130h	Life guard Error

Tab. 3.10 Error codes

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 (000h). 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:

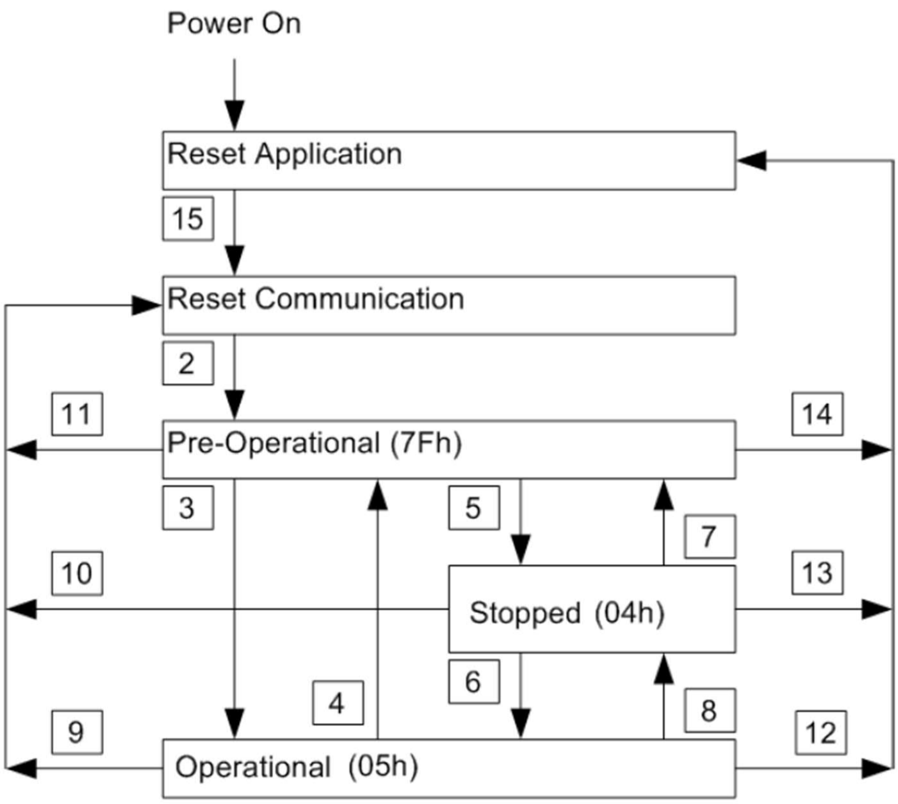
Data length ↓ NodeID ↓

000h	2	CS	NI						

↑ Identifier 000h ↑ Command soecifier

Pic 3.7 —NMT message structure.

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.



Pic. 3.8 – Status diagram

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

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

Tab. 3.11 — 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
Pre-Operational	Communication via SDOs possible; PDOs not active (no sending/evaluating)	V	-	V
Operational	Communication via SDOs possible; all PDOs active (sending/ evaluating)	V	V	V
Stopped	No communication except for heartbeating	-	-	V

Tab. 3.12 — 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).

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.

Data length

701h	1	0							

Identifier:
700h + node number

Bootup message Identifier

Pic 3.9 — Structure of the Bootup Message.

3.8. Heartbeat

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 (chapter. 3.9) are sent with the identifier 700h + 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 700h + node number. It contains only 1 byte of user data, the NMT status of the controller (chapter 3.6).

Data length

701h	1	N							

Identifier:
700h + node number

State NMT

Pic. 3.10 — Structure of the Heartbeat Message.

N	Significance
04h	Stopped
05h	Operational
7Fh	Pre-Operational

Tab. 3.13 —NMT condition.

CANopen access procedure

3.8.3. Description of the Objects

Object 1017h: Producer Heartbeat Time

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

Index	1017h
Name	Producer Heartbeat Time
Object Type	VAR
Data Type	UINT16
Access	RW
PDO Mapping	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.

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 700h + 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 700h + 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.

Remote bit (Remote frames have no data)

↓

701h	R	

↑

Identifier:
700h + node number

Pic 3.11 —Node Guarding structure.

CANopen access procedure

Data length

↓

701h	1	T/N							

↑ Identifier: 700h + node number

↑ Toggle bit / NMT status

Pic 3.12 —Node Guarding responce structure.

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

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

Tab. 3.14 — T/N byte structure.

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 is triggered. The toggle bit is reset through the NMT command Reset Communication.

3.9.3. Description of the Objects

Object 100Ch: 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 (100Ch) and life_time_factor (100Dh). 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	101Ch
Name	Guard Time
Object Type	VAR
Data Type	UINT16
Access	RW
PDO Mapping	NO
Units	ms
Value Range	0...65535
Default Value	0

Object 100Dh: Life Time Factor

The life time factor should be written with 1 in order to specify the guard_time directly.

Index	101Ch
Name	Life Time Factor
Object Type	VAR
Data Type	UINT8
Access	RW
PDO Mapping	NO
Units	-
Value Range	0...255
Default Value	0

3.9.4. Table of Identifiers

The following table gives an overview of the identifiers used:

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

4. 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.

4.1. 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.

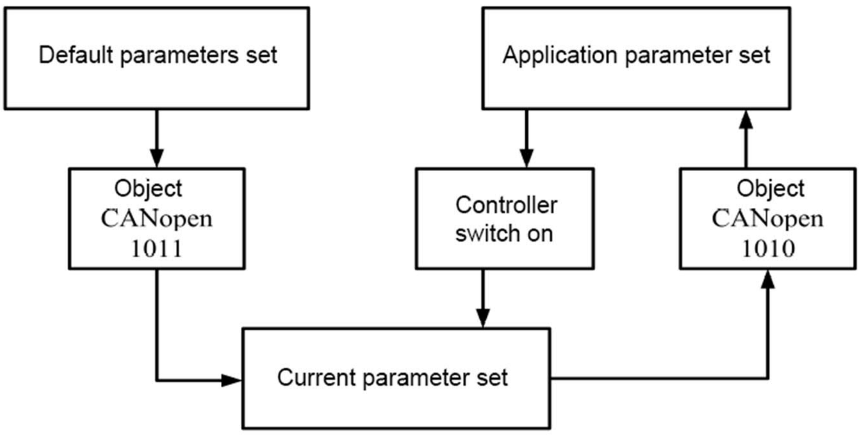
– Default parameters set

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 1011h_01h (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

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 1010h_01h (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



Pic. 4.1. - 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).

Setting parameters

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 1011h_01h (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 reallyincludes the parameters you want. An incorrectly parametrised controller can turn out of control and cause personal injury or property damage.

Index	1011h
Name	Restore Default Parameters
Object Type	ARRAY
Number of entries	3

Sub-Index	01h
Description	Restore all Default Parameters
Data Type	UINT32
Access	WO
PDO Mapping	NO
Units	-
Value Range	64616F6Ch(“load”)
Default Value	-

Sub-Index	02h
Description	Restore Communication Default Parameters
Data Type	UINT32
Access	WO
PDO Mapping	NO
Units	-
Value Range	64616F6Ch(“load”)
Default Value	-

Setting parameters

Sub-Index	03h
Description	Restore Motor Default Parameters
Data Type	UINT32
Access	WO
PDO Mapping	NO
Units	-
Value Range	64616F6Ch(“load”)
Default Value	-

Signature	MSB			LSB
ASCII	d	a	o	l
hex	64h	61h	6Fh	6Ch

Tab. 4.1 — Example for ASCII text “load”.

The object 1011h_01h (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. If the a wrong command was sent or rong identificator is used a eroor message will be generated. The CAN communication parameters (node no., baud rate and operating mode) remain unchanged.

Object 1010h: Store Parameter Field

Index	1010h
Name	Store Parameter Field
Object Type	ARRAY
Number of entries	3

Sub-Index	01h
Description	Save all Parameters
Data Type	UINT32
Access	WO
PDO Mapping	NO
Units	-
Value Range	65766173h(“save”)
Default Value	-

Setting parameters

Sub-Index	02h
Description	Save Communication Parameters
Data Type	UINT32
Access	WO
PDO Mapping	NO
Units	-
Value Range	65766173h(“save”)
Default Value	-

Sub-Index	03h
Description	Save Motor Parameters
Data Type	UINT32
Access	WO
PDO Mapping	NO
Units	-
Value Range	65766173h(“save”)
Default Value	-

Signature	MSB			LSB
ASCII	e	v	a	s
hex	65h	76h	61h	73h

Tab. 4.2 — Example of ASCII “save”.

The commands for saving parameters and loading parameters by default are possible only in the Pre-Operational state.

4.2. 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, hat 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.
6510h_01h	VAR	Compatibility_control	UNSIGNED16	RW

Object 6510h_01h: Compatibility_control

Sub-Index	01h
Description	Compatibility_control
Data Type	UINT16
Access	RW
PDO Mapping	NO
Units	-
Value Range	-
Default Value	19h

Bit	**Value**	**Name**	**Description**
0	0001h	save_type	Bit set - saving to non-volatile memory is performed in a single cycle and the response is received after saving. The bit is reset - the answer is given immediately, while saving continues in the background and takes up to six minutes, you should consider this fact and do not turn off the power immediately after receiving a response.
1	0002h	low_speed_optimization	The set bit includes speed optimization at low speed, only for voltage control mode.
2	0004h	output_inversion	The digital outputs are galvanically isolated and represent an open collector, in some cases it is necessary to invert the signal to them. Implemented by the bit described.
3	0008h	emergence_type	Bit set - the identifier of the error message is generated as 80h + NodeID, if it has not been changed during the work. Bit cleared is an random identifier that is initialized from nonvolatile memory.

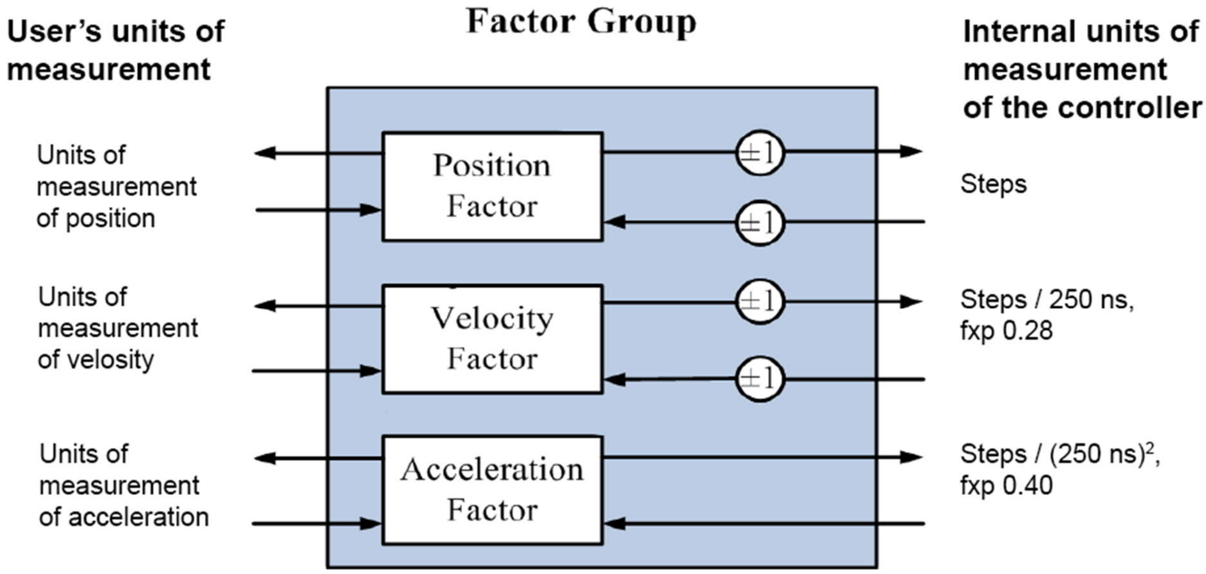
Setting parameters

<table>
<tr><th>Bit</th><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>4,5</td><td>0030h</td><td>input_delay</td><td>The bits are analyzed together and set the processing delay of the signal at the discrete inputs, according to the table below:
<table>
<tr><th>Bit 5</th><th>Bit 4</th><th>Delay value, ms</th></tr>
<tr><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>5</td></tr>
<tr><td>1</td><td>0</td><td>10</td></tr>
<tr><td>1</td><td>1</td><td>20</td></tr>
</table>
Use of this option is relevant when the "bounce" effect of the end and reference sensors is present.</td></tr>
</table>

4.3. 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:



Pic. 4.2. – 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:

Setting parameters

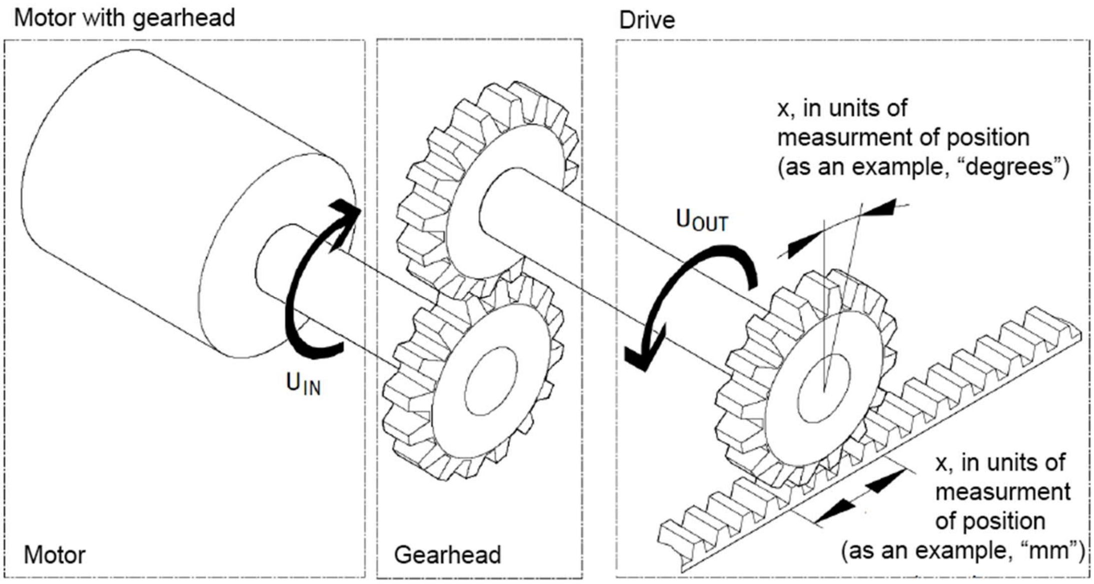
Size	Measurement file	Unit	Explanation
Length	Speed units	Steps (Microsteps)	When using micro-stepping modes, the movement is calculated in them, while the speed and acceleration are in full steps.
Speed	Speed units	Steps/s	Full steps per second.
Acceleration	Acceleration units	Steps/s^2	Rotational speed increase per time unit.

Description of the Objects
Objects treated in this chapter

Index	Object	Name	Type	Attr.
607Eh	VAR	Polarity	UNSIGNED8	RW
6093h	ARRAY	Position_factor	UNSIGNED32	RW
6095h	ARRAY	Velocity_factor	UNSIGNED32	RW
6097h	ARRAY	Acceleration_factor	UNSIGNED32	RW
2041h	ARRAY	Main_velocity_factor	UNSIGNED32	RW
2042h	ARRAY	Main_acceleration_factor	UNSIGNED32	RW
2043h	ARRAY	FS_SPD_factor	UNSIGNED32	RW
2044h	ARRAY	MAX_SPEED_factor	UNSIGNED32	RW

4.3.1. Object 6093h: Position_factor

The Position_factor object is used to recalculate all units of length in this application from positon_units to increments of the internal unit of measure (for example, in full-step mode, the increment value is one step, for the microstep increment, for 1/64 - 1/64 total stepping, etc .). It consists of a numerator and a denominator.



Pic. 4.3. Calculating the positioning units.

Setting parameters

Index	6093h
Name	Position_factor
Object Type	ARRAY
Number of entries	2

Sub-Index	01h
Description	Position_factor_Numerator
Data Type	UINT32
Access	RW
PDO Mapping	YES
Units	-
Value Range	-
Default Value	1

Sub-Index	02h
Description	Position_factor_Divisor
Data Type	UINT32
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 (UIN) and revolutions at the drive-out (UOUT)
feed_constant	Ratio between revolutions at the drive-out (UOUT) nd movement in position_ units (e.g., 25600 microsteps = 360 degrees)

The position_factors is calculated using the following formula:

$$\text{Position\_factor} = \frac{numerator}{divisor} = \frac{gear\_ratio}{feed\_constant}$$

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.

i The position_factor must not be larger than 2^{24}

EXAMPLE

First, the desired unit (column 1) and the desired number of decimal places (NK) 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: In the example, the fragmentation of the complete step 1/128 will be used with a motor that has 200 complete steps per revolution, so 1 revolution is **25600 microsteps (uStep)**.
The values of the 2041h ... 2044h objects are set by default, so the movement, drive speed, acceleration, without taking into account objects 6093h, 6095h, 6097h, is measured in microsteps, steps per second, steps per second per second.

Position factor calculation sequence				
Position units	**Feed constant**	**Gear ratio**	**Formula**	**Result shortened**
Degree, 1NK →1/10 degree (°/10)	1 $U_{OUT} = 3600\frac{°}{10}$	1/1	$\frac{\frac{1}{1}*25600uStep}{3600\frac{°}{10}} = \frac{25600uStep}{3600\frac{°}{10}}$	$\frac{numerator: 64}{divisor: 9}$

Examples of calculating the position factor				
Position units[1]	**Feed constant**[2]	**Gear ratio**	**Formula**	**Result shortened**
Microsteps, 0NK →uStep	1 U_{OUT} = 25600 uStep	1/1	$\frac{\frac{1}{1}*25600uStep}{25600uStep} = \frac{1uStep}{1uStep}$	$\frac{numerator: 1}{divisor: 1}$
Degree, 1NK →1/10 degree (°/10)	1 $U_{OUT} = 3600\frac{°}{10}$	1/1	$\frac{\frac{1}{1}*25600uStep}{3600\frac{°}{10}} = \frac{25600uStep}{3600\frac{°}{10}}$	$\frac{numerator: 64}{divisor: 9}$
Rev., 2NK →1/100 Rev. (R/100)	1 $U_{OUT} = 100\frac{U}{100}$	1/1	$\frac{\frac{1}{1}*25600uStep}{100\frac{1}{100}} = \frac{25600uStep}{100\frac{1}{100}}$	$\frac{numerator: 256}{divisor: 1}$
		2/3	$\frac{\frac{2}{3}*25600uStep}{100\frac{1}{100}} = \frac{51200uStep}{300\frac{1}{100}}$	$\frac{numerator: 512}{divisor: 3}$
mm, 1NK →1/10 mm (mm/10)	1 $U_{OUT} = 325{,}5\frac{mm}{10}$	4/5	$\frac{\frac{4}{5}*25600uStep}{325{,}5\frac{mm}{10}} = \frac{204800uStep}{3255\frac{mm}{10}}$	$\frac{numerator: 40960}{divisor: 651}$

1) Desired unit at the drive-out

2) Positioning units per revolution at the drive-out (U_{OUT}).

Feed constant of the drive* 10^{-NK} (points after the decimal).

4.3.2. Object 6095h: Velocity_factor

The object velocity_factor converts all speed values of the application from speed_units into the internal units. It consists of numerator and denominator.

Index	6095h
Name	Velocity_factor
Object Type	ARRAY
Number of entries	2

Sub-Index	01h
Description	Velocity_factor_Numerator
Data Type	UINT32
Access	RW
PDO Mapping	YES
Units	-
Value Range	-
Default Value	1

Sub-Index	02h
Description	Velocity_factor_Divisor
Data Type	UINT32
Access	RW
PDO Mapping	YES
Units	-
Value Range	-
Default Value	1

Following values are used in The Velocity_factor formula:

Parameters	**Description**
gear_ratio	The ratio between the input revolutions (UIN) and the output revolutions (UOUT)
feed_constant	The ratio between the output revolutions (UOUT) and the movement in position_units (for example, 200 steps = 360 degrees)
time_factor_v	Ratio between the internal unit of time and the user-defined unit of time. (for example, 1 min = 60 s)

The calculation of the velocity_factor uses the following equation:

$$\text{Velocity\_factor} = \frac{numerator}{divisor} = \frac{gear\_ratio \times time\_factor\_v}{feed\_constant}$$

The velocity_encoder_factor must not be greater than 2^{24}

Movement in units of speed, acceleration, and deceleration position_units is specified in full steps.

Setting parameters

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 (NK) 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. As in the previous example, one revolution of the motor shaft corresponds to 25600 microsteps (uStep) and 200 (Step) full steps. Do not forget to take into account the fact that the speed and acceleration of the controller are measured in full steps with the default objects 2041h, 2042h, 2043h, 2044h.

Velocity factor calculation sequence					
SpeedUnits	**Feed Constant**	**Time Constant**	**Gear Ratio**	**Formula**	**Result shortened**
cm/min, 1NK →1/10cm/min (cm/10min)	$32{,}55\frac{mm}{U} \rightarrow$ 1 U_{OUT} = $32{,}55\ \frac{cm}{10}$	$1min=$ =60s	4/5	$\frac{\frac{4}{5}\times\frac{1s}{60s}}{32{,}55\frac{cm}{10}200\ Step}=\frac{800}{9765}$	$\frac{numerator: 160}{divisor: 1953}$

Examples of calculating the speed factor					
Speed units[1)]	**Feed const.**[2)]	**Time Constant**	**Gear Ratio**	**Formula**	**Result Shortened**
Rev/min, 0NK → Rev/min	1 U_{OUT} = 1 U_{OUT}	$1min=$ $60s$	1/1	$\frac{\frac{1}{1}\times\frac{1s}{60s}}{1\,rev./200Step}=\frac{200}{60}$	$\frac{numerator: 10}{divisor: 3}$
Rev/min, 2NK →1/100 Rev/min (Rev/min)	1 U_{OUT} = $100\frac{U}{100}$	$1min=$ $60s$	2/3	$\frac{\frac{2}{3}\times\frac{1s}{60s}}{100\frac{rev.}{100}/200Step}=\frac{400}{18000}$	$\frac{numerator: 1}{divisor: 45}$
°/s, 1NK →1/10 °/s (°/10s)	1 U_{OUT} = $3600\frac{°}{10}$	$1\frac{1}{s}$	1/1	$\frac{\frac{1}{1}\times\frac{1s}{1s}}{3600\frac{°}{10}/200Step}=\frac{200}{3600}$	$\frac{numerator: 1}{divisor: 18}$
cm/min, 1NK →1/10 cm/min (cm/min)	$32{,}55\frac{mm}{U} \rightarrow$ 1 U_{OUT} = $32{,}55\frac{cm}{10}$	1min= $60s$	4/5	$\frac{\frac{4}{5}\times\frac{1s}{60s}}{32{,}55\frac{cm}{10}/200Step}=\frac{800}{9765}$	$\frac{numerator: 160}{divisor: 1953}$

1) Desired unit at the drive-out

2) Positioning units per revolution at the drive-out (U_{OUT}).

Feed constant of the drive * 10^{-NK} (points after the decimal).

4.3.3. Object 6097h: Acceleration_factor

The Object Acceleration_factor converts all acceleration values of the application from acceleration_units into the internal unit. It consists of numerator and denominator.

Index	6097h
Name	Acceleration_factor
Object Type	ARRAY
Number of entries	2

Sub-Index	01h
Description	Acceleration_factor_Numerator
Data Type	UINT32
Access	RW
PDO Mapping	YES
Units	-
Value Range	-
Default Value	1

Sub-Index	02h
Description	Acceleration_factor_Divisor
Data Type	UINT32
Access	RW
PDO Mapping	YES
Units	-
Value Range	-
Default Value	1

In the calculation formula Acceleration_factor the following values are used:

Parameters	Description
gear_ratio	The ratio between the input revolutions (UIN) and the output revolutions (UOUT)
feed_constant	The ratio between the output revolutions (UOUT) and the movement in position_units (for example, 200 steps = 360 degrees)
time_factor_a	the ratio between the internal unit of time squared and the user-defined unit of time squared. (example, 1 min^2 = 3600 s^2)

Calculation of the acceleration_factor uses the following equation:

$$\text{Acceleration\_factor} = \frac{numerator}{divisor} = \frac{gear\_ratio \times time\_factor\_a}{feed\_constant}$$

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 (NK) 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. As in the previous example, one revolution of the motor shaft corresponds to 25600 microsteps (uStep) and 200 (Step) full steps. Do not forget to take into account the fact that the speed and acceleration of the controller are measured in full steps with the default objects 2041h, 2042h, 2043h, 2044h.

Process of calculating the acceleration factor					
Units of acceleration	Feed const.	Time Constant	Gear	Formula	Result Shortened
cm/min^2, 1NK →1/10 cm/min^2 ($cm/10min^2$)	$32{,}55\frac{mm}{U} \rightarrow$ 1 U_{OUT} = $32{,}55\frac{cm}{10}$	$1min^2$= =$3600s^2$	4/5	$\frac{\frac{4}{5}\times\frac{1s^2}{3600s^2}}{32{,}55\frac{mm}{10}/200Step}=\frac{800}{585900}$	$\frac{numerator: 8}{divisor: 5859}$

Examples of calculating the acceleration factor					
Units of acceleration [1]	Feed const[2]	Time Constant	Gear	Formula	Result Shortened
$°/s^2$, 1NK →1/10 $°/s^2$ ($°/10s^2$)	1 U_{OUT} = $3600\frac{°}{10}$	$1s^2$	1/1	$\frac{\frac{1}{1}\times\frac{1s^2}{1s^2}}{3600\frac{°}{10}/200Step}==\frac{200}{3600}$	$\frac{numerator: 1}{divisor: 18}$
Rev./min.^{2}, 2NK →1/100 Rev/min^2 ($Rev./100min^2$)	1 U_{OUT} = $100\frac{U}{100}$	min= $3600s^2$	2/3	$\frac{\frac{2}{3}\times\frac{1s^2}{3600s^2}}{100\frac{1}{100}/200Step}=\frac{400}{1080000}$	$\frac{numerator: 1}{divisor: 2700}$
cm/min^2, 1NK →1/10 cm/min^2 ($cm/10min^2$)	$32{,}55\frac{mm}{U} \rightarrow$ 1 U_{OUT} = $32{,}55\frac{cm}{10}$	$1min^2$= $3600s^2$	4/5	$\frac{\frac{4}{5}\times\frac{1s^2}{3600s^2}}{32{,}55\frac{cm}{10}/200Step}=\frac{800}{585900}$	$\frac{numerator: 8}{divisor: 5859}$

1) Desired unit at the drive-out.

2) Positioning units per revolution at the drive-out (U_{OUT}).

Feed constant of the drive* 10^{-NK} (points after the decimal).

4.3.4. Object 607Eh: 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	607Eh
Name	Polarity
Object Type	VAR
Data Type	UINT8
Access	RW
PDO Mapping	YES
Units	-
Value Range	40h, 80h, C0h
Default Value	0

Bit	Value	Name	Significance
6	40h	velocity_polarity_flag	0: multiply by 1 (default) 1: multiply by -1 (inverse)
7	80h	position_polarity_flag	0: multiply by 1 (default) 1: multiply by -1 (inverse))

Tab. 4.3 — Significance of bit polarity_flag.

4.3.5. Objects 2041h...2044h Overview

Since the internal units of the motor controller have different accuracy for different applications, objects 2041h, 2042h, 2043h, 2044h are entered. Their main purpose: sharing with objects 6093h, 6095h, 6097h and reduction to single units of measurement (objects 2041h, 2043h, 2044h refer to the representation of the unit of measurement of speed, but having different accuracy, they must necessarily be reduced to a single measure). So, by default, the objects 2041h ... 2044h are assigned values that reduce the internal units of the engine controller's speed (steps of 250ns) to steps per second (Step / s), acceleration (steps of 250ns squared) to steps per second in a square ((Step/s^2). If necessary, the values of these objects can be changed .

4.3.6. Object 2041h: Main_velocity_factor

The purpose and structure are similar to the 6095h object. Action extends to objects: 606Bh, 606Ch, 60FFh.

Index	2041h
Name	Acceleration_factor
Object Type	ARRAY
Number of entries	2

Setting parameters

Sub-Index	01h
Description	Main_velocity_factor_numerator
Data Type	UINT32
Access	RW
PDO Mapping	YES
Units	-
Value Range	-
Default Value	1048576

Sub-Index	02h
Description	Main_velocity_factor_divisor
Data Type	UINT32
Access	RW
PDO Mapping	YES
Units	-
Value Range	-
Default Value	15625

For this application, the speed in internal units is measured in Step / Tick and has a record format with a fixed point of 0.28, where Tick is 250 ns. The formula for translating this internal velocity unit in Step / s is as follows:

$$[Step/s] = \frac{SPEED \times 2^{-28}}{Tick},$$

SPEED – internal speed measurment unit.

In steps per second, the speed of the drive varies from 0 to 15625 in steps of 0.015 Step / s. Thus, the gradation in the speed of the drive speed must be taken into account.

4.3.7. Object 2042h: Main_acceleration_factor

The purpose and structure is similar to the object 6097h. The action extends to objects: 6083h, 6084h.

Index	2042h
Name	Main_acceleration_factor
Object Type	ARRAY
Number of entries	2

Setting parameters

Sub-Index	01h
Description	Main_acceleration_factor_numerator
Data Type	UINT32
Access	RW
PDO Mapping	YES
Units	-
Value Range	-
Default Value	16777216

Sub-Index	02h
Description	Main_acceleration_factor_divisor
Data Type	UINT32
Access	RW
PDO Mapping	YES
Units	-
Value Range	-
Default Value	244140625

For this application, the speed in internal units is measured in Step / Tick2 and has a fixed point record format of 0.40, where Tick is 250 ns. The formula for translating this internal velocity unit in Step/s^2 is as follows:

$$[Step/s^2] = \frac{ACC \times 2^{-40}}{Tick},$$

$$[Step/s^2] = \frac{DEC \times 2^{-40}}{Tick},$$

where ACC and DEC are the internal units of acceleration and deceleration.

The values in Step/s^2 range from 14.55 to 59590 in steps of 14.55 Step/s^2. Thus, the gradation of the acceleration / deceleration of the drive must be taken into account.

4.3.8. Object 2043h: FS_SPD_factor

The purpose and structure are similar to the 6095h object. The action extends to the object 6510h_04h.

Index	2043h
Name	FS_SPD_factor
Object Type	ARRAY
Number of entries	2

Setting parameters

Sub-Index	01h
Description	FS_SPD_factor_numerator
Data Type	UINT32
Access	RW
PDO Mapping	YES
Units	-
Value Range	-
Default Value	1024

Sub-Index	02h
Description	FS_SPD_factor_divisor
Data Type	UINT32
Access	RW
PDO Mapping	YES
Units	-
Value Range	-
Default Value	15625

For this application, the speed in internal units is measured in Step / Tick and has a fixed point record format of 0.18, where Tick is 250 ns. The formula for translating this internal velocity unit in Step/s is as follows:

$$[Step/s] = \frac{(FS\_SPD+0{,}5)\times 2^{-18}}{Tick},$$

where FS_SPD is the internal unit for measuring the rate of transition to the full step.

In steps per second, the speed of the drive varies from 7.63 to 15625 in steps of 15.25 Step/s. Thus, the gradation in the speed of the drive speed must be taken into account.

4.3.9. Object 2044h: MAX_SPEED_factor

The purpose and structure are similar to the 6095h object. The action extends to objects:6081h, 6099h_01h, 6099h_02h.

Index	2044h
Name	MAX_SPEED_factor
Object Type	ARRAY
Number of entries	2

Setting parameters

Sub-Index	01h
Description	MAX_SPEED_factor_numerator
Data Type	UINT32
Access	RW
PDO Mapping	YES
Units	-
Value Range	-
Default Value	1024

Sub-Index	02h
Description	MAX_SPEED_factor_divisor
Data Type	UINT32
Access	RW
PDO Mapping	YES
Units	-
Value Range	-
Default Value	15625

For this application, the speed in internal units is measured in Step/Tick and has a fixed point record format of 0.18, where Tick is 250 ns. The formula for translating this internal velocity unit in Step/s is as follows:

$$[Step/s] = \frac{MAX\_SPD \times 2^{-18}}{Tick},$$

where MAX_SPD is the internal unit for measuring the maximum rotation speed.

In steps per second, the drive speed varies from 15.25 to 15610 in steps of 15.25 Step/s. Thus, the gradation in the speed of the drive must be taken into account.

4.4. CANopen data transfer rate

4.4.1. Object 2045h: CAN Baudrate

The value of this parameter specifies the data transfer rate. The condition for the entry into force of the entered data is to reboot the network part of the device, it is also necessary to store the device in the nonvolatile memory of the device to apply the parameter after the power is turned off. The possible parameters are given in Table 4.4. Other parameters are not allowed, otherwise the default speed is 500kbit / s.

Setting parameters

Index	2045h
Name	CAN Baudrate
Object Type	VAR
Data Type	UINT8
Access	RW
PDO Mapping	NO
Units	-
Value Range	5, 12, 25, 50, 100
Default Value	50

Object 2045h value, DEC	**CANopen data transfer rate, kbit/s**
5	50
12	125
25	250
50	500
100	1000

Tab. 4.4 — Data transfer rate.

4.5. Error processing

4.5.1. Overview

During the operation of the engine controller, errors and contingencies can occur, while the processing of a particular error can be determined by an action or by inaction (if possible). The error list is defined by objects 2024h_01h ... 2024h_10h. The value of the object is the action performed when the error is processed (refer to table 4.5)

Value	**Name**	**Description**
0h	No Action	No Action.
1h	Soft Stop	Braking, according to Profile_deceleration, with the subsequent transition to the hold mode (the shaft does not rotate).
2h	Soft HiZ	Braking, according to Profile_deceleration, with subsequent disconnection of the windings (the shaft rotates freely).
3h	Hard Stop	Immediate transition to the hold mode (the shaft does not rotate).
4h	Hard HiZ	Immediate switching off of the windings (the shaft rotates freely).
5h	Power OFF	Turn off the power of the final stage (not recommended for use without special instructions). Applying this action will move the controller to the Pre-Operational state. It should also be noted that after applying the voltage to the final stage, it will require initialization, which takes about one second.

Tab. 4.5 — Error processing actions.

4.5.2. Object 2024h_01h: Fault Event Node Guard out of timeout

Error handling Fault Event Node Guard out of timeout occurs if there is no Node Guard message in the given time period.

Setting parameters

Sub-Index	01h
Description	Fault Event Node Guard out of timeout
Data Type	UINT8
Access	RW
PDO Mapping	NO
Units	-
Value Range	0...4
Default Value	0

4.5.3. Object 2024h_02h: Fault Event Temperature out of limit

Fault Event Temperature out of limit error occurs when the threshold temperature of the final stage of the motor controller is exceeded. (2029h_03h Temperature sensor threshold).

Sub-Index	02h
Description	Fault Event Temperature out of limit
Data Type	UINT8
Access	RW
PDO Mapping	NO
Units	-
Value Range	0...4
Default Value	0

4.5.4. Object 2024h_03h: Fault Event Pulse Mode Error Write Config

Error handling Fault Event Pulse Mode Error Write Config occurs when the final stage in the driver mode is incorrectly configured (incorrect values are recorded), as well as when errors occur in "slow" storage in non-volatile memory (reffer to chapter. 4.2).

Sub-Index	03h
Description	Fault Event Pulse Mode Error Write Config
Data Type	UINT8
Access	RW
PDO Mapping	NO
Units	-
Value Range	3, 4
Default Value	4

4.5.5. Object 2024h_04h: Fault Event Pulse Mode Termal warning

Error Handling Fault Event Pulse Mode Termal warning occurs in case of excess temperature, the motor controller output stage + 125 ° C.

Setting parameters

Sub-Index	04h
Description	Fault Event Pulse Mode Termal warning
Data Type	UINT8
Access	RW
PDO Mapping	NO
Units	-
Value Range	3, 4
Default Value	4

4.5.6. Object 2024h_05h: Fault Event Controller Mode Error Write Config

Error handling Fault Event Controller Mode Error Write Config occurs if the output stage in the controller mode is incorrectly configured (incorrect values are recorded), as well as when errors are caused by "slow" saving to non-volatile memory (see paragraph4.2).

Sub-Index	05h
Description	Fault Event Controller Mode Error Write Config
Data Type	UINT8
Access	RW
PDO Mapping	NO
Units	-
Value Range	0...4
Default Value	0

4.5.7. Object 2024h_06h: Fault Event Controller Mode Termal warning

Error Handling Fault Event Controller Mode Termal warning occurs in case of excess temperature, during the motor controller output stage + 125 ° C.

Sub-Index	06h
Description	Fault Event Controller Mode Termal warning
Data Type	UINT8
Access	RW
PDO Mapping	NO
Units	-
Value Range	0...4
Default Value	0

4.5.8. Object 2024h_07h: Fault Event Controller Mode Termal shutdown

Critical error handling Fault Event Controller Mode Termal shutdown occurs in case of excess temperatureduring the motor controller output stage + 150C.

Setting parameters

Sub-Index	07h
Description	Fault Event Controller Mode Termal shutdown
Data Type	UINT8
Access	RW
PDO Mapping	NO
Units	-
Value Range	4,5
Default Value	4

4.5.9. Object 2024h_08h: Fault Event Controller Mode Stall detection

Error handling Fault Event Controller Mode Stall detection occurs when the motor stop threshold is exceeded (Object 6510h VM_STALL_TH).

Sub-Index	08h
Description	Fault Event Controller Mode Stall detection
Data Type	UINT8
Access	RW
PDO Mapping	NO
Units	-
Value Range	0...4
Default Value	0

4.5.10. Object 2024h_09h: Fault Event Controller Mode UVLO

Processing Fault Event Controller Mode UVLO critical error occurs in the case of final-stage power supply voltage drops below the gate output switches threshold UVLOVAL (Object 6510h_14h:. VM_CONFIG and 6510h_1Ch: SM_CONFIG).

Sub-Index	09h
Description	Fault Event Controller Mode UVLO
Data Type	UINT8
Access	RW
PDO Mapping	NO
Units	-
Value Range	4,5
Default Value	4

4.5.11. Object 2024h_0Ah: Fault Event Controller Mode ADC UVLO

Fault Event Controller Mode ADC UVLO error processing occurs in case the motor controller falling below a predetermined voltage.

Setting parameters

Sub-Index	0Ah
Description	Fault Event Controller Mode ADC UVLO
Data Type	UINT8
Access	RW
PDO Mapping	NO
Units	-
Value Range	0...4
Default Value	0

4.5.12. Object 2024h_0Bh: Fault Event Controller Mode CMD error

Object is reserved for future use. The value is always zero.

4.5.13. Object 2024h_0Ch: Fault Event Controller Mode Overcurrent

Processing Fault Event Controller Mode Overcurrent critical error occurs in the case of excess current flowing through the motor windings threshold (Object. 6510h_03h: OCD_TH).

Sub-Index	0Ch
Description	Fault Event Controller Mode Overcurrent
Data Type	UINT8
Access	RW
PDO Mapping	NO
Units	-
Value Range	4,5
Default Value	4

4.5.14. Object 2024h_0Dh: Fault Event Controller Mode 0 Divisor

Fault Event Controller Mode 0 Divisor is processed in case of division by zero, for example, with incorrectly selected Factor Group coefficients. In this case, the conversion factor, in the calculation of which an error occurred, will be assigned a value of one.

Sub-Index	0Dh
Description	Fault Event Controller Mode 0 Divisor
Data Type	UINT8
Access	RW
PDO Mapping	NO
Units	-
Value Range	0...4
Default Value	0

4.5.15. Object 2024h_0Eh: Fault Event Controller Mode CANopen Frame lost

Fault Event Controller Mode CANopen Frame loss occurs when a CANopen frame is lost when it attempts to transmit it. The error occurs when the line is broken.

Setting parameters

Sub-Index	0Eh
Description	Fault Event Controller Mode CANopen Frame lost
Data Type	UINT8
Access	RW
PDO Mapping	NO
Units	-
Value Range	0...4
Default Value	0

4.5.16. Object 2024h_0Fh: Fault Event Controller Mode Out 5V over current

Error handling Fault Event Controller Mode Out 5V over current occurs when the current consumption is exceeded by 1A on the + 5V bus of the I / O port [4].

Sub-Index	0Dh
Description	Fault Event Controller Mode 0 Divisor
Data Type	UINT8
Access	RW
PDO Mapping	NO
Units	-
Value Range	0...4
Default Value	0

4.5.17. Object 2024h_10h: Fault Event Controller Mode Curve Generator error

Error handling Fault Event Controller Mode Curve Generator error occurs when an error occurs in the motion curve generator.

Sub-Index	10h
Description	Fault Event Controller Mode Curve Generator error
Data Type	UINT8
Access	RW
PDO Mapping	NO
Units	-
Value Range	0...4
Default Value	0

The increment in the deceleration segment should not exceed 2^{24} *uStep*, otherwise the processing of the error event of the displacement curves generator will be triggered

Setting parameters

4.5.18. Object 2024h_11h: Fault Event Controller Mode Limit Switch error

The Fault Event Controller Mode Limit Switch error occurs when a sensor is triggered that is not supposed to trigger when using a particular method for finding the home position, or if several sensors are triggered simultaneously.

Sub-Index	11h
Description	Fault Event Controller Mode Limit Switch error
Data Type	UINT8
Access	RW
PDO Mapping	NO
Units	-
Value Range	3, 4
Default Value	4

4.5.19. Object 2024h_12h: Fault Event Controller Mode Home Method 35

The Fault Event Controller Mode Home Method 35 occurs when the start point search method 35 is started with the moving drive.

Sub-Index	12h
Description	Fault Event Controller Mode Home Method 35
Data Type	UINT8
Access	RW
PDO Mapping	NO
Units	-
Value Range	0...4
Default Value	0

4.5.20. A summary table of error handling methods

Object	**No Action (0h)**	**Soft Stop (1h)**	**Soft HiZ (2h)**	**Hard Stop (3h)**	**Hard HiZ (4h)**	**Power OFF (5h)**
2024h_01h	V	V	V	V	V	-
2024h_02h	V	V	V	V	V	-
2024h_03h	-	-	-	V	V	-
2024h_04h	-	-	-	V	V	-
2024h_05h	V	V	V	V	V	-
2024h_06h	V	V	V	V	V	-
2024h_07h	-	-	-	-	V	V
2024h_08h	V	V	V	V	V	-
2024h_09h	-	-	-	-	V	V
2024h_0Ah	V	V	V	V	V	-

Setting parameters

Object	No Action (0h)	Soft Stop (1h)	Soft HiZ (2h)	Hard Stop (3h)	Hard HiZ (4h)	Power OFF (5h)
2024h_0Bh	V	-	-	-	-	-
2024h_0Ch	-	-	-	-	V	V
2024h_0Dh	V	V	V	V	V	-
2024h_0Eh	V	V	V	V	V	-
2024h_0Fh	V	V	V	V	V	-
2024h_10h	V	V	V	V	V	-
2024h_11h	-	-	-	V	V	-
2024h_12h	V	V	V	V	V	-

Tab 4.6 — A summary table of error handling methods.

4.6. Temperature sensor

The controller is equipped with a temperature sensor for the output stage. With it, you can monitor the temperature at a given time, set the temperature threshold for processing the Fault Event Temperature out of limit error. Exceeding the threshold is duplicated in the Statusword by setting the Warning bit. You can also set the sensor update threshold.

Index	2029h
Name	Temperature sensor
Object Type	ARRAY
Number of entries	3

Sub-Index	01h
Description	Temperature sensor value
Data Type	REAL32
Access	RO
PDO Mapping	YES
Units	ºC
Value Range	-
Default Value	-

Setting parameters

Sub-Index	02h
Description	Temperature sensor period
Data Type	UINT8
Access	RW
PDO Mapping	NO
Units	100ms
Value Range	1...255
Default Value	10

Sub-Index	03h
Description	Temperature sensor threshold
Data Type	REAL32
Access	RW
PDO Mapping	NO
Units	ºC
Value Range	-10...+150
Default Value	80

4.7. Output stage parameter

This section is describing the parameters of the output stage such as output current protection thresholds, crushing modes, modes of operation, control parameters gates of the output transistors, etc.

Table4.7 provides description of the objects

Index	**Object**	**Name**	**Type**	**Attr.**
6510h	RECORD	Drive_data		

Tab. 4.7 — Description of the objects

4.7.1. Object 6510h_02h: ADC_OUT

Object ADC_OUT contains the result of an analog-to-digital conversion of the input voltage. The value of this object is used to compensate the loss of the supply voltage. Thus, when the device is connected to the power circuit, it is necessary to set the ADC_OUT object to 16d using the potentiometer [5] (section 1, page 7). The value and the corresponding parameters are indicated in Table 4.8, where V_S stands for supply voltage.

V_S	Value	ADC_OUT[4...0]					compensating coefficient
More, than $V_{S,nom}$ + 50%	> 24	1	1	X	X	X	0,65625
$V_{S,\ nom}$ + 50%	24	1	1	0	0	0	0,65625
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
$V_{S,nom}$	16	1	0	0	0	0	1
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
$V_{S,nom}$ — 50%	8	0	1	0	0	0	1,968875
Less than $V_{S,nom}$ — 50%	< 8	0	0	X	X	X	1,968875

Tab. 4.8 — 5- bit values of the ADC_OUT object.

Sub-Index	02h
Description	ADC_OUT
Data Type	UNSIGNED8
Access	RO
PDO Mapping	YES
Units	-
Value Range	0...32
Default Value	-

4.7.2. Object 6510h_03h: OCD_TH

In the object OCD_TH, the value of the protection output current of the output keys is indicated. The possible values are given in Table 4.9. When writing other values to the object, they will be rounded to the nearest table value.

Object Value	Protection threshold in:	
	current control mode	voltage control mode
1953	1953	1953
2930	2930	2930
3906	3906	3906
4883	4883	4883
5859	5859	5859
6836	6836	6836
7813	7813	
8789	8789	
9766	9766	

Tab. 4.9 — Current protection thresholds.

Setting parameters

Sub-Index	03h
Description	OCD_TH
Data Type	UNSIGNED16
Access	RW
PDO Mapping	NO
Units	mA
Value Range	1953...9766
Default Value	6836

4.7.3. Object 6510h_04h: FS_SPD

This object specifies the speed value in the user units of measurement, at which the transition from the microstep mode to the full-step mode occurs (reffer to the Factor Group paragraph).

Sub-Index	04h
Description	FS_SPD
Data Type	UNSIGNED32
Access	RW
PDO Mapping	NO
Units	speed units
Value Range	-
Default Value	3000

4.7.4. Object 6510h_05h: STEP_MODE

STEP_MODE structure is described in table 4.10.

Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
				CM_VM	STEP_SEL		

Tab. 4.10 — structure STEP_MODE.

An active CM_VM bit transfers the device to the current control mode, inactive - to the voltage control mode.

The STEP_SEL bits are analyzed together and the total sets the microstepping mode.Microstepping coefficients according to CM_VM bits are listed in table 4.11.

STEP_SEL [2...0]			Voltage control mode (CM_VM = 0)	Current control mode (CM_VM = 1)
0	0	0	1/1	1/1
0	0	1	1/2	1/2
0	1	0	1/4	1/4
0	1	1	1/8	1/8
1	0	0	1/16	1/16
1	0	1	1/32	1/16
1	1	0	1/64	1/16
1	1	1	1/128	1/16

Tab. 4.11 — The values of the STEP_MODE bits.

Sub-Index	05h
Description	STEP_MODE
Data Type	UNSIGNED8
Access	RW
PDO Mapping	NO
Units	-
Value Range	-
Default Value	0Ch

4.7.5. Object 6510h_06h: ALARM_EN

The bits of this object generate an internal alarm signal in the event of abnormal situations specified in Table 4.12, followed by processing via object 2024h. Emergency situation related to the critical discharge, causing the motor to stop, ignore the reset (see addition. 4.5).

ALARM_EN bit	Emergency
0 (LSB)	Overcurrent (OCD_TH) to output transistors (keys).
1	Thermal shutdown, exceeding 150ºC.
2	Thermal warning, exceeding 125ºC.
3	UVLO - the gate voltage of the output transistors (keys) drops below the threshold voltage.
4	ADC UVLO - the voltage drop below the set threshold.
5	STALL DETECTION - stop detector.
6	Reserved.
7 (MSB)	Reserved.

Tab. 4.12 — ALARM_EN bit description.

Setting parameters

Sub-Index	06h
Description	ALARM_EN
Data Type	UNSIGNED8
Access	RW
PDO Mapping	NO
Units	-
Value Range	-
Default Value	0Bh

4.7.6. Object 6510h_07h: GATECFG1

Experienced users only! Control parameters of the gate of the output transistors (keys). The structure of the GANECFG1 object is shown in Table 4.13.

Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
					TBOOST		
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IGATE			TCC				

Tab. 4.13 —STEP_MODE structure.

The IGATE parameter specifies the gate current of the output transistors. Table 4.14 lists 7 possible combinations, which range from 4mA to 96mA.

IGATE [2...0]			**Shutter Current, mA**
0	0	0	4
0	0	1	4
0	1	0	8
0	1	1	16
1	0	0	24
1	0	1	32
1	1	0	64
1	1	1	96

Tab. 4.14 — IGATE bit Value.

The TCC parameter specifies the gate charge control time, according to Table 4.15.

TCC [4...0]					Time constant, ns
0	0	0	0	0	125
0	0	0	0	1	250
⋮	⋮	⋮	⋮	⋮	⋮
1	1	1	0	0	3625
1	1	1	0	1	3750
1	1	1	1	0	3750
1	1	1	1	1	3750

Tab. 4.15 — TCC bit Value.

TBOOST parameter determines the length of additional time off, in accordance with Table 4.16.

TBOOST [2...0]			Time constant, ns
0	0	0	0
0	0	1	62,5
0	1	0	125
0	1	1	250
1	0	0	375
1	0	1	500
1	1	0	750
1	1	1	1000

Tab. 4.16 — TBOOST bit Value.

All other bits of this object are ignored. The dependence of the voltage rise rate to the above parameters are given in Table 4.17.

Increase rate of in the output voltage V_S = 48V, V/ µs	I_{GATE}, ns	t_{CC}, ns	t_{DT}, ns	t_{blank}, ns	t_{boost}, ns
980	96	375	125	500	0
790	64	500	125	375	0
520	32	875	125	250	0
400	24	1000	125	250	0
220	16	1600	125	250	0
114	8	3125	125	250	0

Tab. 4.17 — Rate of rise of the output voltage.

Setting parameters

Sub-Index	07h
Description	GATECFG1
Data Type	UNSIGNED16
Access	RW
PDO Mapping	NO
Units	-
Value Range	-
Default Value	00E2h

4.7.7. Object 6510h_08h: GATECFG2

Experienced users only!. Control parameters of the output transistors gate (keys). The structure of the GATECFG2 object is given in Table 4.18.

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TBLANK			TDT				

Tab. 4.18 —STEP_MODE.

The TDT parameter determines the dead time between shutting down the shutter of the opposing key and turning on the shutter of the current key, according to Table 4.19.

TDT [4...0]					Deadtime, ns
0	0	0	0	0	125
0	0	0	0	1	250
. :	. :	. :	. :	. :	. :
1	1	1	1	0	3875
1	1	1	1	1	4000

Tab. 4.19 — TDT bit description.

The TBLANK parameter specifies the time to turn on the current sensor comparators (stop and current detector) after switching, according to Table 4.20.

TBLANK [2...0]			Blanking time, ns
0	0	0	125
0	0	1	250
. :	. :	. :	. :
1	1	0	875
1	1	1	1000

Tab. 4.20 — TBLANK bit description.

Setting parameters

Sub-Index	08h
Description	GATECFG2
Data Type	UNSIGNED8
Access	RW
PDO Mapping	NO
Units	-
Value Range	-
Default Value	20h

4.7.8. Object 6510h_09h: STATUS

STATUS object structure is given in table 4.21.

Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
STALL_A	STALL_B	OCD	TH_STATUS		UVLO_ADC	UVLO	STCK_MOD
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	MOT_STATUS		DIR			BUSY	HiZ

Tab. 4.21 —STEP_MODE.

The installed HiZ flag informs that the output keys of the bridges are in the closed state (the rotor of the motor rotates freely). A reset bit indicates that there is voltage on the motor windings (the motor is in hold mode or rotates).

The reset UVLO flag indicates a voltage drop below the permissible value in the gate supply circuit of the output keys.

The reset flag UVLO_ADC indicates a voltage drop below the set voltage.

The discarded OCD flag informs that the set current of the output stage is exceeded. The reset flags STALL_A and STALL_B inform you that the stop detector in bridge A or bridge B has been triggered, respectively. These flags are informative only when working in voltage control mode.

The group of flags TH_STATUS indicates the temperature state of the output stage and is analyzed together, according to Table 4.22.

TH_STATUS [1...0]		**Status**
0	0	Standard status.
0	1	The temperature has exceeded + 125°C.
1	0	Disconnection of output transistors. The temperature has exceeded + 150°C.
1	1	Turn off the output stage. The temperature has exceeded + 150°C.

Tab. 4.22 — TH_STATUS bit description.

The reset BUSY flag indicates the impossibility of executing the command of the controller.

he installed STCK_MOD flag indicates that the device is operating in driver mode. The DIR flag indicates the direction of rotation of the motor rotor, set for the output stage by the controller, according to table 4.23.

Setting parameters

DIR	The direction given by the master controller
1	Direct
0	Inverse

Tab. 4.23 — DIR Value.

The MOT_STATUS flag group indicates the current rotation parameters, according to the table 4.24.

MOT_STATUS [1...0]		Motor Status
0	0	Stopped
0	1	Rotation with acceleration
1	0	Rotation with braking
1	1	Rotation at a constant speed

Tab. 4.24 — MOT_STATUS Value.

Sub-Index	09h
Description	STATUS
Data Type	UNSIGNED16
Access	RO
PDO Mapping	YES
Units	-
Value Range	-
Default Value	-

4.7.9. Object 6510h_0Ah: VM_KVAL_HOLD

This object is only used in the voltage control mode. The value corresponds to the duty cycle of the PWM modulator in the hold mode, according to Table 4.25.

Sub-Index	0Ah
Description	VM_KVAL_HOLD
Data Type	UNSIGNED8
Access	RW
PDO Mapping	NO
Units	-
Value Range	-
Default Value	05h

4.7.10. Object 6510h_0Bh: VM_KVAL_RUN

This object is only used in the voltage control mode. The value corresponds to the duty cycle of the PWM modulator in the hold mode, according to Table 4.25

Setting parameters

Sub-Index	0Bh
Description	VM_KVAL_RUN
Data Type	UNSIGNED8
Access	RW
PDO Mapping	NO
Units	-
Value Range	-
Default Value	07h

4.7.11. Object 6510h_0Ch: VM_KVAL_ACC

This object is only used in the voltage control mode. The value corresponds to the duty cycle of the PWM modulator in the rotation mode with acceleration, according to Table 4.25.

Sub-Index	0Ch
Description	VM_KVAL_ACC
Data Type	UNSIGNED8
Access	RW
PDO Mapping	NO
Units	-
Value Range	-
Default Value	07h

4.7.12. Object 6510h_0Dh: VM_KVAL_DEC

This object is only used in the voltage control mode. The value corresponds to the duty cycle of the PWM modulator in the rotational mode with braking, according to Table 4.25.

Sub-Index	0Dh
Description	VM_KVAL_ACC
Data Type	UNSIGNED8
Access	RW
PDO Mapping	NO
Units	-
Value Range	-
Default Value	07h

VM_KVAL_X [7...0]								Output voltage
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	$V_S \times (1/256)$
⋰	⋰	⋰	⋰	⋰	⋰	⋰	⋰	⋰
1	1	1	1	1	1	1	0	$V_S \times (254/256)$
1	1	1	1	1	1	1	1	$V_S \times (255/256)$

Tab. 4.25 — Values of bits VM_KVAL_X.

4.7.13. Object 6510h_0Eh: VM_INT_SPEED

The object is involved in voltage control mode only. The value of this object determibes threshold velocity or motor rotor, when BEMF compensation will be applied. It is measured in internal units of velocity measurement (step/tick) and has format fxp 0.18. To convert it to steps/sec use formula:

$$[Step/s] = \frac{\text{VM\_INT\_SPEED} \cdot 2^{-18}}{Tick},$$

whre VM_INT_SPEED – 14-bit object value, and tick is 250 ns.

It is recommended to use a built-in utility «BEMF compensation» of the configuration software «CANopen Builder» to calculate the value of this object.

Sub-Index	0Eh
Description	VM_INT_SPEED
Data Type	UNSIGNED16
Access	RW
PDO Mapping	NO
Units	-
Value Range	-
Default Value	0x0F05h

4.7.14. Object 6510h_0Fh: VM_ST_SLP

Object is only used when working in voltage control mode. The object value is the 8-bit slope coefficient of the BEMF compensation curve, when the rotation speed is below VM_INT_SPEED. The value ranges from 0 to 0.004, in increments of 0.000015 (see Appendix B for more details). To calculate the value of this object, it is recommended to use the configuration program "CANopen Builder", the built-in utility "BEMF compensation".

Setting parameters

Sub-Index	0Fh
Description	VM_ST_SLP
Data Type	UNSIGNED8
Access	RW
PDO Mapping	NO
Units	-
Value Range	-
Default Value	0Fh

4.7.15. Object 6510h_10h: VM_FN_SLP_ACC

This object is only used when working in voltage control mode. The object value is the 8-bit slope coefficient of the BEMF compensation curve, when the rotational speed is higher than VM_INT_SPEED. The value ranges from 0 to 0.004, in increments of 0.000015 (see Appendix B for more details). To calculate the value of this object, it is recommended to use the configuration program "CANopen Builder", the built-in utility "BEMF compensation".

Sub-Index	10h
Description	VM_FN_SLP_ACC
Data Type	UNSIGNED8
Access	RW
PDO Mapping	NO
Units	-
Value Range	-
Default Value	17h

4.7.16. Object 6510h_11h: VM_FN_SLP_DEC

This object is only used when working in voltage control mode. The object value is the 8-bit slope coefficient of the BEMF compensation curve, when the rotational speed is higher than VM_INT_SPEED. The value ranges from 0 to 0.004, in increments of 0.000015 (see Appendix B for more details). To calculate the value of this object, it is recommended to use the configuration program "CANopen Builder", the built-in utility "BEMF compensation".

Sub-Index	11h
Description	VM_FN_SLP_DEC
Data Type	UNSIGNED8
Access	RW
PDO Mapping	NO
Units	-
Value Range	-
Default Value	17h

4.7.17. Object 6510h_12h: VM_K_THERM

This Object is only used when working in voltage control mode. This object was introduced to compensate for the temperature drift of the resistance of the motor windings. 4-bit object values are presented in Table 4.26. Values range from 1 to 1.46875 in increments of 0.03125.

VM_K_THERM [3...0]				**Compensation factor**
0	0	0	0	1
0	0	0	1	1,03125
....				
1	1	1	0	1,4375
1	1	1	1	1,46875

Tab. 4.26 — VM_K_THERM Values.

Sub-Index	12h
Description	VM_K_THERM
Data Type	UNSIGNED8
Access	RW
PDO Mapping	NO
Units	-
Value Range	-
Default Value	0

4.7.18. Object 6510h_13h: VM_STALL_TH

Object is only used when working in voltage control mode. The values of the object have a gradation (~ 977mA), in this connection it is recommended to write down the desired value into the object, and then to consider the actual value.

Sub-Index	13h
Description	VM_STALL_TH
Data Type	UNSIGNED16
Access	RW
PDO Mapping	NO
Units	mA
Value Range	977...31250
Default Value	16602

4.7.19. Object 6510h_14h: VM_CONFIG

Object is only used when working in voltage control mode. The structure of the VM_CONFIG object is shown in Table 4.27.

Setting parameters

Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
F_PWM_INT			F_PWM_DEC			VCCVAL	UVLOVAL
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		EN_VSCOMP					

Tab. 4.27 —VM_CONFIG Structure.

A set bit EN_VSCOMP activates compensation of voltage from that calculated, the discarded - disconnects.

VCCVAL bit value selects the final-stage control gate voltage of the output transistors according to Table 4.28.

VCCVAL	**V_{CC}, V**
0	7,5
1	15

Tab. 4.28 – VCCVAL Value.

The UVLOVAL bit specifies the protection thresholds according to Table 4.29.

UVLOVAL	**V_{ccthOn}, V**	**V_{ccthOn}, V**	**$DV_{BOOTThOn}$, V**	**$DV_{BOOTThOff}$, V**
0	6,9	6,3	6	5,5
1	10,4	10	9,2	8,8

Tab. 4.29 – UVLOVAL Values.

Parameters F_PWM_INT and F_PWM_DEC set the frequency of the output PWM signal of the controller. The frequencies and parameters are presented in Table 4.30.

	F_PWM_DEC							
F_PWM_INT	000	001	010	011	100	101	110	111
000	19,5	23,4	27,3	31,3	39,1	46,9	54,7	62,5
001	9,8	11,7	13,7	15,6	19,5	23,4	27,3	31,3
010	6,5	7,8	9,1	10,4	13,0	15,6	18,2	20,8
011	4,9	5,9	6,8	7,8	9,8	11,7	13,7	15,6
100	3,9	4,7	5,5	6,3	7,8	9,4	10,9	12,5
101	3,3	3,9	4,6	5,2	6,5	7,8	9,1	10,4
110	2,8	3,3	3,9	4,5	5,6	6,7	7,8	8,9

Tab. 4.30 — Frequency of output PWM signal, kHz.

Sub-Index	14h
Description	VM_CONFIG
Data Type	UNSIGNED16
Access	RW
PDO Mapping	NO
Units	-
Value Range	-
Default Value	5B00h

4.7.20. Object 6510h_15h: CM_TVAL_HOLD

This object is only used in the current control mode. The value of the object corresponds to the current in milliamperes flowing through the winding of the motor in the hold mode and has a gradation of values (~ 217 mA), therefore it is recommended to write down the desired value to the object, and then to consider the actual value.

Sub-Index	15h
Description	CM_TVAL_HOLD
Data Type	UNSIGNED16
Access	RW
PDO Mapping	NO
Units	mA
Value Range	1083...5200
Default Value	1083

4.7.21. Object 6510h_16h: CM_TVAL_RUN

This object is only used in the current control mode. The value of the object corresponds to the current in milliamperes flowing through the winding of the motor in the rotor rotation mode at a constant speed and has a value gradation (~ 217 mA), therefore it is recommended to write down the desired value to the object and then read the actual value.

Sub-Index	16h
Description	CM_TVAL_RUN
Data Type	UNSIGNED16
Access	RW
PDO Mapping	NO
Units	mA
Value Range	1083...5200
Default Value	1083

4.7.22. Object 6510h_17h: CM_TVAL_ACC

This object is only used in the current control mode. The value of the object corresponds to the current in milliamperes flowing through the winding of the motor in the rotor rotation mode with acceleration and has a gradation of values (~ 217 mA), therefore it is recommended to write down the desired value to the object, and then to consider the actual value.

Sub-Index	17h
Description	CM_TVAL_ACC
Data Type	UNSIGNED16
Access	RW
PDO Mapping	NO
Units	mA
Value Range	1083...6067
Default Value	1083

4.7.23. 4.7.23 Object 6510h_18h: CM_TVAL_DEC

This object is only used in the current control mode. The value of the object corresponds to the current in milliamperes flowing through the motor winding in the rotor rotation mode with braking and has a value gradation (~ 217 mA), therefore it is recommended to write down the desired value to the object and then to count the actual value.

Sub-Index	18h
Description	CM_TVAL_DEC
Data Type	UNSIGNED16
Access	RW
PDO Mapping	NO
Units	mA
Value Range	1083...6067
Default Value	1083

4.7.24. Object 6510h_19h: CM_T_FAST

This object is only used in the current control mode. The structure of the object is shown in Table 4.31. and consists of two vaues: TOFF_FAST and FAST_STEP, available values are shown in table 4.32. Please, see appendix B for detailed information.

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TOFF_FAST				FAST_STEP			

Tab. 4.31 —CM_T_FAST structure.

TOFF_FAST [3...0] FAST_STEP [3...0]				**Fast decay time, µs**
0	0	0	0	2
0	0	0	1	4
· :	· :	· :	· :	· :
1	1	1	0	30
1	1	1	1	32

Tab. 4.32 — TOFF_FAST and FAST_STEP Values.

Setting parameters

Sub-Index	19h
Description	CM_T_FAST
Data Type	UNSIGNED8
Access	RW
PDO Mapping	NO
Units	-
Value Range	-
Default Value	53h

4.7.25. Object 6510h_1Ah: CM_TON_MIN

This object is only used in the current control mode. The value of this object corresponds to the minimum value of the key activation time. The possible 7-bit values of the object are given in Table 4.33.

Sub-Index	1Ah
Description	CM_TON_MIN
Data Type	UNSIGNED8
Access	RW
PDO Mapping	NO
Units	-
Value Range	-
Default Value	05h

4.7.26. Object 6510h_1Bh: CM_TOFF_MIN

This object is only used in the current control mode. The value of this object corresponds to the minimum value of the key's turn-off time. The possible 7-bit values of the object are given in Table 4.33.

Sub-Index	1Bh
Description	CM_TOFF_MIN
Data Type	UNSIGNED8
Access	RW
PDO Mapping	NO
Units	-
Value Range	-
Default Value	29h

Setting parameters

CM_TON_MIN [6...0] CM_TOFF_MIN [6...0]							**Time, µs**
0	0	0	0	0	0	0	0,5
0	0	0	0	0	0	1	1
. :	. :	. :	. :	. :	. :	. :	. :
1	1	1	1	1	1	0	63,5
1	1	1	1	1	1	1	64

Tab 4.33 —TON_MIN and TOFF_MIN Values.

4.7.27. Object 6510_1C:CM_CONFIG

Object is only used when operating in the current control mode. The structure of the CM_CONFIG object is shown in Table 4.34.

Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
PRED_EN	TSW					VCCVAL	UVLOVAL
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Tab. 4.34 —CM_CONFIG Structure.

The VCCVAL bit selects the Gate Control voltage of the output stage transistors according to Table 4.35.

VCCVAL	**V_{CC}, V**
0	7,5
1	15

Tab. 4.35 – VCCVAL value.

UVLOVAL value sets the protection thresholds according to Table 4.36.

UVLOVAL	**V_{ccthOn}, V**	**V_{ccthOn}, V**	**$DV_{BOOTThOn}$, V**	**$DV_{BOOTThOff}$, V**
0	6,9	6,3	6	5,5
1	10,4	10	9,2	8,8

Tab. 4.36 –UVLOVAL value.

An activated PRED_EN bit includes the predictive method of current control, reseted - turns off. See Appendix B for more details.

The TSW bit group specifies the period of operation of the current control algorithm. The value is selected according to table 4.37.

TSW [4...0]					**Switching period**
0	0	0	0	0	4 µs (250kHz)
0	0	0	0	1	4 µs (250 kHz)
0	0	0	1	0	8 µs (125 kHz)
. :	. :	. :	. :	. :	. :
1	1	1	1	1	124 µs (8 kHz)

Tab. 4.37 – TSW Values.

Setting parameters

Sub-Index	1Ch
Description	CM_CONFIG
Data Type	UNSIGNED16
Access	RW
PDO Mapping	NO
Units	-
Value Range	-
Default Value	3700h

Warning!
It is not recommended to change the parameters set by default or calculated with the configuration software of the motor controller, without any urgent reasons, such as for example the electromagnetic compatibility of the installation,
This warning applies to objects:
6510h_07h GATECFG1;
6510h_08h GATECFG2;
It is strongly recommended that you review Appendix B before making any changes to objects:**6510h_0Ah … 6510h_11h, 6510h_14h … 6510h_1Ch.**

4.8. Description of device objects

This section will observe objects from Table 4.38.

Index	**Object**	**Name**	**Type**	**Attr.**
605Ah	VAR	Quick_stop_option_code	INTEGER16	RW
6063h	VAR	Position_actual_value	INTEGER32	RO
6064h	VAR	Position_actual_value	INTEGER32	RO
6069h	VAR	Velocity_sensor_actual_value	INTEGER32	RO
60FDh	VAR	Digital_inputs	UNSIGNED32	RO
60FEh	ARRAY	Digital_outputs		

Tab. 4.38 — Object description.

4.8.1. Object 605Ah: Quick_stop_option_code

Using the Quick_stop_option_code parameter, you specify how the motor controller functions during Quick Stop. Object displays the realizable principle of the motor controller, Table 4.39.

Value	**Description**
0	Disconnection of the motor windings (HiZ state).

Tab. 4.39 — braking principles.

For this motor controller it is invariable.

Setting parameters

Index	605Ah
Name	Quick_stop_option_code
Object Type	VAR
Data Type	INTEGER16
Access	RW
PDO Mapping	YES
Units	-
Value Range	-
Default Value	0

4.8.2. Object 6063h: Position_actual_value_s

With this object, you can read the actual position. This object is specified in increments: uStep or Step, depending on the settings for the fragmentation of the full step.

Index	6063h
Name	Position_actual_value_s
Object Type	VAR
Data Type	INTEGER32
Access	RO
PDO Mapping	YES
Units	uStep (Step)
Value Range	-
Default Value	-

4.8.3. Object 6064h: Position_actual_value

With this object, you can read the actual position. This object is specified in user-defined units of measure.

Index	6064h
Name	Position_actual_value
Object Type	VAR
Data Type	INTEGER32
Access	RO
PDO Mapping	YES
Units	position units
Value Range	-
Default Value	-

4.8.4. Object 6069h: Velocity_sensor_actual_value

With Velocity_sensor_actual_value, the speed value can be read in internal units.

Setting parameters

Index	6069h
Name	Velocity_sensor_actual_value
Object Type	VAR
Data Type	INTEGER32
Access	RO
PDO Mapping	YES
Units	Step/Tick (0.28)
Value Range	-
Default Value	-

4.8.5. Object 60FDh: Digital_inputs

With the object 60FDh, it is possible to get response from the discrete inputs.

Index	60FDh
Name	Digital_inputs
Object Type	VAR
Data Type	UNSIGNED32
Access	RO
PDO Mapping	YES
Units	-
Value Range	According to table 4.40
Default Value	-

Bit	**Value**	**Description**
0	00000001h	Negative limit switch
1	00000002h	Negative reference sensor
2	00000004h	Positive Reference Sensor
3	00000008h	Positive limit switch

Tab. 4.40 — Digital_inputs.

If necessary, the inputs of the home sensors can be connected with each other

4.8.6. Object 60FEh: Digital_outputs

With the 60FEh object, you can control the digital outputs. To do this, in the Digital_outputs_Bitmask object, you must specify which discrete outputs to control. Using the Digital_outputs_Physical_outputs object, you can optionally set the selected outputs.

Index	60FEh
Name	Digital_outputs
Object Type	ARRAY
Number of entries	2

Sub-Index	01h
Description	Digital_outputs_Physical_outputs
Data Type	UNSIGNED32
Access	RW
PDO Mapping	YES
Units	-
Value Range	According to table 4.41
Default Value	0

Sub-Index	02h
Description	Digital_outputs_Bitmask
Data Type	UNSIGNED32
Access	RW
PDO Mapping	YES
Units	-
Value Range	-
Default Value	0

Bit	**Value**	**Description**
16	00010000h	OUT_1
17	00020000h	OUT_2

Tab. 4.40 — Digital outputs.

5. Device Control

5.1. State Machine

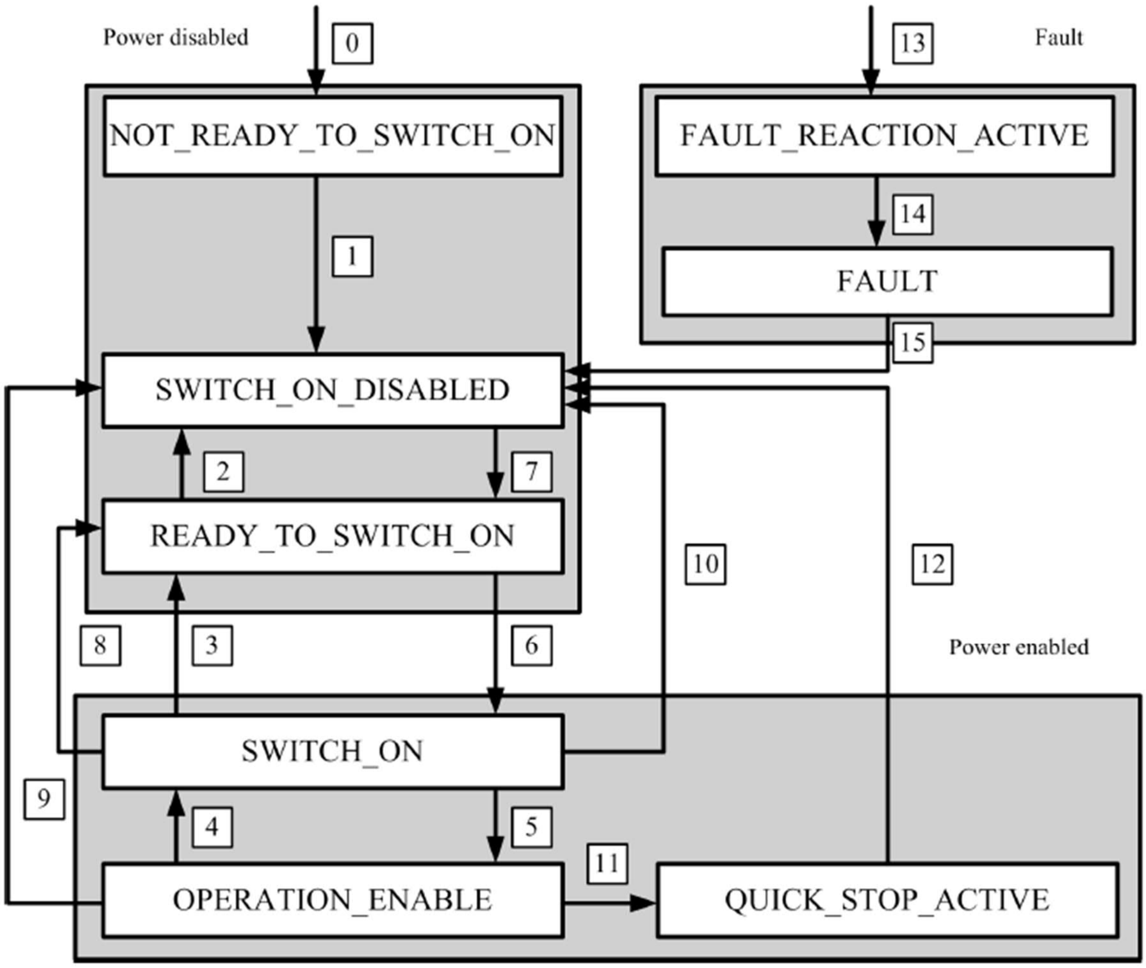
5.1.1. Overview

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)	he 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. 5.1 - Controller regulation terms.

5.1.2. State Machine



Pic. 5.1 — Status diagram of the motor controller (state machine)

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.

After it is switched on, the motor controller initialises itself and then reaches the status READY_TO_SWITCH_ON, when power is applied to the output keys, but there is no control voltage on the motor connection terminals.

Status diagram: statuses

The following table lists all statuses and their meanings:

Name	**Description**
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 output stage is energized and initialized.
SWITCHED_ON	The output stage is switched on.
OPERATION_ENABLE	Voltage to the motor is on, and the motor is

Name	Description
	regulated corresponding to the operating mode.
QUICKSTOP_ACTIVE	The Quick Stop Function is being executed
FAULT_REACTION_ACTIVE	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..

Tab. 5.2 — Statuses.

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 enable + 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	Deceleration starts according

							Quick_stop_option_code.
12	Braking ended without command Disable Voltage	Disable Voltage	x	x	0	x	Braking is initiated in accordance with quick_stop_option_code.
13	Error occurred	Internal transition					For uncritical errors, reaction in accordance with fault_reaction_option_code. With critical errors, transition 14 follows.
14	Error handling has 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)

Tab. 5.3 — Status diagram.

Caution
Output stage— is a set of output transistors (keys) with control elements.
Output transistors— are power semiconductors that work in a key mode, which form the engine control signal.
The closed state of the output transistors - indicates the absence of a control voltage at the motor connection terminals (the motor rotates freely).
The presence of power of the output transistors is not a sufficient condition for the appearance of voltage on the motor terminals (the motor can freely rotate).

5.1.3. Control word (controlword)

Object 6040h: 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, 6 and 8 depends on the current operating mode (modes_of_operation) of the motor controller, which is explained after this chapter.

Index	6040h
Name	Controlword
Object Type	VAR
Data Type	UNSIGNED16
Access	RW
PDO Mapping	YES
Units	-
Value Range	-
Default Value	06h

Bit	Value	Function
0	0001h	Control of the status transitions. (These bits are evaluated together)
1	0002h	
2	0004h	
3	0008h	
4	0010h	new_set_point/start_homing_operation
5	0020h	change_set_immediatly
6	0040h	absolute/relative
7	0080h	reset_fault
8	0100h	halt
9	0200h	reserved – set to 0
10	0400h	reserved – set to 0
11	0800h	reserved – set to 0
12	1000h	reserved – set to 0
13	2000h	reserved – set to 0
14	4000h	reserved – set to 0
15	8000h	reserved – set to 0

Tab. 5.4 — Bit assignment of the controlword

As already comprehensively described, status transitions can be carried out with the bits 0 … 3. Commands required for this are presented in Table 5.5. Command Fault Reset through the positive edge change (from 0 to 1) is generated by bit 7.

Command	**Bit 7**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
	0080h	**0008h**	**0004h**	**0002h**	**0001h**
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. 5.5 — Overview of all commands.

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 modes_of_operation	
	new_set_point	In Profile Position Mode: A rising edge signals to the motor controller that a new positioning task should be undertaken.
	start_homing_operation	In the Homing Mode: A rising edge causes the parametrised reference travel to start. A falling edge interrupts a running reference travel prematurely.
5	change_set_immediatly	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.
6	relative	Only in the Profile Position Mode: If the bit is set, the motor controller obtains the target position (target_position) of the current positioning task relative to the setpoint position (position_demand_value) of the position controller.
7	reset_fault	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.
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 Homing Mode: If the bit is set, the ongoing reference travel is interrupted. Deletion of the bit has no effect.

Tab. 5.6 — controlword bit 4 … 8

5.1.4. 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
	0040h	0020h	0008h	0004h	0002h	0001h		
Not_Ready_To_Switch_On	0	x	0	0	0	0	004Fh	0000h
Switch_On_Disabled	1	x	0	0	0	0	004Fh	0040h
Ready_to_Switch_On	0	1	0	0	0	1	006Fh	0021h
Switched_On	0	1	0	0	1	1	006Fh	0023h
OPERATION_ENABLE	0	1	0	1	1	1	006Fh	0027h
QUICK_STOP_ACTIVE	0	0	0	1	1	1	006Fh	0007h
Fault_Reaction_Active	0	x	1	1	1	1	004Fh	000Fh
Fault	0	x	1	1	1	1	004Fh	0008h
FAULT (in accordance with CiA402)	0	x	1	0	0	0	004Fh	0008h

Tab. 5.7 — Device status.

5.1.5. Status words (statuswords)

Object 6041h: Statusword

Index	6041h
Name	Statusword
Object Type	VAR
Data Type	UNSIGNED16
Access	RO
PDO Mapping	YES
Units	-
Value Range	-
Default Value	-

Bit	Value	Function
0	0001h	Status of the motor controller (Tab. 5.7). (These bits must be evaluated together.)
1	0002h	
2	0004h	
3	0008h	
4	0010h	voltage_enabled
5	0020h	Status of the motor controller (Tab 5.7).
6	0040h	
7	0080h	warning
8	0100h	drive_is_moving
9	0200h	remote

10	0400h	target_reached
11	0800h	internal_limit_active
12	1000h	set_point_acknowledge / speed_0 / homing_attained /
13	2000h	following_error / homing_error
14	4000h	manufacturer_statusbit
15	8000h	manufacturer_statusbit

Tab. 5.8 — Bit allocation in the status word.

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.

Tab. 5.9 — 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
7	warning	This bit shows that a warning is active.
8	drive_is_moving	The drive is moving.
9	remote	This bit shows that the output stage of the motor controller can be enabled via the CAN network.
10	Dependent on modes_of_operation.	
	target_reached	Profile Position Mode: The bit is set when the current target position is reached. In addition, it is set in the event that the drive enters a quiescent state with the given bit Halt. It is deleted as soon as a new target is specified.
		In the Profile Velocity Mode: The bit is set when the velocity speed is reached.
11	internal_limit_active	This bit shows that the limitations are 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.
	speed_0	In the Profile Velocity Mode This bit is set when the drive has stopped.
	homing_attained	In the Homing Mode: This bit is set when homing has been ended without error.
13	Dependent on modes_of_operation.	
	following_error	In the Profile Position Mode: his bit is set when an error occurred during positioning.
	homing_error	In the Homing Mode: This bit is set when the reference travel has been interrupted (Halt bit), both limit switches respond simultaneously.
14	manufacturer_statusbit	Not used
15	manufacturer_statusbit	Not used

Tab. 5.10 — statusword bit 5...15.

6. Operating modes

6.1. Setting the operating mode

6.1.1. Overview

The motor controller can be placed into a number of operating modes. Only some are specified in detail under CANopen:

- profile velocity mode
- homing mode
- profile position mode

6.1.2. Description of the Objects

Objects treated in this chapter

Index	Object	Name	Type	Attr.
6060h	VAR	Modes_of_operation	INTEGER8	RW
6061h	VAR	Modes_of_operation_display	INTEGER8	RO

Tab. 6.1 — Object Description.

Object 6060h: Modes_of_operation

The object modes_of_operation sets the operating mode of the motor controller.

Index	6060h
Name	Modes_of_operation
Object Type	VAR
Data Type	INTEGER8
Access	RW
PDO Mapping	YES
Units	-
Value Range	1, 3, 6
Default Value	1

Value	Description
1	Profile Position Mode
3	Profile Velocity Mode
6	Homing Mode

Tab. 6.2 — Significance methods.

Operating modes

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.

Objectt 6061h: Modes_of_operation_display

In the object modes_of_operation_display, the current operating mode of the motor controller can be read.

Index	6061h
Name	Modes_of_operation_display
Object Type	VAR
Data Type	INTEGER8
Access	RO
PDO Mapping	YES
Units	-
Value Range	1, 3, 6
Default Value	-

Value	**Description**
1	Profile Position Mode
3	Profile Velocity Mode
6	Homing Mode

Tab. 6.3 — control methods.

6.2. Operating mode reference travel (homing mode)

6.2.1. 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. 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).

6.2.2. Description of the Objects

Objects treated in this chapter.

Index	Object	Name	Type	Attr.
607Ch	VAR	Home_offset	INTEGER32	RW
6098h	VAR	Homing_method	INTEGER8	RW
6099h	ARRAY	Homing_speeds	UNSIGNED32	RW

Tab. 6.4 — Description of the Objects.

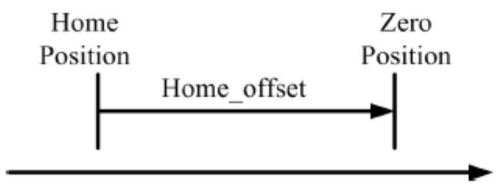
Affected objects from other chapters

Index	Object	Name	Type	Chapter
6040h	VAR	Controlword	UNSIGNED16	5.1.3
6041h	VAR	Statusword	UNSIGNED16	5.1.5
6083h	VAR	Profile_acceleration	UNSIGNED32	6.3.2
6084h	VAR	Profile_deceleration	UNSIGNED32	6.3.2

Tab. 6.5 — Affected objects from other chapters.

Object 607Ch: Home_offset

The object home_offset establishes the shift of the zero position compared to the determined reference Position



Pic. 6.1 — Home Offset

Index	607Ch
Name	Home_offset
Object Type	VAR
Data Type	INTEGER32
Access	RW
PDO Mapping	YES
Units	position unit
Value Range	-
Default Value	0

Object 6098h: 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 three possible homing signals: the negative and positive limit switch and the reference switch (positive and negative sensors can be combined).

Index	6098h
Name	Homing_method
Object Type	VAR
Data Type	INTEGER8
Access	RW
PDO Mapping	YES
Units	-
Value Range	17, 18, 23, 27, 35
Default Value	35

Value	**Motion**	**Objective**	**Point of reference for zero**
17	Negative	Limit switch	Limit switch
18	Positive	Limit switch	Limit switch
23	Positive	Reference switch	Reference switch
27	Negative	Reference switch	Reference switch
35		No travel	Current actual position

Tab. 6.6 — Homing methods.

The homing_method can only be set when homing is not active.

The process of the individual methods is described in detail in chapter 6.2.3.

Object 6099h: Homing_speeds

This object determines the speeds used during the reference travel.

Index	6099h
Name	Homing_speeds
Object Type	ARRAY
Number of entries	2

Sub-Index	01h
Description	Homing_speeds_Speed_during_search_for_switch
Data Type	UNSIGNED32
Access	RW
PDO Mapping	YES
Units	speed units
Value Range	-
Default Value	1000

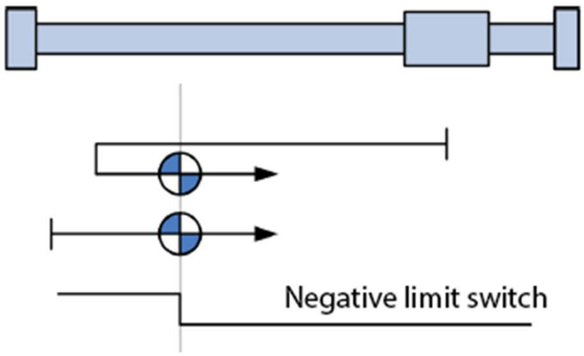
Sub-Index	02h
Description	Homing_speeds_Speed_during_search_for_zero
Data Type	UNSIGNED32
Access	RW
PDO Mapping	YES
Units	speed units
Value Range	-
Default Value	50

6.2.3. Reference Travel Processes

The various reference travel methods are depicted in the following illustrations.

Method 17: Negative limit switch

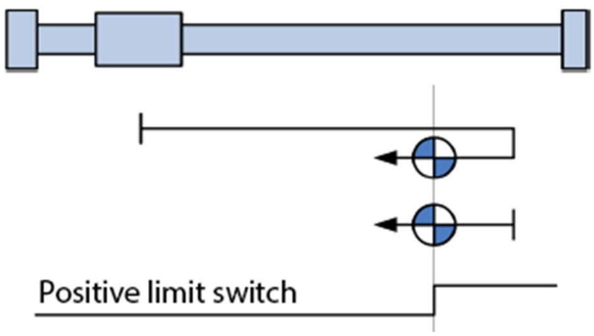
When using this method, the drive first moves relatively quickly in the negative direction until it reaches the negative limit switch. This is shown in the diagram (Picture 6.2) with a rising front. Then, the drive slowly returns and looks for the exact position of the limit switch. The zero position is determined based on the falling front of the negative limit switch signal.



Pic. 6.2. — Travel to the reference point via Negative limit switch

Method 18 Positive limit switch

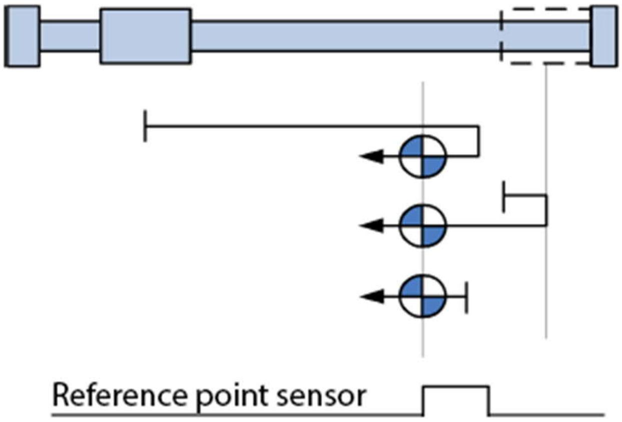
When using this method, the drive first moves relatively quickly in the positive direction until it reaches a positive limit switch. This is shown in the diagram (Picture 6.3) with a rising front. Then, the drive slowly returns and looks for the exact position of the limit switch. The zero position is determined based on the falling front of the positive limit switch signal.



Pic. 6.3. — Travel to the reference point via Positive limit switch

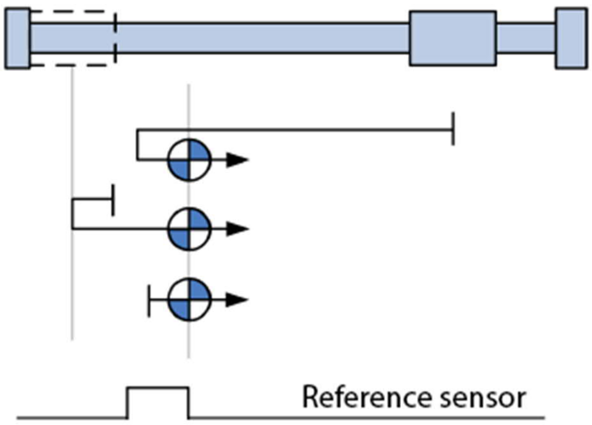
Methods 23 and 27: Moving to the origin of the reference point sensor

In both methods a reference point sensor is used, which is active only on a part of the path segment. This method of determining the origin is particularly suitable for working with circular axes, where the origin sensor is activated once per revolution. With method 23, the drive first moves in the positive direction, and at method 27 - in the negative direction. The zero position is determined from the front of the limit switch signal. This is clearly shown in Pictures 6.4 and 6.5.



Pic. 6.4. — Travel to the reference point via Positive limit switch

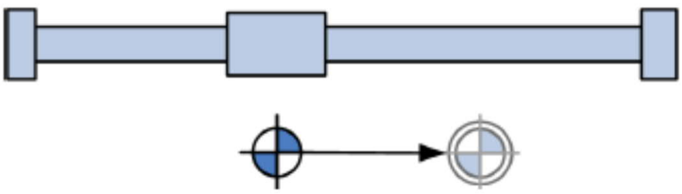
When moving to the origin of the reference point sensor, the limit switches serve, first of all, for reversing the search direction. If an opposing limit switch is reached in connection with this, an error is generated.



Pic. 6.5. — Moves to the origin of the reference sensor with negative initial movement

Method 35: Moving to the origin from the current position

With method 35, the zero position is determined by the current position (Picture 6.6).



Pic.6.6. — Moves to the origin from the current position.

6.2.4. 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.

6.3. Positioning Operating Mode (Profile Position Mode)

6.3.1. Overview

The target position (target_position) is passed on to the curve generator. All input variables of the curve generator are converted with the variables of the factor group (chapter 4.3) into the internal units of the controller.

6.3.2. Description of the objects

Objects treated in this chapter

Index	**Object**	**Name**	**Type**	**Attr.**
607Ah	VAR	Target_position	INTEGER32	RW
6081h	VAR	Profile_velocity	UNSIGNED32	RW
6083h	VAR	Profile_acceleration	UNSIGNED32	RW
6084h	VAR	Profile_deceleration	UNSIGNED32	RW
6086h	VAR	Motion_profile_type	INTEGER16	RW

Tab. 6.7 — Description of the objects.

Affected objects from other chapters

Index	**Object**	**Name**	**Type**	**Chapter**
6040h	VAR	Controlword	UNSIGNED16	5.1.3
6041h	VAR	Statusword	UNSIGNED16	5.1.5
605Ah	VAR	Quick_stop_option_code	INTEGER16	4.8.1
607Eh	VAR	Polarity	UNSIGNED8	4.3.4
6093h	ARRAY	Position_factor	UNSIGNED32	4.3.1
6095h	ARRAY	Velocity_factor	UNSIGNED32	4.3.2
6097h	ARRAY	Acceleration_factor	UNSIGNED32	4.3.3

Tab. 6.8 — Affected objects from other chapters.

Object 607Ah: 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).

Operating modes

Index	607Ah
Name	Target_position
Object Type	VAR
Data Type	INTEGER32
Access	RW
PDO Mapping	YES
Units	position unit
Value Range	-
Default Value	0

Object 6081h: Profile_velocity

Object Profile_velocity indicates the speed, which is usually achieved during positioning at the end of the acceleration. Object Profile_velocity is expressed in units of speed (speed units).

Index	6081h
Name	Profile_velocity
Object Type	VAR
Data Type	UNSIGNED32
Access	RW
PDO Mapping	YES
Units	speed units
Value Range	-
Default Value	1000

Object 6083h: 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 (acceleration units).

Index	6083h
Name	Profile_acceleration
Object Type	VAR
Data Type	UNSIGNED32
Access	RW
PDO Mapping	YES
Units	acceleration unit
Value Range	-
Default Value	1000

Object 6084h: Profile_deceleration

The object profile_deceleration specifies the deceleration (negative) with which the motor is braked. It isspecified in user-defined acceleration units (acceleration units).

Operating modes

Index	6084h
Name	Profile_deceleration
Object Type	VAR
Data Type	UNSIGNED32
Access	RW
PDO Mapping	YES
Units	acceleration unit
Value Range	-
Default Value	1000

Object 6086h: Motion_profile_type

The object motion_profile_type is used to select the type of positioning profile.

Index	6086h
Name	Motion_profile_type
Object Type	VAR
Data Type	INTEGER16
Access	RW
PDO Mapping	YES
Units	-
Value Range	0
Default Value	0 (Linear ramp)

6.3.3. Description of function

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 , the host can start positioning by setting the bit new_set_point in the controlword to “1” . 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. Only when the motor controller can accept a new positioning job does it signal this through a “0” in the set_point_acknowledge bit. Before this, no new positioning may be started by the host.

6.4. Speed Adjustment Operating Mode (Profile Velocity Mode)

6.4.1. Description of the Objects

Objects treated in this chapter.

Index	**Object**	**Name**	**Type**	**Attr.**
6069h	VAR	Velocity_sensor_actual_value	INTEGER32	RO
606Ah	VAR	Sensor_selection_code	INTEGER16	RW
606Bh	VAR	Velocity_demand_value	INTEGER32	RO
606Ch	VAR	Velocity_actual_value	INTEGER32	RO
60FFh	VAR	Target_velocity	INTEGER32	RW

Tab. 6.9 — Description of the Objects

Affected objects from other chapters

Index	**Object**	**Name**	**Type**	**Chapter**
6040h	VAR	Controlword	UNSIGNED16	5.1.3
6041h	VAR	Statusword	UNSIGNED16	5.1.5

Index	Object	Name	Type	Chapter
607Eh	VAR	Polarity	UNSIGNED8	4.3.4
6083h	VAR	Profile_acceleration	UNSIGNED32	6.3.2
6084h	VAR	Profile_deceleration	UNSIGNED32	6.3.2
6086h	VAR	Motion_profile_type	INTEGER16	6.3.2

Tab. 6.10 — Connected Objects from other chapters.

Object 6069h: 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 R272-42 family. Therefore, to determine the actual speed value, the object 606Ch should be used.

Index	6069h
Name	Velocity_sensor_actual_value
Object Type	VAR
Data Type	INTEGER32
Access	RO
PDO Mapping	YES
Units	Step/Tick (0.28)
Value Range	-
Default Value	-

Object 606Ah: Sensor_selection_code

The speed sensor can be selected with this object. Currently, no separate speed sensor is planned. The value is fixed to -1.

Index	606Ah
Name	Sensor_selection_code
Object Type	VAR
Data Type	INTEGER16
Access	RW
PDO Mapping	YES
Units	-
Value Range	-1
Default Value	-1

Object 606Bh: 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.

Index	606Bh
Name	Velocity_demand_value
Object Type	VAR
Data Type	INTEGER32
Access	RO
PDO Mapping	YES
Units	speed units
Value Range	-
Default Value	-

Object 60FFh: Target_velocity

The object target_velocity is the setpoint specification for the ramp generator.

Index	60FFh
Name	Target_velocity
Object Type	VAR
Data Type	INTEGER32
Access	RW
PDO Mapping	YES
Units	speed units
Value Range	-
Default Value	-

7. Driver Mode

7.1. Overview

Configuration and setting up in driver mode is performed via special software (with driver mode active) with subsequent saving of settings in the ROM of the motor controller, the regulation of which is not possible through external controls.

In the driver mode is controlled using the input output port (I / O port), contacts STEP, DIR, ENABLE, RESET, FAULT. In this case, the purpose of some of them can be redefined.

7.2. Description of the Objects

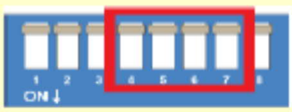
Objects treated in this chapter

Index	**Object**	**Name**	**Type**	**Attr.**
2032h	VAR	Pulse Mode Current	UNSIGNED8	RO
2033h	VAR	Pulse Mode Microstep	UNSIGNED8	RO
2034h	VAR	Pulse Mode Config	UNSIGNED8	RW
2035h	VAR	Pulse Mode Hold Timeout	UNSIGNED16	RW
2036h	VAR	Pulse Mode Hold Current Factor	REAL32	RW
2038h	VAR	Pulse Mode Overcurrent detection threshold	UNSIGNED16	RW
2039h	ARRAY	Pulse Mode Advanced Config		
2040h	VAR	Pulse Mode Position	INTEGER32	RW

Tab. 7.1 — Description of the Objects.

Object 2032h: Pulse Mode Current

The value of this object is the number of the table current value in the motor winding (Table 7.2).

Object Value	Sw. 4	Sw. 5	Sw. 6	Sw. 7	Current value, A	Object Value	Sw. 4	Sw. 5	Sw. 6	Sw. 7	Current value, A
0	OFF	OFF	OFF	OFF	1,2	9	ON	OFF	OFF	ON	3,0
1	ON	OFF	OFF	OFF	1,4	10	OFF	ON	OFF	ON	3,2
2	OFF	ON	OFF	OFF	1,6	11	ON	ON	OFF	ON	3,4
3	ON	ON	OFF	OFF	1,8	12	OFF	OFF	ON	ON	3,6
4	OFF	OFF	ON	OFF	2,0	13	ON	OFF	ON	ON	3,8
5	ON	OFF	ON	OFF	2,2	14	OFF	ON	ON	ON	4,0
6	OFF	ON	ON	OFF	2,4	15	ON	ON	ON	ON	4,2
7	ON	ON	ON	OFF	2,6						
8	OFF	OFF	OFF	ON	2,8						

Tab. 7.2 — Correspondence of the positions of the switches to the currents in the motor windings.

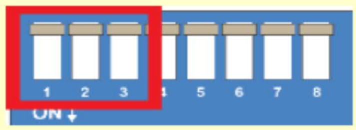
Driver Mode

Index	2032h
Name	Pulse Mode Current
Object Type	VAR
Data Type	UNSIGNED8
Access	RO
PDO Mapping	NO
Units	-
Value Range	0...15
Default Value	-

Object 2033h: Pulse Mode Microstep

The value of this object is the number of the table value of the motor microstepping mode (table 7.3).

Object Value	Sw. 1	Sw. 2	Sw. 3	Microstepping
0	OFF	OFF	OFF	1/1
1	ON	OFF	OFF	1/2
2	OFF	ON	OFF	1/4
3	ON	ON	OFF	1/8
4	OFF	OFF	ON	1/16
⋰	⋰	⋰	⋰	⋰
7	ON	ON	ON	1/16



Tab. 7.3 — Microstepping options according to switch positions.

Index	2033h
Name	Pulse Mode Microstep
Object Type	VAR
Data Type	UNSIGNED8
Access	RO
PDO Mapping	NO
Units	-
Value Range	0...7
Default Value	-

Object 2034h: Pulse Mode Config

The structure of this object is shown in Table 7.4.

Driver Mode

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
				FC	QE	DP	EN

Tab. 7.4 — Structure of Pulse Mode Config.

The set EN bit inverts the ENABLE signal, which allows the control voltage to be applied to the motor windings. Thus, for a given EN bit on the ENABLE logical input, the enable signal is a logical zero.

The DP bit only affects the sign of the displayed value of the current position in the driver mode (object 2040h Pulse Mode Position).

The specified QE bit switches the driver to quadrature mode (gray code). The signals A and B are connected to the STEP and DIR inputs. The reset bit switches the driver to the "step / direction" mode.

The specified FC bit activates the current-reduction function in the holding mode. The transition to hold mode is considered to be carried out after the Pulse Mode Hold Timeout from the last pulse at the STEP input. The current in hold mode is calculated as the product of the setpoint and Pulse Mode Hold Timeout. reduction does not occur current during a reset bit.

Index	2034h
Name	Pulse Mode Config
Object Type	VAR
Data Type	UNSIGNED8
Access	RW
PDO Mapping	NO
Units	-
Value Range	-
Default Value	11h

Object 2035h: Pulse Mode Hold Timeout

The value of the object specifies the time of the transition to the hold mode. The time count starts from the moment the step is taken and is reset when the next step is taken, if the step does not occur during the time Pulse Mode Hold Timeout, the driver goes into the hold mode with a decrease in current according to Pulse Mode Hold Current Factor.

Index	2035h
Name	Pulse Mode Hold Timeout
Object Type	VAR
Data Type	UNSIGNED16
Access	RW
PDO Mapping	NO
Units	ms
Value Range	-
Default Value	1000

Driver Mode

Object 2036h: Pulse Mode Hold Current Factor

The value of the object is the coefficient that determines the proportion of the set current for the hold mode.

Index	2036h
Name	Pulse Mode Hold Current Factor
Object Type	VAR
Data Type	REAL32
Access	RW
PDO Mapping	NO
Units	-
Value Range	0...1
Default Value	0,5

Object 2038h: Pulse Mode Overcurrent detection threshold

Object is only used when working in driver mode and is identical to the object 6510h_03h (OCD_TH) in structure and functional (chapter 4.7.2).

Object 2039h: Pulse Mode Advanced Config

In the Pulse Mode Advanced Config group, additional motor controller settings are displayed in driver mode.

Index	2039h
Name	Pulse Mode Advanced Config
Object Type	ARRAY
Number of entries	6

Since the description of structurally and functionally identical objects has already been cited in this paper, the correspondence of objects is shown in Table 7.5.

Object in driver mode		**Object in controller mode**		**Chapter**
Index_Sub	**Name**	**Index_Sub**	**Name**	
2039h_01h	GATECFG1	6510h_07h	GATECFG1	4.7.6
2039h_02h	GATECFG2	6510h_08h	GATECFG2	4.7.7
2039h_03h	CONFIG	6510h_1Ch	CM_CONFIG	4.7.27
2039h_04h	T_FAST	6510h_19h	CM_T_FAST	4.7.24
2039h_05h	TON_MIN	6510h_1Ah	CM_TON_MIN	4.7.25
2039h_06h	TOFF_MIN	6510h_1Bh	CM_TOFF_MIN	4.7.26

Tab. 7.5 — Correspondence of objects

Object 2040h: Pulse Mode Position

Object displays the current position for the driver mode with the STEP / DIR (Sub-Index 01h) control signals and the quadrature signal (Sub-Index 02h). Measured in steps and microsteps (Step / uStep), depending on the crushing settings.

Driver Mode

Sub-Index	01h
Description	Pulse Mode Position STEP/DIR mode
Data Type	INTEGER32
Access	RW
PDO Mapping	NO
Units	uStep (Step)

Sub-Index	02h
Description	Pulse Mode Position Quadrature mode
Data Type	INTEGER32
Access	RW
PDO Mapping	NO
Units	uStep (Step)

WARNING

It is not recommended to change the parameters set by default or calculated with the configuration software of the motor controller, without any urgent reasons, such as for example the electromagnetic compatibility of the installation,
This warning applies to objects:
2039h_01h GATECFG1;
2039h_02h GATECFG2;
It is strongly recommended that you review Appendix B before changing the objects:
2039h_03h … 2039h_06h.

8. Access Attributes

Additional restrictions are introduced (Table 8.1) for changing the values of objects at certain points in time (Table 8.2).

Attr.	Description
WS	Recording is possible while the motor is stopped (the windings may be energized). The shaft is fixed or freely rotates.
WH	Recording is possible when the motor is stopped and the voltage is removed from the windings (HiZ state). The shaft rotates freely.

Tab. 8.1 — Additional Access Attributes.

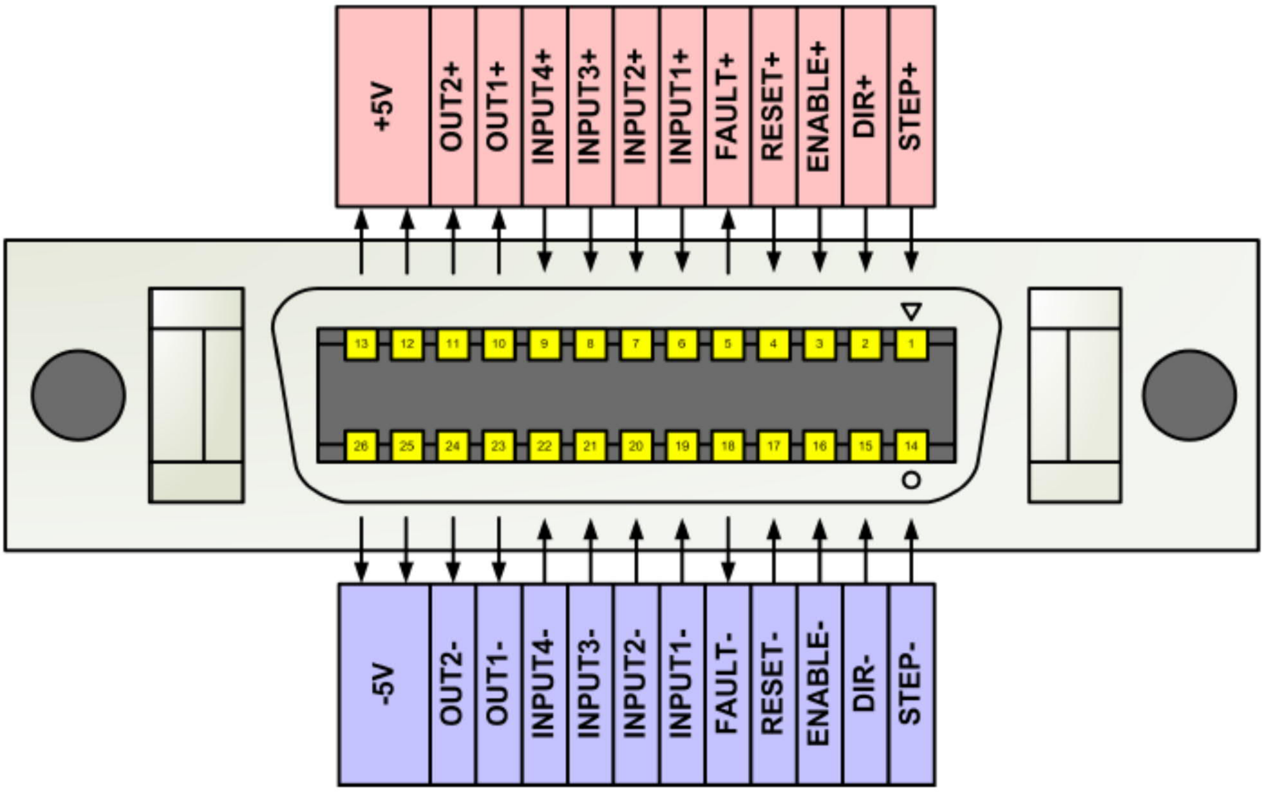
Index	Sub	Name	Attr.
6083h		Profile_acceleration	WS
6084h		Profile_deceleration	WS
6510h	05h	STEP_MODE	WH
6510h	06h	ALARM_EN	WS
6510h	07h	GATECFG1	WH
6510h	08h	GATECFG2	WH
6510h	0Eh	VM_INT_SPEED	WH
6510h	0Fh	VM_ST_SLP	WH
6510h	10h	VM_FN_SLP_ACC	WH
6510h	11h	VM_FN_SLP_DEC	WH
6510h	14h	VM_CONFIG	WH
6510h	19h	CM_T_FAST	WH
6510h	1Ah	CM_TON_MIN	WH
6510h	1Bh	CM_TOFF_MIN	WH
6510h	1Ch	CM_CONFIG	WH

Tab. 8.2 — Objects, to which additional attributes are applied.

9. Controls

9.1. Connector input output

The pin assignment of the I / O connector is shown in Figure 9.1.



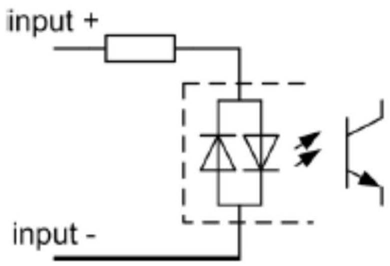
Pic. 9.1 — Pin assignments for the I / O connector

The method of connecting the end sensors and the sensors of the reference point is given in Table 9.1.

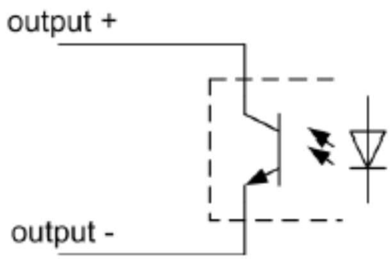
Output	**Sensor**
INPUT1	Negative limit sensor
INPUT2	Negative Reference Sensor
INPUT3	Positive Reference Sensor
INPUT4	Positive limit sensor

Tab. 9.1 — Connecting the limit sensors.

Internal Circuitry



Pic 9.2 — Digital inputs



Pic 9.3 — Digital outputs

9.2. Led Indication

On the front panel of the engine controller there are two LEDs: green and red.

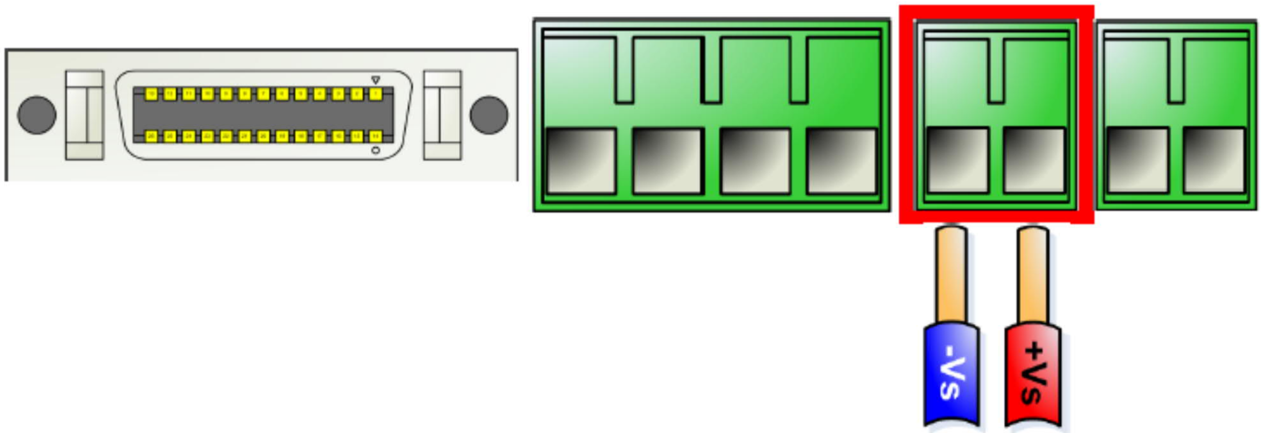
The green LED informs you of the regular, error-free operation of the controller, the red LED indicates that an error has occurred or a warning is active

The red LED next to the USB connector [8] informs you that data is being exchanged with the PC.

The green LED next to the CAN connector [10] [11] informs that data is being received over the CAN network, orange LED indicates that the transmission is in progress.

9.3. Connecting to power supply

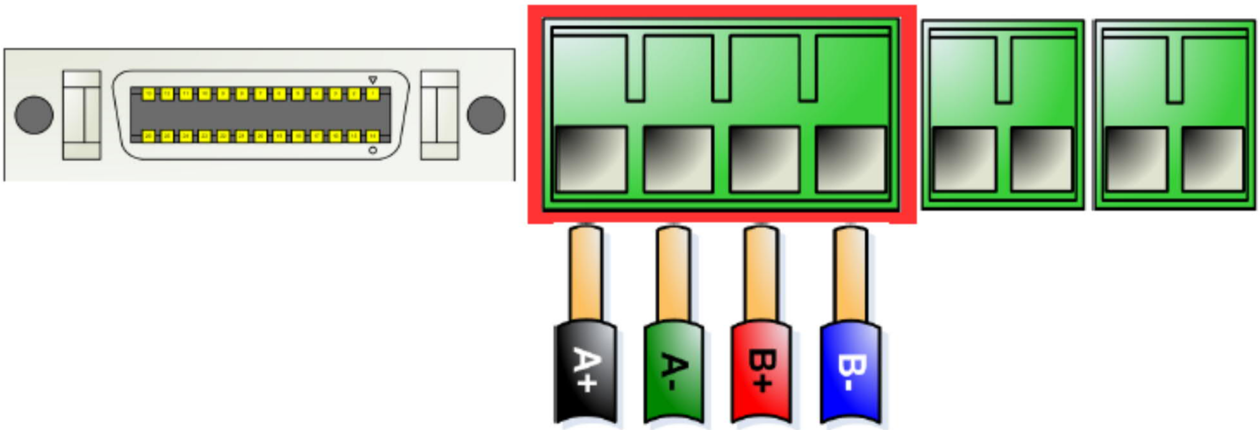
Connection of the motor controller to the power source is carried out according to the wiring diagram shown in Picture 9.4.



Pic 9.4 — Connecting the Power Supply

9.4. Motor Connection

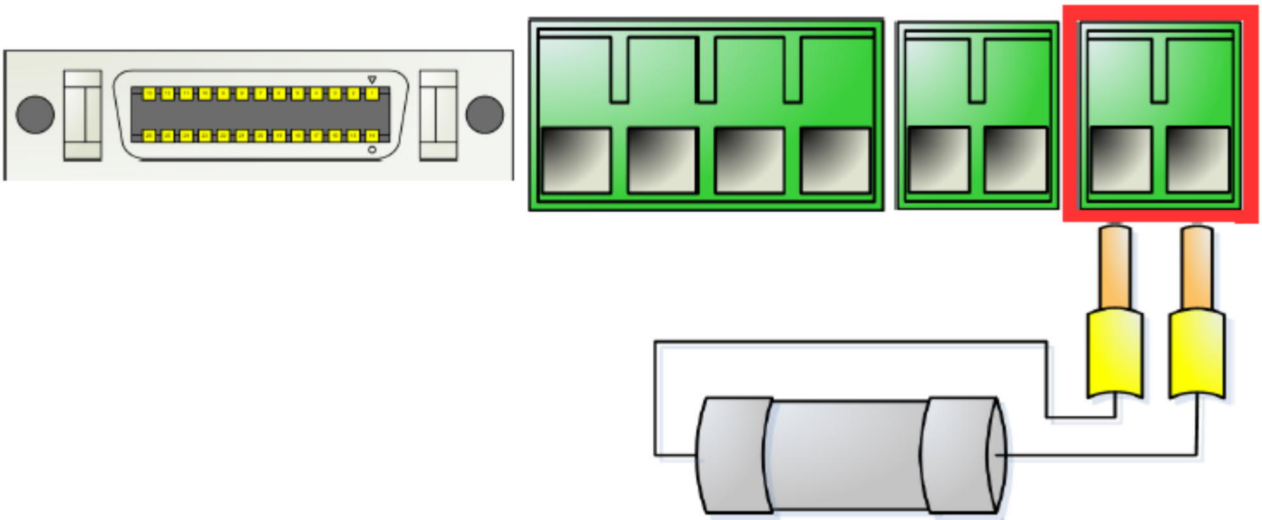
The motor is connected to the controller according to the wiring diagram shown in Picture 9.5.



Pic 9.5 — Motor Connection

9.5. Braking resistor connection

Connection of the braking resistor (if necessary) to the motor controller is carried out according to the wiring diagram shown in Picture 9.6.



Pic 9.6 — Connection of the Braking resistor

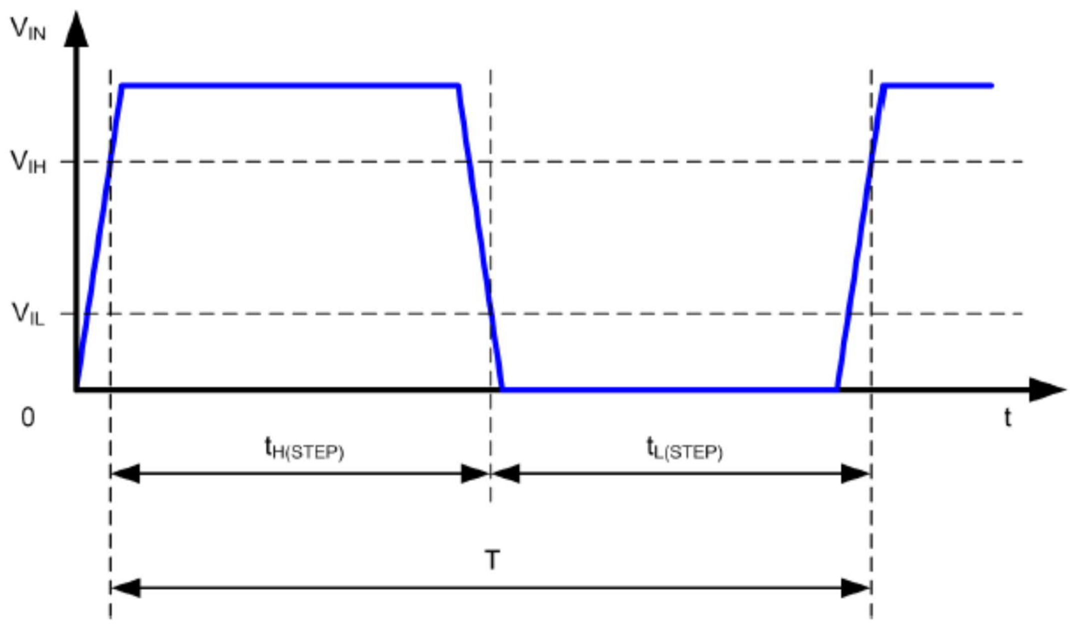
R272-42-CAN already has a built-in braking resistor with a resistance of 33Ω. The external resistor will be connected in parallel with the internal resistor.

10. Technical specifications

The specifications of the motor controller are shown in Table 10.1, the STEP input signal diagram in Picture 10.1.

№	Parameter	Value		
		Min.	**Nom.**	**Max.**
1	Supply voltage, V	12		48
2	Current in the motor windings, A	1	4,2	6,5
3	Voltage at the logic input, V	-6,3		6,3
4	Logic output current, mA			50
5	Output current + 5V, A			1
6	Maximum frequency of STEP signal, kHz			50
7	Maximum frequency of quadrature control signal, kHz			1
8	The signal period STEP (T), µs	20		
9	High Level Duration ($t_{H(STEP)}$), µs	5		
10	Low Level Duration ($t_{L(STEP)}$), µs	5		
11	Low Logic Input Level ($\lvert V_{IL} \rvert$), V			2,2
12	High Logic Input Level ($\lvert V_{IH} \rvert$), V	2,6		
13	Microstepping	1/1, 1/2, 1/4, 1/8, 1/16, 1/32, 1/64, 1/128		

Tab. 10.1 — Technical specifications.



Pic 10.1 — Input Signal

.

Appendix A. Object Dictionary

Object	Object name	Object code	Type	Sub-Index	Attr.	PDO
1000h	Device Type	VAR	UNSIGNED32		RO	
1001h	Error Register	VAR	UNSIGNED8		RO	
1003h	Predefined Error Field	ARRAY				
1003h	Number of Errors	VAR	UNSIGNED8	00h	RW	
1003h	Standard Error Field	VAR	UNSIGNED32	01h	RO	
1003h	Standard Error Field_2	VAR	UNSIGNED32	02h	RO	
1003h	Standard Error Field_3	VAR	UNSIGNED32	03h	RO	
1003h	Standard Error Field_4	VAR	UNSIGNED32	04h	RO	
1003h	Standard Error Field_5	VAR	UNSIGNED32	05h	RO	
1003h	Standard Error Field_6	VAR	UNSIGNED32	06h	RO	
1003h	Standard Error Field_7	VAR	UNSIGNED32	07h	RO	
1003h	Standard Error Field_8	VAR	UNSIGNED32	08h	RO	
1003h	Standard Error Field_9	VAR	UNSIGNED32	09h	RO	
1003h	Standard Error Field_a	VAR	UNSIGNED32	0Ah	RO	
1003h	Standard Error Field_b	VAR	UNSIGNED32	0Bh	RO	
1003h	Standard Error Field_c	VAR	UNSIGNED32	0Ch	RO	
1003h	Standard Error Field_d	VAR	UNSIGNED32	0Dh	RO	
1003h	Standard Error Field_e	VAR	UNSIGNED32	0Eh	RO	
1003h	Standard Error Field_f	VAR	UNSIGNED32	0Fh	RO	
1003h	Standard Error Field_10	VAR	UNSIGNED32	10h	RO	

Object	Object name	Object code	Type	Sub-Index	Attr.	PDO
1005h	COB ID SYNC	VAR	UNSIGNED32		RW	
1006h	Communication Cycle Period	VAR	UNSIGNED32		RW	
1008h	Manufacturer Device Name	VAR	VISIBLE_STRING		CONST	
1009h	Manufacturer Hardware Version	VAR	VISIBLE_STRING		CONST	
100Ah	Manufacturer Software Version	VAR	VISIBLE_STRING		CONST	
100Ch	Guard Time	VAR	UNSIGNED16		RW	
100Dh	Life Time Factor	VAR	UNSIGNED8		RW	
1010h	Store Parameter Field	ARRAY				
1010h	Number of entries	VAR	UNSIGNED8	00h	RO	
1010h	Save all Parameters	VAR	UNSIGNED32	01h	WO	
1010h	Save Communication Parameters	VAR	UNSIGNED32	02h	WO	
1010h	Save Motor Parameters	VAR	UNSIGNED32	03h	WO	
1011h	Restore Default Parameters	ARRAY				
1011h	Number of entries	VAR	UNSIGNED8	00h	RO	
1011h	Restore all Default Parameters	VAR	UNSIGNED32	01h	WO	
1011h	Restore Communication Default Parameters	VAR	UNSIGNED32	02h	WO	
1011h	Restore Motor Default Parameters	VAR	UNSIGNED32	03h	WO	
1014h	COB ID EMCY	VAR	UNSIGNED32		RW	
1017h	Producer Heartbeat Time	VAR	UNSIGNED16		RW	
1018h	Identity Object	RECORD				
1018h	Number of entries	VAR	UNSIGNED8	00h	RO	
1018h	Vendor Id	VAR	UNSIGNED32	01h	RO	
1018h	Product Code	VAR	UNSIGNED32	02h	RO	

Object	Object name	Object code	Type	Sub-Index	Attr.	PDO
1018h	Revision number	VAR	UNSIGNED32	03h	RO	
1018h	Serial number	VAR	UNSIGNED64	04h	RO	
1200h	Server SDO Parameter	RECORD				
1200h	Number of entries	VAR	UNSIGNED8	00h	RO	
1200h	COB ID Client to Server	VAR	UNSIGNED32	01h	RO	
1200h	COB ID Server to Client	VAR	UNSIGNED32	02h	RO	
1400h	Receive PDO Communication Parameter 0	RECORD				
1400h	Number of entries	VAR	UNSIGNED8	00h	RO	
1400h	COB ID	VAR	UNSIGNED32	01h	RW	
1400h	Transmission Type	VAR	UNSIGNED8	02h	RW	
1401h	Receive PDO Communication Parameter 1	RECORD				
1401h	Number of entries	VAR	UNSIGNED8	00h	RO	
1401h	COB ID	VAR	UNSIGNED32	01h	RW	
1401h	Transmission Type	VAR	UNSIGNED8	02h	RW	
1402h	Receive PDO Communication Parameter 2	RECORD				
1402h	Number of entries	VAR	UNSIGNED8	00h	RO	
1402h	COB ID	VAR	UNSIGNED32	01h	RW	
1402h	Transmission Type	VAR	UNSIGNED8	02h	RW	
1403h	Receive PDO Communication Parameter 3	RECORD				
1403h	Number of entries	VAR	UNSIGNED8	00h	RO	
1403h	COB ID	VAR	UNSIGNED32	01h	RW	
1403h	Transmission Type	VAR	UNSIGNED8	02h	RW	
1404h	Receive PDO Communication Parameter 4	RECORD				

Object	Object name	Object code	Type	Sub-Index	Attr.	PDO
1404h	Number of entries	VAR	UNSIGNED8	00h	RO	
1404h	COB ID	VAR	UNSIGNED32	01h	RW	
1404h	Transmission Type	VAR	UNSIGNED8	02h	RW	
1405h	Receive PDO Communication Parameter 5	RECORD				
1405h	Number of entries	VAR	UNSIGNED8	00h	RO	
1405h	COB ID	VAR	UNSIGNED32	01h	RW	
1405h	Transmission Type	VAR	UNSIGNED8	02h	RW	
1406h	Receive PDO Communication Parameter 6	RECORD				
1406h	Number of entries	VAR	UNSIGNED8	00h	RO	
1406h	COB ID	VAR	UNSIGNED32	01h	RW	
1406h	Transmission Type	VAR	UNSIGNED8	02h	RW	
1407h	Receive PDO Communication Parameter 7	RECORD				
1407h	Number of entries	VAR	UNSIGNED8	00h	RO	
1407h	COB ID	VAR	UNSIGNED32	01h	RW	
1407h	Transmission Type	VAR	UNSIGNED8	02h	RW	
1600h	Receive PDO Mapping Parameter 0	RECORD				
1600h	Number of entries	VAR	UNSIGNED8	00h	RW	
1600h	PDO Mapping Entry	VAR	UNSIGNED32	01h	RW	
1600h	PDO Mapping Entry_2	VAR	UNSIGNED32	02h	RW	
1600h	PDO Mapping Entry_3	VAR	UNSIGNED32	03h	RW	
1600h	PDO Mapping Entry_4	VAR	UNSIGNED32	04h	RW	
1600h	PDO Mapping Entry_5	VAR	UNSIGNED32	05h	RW	
1600h	PDO Mapping Entry_6	VAR	UNSIGNED32	06h	RW	

Object	Object name	Object code	Type	Sub-Index	Attr.	PDO
1600h	PDO Mapping Entry_7	VAR	UNSIGNED32	07h	RW	
1600h	PDO Mapping Entry_8	VAR	UNSIGNED32	08h	RW	
1601h	Receive PDO Mapping Parameter 1	RECORD				
1601h	Number of entries	VAR	UNSIGNED8	00h	RW	
1601h	PDO Mapping Entry	VAR	UNSIGNED32	01h	RW	
1601h	PDO Mapping Entry_2	VAR	UNSIGNED32	02h	RW	
1601h	PDO Mapping Entry_3	VAR	UNSIGNED32	03h	RW	
1601h	PDO Mapping Entry_4	VAR	UNSIGNED32	04h	RW	
1601h	PDO Mapping Entry_5	VAR	UNSIGNED32	05h	RW	
1601h	PDO Mapping Entry_6	VAR	UNSIGNED32	06h	RW	
1601h	PDO Mapping Entry_7	VAR	UNSIGNED32	07h	RW	
1601h	PDO Mapping Entry_8	VAR	UNSIGNED32	08h	RW	
1602h	Receive PDO Mapping Parameter 2	RECORD				
1602h	Number of entries	VAR	UNSIGNED8	00h	RW	
1602h	PDO Mapping Entry	VAR	UNSIGNED32	01h	RW	
1602h	PDO Mapping Entry_2	VAR	UNSIGNED32	02h	RW	
1602h	PDO Mapping Entry_3	VAR	UNSIGNED32	03h	RW	
1602h	PDO Mapping Entry_4	VAR	UNSIGNED32	04h	RW	
1602h	PDO Mapping Entry_5	VAR	UNSIGNED32	05h	RW	
1602h	PDO Mapping Entry_6	VAR	UNSIGNED32	06h	RW	
1602h	PDO Mapping Entry_7	VAR	UNSIGNED32	07h	RW	
1602h	PDO Mapping Entry_8	VAR	UNSIGNED32	08h	RW	
1603h	Receive PDO Mapping Parameter 3	RECORD				

Object	Object name	Object code	Type	Sub-Index	Attr.	PDO
1603h	Number of entries	VAR	UNSIGNED8	00h	RW	
1603h	PDO Mapping Entry	VAR	UNSIGNED32	01h	RW	
1603h	PDO Mapping Entry_2	VAR	UNSIGNED32	02h	RW	
1603h	PDO Mapping Entry_3	VAR	UNSIGNED32	03h	RW	
1603h	PDO Mapping Entry_4	VAR	UNSIGNED32	04h	RW	
1603h	PDO Mapping Entry_5	VAR	UNSIGNED32	05h	RW	
1603h	PDO Mapping Entry_6	VAR	UNSIGNED32	06h	RW	
1603h	PDO Mapping Entry_7	VAR	UNSIGNED32	07h	RW	
1603h	PDO Mapping Entry_8	VAR	UNSIGNED32	08h	RW	
1604h	Receive PDO Mapping Parameter 4	RECORD				
1604h	Number of entries	VAR	UNSIGNED8	00h	RW	
1604h	PDO Mapping Entry	VAR	UNSIGNED32	01h	RW	
1604h	PDO Mapping Entry_2	VAR	UNSIGNED32	02h	RW	
1604h	PDO Mapping Entry_3	VAR	UNSIGNED32	03h	RW	
1604h	PDO Mapping Entry_4	VAR	UNSIGNED32	04h	RW	
1604h	PDO Mapping Entry_5	VAR	UNSIGNED32	05h	RW	
1604h	PDO Mapping Entry_6	VAR	UNSIGNED32	06h	RW	
1604h	PDO Mapping Entry_7	VAR	UNSIGNED32	07h	RW	
1604h	PDO Mapping Entry_8	VAR	UNSIGNED32	08h	RW	
1605h	Receive PDO Mapping Parameter 5	RECORD				
1605h	Number of entries	VAR	UNSIGNED8	00h	RW	
1605h	PDO Mapping Entry	VAR	UNSIGNED32	01h	RW	
1605h	PDO Mapping Entry_2	VAR	UNSIGNED32	02h	RW	

Object	Object name	Object code	Type	Sub-Index	Attr.	PDO
1605h	PDO Mapping Entry_3	VAR	UNSIGNED32	03h	RW	
1605h	PDO Mapping Entry_4	VAR	UNSIGNED32	04h	RW	
1605h	PDO Mapping Entry_5	VAR	UNSIGNED32	05h	RW	
1605h	PDO Mapping Entry_6	VAR	UNSIGNED32	06h	RW	
1605h	PDO Mapping Entry_7	VAR	UNSIGNED32	07h	RW	
1605h	PDO Mapping Entry_8	VAR	UNSIGNED32	08h	RW	
1606h	Receive PDO Mapping Parameter 6	RECORD				
1606h	Number of entries	VAR	UNSIGNED8	00h	RW	
1606h	PDO Mapping Entry	VAR	UNSIGNED32	01h	RW	
1606h	PDO Mapping Entry_2	VAR	UNSIGNED32	02h	RW	
1606h	PDO Mapping Entry_3	VAR	UNSIGNED32	03h	RW	
1606h	PDO Mapping Entry_4	VAR	UNSIGNED32	04h	RW	
1606h	PDO Mapping Entry_5	VAR	UNSIGNED32	05h	RW	
1606h	PDO Mapping Entry_6	VAR	UNSIGNED32	06h	RW	
1606h	PDO Mapping Entry_7	VAR	UNSIGNED32	07h	RW	
1606h	PDO Mapping Entry_8	VAR	UNSIGNED32	08h	RW	
1607h	Receive PDO Mapping Parameter 7	RECORD				
1607h	Number of entries	VAR	UNSIGNED8	00h	RW	
1607h	PDO Mapping Entry	VAR	UNSIGNED32	01h	RW	
1607h	PDO Mapping Entry_2	VAR	UNSIGNED32	02h	RW	
1607h	PDO Mapping Entry_3	VAR	UNSIGNED32	03h	RW	
1607h	PDO Mapping Entry_4	VAR	UNSIGNED32	04h	RW	
1607h	PDO Mapping Entry_5	VAR	UNSIGNED32	05h	RW	

Object	Object name	Object code	Type	Sub-Index	Attr.	PDO
1607h	PDO Mapping Entry_6	VAR	UNSIGNED32	06h	RW	
1607h	PDO Mapping Entry_7	VAR	UNSIGNED32	07h	RW	
1607h	PDO Mapping Entry_8	VAR	UNSIGNED32	08h	RW	
1800h	Transmit PDO Communication Parameter 0	RECORD				
1800h	Number of entries	VAR	UNSIGNED8	00h	RO	
1800h	COB ID	VAR	UNSIGNED32	01h	RW	
1800h	Transmission Type	VAR	UNSIGNED8	02h	RW	
1800h	Inhibit Time	VAR	UNSIGNED16	03h	RW	
1801h	Transmit PDO Communication Parameter 1	RECORD				
1801h	Number of entries	VAR	UNSIGNED8	00h	RO	
1801h	COB ID	VAR	UNSIGNED32	01h	RW	
1801h	Transmission Type	VAR	UNSIGNED8	02h	RW	
1801h	Inhibit Time	VAR	UNSIGNED16	03h	RW	
1802h	Transmit PDO Communication Parameter 2	RECORD				
1802h	Number of entries	VAR	UNSIGNED8	00h	RO	
1802h	COB ID	VAR	UNSIGNED32	01h	RW	
1802h	Transmission Type	VAR	UNSIGNED8	02h	RW	
1802h	Inhibit Time	VAR	UNSIGNED16	03h	RW	
1803h	Transmit PDO Communication Parameter 3	RECORD				
1803h	Number of entries	VAR	UNSIGNED8	00h	RO	
1803h	COB ID	VAR	UNSIGNED32	01h	RW	
1803h	Transmission Type	VAR	UNSIGNED8	02h	RW	
1803h	Inhibit Time	VAR	UNSIGNED16	03h	RW	

Object	Object name	Object code	Type	Sub-Index	Attr.	PDO
1804h	Transmit PDO Communication Parameter 4	RECORD				
1804h	Number of entries	VAR	UNSIGNED8	00h	RO	
1804h	COB ID	VAR	UNSIGNED32	01h	RW	
1804h	Transmission Type	VAR	UNSIGNED8	02h	RW	
1804h	Inhibit Time	VAR	UNSIGNED16	03h	RW	
1805h	Transmit PDO Communication Parameter 5	RECORD				
1805h	Number of entries	VAR	UNSIGNED8	00h	RO	
1805h	COB ID	VAR	UNSIGNED32	01h	RW	
1805h	Transmission Type	VAR	UNSIGNED8	02h	RW	
1805h	Inhibit Time	VAR	UNSIGNED16	03h	RW	
1806h	Transmit PDO Communication Parameter 6	RECORD				
1806h	Number of entries	VAR	UNSIGNED8	00h	RO	
1806h	COB ID	VAR	UNSIGNED32	01h	RW	
1806h	Transmission Type	VAR	UNSIGNED8	02h	RW	
1806h	Inhibit Time	VAR	UNSIGNED16	03h	RW	
1807h	Transmit PDO Communication Parameter 7	RECORD				
1807h	Number of entries	VAR	UNSIGNED8	00h	RO	
1807h	COB ID	VAR	UNSIGNED32	01h	RW	
1807h	Transmission Type	VAR	UNSIGNED8	02h	RW	
1807h	Inhibit Time	VAR	UNSIGNED16	03h	RW	
1A00h	Transmit PDO Mapping Parameter 0	RECORD				
1A00h	Number of entries	VAR	UNSIGNED8	00h	RW	
1A00h	PDO Mapping Entry	VAR	UNSIGNED32	01h	RW	

Object	Object name	Object code	Type	Sub-Index	Attr.	PDO
1A00h	PDO Mapping Entry_2	VAR	UNSIGNED32	02h	RW	
1A00h	PDO Mapping Entry_3	VAR	UNSIGNED32	03h	RW	
1A00h	PDO Mapping Entry_4	VAR	UNSIGNED32	04h	RW	
1A00h	PDO Mapping Entry_5	VAR	UNSIGNED32	05h	RW	
1A00h	PDO Mapping Entry_6	VAR	UNSIGNED32	06h	RW	
1A00h	PDO Mapping Entry_7	VAR	UNSIGNED32	07h	RW	
1A00h	PDO Mapping Entry_8	VAR	UNSIGNED32	08h	RW	
1A01h	Transmit PDO Mapping Parameter 1	RECORD				
1A01h	Number of entries	VAR	UNSIGNED8	00h	RW	
1A01h	PDO Mapping Entry	VAR	UNSIGNED32	01h	RW	
1A01h	PDO Mapping Entry_2	VAR	UNSIGNED32	02h	RW	
1A01h	PDO Mapping Entry_3	VAR	UNSIGNED32	03h	RW	
1A01h	PDO Mapping Entry_4	VAR	UNSIGNED32	04h	RW	
1A01h	PDO Mapping Entry_5	VAR	UNSIGNED32	05h	RW	
1A01h	PDO Mapping Entry_6	VAR	UNSIGNED32	06h	RW	
1A01h	PDO Mapping Entry_7	VAR	UNSIGNED32	07h	RW	
1A01h	PDO Mapping Entry_8	VAR	UNSIGNED32	08h	RW	
1A02h	Transmit PDO Mapping Parameter 2	RECORD				
1A02h	Number of entries	VAR	UNSIGNED8	00h	RW	
1A02h	PDO Mapping Entry	VAR	UNSIGNED32	01h	RW	
1A02h	PDO Mapping Entry_2	VAR	UNSIGNED32	02h	RW	
1A02h	PDO Mapping Entry_3	VAR	UNSIGNED32	03h	RW	
1A02h	PDO Mapping Entry_4	VAR	UNSIGNED32	04h	RW	

Object	Object name	Object code	Type	Sub-Index	Attr.	PDO
1A02h	PDO Mapping Entry_5	VAR	UNSIGNED32	05h	RW	
1A02h	PDO Mapping Entry_6	VAR	UNSIGNED32	06h	RW	
1A02h	PDO Mapping Entry_7	VAR	UNSIGNED32	07h	RW	
1A02h	PDO Mapping Entry_8	VAR	UNSIGNED32	08h	RW	
1A03h	Transmit PDO Mapping Parameter 3	RECORD				
1A03h	Number of entries	VAR	UNSIGNED8	00h	RW	
1A03h	PDO Mapping Entry	VAR	UNSIGNED32	01h	RW	
1A03h	PDO Mapping Entry_2	VAR	UNSIGNED32	02h	RW	
1A03h	PDO Mapping Entry_3	VAR	UNSIGNED32	03h	RW	
1A03h	PDO Mapping Entry_4	VAR	UNSIGNED32	04h	RW	
1A03h	PDO Mapping Entry_5	VAR	UNSIGNED32	05h	RW	
1A03h	PDO Mapping Entry_6	VAR	UNSIGNED32	06h	RW	
1A03h	PDO Mapping Entry_7	VAR	UNSIGNED32	07h	RW	
1A03h	PDO Mapping Entry_8	VAR	UNSIGNED32	08h	RW	
1A04h	Transmit PDO Mapping Parameter 4	RECORD				
1A04h	Number of entries	VAR	UNSIGNED8	00h	RW	
1A04h	PDO Mapping Entry	VAR	UNSIGNED32	01h	RW	
1A04h	PDO Mapping Entry_2	VAR	UNSIGNED32	02h	RW	
1A04h	PDO Mapping Entry_3	VAR	UNSIGNED32	03h	RW	
1A04h	PDO Mapping Entry_4	VAR	UNSIGNED32	04h	RW	
1A04h	PDO Mapping Entry_5	VAR	UNSIGNED32	05h	RW	
1A04h	PDO Mapping Entry_6	VAR	UNSIGNED32	06h	RW	
1A04h	PDO Mapping Entry_7	VAR	UNSIGNED32	07h	RW	

Object	Object name	Object code	Type	Sub-Index	Attr.	PDO
1A04h	PDO Mapping Entry_8	VAR	UNSIGNED32	08h	RW	
1A05h	Transmit PDO Mapping Parameter 5	RECORD				
1A05h	Number of entries	VAR	UNSIGNED8	00h	RW	
1A05h	PDO Mapping Entry	VAR	UNSIGNED32	01h	RW	
1A05h	PDO Mapping Entry_2	VAR	UNSIGNED32	02h	RW	
1A05h	PDO Mapping Entry_3	VAR	UNSIGNED32	03h	RW	
1A05h	PDO Mapping Entry_4	VAR	UNSIGNED32	04h	RW	
1A05h	PDO Mapping Entry_5	VAR	UNSIGNED32	05h	RW	
1A05h	PDO Mapping Entry_6	VAR	UNSIGNED32	06h	RW	
1A05h	PDO Mapping Entry_7	VAR	UNSIGNED32	07h	RW	
1A05h	PDO Mapping Entry_8	VAR	UNSIGNED32	08h	RW	
1A06h	Transmit PDO Mapping Parameter 6	RECORD				
1A06h	Number of entries	VAR	UNSIGNED8	00h	RW	
1A06h	PDO Mapping Entry	VAR	UNSIGNED32	01h	RW	
1A06h	PDO Mapping Entry_2	VAR	UNSIGNED32	02h	RW	
1A06h	PDO Mapping Entry_3	VAR	UNSIGNED32	03h	RW	
1A06h	PDO Mapping Entry_4	VAR	UNSIGNED32	04h	RW	
1A06h	PDO Mapping Entry_5	VAR	UNSIGNED32	05h	RW	
1A06h	PDO Mapping Entry_6	VAR	UNSIGNED32	06h	RW	
1A06h	PDO Mapping Entry_7	VAR	UNSIGNED32	07h	RW	
1A06h	PDO Mapping Entry_8	VAR	UNSIGNED32	08h	RW	
1A07h	Transmit PDO Mapping Parameter 7	RECORD				
1A07h	Number of entries	VAR	UNSIGNED8	00h	RW	

Object	Object name	Object code	Type	Sub-Index	Attr.	PDO
1A07h	PDO Mapping Entry	VAR	UNSIGNED32	01h	RW	
1A07h	PDO Mapping Entry_2	VAR	UNSIGNED32	02h	RW	
1A07h	PDO Mapping Entry_3	VAR	UNSIGNED32	03h	RW	
1A07h	PDO Mapping Entry_4	VAR	UNSIGNED32	04h	RW	
1A07h	PDO Mapping Entry_5	VAR	UNSIGNED32	05h	RW	
1A07h	PDO Mapping Entry_6	VAR	UNSIGNED32	06h	RW	
1A07h	PDO Mapping Entry_7	VAR	UNSIGNED32	07h	RW	
1A07h	PDO Mapping Entry_8	VAR	UNSIGNED32	08h	RW	
2014h	TPDO1 Transmit mask	ARRAY				
2014h	Number of entries	VAR	UNSIGNED8	00h	RO	
2014h	TPDO1 Transmit mask low	VAR	UNSIGNED32	01h	RW	
2014h	TPDO1 Transmit mask high	VAR	UNSIGNED32	02h	RW	
2015h	TPDO2 Transmit mask	ARRAY				
2015h	Number of entries	VAR	UNSIGNED8	00h	RO	
2015h	TPDO2 Transmit mask low	VAR	UNSIGNED32	01h	RW	
2015h	TPDO2 Transmit mask high	VAR	UNSIGNED32	02h	RW	
2016h	TPDO3 Transmit mask	ARRAY				
2016h	Number of entries	VAR	UNSIGNED8	00h	RO	
2016h	TPDO3 Transmit mask low	VAR	UNSIGNED32	01h	RW	
2016h	TPDO3 Transmit mask high	VAR	UNSIGNED32	02h	RW	
2017h	TPDO4 Transmit mask	ARRAY				
2017h	Number of entries	VAR	UNSIGNED8	00h	RO	
2017h	TPDO4 Transmit mask low	VAR	UNSIGNED32	01h	RW	

Object	Object name	Object code	Type	Sub-Index	Attr.	PDO
2017h	TPDO4 Transmit mask high	VAR	UNSIGNED32	02h	RW	
2024h	Fault Event	ARRAY				
2024h	Number of entries	VAR	UNSIGNED8	00h	RO	
2024h	Fault Event Node Guard out of timeout	VAR	UNSIGNED8	01h	RW	
2024h	Fault Event Temperature out of limit	VAR	UNSIGNED8	02h	RW	
2024h	Fault Event Pulse Mode Error Write Config	VAR	UNSIGNED8	03h	RW	
2024h	Fault Event Pulse Mode Termal warning	VAR	UNSIGNED8	04h	RW	
2024h	Fault Event Controller Mode Error Write Config	VAR	UNSIGNED8	05h	RW	
2024h	Fault Event Controller Mode Termal warning	VAR	UNSIGNED8	06h	RW	
2024h	Fault Event Controller Mode Termal shutdown	VAR	UNSIGNED8	07h	RW	
2024h	Fault Event Controller Mode Stall detection	VAR	UNSIGNED8	08h	RW	
2024h	Fault Event Controller Mode UVLO	VAR	UNSIGNED8	09h	RW	
2024h	Fault Event Controller Mode ADC UVLO	VAR	UNSIGNED8	0Ah	RW	
2024h	Fault Event Controller Mode CMD error	VAR	UNSIGNED8	0Bh	RW	
2024h	Fault Event Controller Mode Overcurrent	VAR	UNSIGNED8	0Ch	RW	
2024h	Fault Event Controller Mode 0 Divisor	VAR	UNSIGNED8	0Dh	RW	
2024h	Fault Event Controller Mode CANopen Frame lost	VAR	UNSIGNED8	0Eh	RW	
2024h	Fault Event Controller Mode Out 5V over current	VAR	UNSIGNED8	0Fh	RW	
2024h	Fault Event Controller Mode Curve Generator error	VAR	UNSIGNED8	10h	RW	
2024h	Fault Event Controller Mode Limit Switch error	VAR	UNSIGNED8	11h	RW	
2024h	Fault Event Controller Mode Home Method 35	VAR	UNSIGNED8	12h	RW	
2029h	Temperature sensor	ARRAY				
2029h	Number of entries	VAR	UNSIGNED8	00h	RO	

Object	Object name	Object code	Type	Sub-Index	Attr.	PDO
2029h	Temperature sensor value	VAR	REAL32	01h	RO	V
2029h	Temperature sensor period	VAR	UNSIGNED8	02h	RW	
2029h	Temperature sensor threshold	VAR	REAL32	03h	RW	
2032h	Pulse Mode Current	VAR	UNSIGNED8		RO	
2033h	Pulse Mode Microstep	VAR	UNSIGNED8		RO	
2034h	Pulse Mode Config	VAR	UNSIGNED8		RW	
2035h	Pulse Mode Hold Timeout	VAR	UNSIGNED16		RW	
2036h	Pulse Mode Hold Current Factor	VAR	REAL32		RW	
2038h	Pulse Mode Overcurrent detection threshold	VAR	UNSIGNED16		RW	
2039h	Pulse Mode Advanced Config	ARRAY				
2039h	Number of entries	VAR	UNSIGNED8	00h	RO	
2039h	GATECFG1	VAR	UNSIGNED16	01h	RW	
2039h	GATECFG2	VAR	UNSIGNED8	02h	RW	
2039h	CONFIG	VAR	UNSIGNED16	03h	RW	
2039h	T_FAST	VAR	UNSIGNED8	04h	RW	
2039h	TON_MIN	VAR	UNSIGNED8	05h	RW	
2039h	TOFF_MIN	VAR	UNSIGNED8	06h	RW	
2040h	Pulse Mode Position	ARRAY				
2040h	Number of entries	VAR	UNSIGNED8	00h	RO	
2040h	Pulse Mode Position STEP/DIR mode	VAR	INTEGER32	01h	RW	
2040h	Pulse Mode Position Quadrature mode	VAR	INTEGER32	02h	RW	
2041h	Main_velocity_factor	ARRAY				
2041h	NrOfObjects	VAR	UNSIGNED8	00h	RO	

Object	Object name	Object code	Type	Sub-Index	Attr.	PDO
2041h	Main_velocity_factor_numerator	VAR	UNSIGNED32	01h	RW	V
2041h	Main_velocity_factor_divisor	VAR	UNSIGNED32	02h	RW	V
2042h	Main_acceleration_factor	ARRAY				
2042h	NrOfObjects	VAR	UNSIGNED8	00h	RO	
2042h	Main_acceleration_factor_numerator	VAR	UNSIGNED32	01h	RW	V
2042h	Main_acceleration_factor_divisor	VAR	UNSIGNED32	02h	RW	V
2043h	FS_SPD_factor	ARRAY				
2043h	NrOfObjects	VAR	UNSIGNED8	00h	RO	
2043h	FS_SPD_factor_numerator	VAR	UNSIGNED32	01h	RW	V
2043h	FS_SPD_factor_divisor	VAR	UNSIGNED32	02h	RW	V
2044h	MAX_SPEED_factor	ARRAY				
2044h	NrOfObjects	VAR	UNSIGNED8	00h	RO	
2044h	MAX_SPEED_factor_numerator	VAR	UNSIGNED32	01h	RW	V
2044h	MAX_SPEED_factor_divisor	VAR	UNSIGNED32	02h	RW	V
2045h	CAN Baudrate	VAR	UNSIGNED8		RW	
6040h	Controlword	VAR	UNSIGNED16		RW	V
6041h	Statusword	VAR	UNSIGNED16		RO	V
605Ah	Quick_stop_option_code	VAR	INTEGER16		RW	V
6060h	Modes_of_operation	VAR	INTEGER8		RW	V
6061h	Modes_of_operation_display	VAR	INTEGER8		RO	V
6063h	Position_actual_value_s	VAR	INTEGER32		RO	V
6064h	Position_actual_value	VAR	INTEGER32		RO	V
6069h	Velocity_sensor_actual_value	VAR	INTEGER32		RO	V

Object	Object name	Object code	Type	Sub-Index	Attr.	PDO
606Ah	Sensor_selection_code	VAR	INTEGER16		RW	V
606Bh	Velocity_demand_value	VAR	INTEGER32		RO	V
606Ch	Velocity_actual_value	VAR	INTEGER32		RO	V
607Ah	Target_position	VAR	INTEGER32		RW	V
607Ch	Home_offset	VAR	INTEGER32		RW	V
607Eh	Polarity	VAR	UNSIGNED8		RW	V
6081h	Profile_velocity	VAR	UNSIGNED32		RW	V
6083h	Profile_acceleration	VAR	UNSIGNED32		RW	V
6084h	Profile_deceleration	VAR	UNSIGNED32		RW	V
6086h	Motion_profile_type	VAR	INTEGER16		RW	V
6093h	Position_factor	ARRAY				
6093h	Position_factor_number_of_entries	VAR	UNSIGNED8	00h	RO	
6093h	Position_factor_Numerator	VAR	UNSIGNED32	01h	RW	V
6093h	Position_factor_Divisor	VAR	UNSIGNED32	02h	RW	V
6095h	Velocity_factor	ARRAY				
6095h	Velocity_factor_number_of_entries	VAR	UNSIGNED8	00h	RO	
6095h	Velocity_factor_Numerator	VAR	UNSIGNED32	01h	RW	V
6095h	Velocity_factor_Divisor	VAR	UNSIGNED32	02h	RW	V
6097h	Acceleration_factor	ARRAY				
6097h	Acceleration_factor_number_of_entries	VAR	UNSIGNED8	00h	RO	
6097h	Acceleration_factor_Numerator	VAR	UNSIGNED32	01h	RW	V
6097h	Acceleration_factor_Divisor	VAR	UNSIGNED32	02h	RW	V
6098h	Homing_method	VAR	INTEGER8		RW	

Object	Object name	Object code	Type	Sub-Index	Attr.	PDO
6099h	Homing_speeds	ARRAY				
6099h	Homing_speeds_number_of_entries	VAR	UNSIGNED8	00h	RO	
6099h	Homing_speeds_Speed_during_search_for_switch	VAR	UNSIGNED32	01h	RW	V
6099h	Homing_speeds_Speed_during_search_for_zero	VAR	UNSIGNED32	02h	RW	V
60FDh	Digital_inputs	VAR	UNSIGNED32		RW	V
60FEh	Digital_outputs	ARRAY				
60FEh	Digital_outputs_number_of_entries	VAR	UNSIGNED8	00h	RO	
60FEh	Digital_outputs_Physical_outputs	VAR	UNSIGNED32	01h	RW	V
60FEh	Digital_outputs_Bitmask	VAR	UNSIGNED32	02h	RW	V
60FFh	Target_velocity	VAR	INTEGER32		RW	V
6504h	Drive_manufacturer	VAR	VISIBLE_STRING		RO	
6505h	http_drive_catalog_address	VAR	VISIBLE_STRING		RO	
6510h	Drive_data	RECORD				
6510h	Drive_data_number_of_entries	VAR	UNSIGNED8	00h	RO	
6510h	Compatibility_control	VAR	UNSIGNED16	01h	RW	
6510h	ADC_OUT	VAR	UNSIGNED8	02h	RW	V
6510h	OCD_TH	VAR	UNSIGNED16	03h	RW	
6510h	FS_SPD	VAR	UNSIGNED32	04h	RW	
6510h	STEP_MODE	VAR	UNSIGNED8	05h	RW	
6510h	ALARM_EN	VAR	UNSIGNED8	06h	RW	
6510h	GATECFG1	VAR	UNSIGNED16	07h	RW	
6510h	GATECFG2	VAR	UNSIGNED8	08h	RW	
6510h	STATUS	VAR	UNSIGNED16	09h	RO	V

Object	Object name	Object code	Type	Sub-Index	Attr.	PDO
6510h	VM_KVAL_HOLD	VAR	UNSIGNED8	0Ah	RW	
6510h	VM_KVAL_RUN	VAR	UNSIGNED8	0Bh	RW	
6510h	VM_KVAL_ACC	VAR	UNSIGNED8	0Ch	RW	
6510h	VM_KVAL_DEC	VAR	UNSIGNED8	0Dh	RW	
6510h	VM_INT_SPEED	VAR	UNSIGNED16	0Eh	RW	
6510h	VM_ST_SLP	VAR	UNSIGNED8	0Fh	RW	
6510h	VM_FN_SLP_ACC	VAR	UNSIGNED8	10h	RW	
6510h	VM_FN_SLP_DEC	VAR	UNSIGNED8	11h	RW	
6510h	VM_K_TERM	VAR	UNSIGNED8	12h	RW	
6510h	VM_STALL_TH	VAR	UNSIGNED16	13h	RW	
6510h	VM_CONFIG	VAR	UNSIGNED16	14h	RW	
6510h	CM_TVAL_HOLD	VAR	UNSIGNED16	15h	RW	
6510h	CM_TVAL_RUN	VAR	UNSIGNED16	16h	RW	
6510h	CM_TVAL_ACC	VAR	UNSIGNED16	17h	RW	
6510h	CM_TVAL_DEC	VAR	UNSIGNED16	18h	RW	
6510h	CM_T_FAST	VAR	UNSIGNED8	19h	RW	
6510h	CM_TON_MIN	VAR	UNSIGNED8	1Ah	RW	
6510h	CM_TOFF_MIN	VAR	UNSIGNED8	1Bh	RW	
6510h	CM_CONFIG	VAR	UNSIGNED16	1Ch	RW	

Appendix B. CANopen Builder user guide

In this appendix you will find a user guide for working with the software of the configurator CANopen Builder Limited v1.0 (hereinafter referred to as the configurator) of the engine controller R272-42-CAN.

System requirements

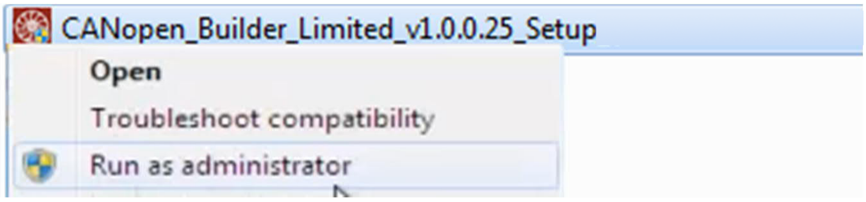
Hardware: The minimum for installing Windows 7; 65MB of free disk space.

Software environment: Windows 7 x86, Windows 7 x64, Windows 8 x86, Windows 8 x64, Windows 8.1 x86, Windows 8.1 x64.

Installing the software

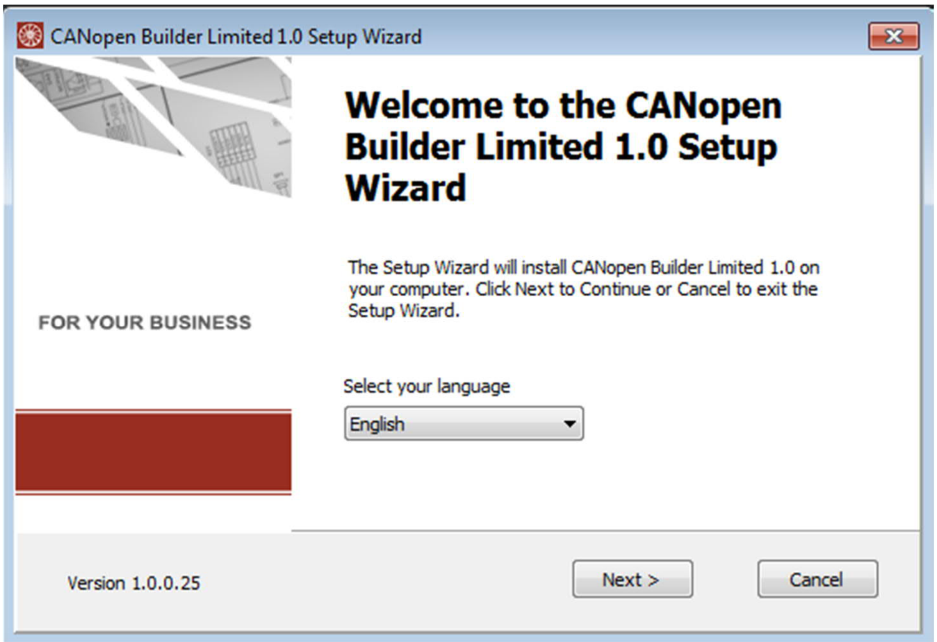
Connect the motor controller to the power source, according to the technical specifications and requirements, and then to the USB port of the computer.

1) Run the software installation application (executable file with the exe extension) on the computer that you want to install the configurator. The application requires administrator rights.



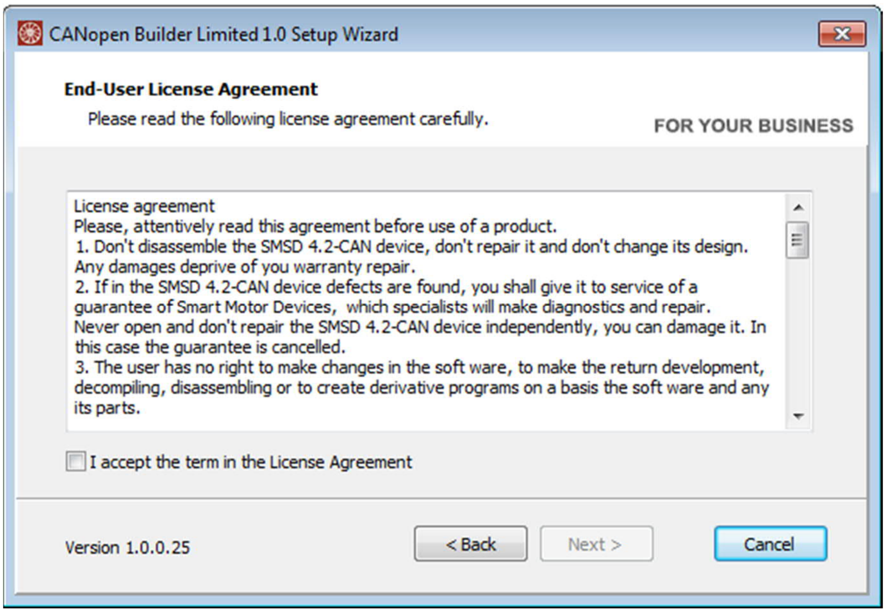
Picture B.1 — Starting application using administrator rights.

2) After launching the application, you will be prompted to select the language of the installation wizard (see Picture B.2).



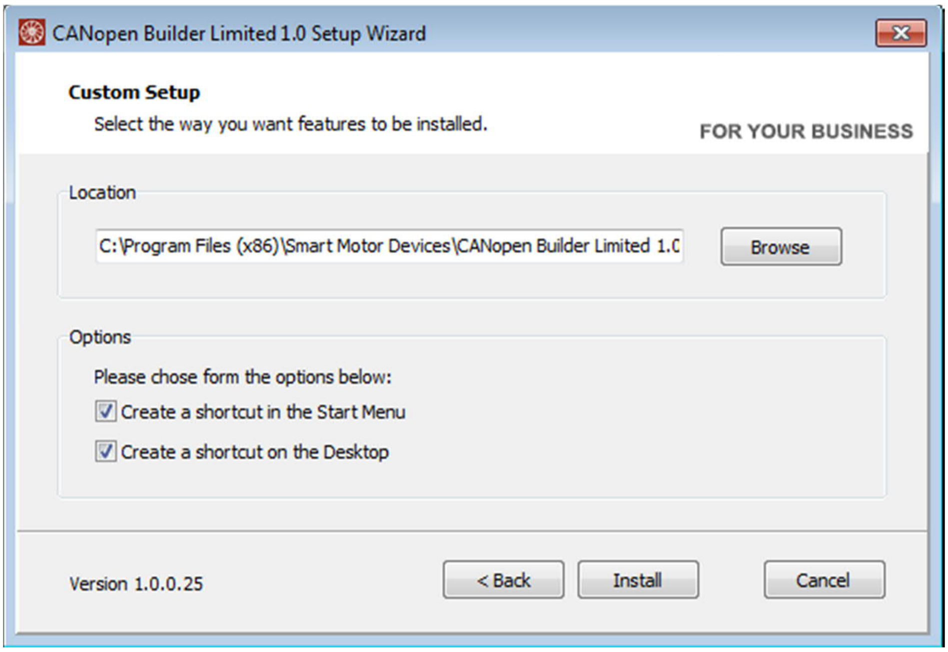
Pic B.2 — Language Select

3) On the next page, you are invited to read the license agreement, Picture B.3, after acceptance (in case of consent), the transition to the next stage will be available.



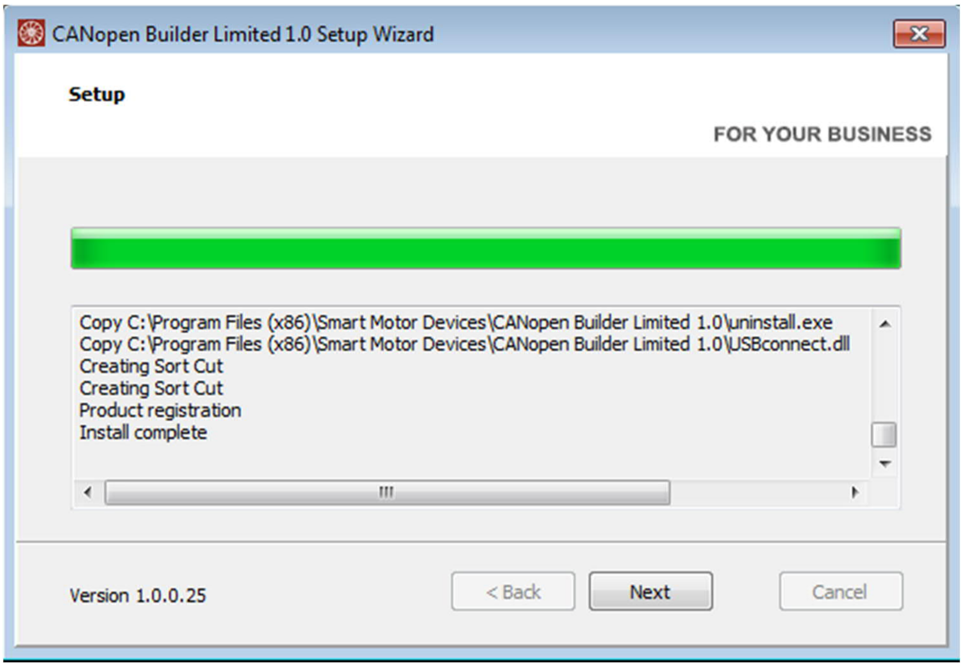
Picture B.3 — License Agreement.

4) Next, the program installation options are available: path and shortcuts (see Picture B.4).



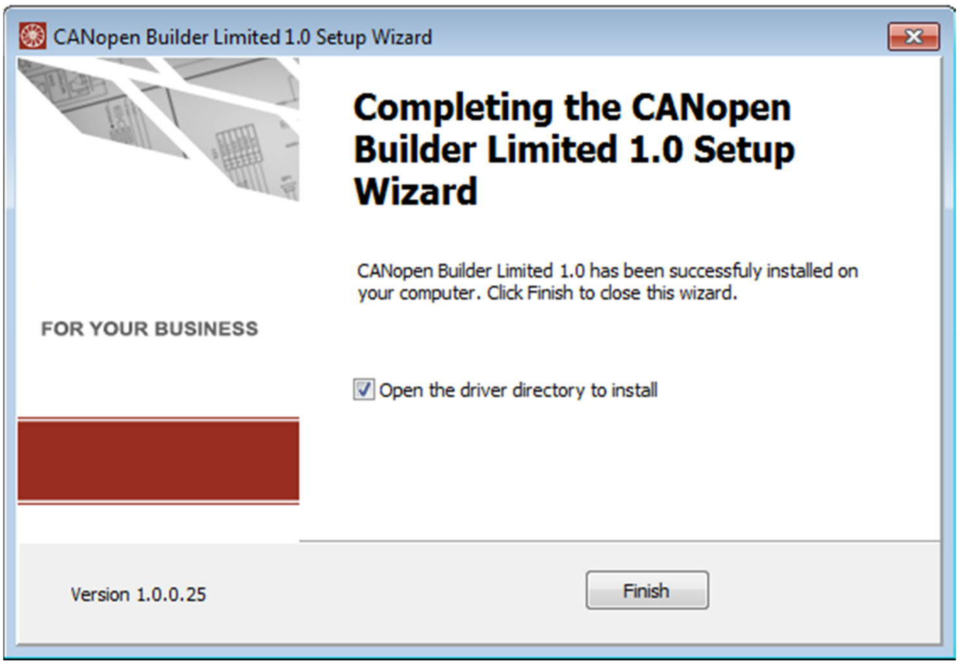
Pic B.4 — Installation options.

5) After clicking the "Install" button, the configurator will be installed according to the selected parameters. The installation progress and its log can be seen on the next page of the installation wizard (see Picture B.5).



Pic B.5 — Installation.

6) At the end of the installation, the "Next" button will be available. When you move on to the next step, you will be prompted to install the driver for the controller. Driver installation is required if it was not previously installed (see Picture B.6).



Pic B.6 — Installation Complete.

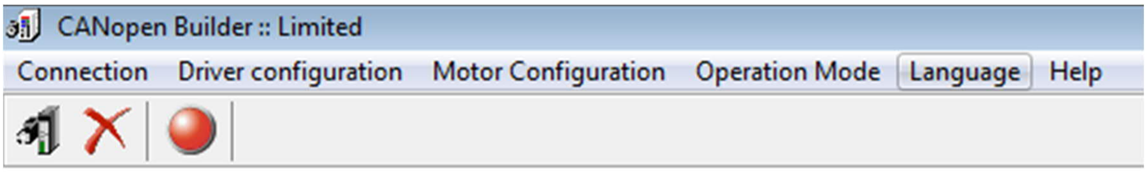
7) After the installation of the configurator and the selected driver installation point are completed, the driver directory will be opened in Windows Explorer. You must select a driver that meets the system requirements (for 32-bit systems, run the dpinst_x86.exe installation wizard from the appropriate operating system directory, dpinst_amd64.exe for 64-bit systems) and run the device driver installation wizard. If during the installation the "Windows Security" window appears with the message that it is impossible to verify the publisher, you must select the "Install this driver anyway" option, otherwise the configuration of the engine controller will not be possible. The successful completion of the

driver installation will be indicated by the wizard window with the message "Ready for operation".

Operating CANopen Builder

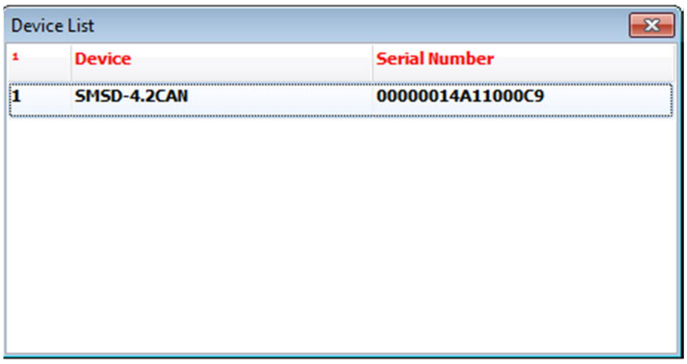
Administrator rights are required to operate the program.

At the first start-up, select the interface language of the configurator program. The default language is English. The choice is made through Menu-> Language, as shown in Picture B.7.



Pic B.7 — Language Select.

Select Menu-> Connection-> Device list (a shortcut button is also available ). A window will open with a list of compatible devices, as shown in Picture B.8.



Pic B.8 — a list of compatible devices.

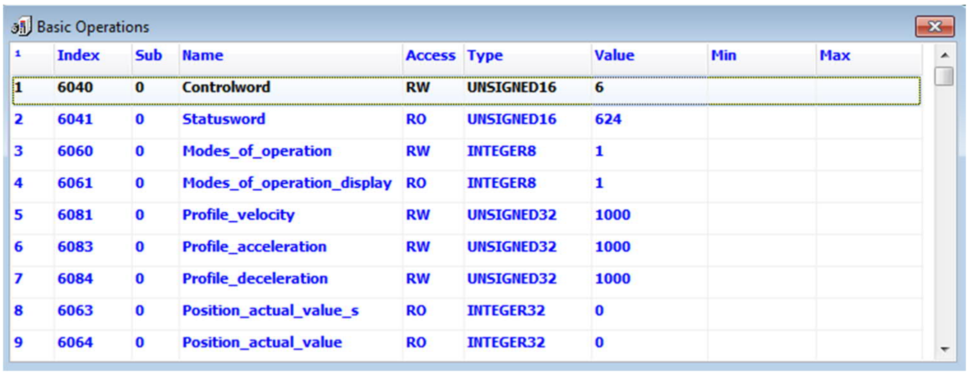
The connection is made by double left clicking the device in the list. If the operation is successful, the indicator changes color from red  to a shimmering blue-green one  . With a single left click on it, the connection details are shown.

The connected device is also displayed in the status bar (Picture B.9).

Pic. B.9 — status bar.

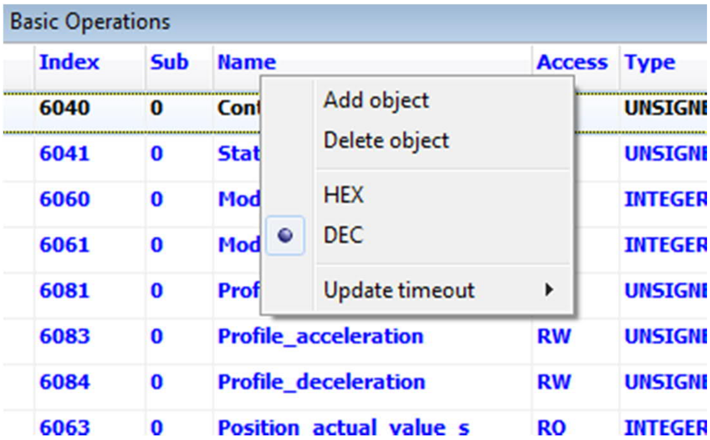
Disconnecting is done through a double left click on the device from the list of compatible devices, either through Menu-> Connection-> Disconnect, or through the shortcut button .

All objects of the object dictionary are structured according to the main menu items: driver configuration, engine configuration, operation mode. The subdivision windows are of the same type, differ only in the set of displayed objects, therefore the functional of only one window will be considered below. The "Basic Operations" window has the form shown in picture B.10.



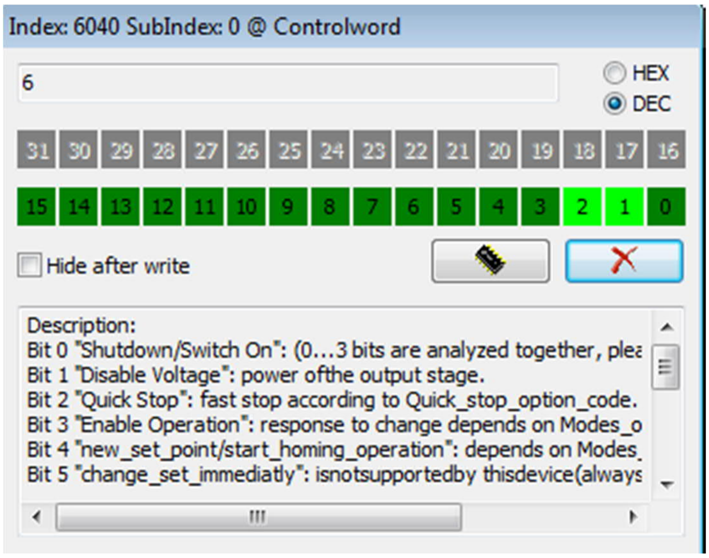
Pic. B.10 — Basic Operations.

For such windows, you can change the calculus system to display object values, add / remove an object from the list, and change the timeout for updating the values, by calling the context menu (one right click with the mouse) (see Picture B.11).



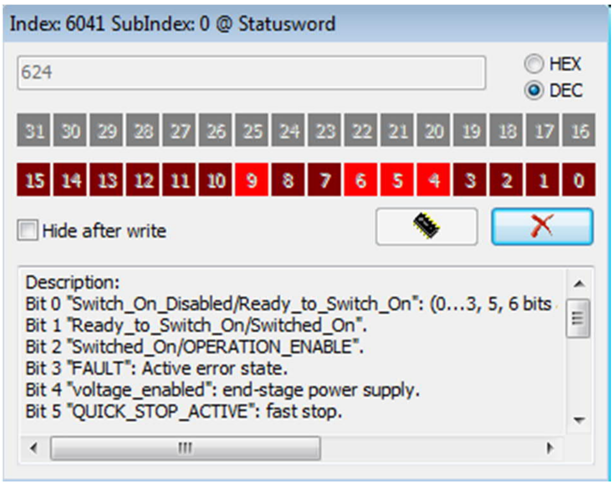
Pic. B.11 — Context menu.

Calling the window for editing the value of an object is done with a double left click of the mouse. The editing window is shown in Picture B.12.



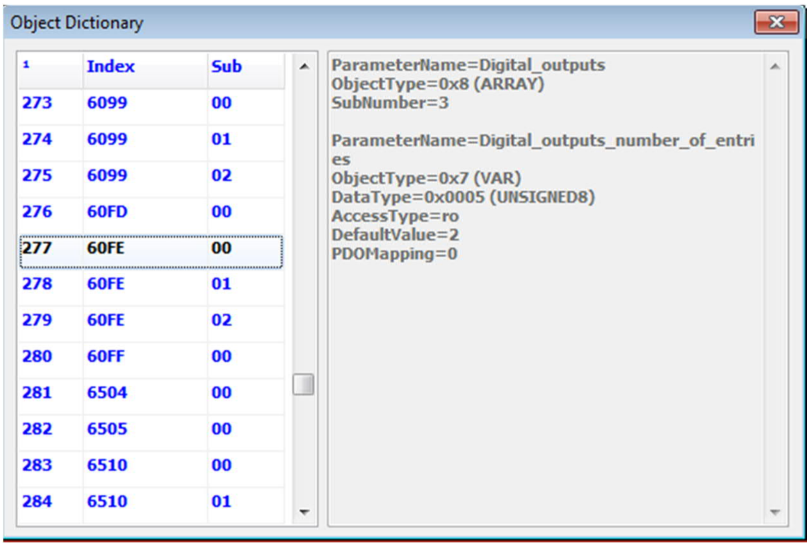
Pic. B.12 — window for editing the value of an object.

You can enter the value in decimal or hexadecimal notation, pre-marking the required one, you can also set / clear the required bits. If there are hints to the object, they are displayed at the bottom of the window. The window of the object that can not be edited is shown in picture B.13.



Pic B.13 — window for editing the value of an object.

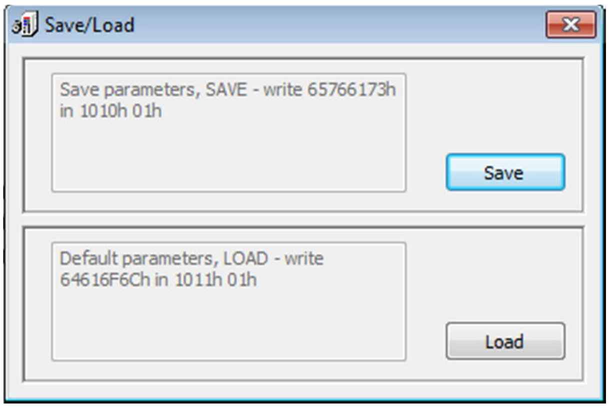
If you select the item "Add Object" in the context menu, the "Object Dictionary" window will open (Picture B.14), where you will be prompted to select a dictionary object to add to the display list (the object will be added to the end of the list; and are deleted during reconnection). Addition is made by double left mouse click on the desired object.



Pic B.14 — Object Dictionary.

i The device's EDS file is located in the directory of the installed program in the EDS directory. This file is required to configure the PLC.

For convenience of use, the window for saving parameters and restoring default values has an interface different from the above parameterization windows of the engine controller (Picture B.15). Menu-> Driver Configuration-> Initialize / Save.

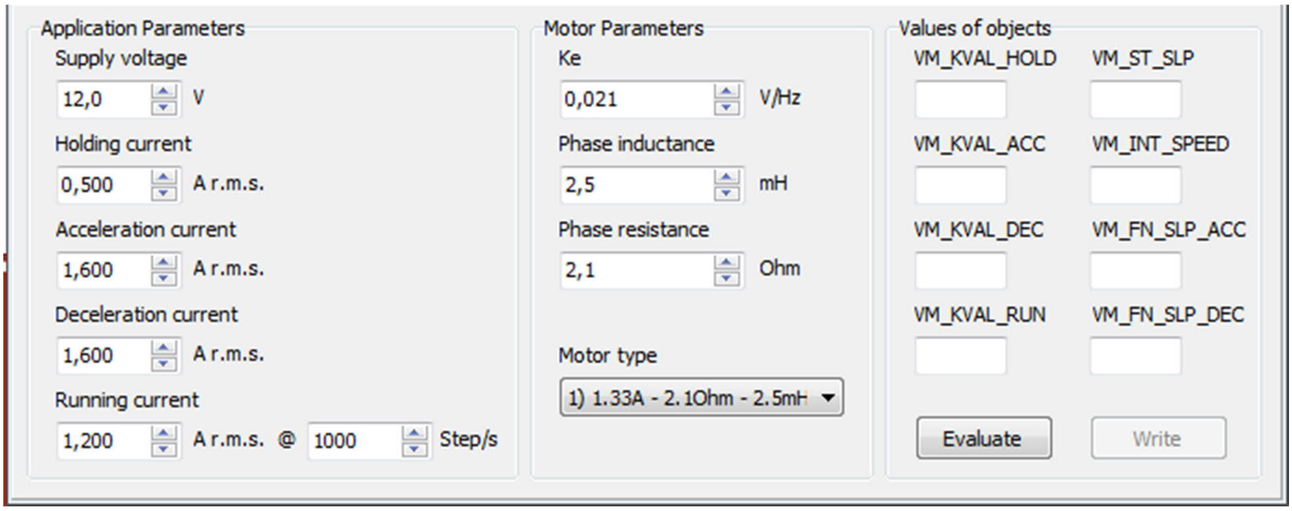


Pic. B.15 — Save and initialize window.

It should be noted that when using this method, both network parameters and driver parameters will be saved or restored.

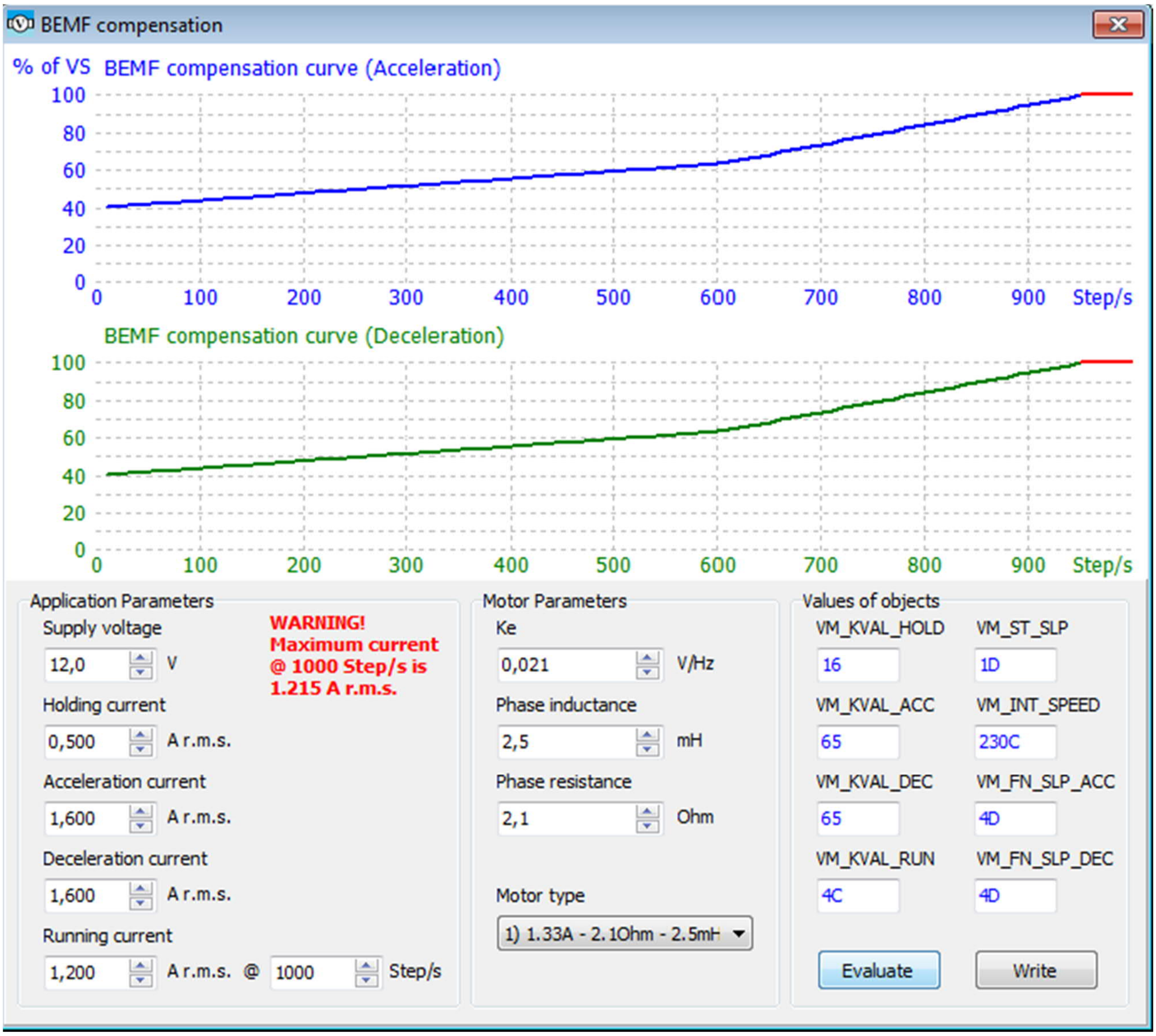
BEMF Compensation

As part of the configurator, there is a subroutine for calculating the values of the parameters of the motor controller for operation in the voltage control mode - BEMF Compensation. It is started via Menu-> Motor Configuration-> BEMF Compensation. After starting the program, it is required to enter drive parameters and motor parameters (you can use the engine presets, for this select the used engine from the drop-down list, as shown in Picture B.16).



Pic B.16 — BEMF Compensation.

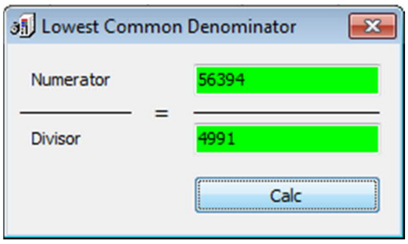
When you click on the button "Calculate", the calculation of the value of the objects will be performed and the necessary graphs illustrating the areas of torque loss will be constructed (if such exist, see picture. B.17).



Pic B.17 — BEMF Compensation.

The values obtained can be written to the motor controller by pressing the "Record" button.

For convenience in calculating Factor Group coefficients, the configurator contains a subroutine for finding the lowest common divisor by Euclid's algorithm (see Picture B.18). Runs via Menu-> Help-> Lowest common denominator.



Pic. B.18 — Search for lowest common divisor.

Appendix C. Calculation examples for Voltage and Current modes

In this appendix, calculations of the values of the engine controller dictionary objects for the voltage and current modes are briefly presented.

Voltage Control mode

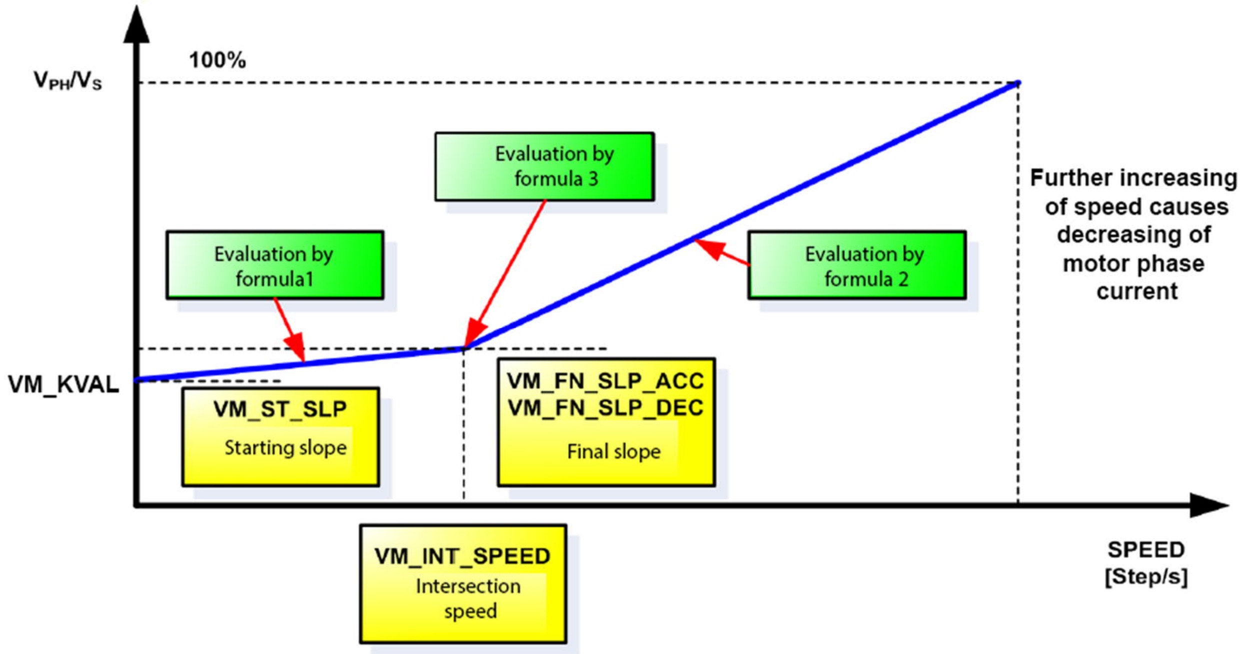
Below are the main calculation formulas used by the BEMF Compensation subroutine (see Figure C.1).

$$|V_{ph}| = \begin{cases} R \cdot |I_{ph}| + K_e \cdot f, (1) \\ 2\pi f \cdot L \cdot |I_{ph}| + K_e \cdot f (2) \end{cases}$$

$$f = \frac{R}{2\pi L} (3)$$

where R is the motor winding resistance [Ohm], Iphase is the amplitude value to be set in the winding [A], f is the rotation speed at the intersection of the graphs [4Step / s], Ke is the motor constant [V / Hz], L is the inductance windings [H].

The coefficient Ke is calculated experimentally or taken from a reference book and is the ratio of the peak amplitude to the frequency of the alternating voltage arising when the rotor of the stepper motor is unwound at a constant speed by another motor or manually.



Pic C.1 — BEMF compensation curve.

Further, the calculation of the values of objects for operation in the voltage control mode will be presented as an example.

Input:

$V_S = 50$ V
$I_{HOLD} = 2$ A
$I_{RUN} = 5$ A
$I_{ACC} = I_{DEC} = 5$ A
$K_e = 0{,}024$ V/Hz
$L = 3{,}2$ mH
$R = 1{,}65$ ohm

Calculate:

VM_KVAL_HOLD
VM_KVAL_RUN
VM_KVAL_ACC
VM_KVAL_DEC
VM_INT_SPEED
VM_ST_SLP
VM_FN_SLP_ACC
VM_FN_SLP_DEC

Solution:

$$\text{VM\_KVAL\_HOLD} = (\frac{R \cdot |I_{ph}|}{V_S}) \cdot 2^8 = (1{,}65 \cdot 2/50) \cdot 2^8 \approx 17 = 11\text{h};$$

$$\text{VM\_KVAL\_RUN} = \text{VM\_KVAL\_ACC} = \text{VM\_KVAL\_DEC} = (\frac{R \cdot |I_{ph}|}{V_S}) \cdot 2^8 = (1{,}65 \cdot 5 \cdot \sqrt{2}/50) \cdot 2^8 \approx\approx 60 = 3\text{Ch};$$

$$\text{VM\_INT\_SPEED} = (4 \cdot \frac{R}{2\pi L}) \cdot Tick \cdot 2^{26} = ((4 \cdot 1{,}65)/(2\pi \cdot 3{,}2 \cdot 10^{-3})) \cdot 250 \cdot 10^{-9} \cdot 2^{26} \approx 5507 = 1583\text{h};$$

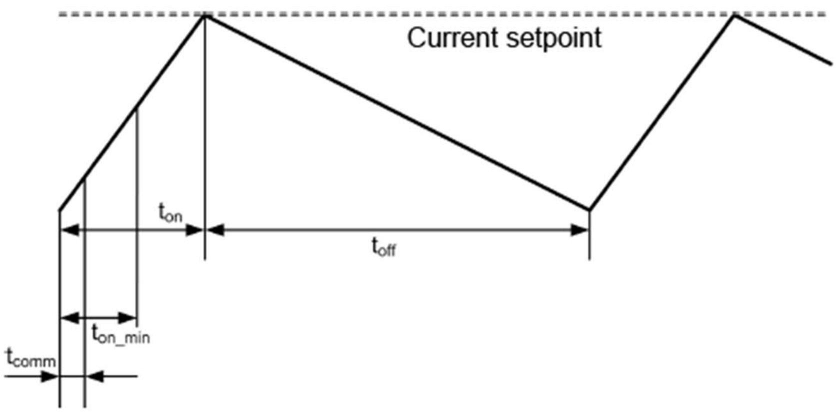
$$\text{VM\_ST\_SLP} = \frac{K_e/4}{V_S} \cdot 2^{16} = \frac{(0{,}024/4)}{50} \cdot 2^{16} \approx 8 = 8\text{h};$$

$$\text{VM\_FN\_SLP\_ACC} = \text{VM\_FN\_SLP\_DEC} = ((\frac{2\pi \cdot L \cdot |\, I_{PH} \,| + K_e}{4})/V_S) \cdot 2^{16} =$$

$$= ((\frac{2\pi \cdot 3{,}2 \cdot 10^{-3} \cdot 5 \cdot \sqrt{2} + 0{,}024}{4})/50) \cdot 2^{16} \approx 54 = 36\text{h}.$$

Current mode

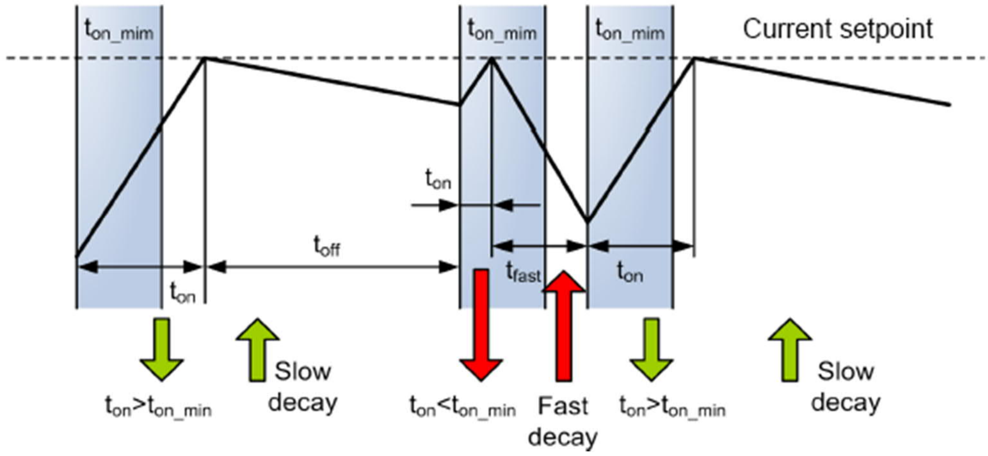
In this section, the objects CM_T_FAST (TOFF_FAST, FAST_STEP), CM_TON_MIN, CM_TOFF_MIN will be presented in detail. Picture C.2 shows the timing diagram of the output transistor.



Pic C.2 — Current setpoint monitoring.

For each control cycle, the on-time time ton is measured and compared with a predetermined threshold value ton_min. If the measured switching time is longer than the specified minimum threshold, then a fast decay mode is not required, and a slow decay is used (Picture C.2).

When the turn-on time is less than the set threshold, the system switches to the fast decay mode, at the first occurrence of such an event, the rapid drop time is 1/8 from the set maximum tOFF_FAST. If the time of the next measured inclusion is greater than the minimum set ton_min, the system returns to work with a slow decay mode (Picture C.3)



Pic B.3 — Selection of the current reduction mode and the duration of the on-time.

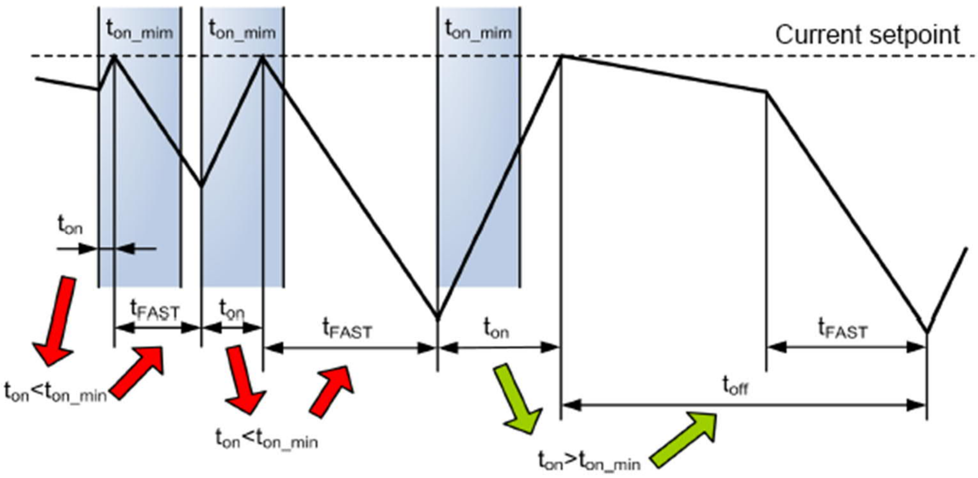
The tFAST value is adjusted according to the following rules:

- At the first occurrence of the situation, when ton <ton_min, the duration (tFAST) of fast decay is set to 1/8 from tOFF_FAST.

- In subsequent cases, the duration is doubled tFAST = 2tFAST.

- The maximum value of tFAST is limited to tOFF_FAST (see Picture C.4).

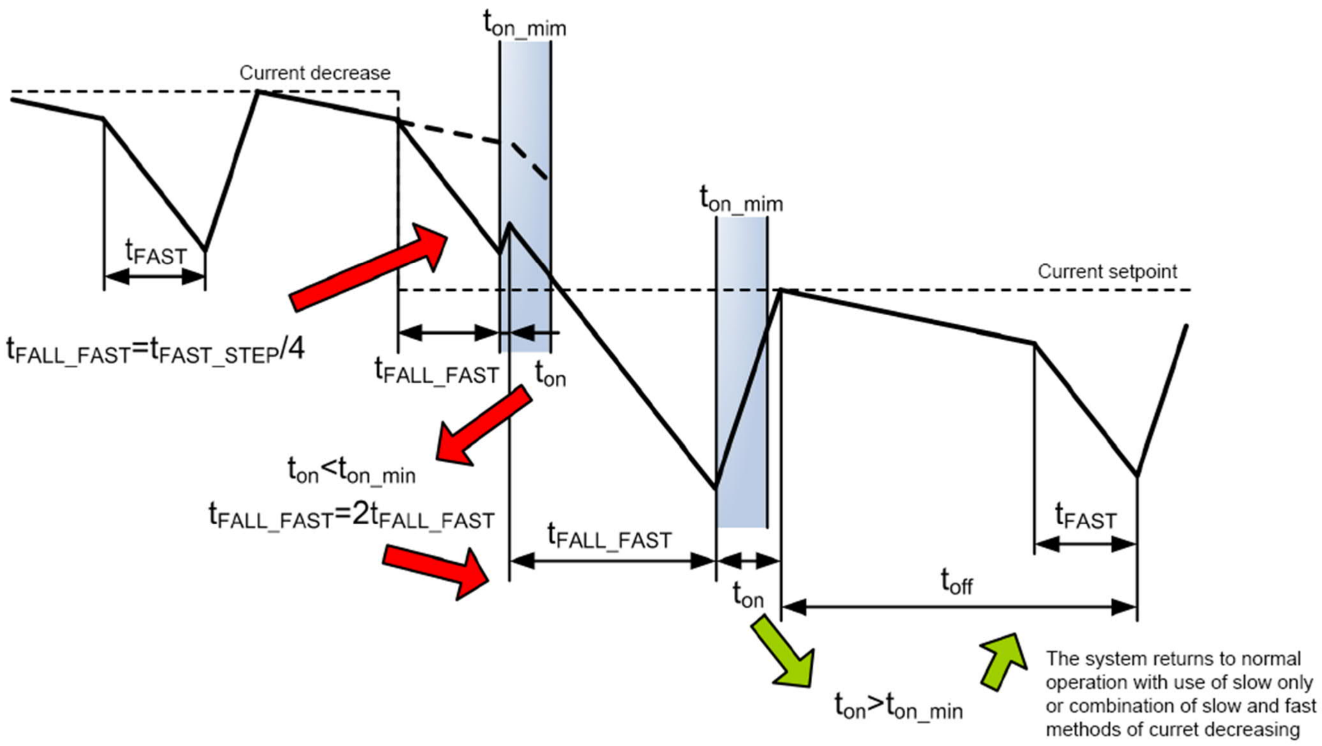
If the event Ton <Ton_min reappears, after the slow decay mode, the last value of tFAST is used



Pic B.4 — Regulation of the time of fast current decay (fast decay).

When working in microstep mode, when the set current decreases within one full step (the current decreases from the maximum set value to zero), the monitoring system immediately starts using the fast decay mode for faster decay. The algorithm used in this case is identical to the one described above.

At the first occurrence of the current reduction event (a microstep is made in the direction of decreasing current in the winding), the time of fast reduction of the current tFAST_FALL is equal to 1/4 from the maximum set value tFAST_STEP. If the next turn-on time tON turns out to be less than tON_MIN, then tFAST_FALL is doubled, and will be limited with further increase in tFAST_STEP. When the zero current mark crosses, the tFAST_FALL value takes a minimum value of 1/4 from tFAST_STEP. More clearly shown in Picture B.5.



Pic C.5 — Reduction of current in microstep mode.

When working in the predictive mode (the PRED_EN bit is set), the time of switching on the output top keys tpred is calculated by the formula:

$$t_{pred}(n) = \frac{t_{on}(n)+t_{on}(n-1)}{2},$$

where n is the PWM sequence number, ton is the on-time.

Thus, when using this method, the on-time is averaged between the current and previous PWM cycle, which ensures the smoothness of the current envelope. When the system comes to a state of stability, the time tpred becomes equivalent to ton.

CM_TON_MIN

Minimum switching time ton. It should be noted that this time is composed of the switching time and the on-time. In turn, the switching time tcomm consists of the time to turn off the lower key, the dead time and the time of turning on the upper key.

CM_TOFF_MIN

Off time toff. This value determines the maximum frequency of the PWM signal, which is calculated by the formula:

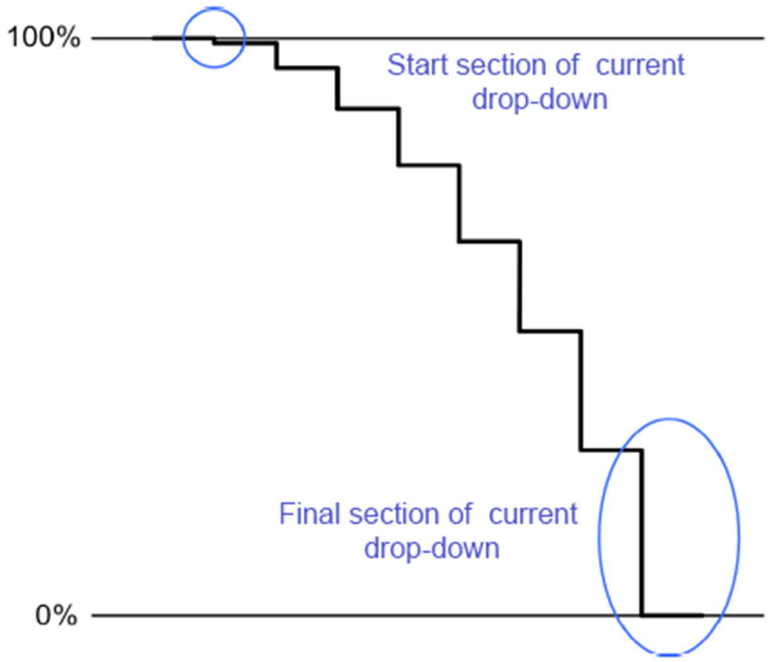
$$f_{SW\_max} = 1/(t_{on\_min} + t_{off}).$$

TOFF_FAST

Large current ripples increase noise, which in turn reduces the positioning accuracy within the microstep, so it is important to achieve minimum current ripple. The minimum value of fast decay, which is 1/8 of TOFF_FAST, must be greater than TON_MIN. Failure to comply with this requirement will result in a turn-on time ton less than the specified minimum, and as a result will lead to an increase in the tFAST time by half.

FAST_STEP

A large value of this parameter can lead to an excessively fast current drop at the initial stage of the current sine wave decay (see Figure B.6), but at the final stage it will be the most effective. The small value on the contrary will have a maximum of efficiency in the initial part of the recession and a minimum in the final. The best is a compromise between the initial and final sections of the current sinusoid and requires fine tuning. In either case, the minimum value of the parameter (1/4 of FAST_STEP) must be greater than TON_MIN.



Picture C.6 — Current drop on the example of microstepping mode 1/8

PRED_EN

With the prognostic mode active, the calculation of the parameters should be carried out, guided by the following rules:

- TOFF_MIN < TSW
- TOFF_MIN > TOFF_FAST
- TOFF_MIN + 2TON_MIN > $1/f_{SW\_max}$