Inverter

i700

E70ACM...

i700 servo inverter

Reference manual EN

13567282

Lenze

Contents

Contents

Contents

Contents

1 About this documentation

> **Danger!**
>
> The i700 servo inverters is a source of danger which may lead to death or severe injury of persons.
>
> In order to ensure protection against these dangers, observe the safety instructions before switching on the i700 servo inverters.
>
> Please read the safety instructions in the **mounting instructions** and **hardware manual** of the i700 servo inverters. Both instructions are included in the scope of supply of the i700 servo inverters.

Target group

This documentation addresses to all persons who want to parameterise, configure, and diagnose the i700 servo inverters.

Information regarding the validity

The information in this documentation are valid for the following standard devices:

Product series	Type designation	From software version
i700 servo inverters	E70ACM...	01.00

Screenshots/application examples

All screenshots in this documentation are application examples. Depending on the firmware version of the i700 servo inverters and the software version of the engineering tools installed (»PLC Designer« or »EASY Starter«), the screenshots in this documentation may deviate from the screen representation.

Document history

Version			Description
7.0	05/2019	TD06	Extensions & corrections; adaptation to software version V02.13 Extension: ▸ Wiring test by means of the "Cable Check" function (📖 71) (already available from software version V02.11 onwards)
6.0	11/2018	TD06	Extensions & corrections; adaptation to software version V02.12
5.0	06/2018	TD06	Extensions & corrections; adaptation to software version V02.11
4.0	03/2017	TD06	Extensions & corrections; Adaptations to software version V01.10 / V02.10
3.0	06/2016	TD06	Extensions & corrections: • Resetting the "Undervoltage (LU)" error message (📖 58)
2.0	11/2015	TD06	Extensions & corrections; Adaptation to software version V01.09
1.6	12/2014	TD06	Extensions & corrections; Adaptation to software version V01.06
1.5	03/2014	TD05	Extensions & corrections; Adaptation to software version V01.06
1.4	10/2013	TD05	Extensions & corrections; Adaptation to software versions V01.04 and V01.05
1.3	03/2013	TD05	Extensions & corrections; Adaptation to software version V01.03
1.2	10/2012	TD05	Extensions & corrections; Adaptation to software version V01.02
1.1	07/2012	TD05	First edition

Tip!

Information and tools regarding the Lenze products can be found on the Internet:

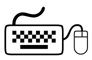
http://www.lenze.com → Download

1.1 Conventions used

This documentation uses the following conventions to distinguish between different types of information:

Type of information	Highlighting	Examples/notes
Spelling of numbers		
Decimal separator	Point	The decimal point is always used. Example: 1234.56
Hexadecimal number	0x	For hexadecimal numbers, the "0x" prefix is used. Example: 0x60F4
Binary number	0b	For binary numbers, the "0b" prefix is used. Example: 0b00010111
Text		
Version information	Blue text colour	Information that is only valid for or from a certain software version of the i700 servo inverters is indicated accordingly in this documentation. Example: This function extension is available from software version V3.0!
Program name	» «	The Lenze PC software »PLC Designer«...
Window	*italics*	The *Message* window ... / The *Options* dialog box...
Variable names		By setting *bEnable* to TRUE...
Control element	**Bold**	The **OK** button... / The **Copy** command... / The **Properties** tab... / The **Name** input field...
Sequence of menu commands		If several commands must be used in sequence to carry out a function, the individual commands are separated by an arrow: Select **File→Open** to...
Shortcut	**<bold>**	Use **<F1>** to open the online help.
		If a shortcut is required for a command to be executed, a "+" has been put between the key identifiers: With **<Shift>+<ESC>** ...

Program code	`Courier`	`IF var1 < var2 THEN`
		`a = a + 1`
		`END IF`

Keyword	**`Courier bold`**	
Hyperlink	<u>underlined</u>	Optically highlighted reference to another topic. It is activated with a mouse-click in this online documentation.
Icons		
Page reference	(🕮 16)	Optically highlighted reference to another page. It is activated with a mouse-click in this online documentation.
Step-by-step instructions		Step-by-step instructions are indicated by a pictograph.

1.2 Terminology used

Term	Meaning
Engineering tools	Software solutions for easy engineering in all project stages
	»EASY Navigator« – ensures easy operator guidance • All convenient Lenze Engineering tools at a glance • Tools can be quickly selected • The clear structure simplifies the engineering process from the start
	»EASY Starter« – easy-to-use tool for service technicians • Specifically designed for commissioning and maintaining Lenze devices • Graphic user interface with very few icons • Easy to run online diagnostics, set parameters and perform commissioning • No risk of accidentally changing an application • Loading off-the-shelf applications onto the device
	»PLC Designer« – for programming processes • Creating your own programs • Programming of Logic & Motion according to IEC 61131-3 (AWL, KOP, FUP, ST, AS and CFC Editor), based on CoDeSys V3 • Certified function blocks according to PLCopen Part 1 + 2 • Graphical DIN 66025 Editor (G code) with DXF import • Integrated visualisation for a simple process representation • All important information at a glance for commissioning
Lenze Controller	The Lenze Controller (briefly: "Controller") is the central component of the automation system which controls the Logic and Motion functionalities (by means of the runtime software). The Lenze Controller communicates with the field devices via the fieldbus.
Engineering PC	The Engineering PC and the installed Engineering tools serve to configure and parameterise the system. The Engineering PC uses Ethernet to communicate with the Lenze Controller.
EtherCAT	EtherCAT® is a real-time capable Ethernet system with top performance. EtherCAT® is a registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany.
HIPERFACE®	HIPERFACE® stands for High Performance Interface and is a universal interface between motor feedback system and i700 servo inverters. HIPERFACE® is a registered trademark of the SICK STEGMANN GmbH.
Object	"Container" for one or more parameters with which you can parameterise or monitor the i700 servo inverters.
Index	Each object has a unique index for addressing purposes. In this documentation the index is represented as a hexadecimal value and is identified by a prefixed "0x", e.g. "0x1000".
Subindex	If a code contains several parameters, they are stored in so-called "subindexes". In this documentation the colon is used as a separator between the index and the subindex, e.g. "0x1018:1".
Touch probe	A "touch probe" is an event that, for instance, can be triggered edge-controlled via a digital input in order to detect a (quickly changing) actual value at the time of triggering and process it in the program.

1.3 Definition of the notes used

The following signal words and symbols are used in this documentation to indicate dangers and important information:

Safety instructions

Structure of the safety instructions:

Danger!

(characterises the type and severity of danger)

Note

(describes the danger and gives information about how to prevent dangerous situations)

Pictograph	Signal word	Meaning
	Danger!	**Danger of personal injury through dangerous electrical voltage** Reference to an imminent danger that may result in death or serious personal injury if the corresponding measures are not taken.
	Danger!	**Danger of personal injury through a general source of danger** Reference to an imminent danger that may result in death or serious personal injury if the corresponding measures are not taken.
STOP	**Stop!**	**Danger of property damage** Reference to a possible danger that may result in property damage if the corresponding measures are not taken.

Application notes

Pictograph	Signal word	Meaning
	Note!	Important note to ensure trouble-free operation
	Tip!	Useful tip for easy handling
		Reference to another document

1.4 Structure of the parameter descriptions

All parameters which you can use to parameterise or monitor the i700 servo inverters are stored within "objects".

- For the purpose of addressing, each object is provided with a unique index. In this documentation the index is represented as a hexadecimal value and is identified by a prefixed "0x", e.g. "0x1000".
- If an object contains several parameters, they are stored in "subindexes". In this documentation the colon is used as a separator between the index and the subindex, e.g. "0x1018:1".

Note!

This documentation is valid for the i700 servo inverters in the single axis version (single inverter) and also as double axis (double inverter).

For parameters referring to one axis, both indexes (for axis A and axis B) are listed in the parameter description. For a single axis, only the first index is relevant in this case.

Each parameter description is structured according to the following pattern:

Example: Structure of the parameter descriptions in this documentation

❶ ❷ ❸
0x2942 | 0x3142 - Stromreglerparameter

❹

Sub.	Name	Lenze-Einstellung	Datentyp
▸ 1	Stromregler: Verstärkung	148.21 V/A	UNSIGNED_32
▸ 2	Stromregler: Nachstellzeit	3.77 ms	UNSIGNED_32

❺

Subindex 1: **Stromregler: Verstärkung**				
Einstellbereich (min. Wert | Einheit | max. Wert)			Lenze-Einstellung	
0.00	V/A	750.00	148.21 V/A	
☑ Schreibzugriff ☐ RSP ☐ OSC ☑ P ☐ RX ☐ TX			Skalierung: 1/100	UNSIGNED_32

❺

Subindex 2: **Stromregler: Nachstellzeit**				
Einstellbereich (min. Wert | Einheit | max. Wert)			Lenze-Einstellung	
0.01	ms	2000.00	3.77 ms	
☑ Schreibzugriff ☐ RSP ☐ OSC ☑ P ☐ RX ☐ TX			Skalierung: 1/100	UNSIGNED_32

❶	Object index for axis A
❷	Object index for axis B (only relevant for double axis)
❸	Parameter or object name
❹	If the object contains several parameters: Overview table with list of all subindexes
❺	Table with detailed information about the corresponding parameter: • Explanations & references (optional) • Display options/possible settings, Lenze setting, attributes (for the meaning see the following table)

Parameter attributes

Name	Meaning	
Write access	☑ = Parameter can be written to. ☐ = Parameter can only be read.	
CINH	☑ = Parameter can only be written to if controller inhibit is set.	
OSC	☑ = Parameter can be recorded by means of the oscilloscope function.	
P	☑ = Parameter can be persisted.	
Tx	☑ = Parameter can be mapped into the TPDO.	
Rx	☑ = Parameter can be mapped into the RPDO.	
Data type	Data type of the parameter:	
	INTEGER_8	1 byte, with sign
	INTEGER_16	2 bytes with sign
	INTEGER_32	4 bytes with sign
	UNSIGNED_8	1 byte without sign
	UNSIGNED_16	2 bytes without sign
	UNSIGNED_32	4 bytes without sign
	UNSIGNED_64	8 bytes without sign
	STRING(xx)	ASCII string (with character length xx)
	ARRAY [] OF...	ARRAY
Scaling	Scaling of the parameter	

Tip!

To find a specific object or parameter in this documentation, you can use the following navigation helps:

- At the beginning of each main chapter, all objects which are described in the respective chapter are listed in a table.
- In the Table of attributes and the Index, all objects/parameters are listed with a reference to the detailed description.

2 Parameter handling

2.1 Parameter transfer during initialisation

During the acceleration of the system, the Controller and i700 servo inverters exchange configuration data (e.g.bus cycle and PDO mapping). With regard to this, observe the following particular features for the i700 servo inverters:

Parameter download

The i700 servo inverters itself does not store parameter settings safe against mains failure. All settings deviating from the i700 servo inverters "Lenze setting" (default) are maintained centrally in the controller and are stored there permanently (persistently). During the initialisation at run-up, only these deviations are transferred to the i700 servo inverters by the controller. Like this it is ensured that the i700 servo inverters works with the parameter settings provided for it.

Firmware download (optional)

If required, the firmware of the i700 servo inverters can be stored together with the »PLC Designer« project. During the run-up, the controller then checks whether the firmware version in the i700 servo inverters complies with the firmware version stored in the project for this device. If this is not the case, the controller loads the firmware version stored in the project to the i700 servo inverters. Like this, it can be ensured for "Device replacement" service work that the replacement device also works with the same firmware version as the original device.

2.2 Storage parameter set (par001.*) and total parameter set (par000.*)

For the storage of the i700 servo inverters parameters, two different parameter sets are provided, which are stored in different parameter set files in the higher-level controller:

Storage parameter set (par001.i7psf)

- Only contains the parameters of the i700 servo inverters which are writable and identified with the "P" attribute (persistent).
- Parameters can be read out from the i700 servo inverters and saved to the file. Conversely, the parameters stored can be written to the i700 servo inverters again.
- Recommended for storage and archiving the i700 servo inverters settings.

Total parameter set (par000.i7psf)

- Contains all parameters of the i700 servo inverters, including the mere display parameters which vary permanently during operation.
- The total parameter set can only be read from the i700 servo inverters.
- Use for purposes of service and diagnostics.

The structure of the parameter set file is described in the appendix.

▶ Structure of the parameter set file (📖 379)

2.2.1 Saving a parameter set from the i700 to a file (export)

Reading out and storing the parameters from the i700 servo inverters is initiated and controlled from the higher-level controller. For reading out a parameter set, the corresponding parameter values are summarised in a parameter set file (par001.* or par000.*) in the i700 servo inverters and are then transferred as a file to the controller via EtherCAT.

2.2.2 Loading the stored parameter set to the i700 (import)

Loading a stored parameter set file to the i700 servo inverters is initiated and controlled from the higher-level controller, just like in the case of the read-out. The storage parameter set file is transferred to the i700 servo inverters as a file via EtherCAT, and the parameter settings are loaded (imported).

For logical reasons, it is only possible to transfer "storage parameter set files" (par001.*) to the i700 servo inverters. Display parameters are updated under normal operating conditions and do not need to be loaded.

Note!

In the case of Lenze controllers, the parameter set files are transferred as "files" via GCI ("Generic Communication Interface").

- GCI is a Lenze-specific application protocol which uses CoE ("CANopen over EtherCAT").
- Other control manufacturers may also use FoE ("File Access over EtherCAT") for the transfer.

2.2.3 Monitoring of the parameter import (error report)

When the parameter import has been completed, the "parErr.i7psf" error report file is created. This file is recreated automatically during every import.

On the basis of the error report it can be determined whether errors have occurred during the parameter import:

- Parameters that have been transferred with errors are listed there together with their index, subindex, and error code (SDO abort code).
- If no errors have occurred, this data area of the file is blank. However, the headers including the checksum are available and valid nevertheless.

2.3 Cyclic redundancy check (CRC) - parameter set comparison on the basis of the checksum

Each parameter set features an individual checksum (CRC32), which is composed of the settings of all storable indexes (P-flag) of the parameter set. When a setting changes, there is also a change in the checksum.

Cyclic redundancy check (CRC):

By comparison of the checksums of parameter sets it can be determined very quickly whether parameter sets are identical or not. In particular, it is thus determined for the i700 servo inverters whether the parameter set stored in the controller is identical to that in the i700 servo inverters and therefore does not have to be loaded again.

0x2030 - Parameter set: Validity check (CRC)

Checksum for quick comparison of the storage parameter set (par**001**.i7psf)
- If the checksum shown here is identical to the checksum of the parameter set file stored in the L-force Controller, it is not necessary to write the parameter set file to the i700 servo inverter.
- If the checksums are different, however, there are deviations between the parameter set available in the L-force Controller and that in the i700 servo inverter.
- The checksum cannot be used for comparison of the total parameter set (par**000**.i7psf).

Display range (min. value | unit | max. value)			Initialisation	
0		4294967295		
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX				UNSIGNED_32

3 Communication with the controller

"EtherCAT control technology" communication manual

Here you will find detailed information on the EtherCAT configuration and commissioning of Lenze devices in the EtherCAT network.

Objects described in this chapter

Object		Name	Data type
Axis A	Axis B		
0x2020		EoE information	RECORD
0x2824	0x3024	Device control via PDO: Activation	UNSIGNED_8
0x2830	0x3030	Lenze control word	UNSIGNED_16
0x2831	0x3031	Lenze status word	UNSIGNED_16
0x2833	0x3033	Lenze statusword 2	UNSIGNED_16

Tip!

A description of the communication objects relevant to the integration of the servo inverter with the use of a third-party control can be found in the annex.

▶ Communication objects (📖 382)

3.1 Acceleration of the system (initialisation)

Stop!

Before switching on the i700 servo inverters for the first time, check the entire wiring with regard to completeness, short circuit, and earth fault.

To establish communication, the i700 servo inverters must be supplied with voltage. During the acceleration of the system, the controller and the drive exchange configuration data.

The controller transmits the following configuration data to the i700 servo inverter:

- Bus cycle
  - This is the basic cycle within which the EtherCAT bus is actuated. The bus cycle is given as a multiple of 125 µs.
  - The bus cycle equals the communication cycle within which the process data are exchanged cyclically. New process data are only accepted and generated in the Servo-Inverter i700 maximally every 250 µs.
- Parameter set determined during commissioning.
  - Among other things, it includes information about feedforward control values, the mains voltage and the switching frequency as well as controller parameters adapted to the motor module which are used for the motor control.
- Configuration of the process data transmitted cyclically via EtherCAT (PDO mapping)

Note!

If no data for the initialisation of the controller are transmitted, the i700 servo inverters uses the "Lenze setting" for the parameters.

Power up/Power down

In the case of "Power up" and "Power down", no undefined states can occur that cause damage to the device or motor movements which are not requested or not braked.

In the case of a voltage failure/dip, there is no immediate response by the i700 servo inverters. The functionality is to be maintained for as long as possible.

▶ 24-V supply voltage monitoring (📖 290)

Persistent data storage

A complete, persistent storage of the drive configuration which remains the same even though the servo inverter is switched off from time to time is carried out in the higher-level controller instead of in the i700 servo inverters. Hence, the controller needs to transmit the configuration data again to the i700 servo inverters after a power up.

However, in the case of a power down, the following device data are stored persistently in the i700 servo inverters:

- Power-on and elapsed-hour meter (0x2D81 or 0x3581 for axis B)
- History buffer (📖 323)

3.2 Process data (cyclic PDO transfer) and PDO mapping

"EtherCAT control technology" communication manual

Here you will find some detailed information on the configuration of the process data objects (PDO mapping) with the »PLC Designer«.

Cyclic process data are transferred cyclically between the controller (master) and the servo inverters (slaves) as so-called *Process Data Objects* (PDOs).

- The i700 servo inverters supports the following bus cycle times of the EtherCAT:
  0.250 ms
  0.500 ms
  1.000 ms
  ... (only integer multiple of 1 ms)
  10.000 ms (max. cycle time)
- The processing time of a process date through the drive is t=max[bus cycle, 0.250 ms]. If the entire chain from the control via the drive back to the control is considered, a process data needs 2 bus cycles in addition (for Lenze C3200) . Thus, the turnaround time for a PDO can be indicated with 3 bus cycles.
- For the process data communication, the i700 servo inverters supports the mapping of max. 32 process data objects (PDOs) with a total max. size of 100 bytes per direction of transmission.
- A fixed PDO mapping preconfigured by Lenze is available for every CiA402 operating mode supported by the i700 servo inverters. Every PDO mapping includes of several objects from the Object directory.
- For every axis, further fixed, preconfigured PDO mappings can be used for touch probe functionality.
- In addition to the fixed, preconfigured PDO mappings, freely configurable PDO mappings are available for every axis which can be used for individual PDO mapping. A maximum of 8 objects from the Object directory can be configured per direction of transmission.
- The configuration of the PDOs actually transmitted between the controller and the i700 servo inverters is carried out via the »PLC Designer«.
  - Experience shows that most i700 servo inverterss are operated in one of the available CiA402 operating modes (csp, csv, cst or vl). We therefore recommend to use and activate the fixed PDO mapping preconfigured by Lenze for the selected operating mode. In doing so all parameters that usually need to be replaced during a cyclic PDO transfer can be accessed.
  - If you wish to use touch probe functionality in addition, optionally and additionally activate the fixed, preconfigured PDO mapping for touch probe functionality.
  - Moreover, the PDO mappings that can be freely configured by the user can be optionally activated. However, we recommend to use these freely configurable PDO mappings in special cases only, if no standard PDO mapping is suitable for the case of application at hand. Special thought should be given to the aspect of traceability in case of service.

Object		**Info**
Axis A	Axis B	
RPDO mapping – configuration of the process data (setpoints) from the controller to the i700 servo inverters		
0x1600	0x1610	Fixed, preconfigured PDO mapping object for "Cyclic sync position mode (csp)"
0x1601	0x1611	Fixed, preconfigured PDO mapping object for "Cyclic sync torque mode (cst)"
0x1602	0x1612	Fixed, preconfigured PDO mapping object for "Cyclic sync velocity mode (csv)"
0x1603	0x1613	Fixed, preconfigured PDO mapping object for "Velocity mode (vl)"

Object Axis A	 Axis B	Info
0x1604	0x1614	Fixed, preconfigured PDO mapping object for "Touch probe (TP)"
0x1605	0x1615	Freely configurable PDO mapping object
0x1606	0x1616	Fixed, preconfigured PDO mapping object for torque limit values
TPDO mapping – configuration of process data (actual values) from the i700 servo inverters to the controller		
0x1A00	0x1A10	Fixed, preconfigured PDO mapping object for "Cyclic sync position mode (csp)"
0x1A01	0x1A11	Fixed, preconfigured PDO mapping object for "Cyclic sync torque mode (cst)"
0x1A02	0x1A12	Fixed, preconfigured PDO mapping object for "Cyclic sync velocity mode (csv)"
0x1A03	0x1A13	Fixed, preconfigured PDO mapping object for "Velocity mode (vl)"
0x1A04	0x1A14	Fixed, preconfigured PDO mapping object for "Touch probe (TP)"
0x1A05	0x1A15	Freely configurable PDO mapping object

3.2.1 Synchronisation with "Distributed clocks" (DC)

The "Distributed clocks" (DC) function enables an exact time adjustment for applications where several auxiliary axes carry out a coordinated movement at the same time. The data is accepted synchronously with the PLC program. In the case of the DC synchronisation, all slaves are synchronised with a reference clock, called the "DC master".

"EtherCAT control technology" communication manual

Here you will find some detailed information about DC synchronisation.

3.3 Parameter data transfer (SDO communication)

In addition to the cyclic process data transfer, parameter data can be transferred as so-called SDOs (Service Data Objects) in a non-cyclic manner within an individual datagram between the controller (master) and the servo inverters (slaves).

- SDO communication is implemented according to the EtherCAT-CoE protocol, using a mailbox.
- SDO communication enables read or write access to all indices contained in the object directory of the i700 servo inverters.
- The turnaround time for an SDO (request by the control, transport via the bus, processing in the drive, transport of the response back to the control) is between 1 ms and 100 ms plus 2 * bus cycle time (typically 10 ms).

3.3.1 Object directory

The object directory contains the specific indices for all axes. The object directory is structured according to specifications of the EtherCAT Technology Group (ETG):

Range	**Index area**		
	Device	Axis A	Axis B
Communication Area	**0x1000 - 0x1FFF**		
Identification data	0x1000 - 0x1018	-	-
Sync manager	0x1C00 - 0x1C33	-	-
PDO mapping	-	0x1600 - 0x1606 0x1A00 - 0x1A05	0x1610 - 0x1616 0x1A10 - 0x1A15
Manufacturer specific area	**0x2000 - 0x5FFF**		
Device settings	0x2000 - 0x27FF	-	-
Axis identification	-	0x2800 - 0x281F	0x3000 - 0x301F
Axis control	-	0x2820 - 0x283F	0x3020 - 0x303F
Error management	-	0x2840 - 0x28FF	0x3040 - 0x30FF
Motor control & motor settings	-	0x2900 - 0x2CFF	0x3100 - 0x34FF
Touch probe	-	0x2D00 - 0x2D3F	0x3500 - 0x353F
Monitoring functions	-	0x2D40 - 0x2D7F	0x3540 - 0x357F
Diagnostics	-	0x2D80 - 0x2DBF	0x3580 - 0x35BF
Service/internal	-	0x2DC0 - 0x2E3F	0x35C0 - 0x363F
Reserved	-	0x2E40 - 0x2FFF	0x3640 - 0x37FF
CiA402 profile specific area	**0x6000 - 0xDFFF**		
Device profile CiA402	-	0x6000 - 0x67FF	0x6800 - 0x6FFF

3.3.2 SDO abort codes (Abort codes)

If an SDO request is evaluated negatively, a corresponding abort code is output:

SDO abort code	Description
0x0000 0000	No error
0x0503 0000	The status of the toggle bit has not changed.
0x0504 0000	SDO protocol time-out
0x0504 0005	The space in the main memory is not sufficient.
0x0601 0000	Access to object not supported.
0x0601 0001	Read access to a write-protected object.
0x0601 0002	Write access to a write-protected object.
0x0602 0000	An object does not exist in the object directory
0x0604 0041	An object cannot be mapped into the PDO
0x0604 0042	The number and/or length of the objects mapped would exceed the PDO length
0x0604 0043	General parameter incompatibility
0x0604 0047	General internal device incompatibility
0x0606 0000	Access has failed due to a fault in the hardware
0x0607 0010	The data type or the parameter length does not correspond
0x0607 0012	Incorrect data type (The parameter length is too large)
0x0607 0013	Incorrect data type (The parameter length is too small)
0x0609 0011	A subindex is not available
0x0609 0030	The value range for parameters is too great (only for write access)
0x0609 0031	The parameter value is too high
0x0609 0032	The parameter value is too low
0x0800 0000	General error
0x0800 0020	Data cannot be transferred or saved to the application.
0x0800 0021	Data cannot be transferred or saved to the application because of local control.
0x0800 0022	Due to the current device state, data cannot be transferred to the application or stored in the application

3.3.3 ESI: EtherCAT Slave Information file (device description)

The EtherCAT Slave Information file (EtherCAT Device Description file) contains all information about the device (operating modes, parameters, ...).

- The EtherCAT Slave Information file is integrated by the EtherCAT network configuration tool in order to be able to configure and commission the devices.
- Part of the information contained in the EtherCAT Slave Information file can be uploaded online by the EtherCAT master by accessing the EtherCAT EEPROM of the device. The description of the object directory can also be identified online.

3.4 Activating the control via PDO

0x2824 | 0x3024 - Device control via PDO: Activation

This object serves to switch off all RPDOs (from the device's point of view) so that the device is exclusively controlled via SDOs.

- This is, for instance, required for manual enable of commissioning functions and test modes via the (0x6040 control word 0x6840 for axis B). ▶ Enable/inhibit via control word

Selection list(Lenze setting printed in bold)			
0	Off		
1	**Activate**		
☑ Write access ☑ CINH ☐ OSC ☑ P ☐ RX ☐ TX			UNSIGNED_8

3.5 Lenze control and status word

0x2830 | 0x3030 - Lenze control word

Via the Lenze control word, the control functions can be influenced.

Setting range (min. value | unit | max. value)			Lenze setting
0x0000		0xFFFF	0x0000
Value is bit-coded: (☑ = bit set)			Info
Bit 0 ☐	Flying restart: Completed		Via this bit, the control reports the acceptance of the speed found to the "Flying restart" function. The flying restart process is now completed.
Bit 1 ☐	Flying restart: Blocked		"1" ≡ Block flying restart process
Bit 2 ☐	Reserved		
Bit 3 ☐	Reserved		
Bit 4 ☐	Speed controller: Load I component		"1" ≡ Set starting value of the torque • In case of servo control, this corresponds to the I component of the speed controller, in case of V/f operation to the modulation of the slip compensation. • As long as this bit is set to "1", the I component and the slip compensation are set to the starting value set in 0x2902 (or 0x3102 for axis B).
Bit 5 ☐	Position: Select new actual position		"1" ≡ Set/relatively shift actual position • Axis A: Set the actual position (0x6064) under consideration of the set resolution (0x608F) to the value set in 0x2983 (0x2984 = 0), or shift it by the value set in 0x2983 (0x2984 = 1). • Axis B: Set the actual position (0x6864) under consideration of the set resolution (0x688F) to the value set in 0x3183 (0x3184 = 0), or shift it by the value set in 0x3183 (0x3184 = 1).
Bit 6 ☐	Activate DC-injection braking or short-circuit braking		"1" ≡ Trigger DC-injection braking for asynchronous motor or short-circuit braking for synchronous motor
Bit 7 ☐	Velocity and position monitoring - deactivated		
Bit 8 ☐	Reserved		
Bit 9 ☐	Reserved		
Bit 10 ☐	Activate SLS limitation		
Bit 11-15 ☐	Reserved		
☑ Write access ☐ CINH ☑ OSC ☐ P ☑ RX ☐ TX			UNSIGNED_16

0x2831 | 0x3031 - Lenze status word

In the Lenze status word, messages are combined that go beyond the CiA specification.

Display range (min. value | unit | max. value)			Initialisation
0x0000		0xFFFF	
Value is bit-coded:			Info
Bit 0	Position controller: In limitation		Position mode: Output of the position controller in limitation
Bit 1	Speed: Limited speed setpoint 1		Input of speed controller 1 in limitation
Bit 2	Speed controller: In limitation		Output of speed controller 1 in limitation
Bit 3	Torque: Limited target torque		Target torque in limitation
Bit 4	Motor: Limited current setpoint		Setpoint current in limitation
Bit 5	Speed: Limited speed setpoint 2		Torque mode: Input of speed controller 2 in limitation
Bit 6	Upper speed limit is active		Torque mode: Speed is limited to the upper speed limit (0x2946:1 or 0x3146:1 for axis B)
Bit 7	Lower speed limit is active		Torque mode: Speed is limited to the lower speed limit (0x2946:2 or 0x3146:2 for axis B)
Bit 8	Flying restart in progress ...		V/f operation: "Flying restart process" function is active
Bit 9	Flying restart: Ready for operation		V/f operation: "Flying restart process" function has acquired speed
Bit 10	Limited output frequency		Setpoint frequency in limitation
Bit 11	Asynchronous motor magnetised		In case of servo control for synchronous motor (SM): • Bit is always set. In case of servo control for asynchronous motor (ASM): • Bit is set if the difference between setpoint and actual flux is smaller than 10 % of setpoint flux. • Bit is reset if the difference has increased to 15 % of setpoint flux. In case of V/f operation: • Bit is set if the rotor time constant has been passed seven times, calculated from the time the controller has been enabled and no restart on the fly has been active and the total motor current has reached 20 % of the rated motor current for the first time. Otherwise 0.
Bit 12	Motor phase failure detection in progress ...		Motor phase failure detection is active
Bit 13	Feedback: Open circuit		Position feedback is interrupted
Bit 14	Delay time: 'Reset error' is active		The error cannot be reset before the delay time has expired. The remaining delay time is displayed in 0x2840 (or 0x3040 for axis B).
Bit 15	Clamp is active		V/f operation: Clamp (short-time inhibit of the inverter) is active
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			UNSIGNED_16

0x2833 | 0x3033 - Lenze statusword 2

From software version V01.03.xx onwards

Display range (min. value | unit | max. value)			Initialisation
0x0000		0xFFFF	
Value is bit-coded:			Info
Bit 0	Feedback modification		Changes in the settings of the feedback system have been executed. The status bit is reset at controller enable.
Bit 1	Manual test mode active		A manual test mode is active (0x2825 or 0x3025 = 1, 2 or 3)
Bit 2	Manual control mode active		Manual control mode is active (0x2825 or 0x3025 = 4)
Bit 3	Angle tracking control: Active		Reserved
Bit 4	Absolute value encoder selected		From software version V01.06.xx onwards Absolute feedback system selected (parameterised) • In case of the "resolver" device variant, this bit is set if the resolver pole pair number "1" is set in 0x2C43 (or 0x3443 for axis B). • In case of the "encoder" device variant, this bit is set if the encoder type "2: Hiperface absolute value encoder" is set in 0x2C40 (or 0x3440 for axis B).
Bit 5	Absolut position available		From software version V01.06.xx onwards Initialisation of the selected feedback system is completed. All information of the feedback system have been transmitted. • In case of an error in the feedback system, this bit is set to "0" as long as the error exists.
Bit 6	DC-injection braking or short-circuit braking: Active		From software version V01.06.xx onwards DC-injection braking for asynchronous motor or short-circuit braking for synchronous motor is active
Bit 7	Value for angular drift is not valid		From software version V01.06.xx onwards Value of the angular drift in 0x2C42 is not valid due to communication faults.
Bit 8-12	Reserved		
Bit 13	Cable check running		
Bit 14-15	Reserved		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			UNSIGNED_16

3.6 Ethernet over EtherCAT (EoE)

From software version V01.05.xx onwards, the i700 servo inverters supports the "Ethernet over EtherCAT (EoE)" protocol.

The "Ethernet over EtherCAT (EoE)" protocol serves to send standard Ethernet telegrams via the EtherCAT network without the real-time communication of the EtherCAT process data being affected.

This extension allows for establishing a parameter communication (SDO communication) with the Servo-Inverters i700 on the EtherCAT bus via a standard Ethernet connection (e.g.from a PC with »EASY Starter«).

Note!

For establishing an EoE communication, a MAC address must be transferred in the EoE initialisation telegram since the Servo Inverter i700 does not have an own MAC address.

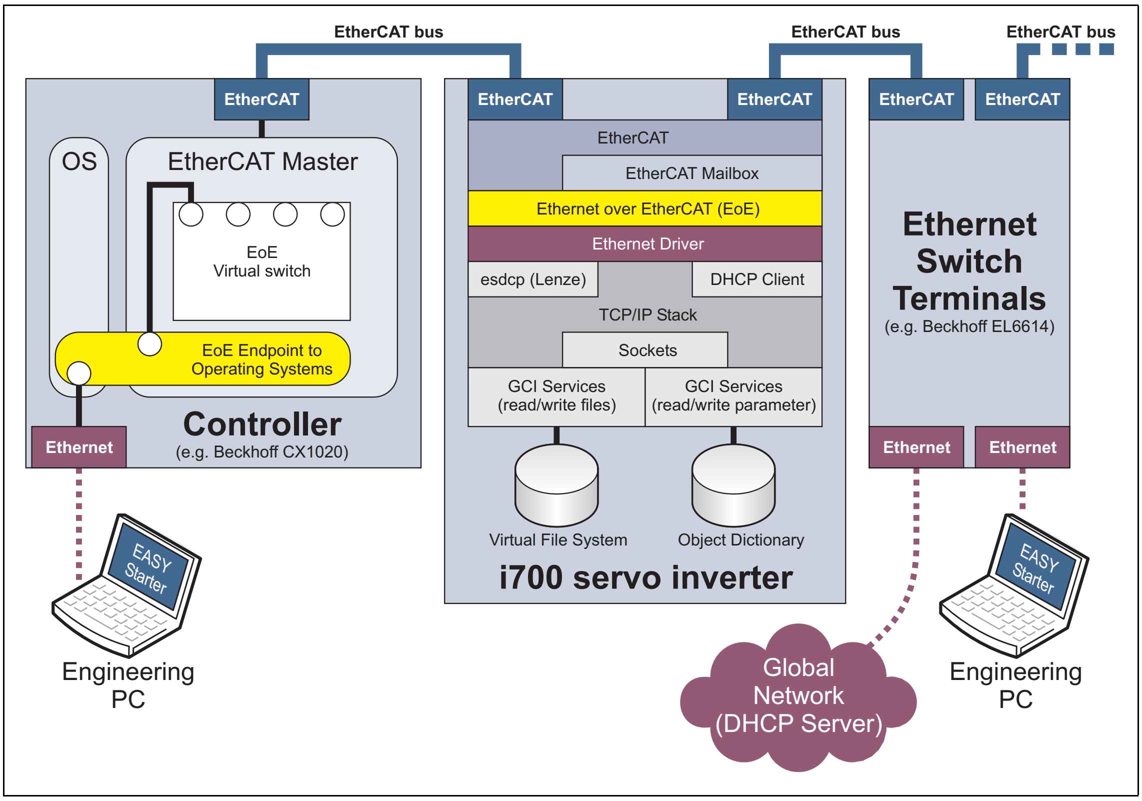
- In case of a static IP assignment, an IP differing from 0.0.0.0, subnet mask and gateway IP must be transmitted in addition.
- In case of a dynamic IP assignment (if the condition is met by external wiring)
  - the transmission of the IP address, subnet mask and gateway IP in the EoE initialisation telegram might not be necessary or
  - the IP address must be transmitted with 0.0.0.0.

3.6.1 System architecture

The following illustration shows the system architecture for an EtherCAT network with EoE nodes. This network provides access per PC with »EASY Starter« via two interfaces:

A. Access via an Ethernet-Switchport terminal (e.g. Beckhoff EL6614)

B. Access via control (e.g. Beckhoff CX1020)
For this purpose, the control has to support IP routing and the "EoE Endpoint to Operating Systems" functionality according to the "ETG.1500 master classes" specification.



[3-1] System architecture for an EtherCAT network with EoE nodes

The IP address allocation in the network can both be made statically by the EtherCAT master via configuration and via a higher-level infrastructure which can be connected to the Ethernet Switchport terminal.

The PC used for diagnostic purposes can get the IP address via the Ethernet Switchport terminal if this supports a DHCP server or BOOTP mechanism or it must feature a statically allocated IP itself in the same subnetwork.

3.6.2 Supported protocols and services

- ARP
- DHCP Client
- ESDCP
- ICMP (ping)
- IP
- UDP/TCP
- GCI-SDO communication (TCP port 9410)

3.6.3 Display of EoE-specific information

The following object serves to read out the EoE-specific information for diagnostic purposes.

0x2020 - EoE information

From software version V01.05.xx onwards
- The subcodes 1 ... 6 display the IP and Ethernet settings configured by the control during initialisation.
- The subcodes 7 and 8 display the traffic via EoE in Rx and Tx direction. The displayed values can be reset to "0" by entering any value.

Sub.	Name	Lenze setting	Data type
1	Virtual MAC address		STRING(32)
2	IP address		STRING(32)
3	Subnet mask		STRING(32)
4	Standard gateway		STRING(32)
5	DNS server		STRING(32)
6	DNS name		STRING(50)
7	Data packets received		UNSIGNED_32
8	Data packets transmitted		UNSIGNED_32
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

3.6.4 GCI-SDO communication (TCP port 9410)

The TCP port 9410 serves to establish a parameter communication with the i700 servo inverters. The PC (client) used for parameter setting and the i700 servo inverters (server) communication with each other by exchanging data via the Ethernet telegrams which are fed into the cyclic EtherCAT telegrams (see also System architecture).

- The parameter data of the i700 servo inverters are stored in "objects". The objects serves, for instance, to set operating parameters and motor data or query diagnostic information.
- The parameter data are transmitted as SDOs (Service Data Objects) and confirmed by the receiver, i.e. the transmitter receives a feedback whether the transmission was successful.
- The transmission of the parameter data usually is not time-critical.
- The parameter communication enables the writing and reading access to the object directory of the i700 servo inverters.

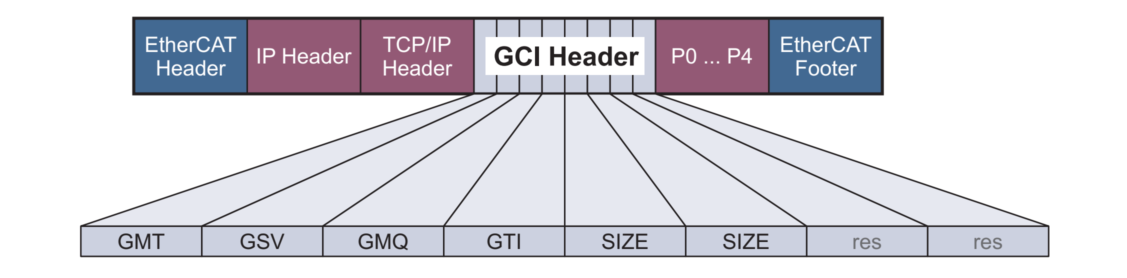
Note!

- Only one communication connection at a time is possible via the TCP port 9410, i.e. only one client at a time can be connected to the i700 servo inverters.
- In case of a writing access to parameter data, make sure that the changes made are not automatically saved in the i700 servo inverters.
  The »EASY Starter« serves to upload the parameters of the i700 servo inverters and save them as a file. This file can then be imported to the Engineering tool (e.g. »PLC Designer«).

3.6.4.1 Structure of the EtherCAT data telegram

The GCI protocol is used for communication.

The EtherCAT data telegram is shown below. Here, the GCI header represents the part of the program that is independent of the type of command transmitted.



[3-2] Structure of the GCI header within the EtherCAT frame

Field	Size	Description	
GMT	1 byte	GCI message type	
		0x01	Reserved
GSV	1 byte	GCI service identification	
		0x82	Read parameters
		0x83	Write parameters
GMQ	1 byte	GCI message qualifier	
		Bit 7	rsp (Request/Response)
		0	Request
		1	Response
		Bit 6	a (Abort)
		0	Data transfer OK
		1	Abort of the data transfer • The transfer is either aborted by the client or the server of a parameter data telegram. • A message is aborted without any confirmation. If the client waits for its message to be confirmed, it will receive the abort notice instead.
		Bit 5 ... bit 0	res (reserved)
		0b000000	Data contents = 0
GTI	1 byte	GCI transaction ID	
		0x00 ... 0xFF	Serial number (transaction identification) • For each client a definite serial number (0 ... 255) is allocated. • The serial number in the multitasking environment is used for referencing to the calling tasks (reverse transaction).
SIZE	2 bytes	Length of the user data • The user data area or the data telegram contains the parameter data. ▸ Assignment of user data areas P0 ... P4	
		0x14	20 bytes
		...	...
		0x114	276 bytes
res	2 bytes	Reserved	
		0x0000	Data contents = 0

Read parameters

The service identification (GSV) = 0x82 in the GCI header serves to read parameter data from the i700 servo inverters:

Client (PC) — Server (i700)

Request

Client (PC)	GMT	GSV	GMQ	GTI	SIZE	SIZE	res	res		Server (i700)
Request ►	0x01	0x82	0x00	0xXX	0xXX	0xXX	0x00	0x00	User data	► Indication

Answer

Client (PC)	GMT	GSV	GMQ	GTI	SIZE	SIZE	res	res		Server (i700)
Confirmation ◄	0x01	0x82	0x80	0xXX	0xXX	0xXX	0x00	0x00	User data (error code)	◄ Response

Write parameters

The service identification (GSV) = 0x83 in the GCI header serves to write parameter data to the i700 servo inverters:

Client (PC) — Server (i700)

Request

Client (PC)	GMT	GSV	GMQ	GTI	SIZE	SIZE	res	res		Server (i700)
Request ►	0x01	0x83	0x00	0xXX	0xXX	0xXX	0x00	0x00	User data	► Indication

Answer

Client (PC)	GMT	GSV	GMQ	GTI	SIZE	SIZE	res	res		Server (i700)
Confirmation ◄	0x01	0x83	0x80	0xXX	0xXX	0xXX	0x00	0x00	User data (error code)	◄ Response

3.6.4.2 Assignment of user data areas P0 ... P4

Range	Byte 1	Byte 2	Byte 3	Byte 4
P0	Status/error code		Data type	Reserved
P1	Index		Reserved	Reserved
P2	Subindex		Reserved	Reserved*
P3	Parameter value			
P4	Parameter value			
* When the data type VISIBLE_STRING is transmitted, byte 4 contains the number of the characters attached.				

Data type in P0 / byte 3

ID	Data type	Data length
0x01	INTEGER_8	1 byte
0x02	INTEGER_16	2 bytes
0x03	INTEGER_32	4 bytes
0x04	INTEGER_64	8 bytes
0x05	UNSIGNED_8	1 byte
0x06	UNSIGNED_16	2 bytes
0x07	UNSIGNED_32	4 bytes
0x08	UNSIGNED_64	8 bytes
0x09	FLOATING_POINT	4 bytes
0x0A	VISIBLE_STRING	max. 256 bytes
0x0B	OCTET_STRING	max. 256 bytes

Parameter value in P3 and P4

Depending on the data type, the parameter value assigns 1 ... 8 bytes. The data is stored in the Little-Endian format, i.e. first the low byte or low word, then the high byte or high word:

Data length of parameter value	Data area P3				Data area P4			
	Byte 1	Byte 2	Byte 3	Byte 4	Byte 1	Byte 2	Byte 3	Byte 4
1 byte	**Value**	0x00	0x00	0x00	0x00	0x00	0x00	0x00
2 bytes	Low byte	High byte	0x00	0x00	0x00	0x00	0x00	0x00
	Value							
4 bytes	Double word				0x00	0x00	0x00	0x00
	Low word		High word					
	Low byte	High byte	Low byte	High byte				
	Value							
8 bytes	Lower-order double word				Higher-order double word			
	Low word		High word		Low word		High word	
	Low byte	High byte	Low byte	High byte	Low byte	High byte	Low byte	High byte
	Value							

3.6.4.3 Error codes

The error code is located in the User data area P0, byte 1 and byte 2.

User data area P0			
Byte 1	Byte 2	Byte 3	Byte 4
Error code		Data type	Reserved
Example error code 0x9002			
Low byte	High byte		
0x02	0x90		

Note!

The other user data contents correspond to those of an error-free message.

Possible error codes

Error code		Definition	Description
Decimal	hex		
33803	0x840B	Invalid type	Invalid parameter type
33804	0x840C	Limit violation	Invalid parameter value
33806	0x840E	Unknown parameter	Invalid parameter index
33812	0x8414	Invalid size	Invalid parameter format
33813	0x8415	Not in select list	Parameter is not in the selection list
33814	0x8416	Read not allowed	Parameter read is not allowed
33815	0x8417	Write not allowed	Parameter write is not allowed
33816	0x8418	CINH not set	Controller inhibit is not set
33829	0x8425	Invalid subindex	Invalid parameter subindex
33837	0x842D	Access not allowed	Parameter access not allowed
36873	0x9009	Wrong GMT received	The general telegram identification does not correspond to the GCI communication.
36874	0x900A	Unknown server request	Internal error in the GCI
36878	0x900E	SRV timeout	

3.6.4.4 Telegram example 1: Querying the heatsink temperature (read request)

The heatsink temperature of the i700 servo inverters is to be read.

- Object to be read: 0x2D84:1
- Assumption: ϑ = 43°C

Request

- SDO command (GSV) = 0x82 = "Read parameter"
- GCI Message Qualifier (GMQ) = 0x00 = 0b00000000 = "Request"
- Transaction ID (GTI) here "0" (optional consecutive number 0 ... 255)
- Length of the user data (SIZE) = 0x0014 = 20 bytes

GCI header							
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
GMT	GSV	GMQ	GTI	SIZE	SIZE	res	res
0x01	0x82	0x00	0x00	0x14	0x00	0x00	0x00
Fixed	Read parameters	Request	Transactions ID	Length of the user data = 20 bytes		Reserved	

User data area P0			
Byte 1	Byte 2	Byte 3	Byte 4
Reserved		Data type	Reserved
0x00	0x00	0x00	0x00
		Optional for read request	

User data area P1				User data area P2			
Byte 1	Byte 2	Byte 3	Byte 4	Byte 1	Byte 2	Byte 3	Byte 4
Index		Reserved		Subindex		Reserved	
0x84	0x2D	0x00	0x00	0x01	0x00	0x00	0x00
Index = 0x2D84				Subindex = 1			

User data area P3				User data area P4			
Byte 1	Byte 2	Byte 3	Byte 4	Byte 1	Byte 2	Byte 3	Byte 4
Reserved				Reserved			
0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00

Answer

GCI Message Qualifier (GMQ) = 0x80 = 0b10000000 = "Response"

GCI header							
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
GMT	GSV	GMQ	GTI	SIZE	SIZE	res	res
0x01	0x82	0x80	0x00	0x14	0x00	0x00	0x00
Fixed	Read parameters	Response	Transactions ID	Length of the user data = 20 bytes		Reserved	

User data area P0			
Byte 1	Byte 2	Byte 3	Byte 4
Reserved		Data type	Reserved
0x00	0x00	0x02	0x00
		INTEGER_16	

User data area P1				User data area P2			
Byte 1	Byte 2	Byte 3	Byte 4	Byte 1	Byte 2	Byte 3	Byte 4
Index		Reserved		Subindex		Reserved	
0x84	0x2D	0x00	0x00	0x01	0x00	0x00	0x00
Index = 0x2D84				Subindex = 1			

User data area P3				User data area P4			
Byte 1	Byte 2	Byte 3	Byte 4	Byte 1	Byte 2	Byte 3	Byte 4
Parameter value		Reserved					
0xAE	0x01	0x00	0x00	0x00	0x00	0x00	0x00
Read value = 0x01AE = 430 = 43.0 [°C]							

3.6.4.5 Telegram-example 2: Querying the software version of the i700 (read request)

The software version of the i700 servo inverters is to be read.

- Object to be read: 0x100A
- Assumption: Software version = "1.5.0.9999 (release)"

Request

- SDO command (GSV) = 0x82 = "Read parameter"
- GCI Message Qualifier (GMQ) = 0x00 = 0b00000000 = "Request"
- Transaction ID (GTI) here "1" (optional consecutive number 0 ... 255)
- Length of the user data (SIZE) = 0x0014 = 20 bytes

GCI header							
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
GMT	GSV	GMQ	GTI	SIZE	SIZE	res	res
0x01	0x82	0x00	0x01	0x14	0x00	0x00	0x00
Fixed	Read parameters	Request	Transactions ID	Length of the user data = 20 bytes		Reserved	

User data area P0			
Byte 1	Byte 2	Byte 3	Byte 4
Reserved		Data type	Reserved
0x00	0x00	0x00	0x00
		Optional for read request	

User data area P1				User data area P2			
Byte 1	Byte 2	Byte 3	Byte 4	Byte 1	Byte 2	Byte 3	Byte 4
Index		Reserved		Subindex		Reserved	
0x0A	0x10	0x00	0x00	0x00	0x00	0x00	0x00
Index = 0x100A				No subindex			

User data area P3				User data area P4			
Byte 1	Byte 2	Byte 3	Byte 4	Byte 1	Byte 2	Byte 3	Byte 4
Reserved				Reserved			
0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00

Answer

- GCI Message Qualifier (GMQ) = 0x80 = 0b10000000 = "Response"
- Length of the user data =
  20 bytes of standard user data area P0 ... P4 **plus**
  21 bytes of attached string (incl. \0 termination) = 41 bytes

GCI header							
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
GMT	GSV	GMQ	GTI	SIZE	SIZE	res	res
0x01	0x82	0x80	0x01	0x29	0x00	0x00	0x00
Fixed	Read parameters	Response	Transactions ID	Length of the user data = 41 bytes		Reserved	

User data area P0			
Byte 1	Byte 2	Byte 3	Byte 4
Reserved		Data type	Reserved
0x00	0x00	0x0A	0x00
		VISIBLE_STRING	

User data area P1				User data area P2			
Byte 1	Byte 2	Byte 3	Byte 4	Byte 1	Byte 2	Byte 3	Byte 4
Index		Reserved		Subindex		Reserved	
0x0A	0x10	0x00	0x00	0x00	0x00	0x00	0x00
Index = 0x100A				No subindex			

User data area P3				User data area P4			
Byte 1	Byte 2	Byte 3	Byte 4	Byte 1	Byte 2	Byte 3	Byte 4
Reserved				Reserved			
0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00

Note!

The read parameter value of data type VISIBLE_STRING (here: "1.5.0.9999 (release)") follows subsequent to the standard user data area.

3.6.4.6 Telegram example 3: Setting the LV warning threshold in the i700 (write request)

The warning threshold for the low-voltage detection (LV) is to be set in the i700 servo inverters to 400 V.

- Object to be written: 0x2540:2

Request

- SDO command (GSV) = 0x83 = "Write parameter"
- GCI Message Qualifier (GMQ) = 0x00 = 0b00000000 = "Request"
- Transaction ID (GTI) here "42" (optional consecutive number 0 ... 255)
- Length of the user data (SIZE) = 0x0014 = 20 bytes

<table>
<tr><th colspan="8">GCI header</th></tr>
<tr><th>Byte 1</th><th>Byte 2</th><th>Byte 3</th><th>Byte 4</th><th>Byte 5</th><th>Byte 6</th><th>Byte 7</th><th>Byte 8</th></tr>
<tr><td>GMT</td><td>GSV</td><td>GMQ</td><td>GTI</td><td>SIZE</td><td>SIZE</td><td>res</td><td>res</td></tr>
<tr><td>0x01</td><td>0x83</td><td>0x00</td><td>0x2A</td><td>0x14</td><td>0x00</td><td>0x00</td><td>0x00</td></tr>
<tr><td>Fixed</td><td>Write parameters</td><td>Request</td><td>Transactions ID</td><td colspan="2">Length of the user data = 20 bytes</td><td colspan="2">Reserved</td></tr>
</table>

<table>
<tr><th colspan="4">User data area P0</th></tr>
<tr><th>Byte 1</th><th>Byte 2</th><th>Byte 3</th><th>Byte 4</th></tr>
<tr><td colspan="2">Reserved</td><td>Data type</td><td>Reserved</td></tr>
<tr><td>0x00</td><td>0x00</td><td>0x06</td><td>0x00</td></tr>
<tr><td colspan="2"></td><td>UNSIGNED_16</td><td></td></tr>
</table>

<table>
<tr><th colspan="4">User data area P1</th><th colspan="4">User data area P2</th></tr>
<tr><th>Byte 1</th><th>Byte 2</th><th>Byte 3</th><th>Byte 4</th><th>Byte 1</th><th>Byte 2</th><th>Byte 3</th><th>Byte 4</th></tr>
<tr><td colspan="2">Index</td><td colspan="2">Reserved</td><td colspan="2">Subindex</td><td colspan="2">Reserved</td></tr>
<tr><td>0x40</td><td>0x25</td><td>0x00</td><td>0x00</td><td>0x02</td><td>0x00</td><td>0x00</td><td>0x00</td></tr>
<tr><td colspan="2">Index = 0x2540</td><td colspan="2"></td><td colspan="2">Subindex = 2</td><td colspan="2"></td></tr>
</table>

<table>
<tr><th colspan="4">User data area P3</th><th colspan="4">User data area P4</th></tr>
<tr><th>Byte 1</th><th>Byte 2</th><th>Byte 3</th><th>Byte 4</th><th>Byte 1</th><th>Byte 2</th><th>Byte 3</th><th>Byte 4</th></tr>
<tr><td colspan="2">Parameter value</td><td colspan="6">Reserved</td></tr>
<tr><td>0x90</td><td>0x01</td><td>0x00</td><td>0x00</td><td>0x00</td><td>0x00</td><td>0x00</td><td>0x00</td></tr>
<tr><td colspan="2">Value to be written = 0x0190
= 400 [V]</td><td colspan="6"></td></tr>
</table>

Answer

GCI Message Qualifier (GMQ) = 0x80 = 0b10000000 = "Response"

GCI header							
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
GMT	GSV	GMQ	GTI	SIZE	SIZE	res	res
0x01	0x83	0x80	0x2A	0x14	0x00	0x00	0x00
Fixed	Write parameters	Response	Transactions ID	Length of the user data = 20 bytes		Reserved	

User data area P0			
Byte 1	Byte 2	Byte 3	Byte 4
Reserved		Data type	Reserved
0x00	0x00	0x06	0x00
		UNSIGNED_16	

User data area P1				User data area P2			
Byte 1	Byte 2	Byte 3	Byte 4	Byte 1	Byte 2	Byte 3	Byte 4
Index		Reserved		Subindex		Reserved	
0x40	0x25	0x00	0x00	0x02	0x00	0x00	0x00
Index = 0x2540				Subindex = 2			

User data area P3				User data area P4			
Byte 1	Byte 2	Byte 3	Byte 4	Byte 1	Byte 2	Byte 3	Byte 4
Parameter value		Reserved					
0x90	0x01	0x00	0x00	0x00	0x00	0x00	0x00
Written value (reflected)							

4 Device settings

Objects described in this chapter

Object	Name	Data type
0x10F1	ECAT: Behaviour in case of error	RECORD
0x2000	Device: Data	RECORD
0x2001	Device: Name	STRING(128)
0x2021	Device: Optical recognition	RECORD
0x2022	Device command	UNSIGNED_32
0x2100	Brand protection	INTEGER_32
0x2105	Set master password	STRING(16)
0x2540	Device: Voltage values	RECORD
0x2580	ECAT DC: Real-time information	RECORD

4.1 Behaviour in case of error

The i700 servo inverters features three different EtherCAT monitoring modes:

- Sync0 monitoring when DC mode is used
- PDO frame failure detection when DC mode is used
- Monitoring to EtherCAT line interruption

Sync0 monitoring when DC mode is used

This monitoring mode checks whether the Sync0 Signals are generated at the correct time in the i700 servo inverters if the "Distributed Clock mode" (DC mode) has been selected and the i700 servo inverters is in the "Operational" status.

- If no Sync0 signals arrive anymore during double the Sync0 cycle time, the i700 servo inverters changes to the "Safe-Operational" status and triggers an error (CiA402 error code 0x8700). 0x32 is returned as bus status (AL status code).
- After "Pre-Operational" has changed to "Safe-Operational", the generation of Sync0 pulses has to be started within 5 seconds. If this is not the case, or if a change from "Safe-Operational" to "Operational" is requested without the signals being generated accordingly, this error is triggered as well.
- This monitoring mode cannot be configured.

PDO frame failure detection when DC mode is used

This monitoring mode checks whether an EtherCAT-PDO telegram (Sync Manager 2 Event) has arrived between two Sync0 signals if the "Distributed Clock mode" (DC mode) has been selected. For this purpose, the i700 servo inverters is provided with an internal frame failure error counter which is increased by the value "3" in case of a frame failure. For every PDO received correctly, the error counter is reduced by the value "1".

This monitoring mode can be configured via the 0x10F1:2 object:

- From software version V01.04.xx onwards, monitoring is activated in the Lenze setting (0x10F1:2 = "20").
- If a value higher than "0" is set in 0x10F1:2: If the internal frame failure error counter reaches the set value, the i700 servo inverters changes to the "Safe-Operational" status and triggers an error (CiA402 error code 0x8700).

Tip!

It is reasonable to set a value ≥"4" in 0x10F1:2 to tolerate a failed PDO and prevent two PDO failures in a row.

The following table lists some possible settings:

Permitted PDO failures in a row	Monitoring threshold (0x10F1:2)
0	1 ... 2
1	4 ... 5
2	7 ... 8
3	10 ... 11

From software version V01.02.xx onwards, an extrapolator is available in the i700 servo inverters for the motor control, which can extrapolate the setpoints for one cycle. If two PDO frames fail in a row, no extrapolation takes place anymore. The last value is frozen.

Monitoring to EtherCAT line interruption

This monitoring mode generally checks whether there is an EtherCAT line interruption.

- If this is the case, the i700 servo inverters changes to the "Safe-Operational" status and triggers an error (CiA402 error code 0x8181).
- This monitoring mode cannot be configured and functions with and without "Distributed Clock mode" (DC mode).

0x10F1 - ECAT: Behaviour in case of error

Response of the device in the event of an error

Sub.	Name	Lenze setting	Data type
▸ 1	Local Error Reaction	2: Device-specific status	UNSIGNED_32
▸ 2	Synchronisation: Error threshold	20	UNSIGNED_32

Subindex 1: **Internal device response**			
Selection list(Lenze setting printed in bold)		Info	
2	**Device-specific status**	An error response is carried out by the i700 servo inverter only.	
☑ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			UNSIGNED_32

Subindex 2: **Synchronisation: Error threshold**				
Setting for PDO frame failure detection • If the internal frame failure error counter reaches the value set here, the Servo-Inverter i700 changes to the "Safe-Operational" status and triggers an error (CiA402 error code 0x8700). • From software version V01.04.xx onwards, the PDO frame failure detection is activated in the Lenze setting "20".				
Setting range (min. value | unit | max. value)			Lenze setting	
0		32	20	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_32

4.2 Device identification data

0x2000 - Device: Data

Type code (Lenze ID), serial number and manufacturing date of the device

Sub.	Name	Lenze setting	Data type
1	Device: Product designation		STRING(50)
2	Device: Serial number		STRING(50)
3	Device: Manufacturing date		STRING(50)
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

0x2001 - Device: Name

Any device name (e.g. "Wheel drive") can be set in this object for the purpose of device identification.

☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX		STRING(128)

0x2100 - brand protection

Sub.	Name	Lenze setting	Data type
1	Brand protection: set pin	0	INTEGER_32
2	Brand protection: enter pin	0	INTEGER_32
3	Brand protection: encryption	0	UNSIGNED_8

Subindex 1: **Brand protection: Set pin**				
Setting range (min. value | unit | max. value)			Lenze setting	
-1		9999999	0	
☑ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX				INTEGER_32

Subindex 2: **Brand protection: Enter pin**				
Setting range (min. value | unit | max. value)			Lenze setting	
-2		9999999	0	
☑ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX				INTEGER_32

Subindex 3: **Brand protection: Encryption**				
Setting range (min. value | unit | max. value)			Lenze setting	
0		9	0	
☑ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX				UNSIGNED_8

0x2105 - Set master password

Sub.	Name	Lenze setting	Data type
▸ 1	Set master password		STRING(16)
▸ 2	Log-in with master password		STRING(16)

Subindex 1: **Set master password**		
☑ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX		STRING(16)

Subindex 2: **Log-in with master password**		
☑ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX		STRING(16)

4.3 Function "Optical device recognition"

0x2021 - Device: Optical recognition

In the case of applications with multiple interconnected servo inverters it may be difficult to locate a device that has been connected online. The "Optical device recognition" function serves to locate the Servo-Inverter i700 by means of blinking LEDs.

- A setting of "1: Start" in subindex 1 activates the function:
  - The two LEDs "RDY" and "ERR" on the front of the i700 servo inverter will blink for the time period set in subindex 2 with a blinking frequency of 20 Hz. Then the function is deactivated automatically.
  - The LEDs on the RJ45 sockets are not used for this function.
  - If the function is reactivated within the time set, the time is extended correspondingly.
- With the setting "0: Stop" in subindex 1 the function can be aborted/deactivated prematurely.

Sub.	Name	Lenze setting	Data type
▸ 1	Start optical recognition	0: Stop	UNSIGNED_8
▸ 2	Optical recognition: Blinking time	5 s	UNSIGNED_16

Subindex 1: **Start optical recognition**			
Selection list(Lenze setting printed in bold)			
0	**Stop,**		
1	Start		
☑ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			UNSIGNED_8

Subindex 2: **Optical recognition: Blinking time**				
Setting range (min. value | unit | max. value)			Lenze setting	
0	s	6000	5 s	
☑ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX				UNSIGNED_16

4.4 Device commands

Note!

The execution of a device command may lead to an interruption of the EtherCAT communication with the master and to a standstill of the axis!

0x2022 - Device command

Selection list(Lenze setting printed in bold)		Info	
0	**No command**		
1000	Load Lenze setting	Load default parameter set	
1100	Restart device	Device restart	
1110	Restart safety		
☑ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			UNSIGNED_32

4.5 Monitoring of the DC-bus voltage

In the case of a single axis, the DC-bus voltage is monitored for undervoltage (LU) and overvoltage (OU).

In the case of a double axis, the DC-bus voltage of both axes is monitored for undervoltage (LU) and overvoltage (OU).

Before a critical undervoltage or overvoltage is reached, threshold values can be defined, indicating a warning or triggering an error.

Warning thresholds

The upper and lower warning threshold can be set separately. They are preset in the Lenze setting in compliance with the rated mains voltage of 400 V_{eff}. If the rated mains voltage deviates, the upper and lower warning threshold must be adapted manually.

The warning thresholds are active in the "Ready to switch on", "Switched on", "Operation enabled" and "Quick stop active" device states.

Error thresholds

The error thresholds and the thresholds for "Reset error" are not adjustable, however, they can be displayed using the objects specified in the following table.

- Up to and including software version V01.06.xx:
  - The error thresholds are active in the "Ready to switch on", "Switched on", "Operation enabled" and "Quick stop active" device states.
- From software version V01.07.xx onwards:
  - The error thresholds are active in the "Operation enabled" and "Quick stop active" device states.

Mains	Undervoltage thresholds			Overvoltage thresholds		
Rated voltage 0x2540:1	Warning threshold 0x2540:2	Error threshold 0x2540:3	Threshold "Reset error" 0x2540:4	Warning threshold 0x2540:5	Error threshold 0x2540:6	Threshold "Reset error" 0x2540:7
0: 230 V_{eff}	430 V	200 V	225 V	795 V	800 V	790 V
1: 400 Veff		285 V	430 V		800 V	790 V
2: 480 V_{eff}		490 V	535 V		800 V	790 V
4: 60 V *)		10 V	15 V		60 V	50 V
10: 230 V_{eff} Reduced LU threshold		50 V	100 V		800 V	790 V
11: 400 V_{eff} Reduced LU threshold		50 V	200 V		800 V	790 V
12: 480 V_{eff} Reduced LU threshold		50 V	250 V		800 V	790 V
Greyed out = read access only or threshold specified						

*) Only permissible for setting up a system! For all other operating states, this rated voltage is impermissible.

0x2540 - Device: Voltage values

Sub.	Name	Lenze setting	Data type
▸ 1	Mains: Rated voltage	1: 400 Veff	UNSIGNED_8
▸ 2	Undervoltage (LU): Warning threshold	430 V	UNSIGNED_16
▸ 3	Undervoltage (LU): Error threshold	285 V	UNSIGNED_16

Sub.	Name	Lenze setting	Data type
▸ 4	Undervoltage (LU): Threshold 'Reset error'	430 V	UNSIGNED_16
▸ 5	Overvoltage (OU): Warning threshold	795 V	UNSIGNED_16
▸ 6	Overvoltage (OU): Error threshold	800 V	UNSIGNED_16
▸ 7	Overvoltage (OU): Threshold 'Reset error'	790 V	UNSIGNED_16
▸ 8	DC-bus voltage is critical		UNSIGNED_8

Subindex 1: **Mains: Rated voltage**			
Selection of the mains voltage used with which the i700 servo inverter is operated.			
Selection list(Lenze setting printed in bold)			
0	230 Veff		
1	**400 Veff**		
2	480 Veff		
3	Reserved		
4	60 V (setting-up operation)		
10	230 Veff - low undervoltage threshold		
11	400 Veff - low undervoltage threshold		
12	480 Veff - low undervoltage threshold		
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			UNSIGNED_8

Subindex 2: **Undervoltage (LU): Warning threshold**				
Warning threshold for monitoring with regard to undervoltage • If the DC-bus voltage of the i700 servo inverter falls below the threshold value set here, the device reports a warning. Reset is effected with a hysteresis of 10 V.				
Setting range (min. value | unit | max. value)			Lenze setting	
0	V	800	430 V	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 3: **Undervoltage (LU): Error threshold**				
Display of the error threshold for monitoring with regard to undervoltage • If the DC-bus voltage of the i700 servo inverter falls below the threshold value shown here, the device reports an error and changes to the "Fault" error status. A restart is only possible after mains recovery and error reset.				
Display range (min. value | unit | max. value)			Initialisation	
0	V	800	285 V	
☐ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 4: **Undervoltage (LU): Threshold 'Reset error'**				
Display of the error reset threshold for monitoring with regard to undervoltage				
Display range (min. value | unit | max. value)			Initialisation	
0	V	800	430 V	
☐ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 5: **Overvoltage (OU): Warning threshold**				
Warning threshold for monitoring with regard to overvoltage • If the DC-bus voltage of the i700 servo inverter exceeds the threshold value set here, the device reports a warning. Reset is effected with a hysteresis of 10 V.				
Setting range (min. value | unit | max. value)			Lenze setting	
0	V	800	795 V	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 6: **Overvoltage (OU): Error threshold**				
Display of the error threshold for monitoring with regard to overvoltage • If the DC-bus voltage of the i700 servo inverter exceeds the threshold value shown here, the device reports an error and the motor module changes to the "Fault" error status.				
Display range (min. value | unit | max. value)			Initialisation	
0	V	800	800 V	
☐ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 7: **Overvoltage (OU): Threshold 'Reset error'**				
Display of the error reset threshold for monitoring with regard to overvoltage				
Display range (min. value | unit | max. value)			Initialisation	
0	V	800	790 V	
☐ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 8: **Critical DC-bus voltage**				
Display range (min. value | unit | max. value)			Initialisation	
0		1		
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☑ TX				UNSIGNED_8

Related topics:

▸ 24-V supply voltage monitoring (📖 290)

4.5.1 Resetting the "Undervoltage (LU)" error message

Available up to and including software version V01.06.xx !

Operation of the servo inverter can only be enabled if the "Undervoltage (LU)" error message has been reset after connecting the mains voltage.

Triggering the "Undervoltage (LU)" error message

The "Undervoltage (LU)" error message, 0x2540:2, triggers a fault if the DC-bus voltage is below the error threshold while the 24 V supply voltage is connected.

This response occurs if

- the DC-bus voltage has not reached the error threshold yet when the system is switched on.
- the DC-bus voltage has fallen below the error threshold when the mains voltage fails.

Available from software version V01.07.xx onwards

The "Undervoltage (LU)" error message can be already reset before the mains voltage is connected.

- The "Undervoltage (LU)" warning that is indicated, 0x2540:3, is reset automatically when the mains voltage is connected.

Note!

If the mains voltage is connected, controller inhibit of the servo inverter must be activated.

Triggering the "Undervoltage (LU)" error message

The "Undervoltage (LU)" error message, 0x2540:2, only triggers a fault if

- the "Operation enabled" or "Quick stop active" device state is to become effective at initital switch-on.
- the servo inverter is in the "Operation enabled" or "Quick stop active" device state if the mains voltage fails.

Diagnostics in the master control

If the servo inverter is in the "trouble" device state due to an undervoltage (LU), this state can be diagnosed in the master control:

- An error bit is set in the status word.
- By output of the error number, an evaluation can be carried out.
- A status bit for undervoltage is set in the CiA status word.

4.6 Real-time information (distributed clock)

0x2580 - ECAT DC: Real-time information

All time information provided in this object is based on UTC and transmitted in the format defined by EtherCAT for this purpose: In nanoseconds, at a width of 64 bits, based on a date of January 01, 2000, (2000-01-01) and a time of 00:00.

There are various ways for the i700 servo inverter to receive its real-time information from the outside:
- From the (EtherCAT) master (DC synchronous in the EtherCAT state "Safe-Operational" or "Operational").
- By writing to subindex 4 (ECAT DC: Current time).

Sub.	Name	Lenze setting	Data type
1	ECAT DC: Status "real-time information"	0: No real-time information received yet	UNSIGNED_8
2	ECAT DC: Time stamp first real-time information		UNSIGNED_64
3	ECAT DC: Time stamp last real-time information		UNSIGNED_64
4	ECAT DC: Current time		UNSIGNED_64

Subindex 1: **ECAT DC: Status real-time information**		
Selection list (read only)		Info
0	**No real-time information received yet**	The i700 servo inverter has not yet received any real-time information from the outside since switch-on. • The i700 servo inverter is still operating with a time which is based on the time when the device was switched off last or which is based on the time stamp of the firmware (whichever time is later).
1	Receive at least one piece of real-time information	The i700 servo inverter has at least once received one piece of real-time information from the outside since switch-on. • The first and the last time the i700 servo inverter has received real-time information are displayed in subindices 2 and 3.
2	Synchronous - cyclically updated	The i700 servo inverter receives cyclically updated real-time information from the outside. • The i700 servo inverter reports this status if it is e.g. DC synchronous in the EtherCAT state "Safe-Operational" or "Operational".
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX		UNSIGNED_8

Subindex 2: **ECAT DC: Time stamp first real-time information**			
Display of the time when the i700 servo inverter received its first real-time information from the outside since switch-on. • In the "No real-time information received yet" status, a value of "0" is displayed.			
Display range (min. value | unit | max. value)			Initialisation
0	ns	2^{64}-1	
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			UNSIGNED_64

Subindex 3: **ECAT DC: Time stamp last real-time information**			
Display of the time when the i700 servo inverter received its last real-time information from the outside.			
Display range (min. value | unit | max. value)			Initialisation
0	ns	2^{64}-1	
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			UNSIGNED_64

Subindex 4: **ECAT DC: Current time**				

Display of the time information the i700 servo inverter is currently using (time of the device if you will).
• The contents are updated every time this subindex is read.
• Depending on the status (see subindex 1), this time information is more or less precise:
  • In the "No real-time information received yet" status, the clock is generally slow compared to real time. This may mean weeks or even months if the i700 servo inverter was switched off for a long time.
  • In the "Receive at least one piece of real-time information" status, the clock is accurate to the second unless the time of the last time information (subindex 3) does not date back several days.
  • In the "Synchronous - cyclically updated" status, the clock is accurate to the µs with the clock of the higher-level master.
  • Due to the synchronisation with an external master, this clock is also able to go backwards. TwinCAT uses the PC clock e.g. as its time base!
Setting the time:
• The clock is set by writing a value.
  • If the i700 servo inverter is in the "No real-time information received yet" status before the writing process, it will then change to the "Receive at least one piece of real-time information" status.
  • If the i700 servo inverter is in the "Synchronous - cyclically updated" status during the writing process, it will ignore the written value.				

Setting range (min. value | unit | max. value)			Lenze setting	
0	ns	2^{64}-1		
☑ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX				UNSIGNED_64

5 Motor control & motor settings

Stop!

From firmware V 2.x and a hardware version of the Servo inverter i700 ≥ 1C, the maximum output frequency for ordered standard devices is limited to ± 599 Hz. If the setpoint requests a higher output frequency, this will cause an incrementing following error that can only be reduced if the requested setpoint falls below ± 599 Hz.

An upgrade of the firmware V 1.x ➞ V 2.x or a downgrade of the firmware V 2.x ➞ V 1.x is not permissible. In both cases, the FoE error code 0x800B appears. In addition, the 0x6080 error code with the "Error" response is generated in the i700 Servo Inverter and logged in the logbook. The error can be reset immediately.

This chapter describes the commissioning of the motor control in a recommended sequence.

Objects described in this chapter

Object		Name	Data type
Axis A	Axis B		
Axis control			
0x2822	0x3022	Axis command	UNSIGNED_32
0x2823	0x3023	Axis command: Progress	UNSIGNED_8
0x2825	0x3025	Modes of operation	UNSIGNED_8
0x2832	0x3032	Identification: Status word	UNSIGNED_16
0x2835	0x3035	Manual test mode: Settings	RECORD
0x2836	0x3036	Manual jog: Settings	RECORD
Controller settings			
0x2900	0x3100	Speed controller: Parameter	RECORD
0x2901	0x3101	Speed controller: Gain - adaptation	UNSIGNED_16
0x2902	0x3102	Speed controller: Load I component	INTEGER_16
0x2903	0x3103	Speed: Speed setpoint - filter time	UNSIGNED_16
0x2904	0x3104	Speed: Actual speed - filter time	UNSIGNED_16
0x2910	0x3110	Moments of inertia	RECORD
0x2939	0x3139	Switching frequency	UNSIGNED_8
0x2941	0x3141	Current controller: Feedforward control	UNSIGNED_8
0x2942	0x3142	Current controller: Parameter	RECORD
0x2943	0x3143	Motor: Current setpoint - filter time	UNSIGNED_16
0x2944	0x3144	Torque: Notch filter torque setpoint	RECORD
0x2945	0x3145	Torque: Jerk limitation setpoint	UNSIGNED_16
0x2947	0x3147	Inverter characteristic: Voltage grid points (y)	RECORD
0x2980	0x3180	Position controller: Gain	UNSIGNED_32
0x2981	0x3181	Position controller: Gain - adaptation	UNSIGNED_16
0x2982	0x3182	Position controller: Output signal limitation	UNSIGNED_32
0x2983	0x3183	Position: Select new actual position	INTEGER_32
0x2984	0x3184	Determine target position: Mode	UNSIGNED_8

Object		Name	Data type
Axis A	Axis B		
0x2986	0x3186	Position controller: Resulting gain adaptation	UNSIGNED_16
0x29C0	0x31C0	Field controller: Parameter	RECORD
0x29E0	0x31E0	Field weakening controller: Parameter	RECORD
0x29E1	0x31E1	Field set value limitation	UNSIGNED_16
0x29E2	0x31E2	DC link circuit voltage: Filter time	UNSIGNED_16
0x29E3	0x31E3	Motor voltage act. value: Filter time	UNSIGNED_16
0x29E4	0x31E4	Voltage reserve	UNSIGNED_8
V/f operation			
0x2B00	0x3300	VFC: V/f characteristic - shape	UNSIGNED_8
0x2B01	0x3301	VFC: V/f characteristic - define reference point	RECORD
0x2B02	0x3302	VFC: User-definable V/f characteristic - frequency grid points (x)	RECORD
0x2B03	0x3303	VFC: User-definable V/f characteristic - voltage grid points (y)	RECORD
0x2B04	0x3304	VFC: Voltage vector control - current setpoint	UNSIGNED_32
0x2B05	0x3305	VFC: Voltage vector control parameter	RECORD
0x2B06	0x3306	VFC: Voltage boost	UNSIGNED_16
0x2B07	0x3307	VFC: Load adjustment - parameter	RECORD
0x2B08	0x3308	VFC: Imax controller - parameter	RECORD
0x2B09	0x3309	VFC: Slip compensation - parameter	RECORD
0x2B0A	0x330A	VFC: Oscillation damping - parameter	RECORD
0x2B0B	0x330B	VFC: Setpoint frequency	INTEGER_16
0x2B0C	0x330C	VFC: Override point of field weakening	INTEGER_16
0x2B80	0x3380	DC-injection braking: Current	UNSIGNED_16
0x2BA0	0x33A0	Flying restart: Activate	UNSIGNED_8
0x2BA1	0x33A1	Flying restart: Current	UNSIGNED_16
0x2BA2	0x33A2	Flying restart: Start frequency	INTEGER_16
0x2BA3	0x33A3	Flying restart: Integration time	UNSIGNED_16
0x2BA4	0x33A4	Flying restart: Min. deviation	UNSIGNED_16
0x2BA5	0x33A5	Flying restart: Delay time	UNSIGNED_16
0x2BA6	0x33A6	Flying restart: Result	RECORD
Motor settings			
0x2C00	0x3400	Motor control	UNSIGNED_8
0x2C01	0x3401	Motor: Common parameters	RECORD
0x2C02	0x3402	Motor (ASM): Parameter	RECORD
0x2C03	0x3403	Motor (SM): Parameter	RECORD
0x2C04	0x3404	Motor: Lss saturation characteristic - inductance grid points (y)	RECORD
0x2C05	0x3405	Motor: Lss saturation characteristic - reference for current grid points (x)	UNSIGNED_16
0x2C06	0x3406	Motor (SM): Magnet characteristic (current) - grid points	RECORD
0x2C07	0x3407	Motor (ASM): Lh saturation characteristic - inductance grid points (y)	RECORD
0x2C08	0x3408	Motor: Motor parameter setting method	UNSIGNED_8
0x6075	0x6875	Motor rated current	UNSIGNED_32
0x6076	0x6876	Motor rated torque	UNSIGNED_32
Feedback system			
0x2C40	0x3440	Encoder: Type	UNSIGNED_8

Object		Name	Data type
Axis A	Axis B		
0x2C41	0x3441	Hiperface: Parameter	RECORD
0x2C42	0x3442	Encoder: Parameter	RECORD
0x2C43	0x3443	Resolver: Number of pole pairs	UNSIGNED_8
0x2C44	0x3444	Resolver error compensation: Parameter	RECORD
0x2C45	0x3445	Open circuit in feedback system: Response	UNSIGNED_8
0x2C46	0x3446	Feedback system: Specifiable number of revolutions	UNSIGNED_16
0x2C5F	0x345F	Feedback system: Parameter CRC	UNSIGNED_32
Pole position identification			
0x2C60	0x3460	Monitoring pole position identification: Response	UNSIGNED_8
0x2C61	0x3461	Pole position identification PPI (360°)	RECORD
0x2C62	0x3462	Pole position identification PPI (min. movement)	RECORD
0x2C63	0x3463	Pole position identification PPI (without movement)	RECORD
Advanced settings			
0x2DE0	0x35E0	Advanced settings	RECORD

5.1 Required commissioning steps (short overview)

The following subchapters provide information on the individual commissioning steps required for a specific control mode/motor type combination.

5.1.1 Servo control for synchronous motor (SM)

Required commissioning steps	
1	From software version V02.11.xx onwards: Wiring test by means of the "Cable Check" function Up to and including software version V02.10.xx: Wiring check by means of manual test modes
2	Setting the control mode
3	Accepting/adapting plant parameters
4	Setting the motor parameters for the servo control
5	Set motor monitoring: • Monitoring of the motor utilisation (I^2xt) • Motor temperature monitoring
6.	Setting the feedback system for the servo control
7	Only required for other manufacturers' motors: • Setting and optimising the current controller • Correction of the stator leakage inductance (Lss)... • Synchronous motor (SM): Pole position identification
8	Only required for the automatic calculation of the speed controller parameters: Determining the total moment of inertia
9	Setting the speed controller
10.	Setting the position controller

Optional commissioning steps (additional function or fine adjustment)	
A.	Compensating for inverter influence on output voltage
B.	Synchronous motor (SM): Compensating for temperature and current influences
C.	Jerk limitation
D.	Notch filters (band-stop filters)
E.	"Short-circuit braking" function

5.1.2 Servo control for asynchronous motor (ASM)

Required commissioning steps	
1	From software version V02.11.xx onwards: Wiring test by means of the "Cable Check" function Up to and including software version V02.10.xx: Wiring check by means of manual test modes
2	Setting the control mode
3	Accepting/adapting plant parameters
4	Setting the motor parameters for the servo control
5	Set motor monitoring: • Monitoring of the motor utilisation (I^2xt) • Motor temperature monitoring
6.	Setting the feedback system for the servo control
7	Only required for other manufacturers' motors: Setting and optimising the current controller
8	Only required for the automatic calculation of the speed controller parameters: Determining the total moment of inertia
9	Setting the speed controller
10.	Setting the position controller
11.	Only required for other manufacturers' motors: • Setting the field controller (ASM) • Setting the field weakening controller (ASM)

Optional commissioning steps (additional function or fine adjustment)	
A.	Compensating for inverter influence on output voltage
B.	Correction of the stator leakage inductance (Lss)...
C.	Asynchronous motor (ASM): Identifying the Lh saturation characteristic
D.	Estimating the optimal magnetising current
E.	Jerk limitation
F.	Notch filters (band-stop filters)
G.	"DC-injection braking" function

5.1.3 V/f characteristic control for asynchronous motor (ASM)

Required commissioning steps	
1	From software version V02.11.xx onwards: Wiring test by means of the "Cable Check" function Up to and including software version V02.10.xx: Wiring check by means of manual test modes
2	Setting the control mode
3	Accepting/adapting plant parameters
4	Compensating for inverter influence on output voltage
5	Only required if voltage vector control, DC-injection braking, or flying restart process is activated: Setting and optimising the current controller
6.	Defining the V/f characteristic shape
7	Setting the voltage boost
8	Setting the load adjustment
9	Activating the voltage vector control (Imin controller)
10.	Defining the behaviour at the current limit (Imax controller)
11.	Deactivating or parameterising feedbacks

Optional commissioning steps (additional function or fine adjustment)	
A.	"Flying restart" function
B.	"DC-injection braking" function
C.	Setting the slip compensation
D.	Setting the oscillation damping

5.2 Commissioning functions (short overview)

For quick commissioning, the i700 servo inverters provides various functions which serve to automatically calculate and set the parameters. These functions can be executed via the following parameter:

0x2822 | 0x3022 - Axis command

Via the cross-reference in the information column, a detailed description of the respective function is provided.

Selection list(Lenze setting printed in bold)		Info
0	**No command**	
1	Inverter characteristic: Load Lenze setting	▸ Loading the standard inverter characteristic
2	Estimating the optimal magnetising current	▸ Estimating the optimal magnetising current
3	Motor model parameters: Determine from motor data by approximation	▸ Enter nameplate data and have motor model parameters determined automatically
4	Current controller: Calculate controller parameters	▸ Calculate current controller parameters
5	Speed controller: Calculate controller parameters	▸ Calculate speed controller parameters
6	Position controller: Calculate controller parameters	▸ Calculate position controller parameters
7	Field controller: Calculate controller parameters	▸ Calculate field controller parameters
8	Field weakening controller: Calculate controller parameters	▸ Calculate field weakening controller parameters
9	Identify resolver error	▸ Determine resolver error
10	VFC: Calculate Imin controller parameters	V/f operation: Calculate minimum current controller parameters ▸ Activate voltage vector control
11	VFC: Calculate Imax controller parameters	V/f operation: Calculate current limiting controller parameters ▸ Defining the behaviour at the current limit (Imax controller)
12	VFC: Calculate flying restart controller parameters	V/f operation: Calculate flying restart controller parameters ▸ Additional flying restart function
13	Lh saturation characteristic : Load Lenze setting	▸ Load standard Lh saturation characteristic
14	Get Hiperface information from encoder	From software version V01.03.xx onwards
15	Motor model parameters: Determine completely from motor data by approximation	▸ Enter nameplate data and have motor model parameters determined automatically
☑ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX		UNSIGNED_32

0x2823 | 0x3023 - Axis command: Progress

Display of the current progress of the activated commissioning function

Display range (min. value | unit | max. value)			Initialisation
0		100	0
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			UNSIGNED_8

0x2825 | 0x3025 - Axis: Operating mode

This object serves to activate different commissioning test modes and procedures for the automatic parameter identification.
Via the cross-reference in the information column, a detailed description of the respective function is provided.

Selection list(Lenze setting printed in bold)		Info
0	**CiA402 mode active**	Normal operation (CiA402 operating modes active) ▶ CiA402 device profile
1	Manual test mode: Voltage/frequency	▶ Test mode "Voltage/frequency"
2	Manual test mode: Current/frequency	▶ Test mode "Current/frequency"
3	Manual test mode: Current pulse	▶ Test mode "Current pulse"
4	Manual control mode	▶ Manual control
5	Pole position identification PPI (360°)	▶ Synchronous motor (SM): Pole position identification
6	Pole position identification PPI (min. movement)	
7	Pole position identification PPI (without movement)	From software version V01.03.xx onwards ▶ Synchronous motor (SM): Pole position identification
8	Identify inverter characteristic	▶ Compensating for inverter influence on output voltage
9	Identify motor parameters	▶ Have motor parameters determined by the i700 servo inverter
10	Identify Lh saturation characteristic	▶ Asynchronous motor (ASM): Identifying the Lh saturation characteristic
11	PRBS excitation: Mechanical path	for internal use only
12	PRBS excitation: Current controller loop	
13	PRBS excitation: Speed controller loop	
14	PRBS excitation: Position controller loop	
15	Cable check	
☑ Write access ☑ CINH ☐ OSC ☐ P ☐ RX ☐ TX		UNSIGNED_8

0x2832 | 0x3032 - Identification: Status word

Display of the status of an activated function for automatic parameter identification.

Display range (min. value | unit | max. value)			Initialisation
0		65535	0
Value is bit-coded:			Info
Bit 0	Enable identification		An identification was activated via 0x2825 (or 0x3025 for axis B) with a value > 4.
Bit 1	Identification in progress		The identification selected via 0x2825 (or 0x3025 for axis B) is carried out (changes to 1 after the controller enable).
Bit 2	Identification complete		Identification has been completed
Bit 3	Identification failed		The identification failed or has been aborted by controller inhibit.
Bits 4-15	Reserved		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☐ TX			UNSIGNED_16

5.2.1 Enable/inhibit operation via control word

The operating mode can only be selected in 0x2825 (or 0x3025f or axis B) when operation is inhibited (pulse inhibit). In order to start the corresponding procedure after the selection, the operation must be enabled explicitly.

Required steps to enable the operation after an STO:

1. Apply the STO terminals of the axis to 24 V.
2. Use the control word (0x6040 or 0x6840 for axis B) to execute the following commands one after another:

Command	Setting in the control word	Notes
Shutdown	6	A changeover from the "Switch on disabled" device status to the "Ready to switch on" device status takes place.
Switching on	7	The switch-on inhibit which is active after switch-on or reset of an error (acknowledgement) is deactivated. A changeover to the "Switched on" device status takes place.
Enable operation	15	The operation is enabled and an active quick stop is quit again. A changeover to the "Operation enabled" device status takes place.

Access to the control word via SDO access

The control word is part of the standard mapping and is preset in the cyclic interface. In this configuration, the control word commands have to be initiated by the controller via PDO access. If this is not possible, for instance, since a manual intervention through the controller is not intended, it is possible to switch off the control via PDO and carry out the enable via an SDO access:

1. Set the object 0x2824 (or 0x3024 for axis B) to "0: Off" to switch off the control via PDO.
2. An SDO access to the control word (0x6040 or 0x6840 for axis B) serves to execute the commands given above one after another.
3. After having executed the commands in the object 0x2824 (or 0x3024 for axis B), select "1: Activate" to re-activate the control via PDO.

Inhibit operation via control word (pulse inhibit)

In order to inhibit the enabled operation without using the STO terminals, use the control word (0x6040 or 0x6840 for axis B) to execute the following command:

Command	Setting in the control word	Notes
Disable operation	7	The pulse inhibit is set. When the automatic brake operation is activated, it is waited for the parameterised brake closing time until the brake is applied before the pulse inhibit is set. It is changed back to the "Switched on" device status.

In this case, a subsequent renewed enable only requires (without using the STO terminals) the control word command "Enable operation" (setting "15").

5.2.2 Saving changed parameters safe against mains failure

If control parameters are changed during the commissioning phase, e.g. by the functions for automatic parameter identification, the changed control parameters must be uploaded from the i700 servo inverters into the controller for permanent storage.

Note!

Currently, the changed controller parameters cannot be transferred directly into the parameter list of the corresponding axis using the »PLC Designer« but only indirectly via the »EASY Starter« (see the following instructions).

How to transfer parameters changed in the i700 servo inverters into the PLC project:

In the »EASY Starter«:

1.  Upload parameter set and save as Lenze parameter file (*.gdc).

In the »PLC Designer«:

2. Go to the *Device view* and select the corresponding i700 servo inverters.
3. Go to the **Project** menu and select the **Device parameters→Import device parameters** command to import the Lenze parameter file (*.gdc) into the selected device.
4. Save PLC project.

5.3 Wiring test by means of the "Cable Check" function

This function extension is available from software version V02.11.xx onwards!

This function serves to detect wiring errors or cable damage which can cause uncontrolled movements of the machine during the commissioning phase or during operation. The function therefore helps to prevent machine damage and serves to correct these errors as quickly as possible.

Use of the function in the life cycle of the machine

During the commissioning phase, the machine can be checked for installation errors on the motor and motor encoder.

▸ Executing the Cable Check function during the commissioning phase (📖 73)

We furthermore recommend always executing this function immediately after initial switch-on of the mains voltage or after servicing when the commissioning phase has been completed.

▸ Executing the Cable Check function during operation (📖 77)

Note!

Load encoders are not supported by the Cable Check function.

Parameters

0x2C64 | 0x3464 - Cable check

From software version V02.11.xx onwards

Sub.	Name	Lenze setting	Data type
▸ 1	Cable check: Behaviour after switch-on	0: No operation	UNSIGNED_8
▸ 2	Cable check: Status word		UNSIGNED_16

Subindex 1: **Cable check: Behaviour after switch-on**			
Selection list(Lenze setting printed in bold)			
0	**No Operation**		
1	Test at initial switch-on and after encoder error only		
2	Test after every switch-on		
☑ Write access ☑ CINH ☐ OSC ☑ P ☐ RX ☐ TX			UNSIGNED_8

Subindex 2: **Cable check: Status word**			
Display range (min. value | unit | max. value)			Initialisation
0		65535	
Value is bit-coded:			Info
Bit 0	Enable cable check		
Bit 1	Cable check running		
Bit 2	Cable check completed		
Bit 3	Cable check failed		
Bits 4-15	Reserved		
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			UNSIGNED_16

5.3.1 Executing the Cable Check function during the commissioning phase

The **Cable Check** function should be executed manually by the user while the machine is commissioned, in order to identify typical errors that may occur during the machine installation.

Typical errors

1. The motor encoders are connected to the wrong device before initial switch-on, whereas the motor is connected to the right device.
   If the motor encoders that are incorrectly connected are of the same type, the motor encoder monitoring does not detect any error.
2. The motor encoders are connected to the right device, however, the motor is connected to the wrong device.
3. The connection of the motor phases to the device is reversed.
4. Individual wires of the rotary transducer are connected incorrectly.

Before executing the **Cable Check** function manually, observe the notes given in

▶ Preconditions for the manual execution of the Cable Check function (📖 74)

Then perform the test

▶ How to manually execute the Cable Check function (📖 75)

Preconditions for the manual execution of the Cable Check function

- The motor can remain coupled to the kinematics.
  - In the case of drives without a motor holding brake, the rotor must be able to move by 20 ° (electrically). This makes it necessary to set the operating mode of the motor holding brake to "No brake connected" (0x2820:1 = 2).
  - In the case of drives with a motor holding brake, the test is performed against the closed motor holding brake. With 0x2820:1, the operating mode must be set to the value **0** or **1**.
- The device must not be in the **Fault** state.
  ▶ Device states (📖 230)
- 24 V supply voltage must be available.
  The display on the front of the blue LED shows **ON** or is blinking.
  ▶ LED status display (📖 318)
- The i700 servo inverter must be supplied with mains voltage.
  Status: 0x6041, bit 4 = TRUE
- The i700 servo inverter must not be in the **STO** state. The safety functions must be parameterised.
  Status: 0x6041, bit 15 = FALSE
  ▶ STO ("Safe Torque Off") (📖 240)
- The motor data must be set correctly.
  ▶ Setting the motor parameters for the servo control (📖 100)
- The data of the motor encoder must be set correctly.
  ▶ Settings for "resolver" version (📖 115)
  ▶ Settings for "encoder" version (📖 121)
- No error message must be active.
  Status: 0x6041, bit 3 = FALSE
  If an error message is active, eliminate these errors first and reset the error message.
- The motor control (0x2C00 or 0x3400) must be set to
  - Servo control - synchronous motor (SM) or
  - Servo control - asynchronous motor (ASM).

  The function only supports these two motor controls.
- The behaviour after switch-on must be set: 0x2C64:1= **0** ("No action").

How to manually execute the Cable Check function

1. Open the »EASY Starter« engineering tool.
2. Establish an online connection to the i700 servo inverter.
3. Call the **Motor commissioning**tab in the workspace of the »EASY Starter«.
4. Call the **Feedback** tab there.
5. In the jalousie **Cable check: Motor and motor encoder** , execute the **Cable Check** function via
   - parameter 0x2825, selection **15**, or
   - by pressing the **Execute Cable Check** button.

   The **Cable Check** function is now activated, i.e.
   - 0x2C64:2, bit 0 = TRUE
   - 0x2832, bit 0 = TRUE
6. Activate the i700 servo inverter via the CiA402 control word (0x6040 or 0x6840 for axis B).

   Execution of the function is indicated as completed in the »EASY Starter« after approx. one second.

   Additionally, the »EASY Starter« indicates whether an error has occurred during the check:

   ▶ Result of the Cable Check

Tip!

The **Cable Check** function can always be started via the **Execute Cable Check** button if the preconditions for this are met.

▶ Preconditions for the manual execution of the Cable Check function (📖 74)

Result of the Cable Check

Status bit	Result of the Cable Check		INFO
	No error	Error	
0x2C64, Cable Check			
Bit 00	FALSE		Cable check enabled
Bit 01	FALSE		Cable check running
Bit 02	TRUE	TRUE	Cable check completed The bit remains TRUE until the 24V supply is switched off (blue and red LED = OFF) or the cable check function is executed again.
Bit 03	FALSE	TRUE	Cable check failed In the event of an error, the bit remains TRUE until the function is executed again or the 24V supply is switched off (blue and red LED = OFF).
0x2832, identification			
Bit 00	FALSE		Identification enabled
Bit 01	FALSE		Identification running
Bit 02	TRUE	TRUE	Identification completed
Bit 03	FALSE		Identification failed

No error detected by the Cable Check function

The i700 servo inverter then goes to the **Operation enabled** state.

▸ Operation enabled (📖 236)

- Deactivate the i700 servo inverter, e.g. via the control word. You can do this using keys F8 / F9 in the EASY Starter.
- Deactivate the **Cable Check** function via parameter 0x2825 and set a selection other than **15** .

Error detected by the Cable Check function

If the **Cable Check** function detects an error, the i700 servo inverter automatically changes to the **Fault** status.

▸ Fault (📖 239)

Error correction

1. Consult the logbook to identify the error causes.
2. Switch off the power supply and 24V supply of the device.
3. Check the wiring and correct it, if necessary.
4. Execute the test again.

Note!

The "Identification" status word (0x2832) is also used by other functions. The display is therefore only valid as long as no other function using this status word is active.

5.3.2 Executing the Cable Check function during operation

During operation, the **Cable Check** function can be automatically executed by the i700 servo inverter itself, in order to detect damage on the motor or motor encoder already when the machines are switched on.

Typical damage that may occur during operation:

- Wire breakage on the motor encoder and / or motor cable
- Whole motor encoder cable and / or motor cable torn off
- Screwed connections that are loosening (e.g. on the motor encoder plug)
- Loose contacts

Starting condition for the Cable Check

The following starting conditions can be set for the automatic **Cable Check** function:

- Check only at initial switch-on or after motor encoder error (0x2C64:1 = **1**) or
- Check after every switch-on (0x2C64:1 = **2**).

Tip!

Observe the line diagrams for configuring the controller.

Preconditions for the automatic execution of the Cable Check function

- In the case of drives without a motor holding brake, the rotor must be able to move by 20 ° (electrically). This makes it necessary to set the operating mode of the motor holding brake to "No brake connected" (0x2820:1 = 2).
  A different setting is interpreted as "Brake available" interpretiert. A testing procedure is then applied, which causes the **Cable Check** function to fail with brakeless motors.
- 24 V supply voltage must be available.
  The display on the front of the blue LED shows **ON** or is blinking.
  ▸ LED status display (📖 318)
- The i700 servo inverter must be supplied with mains voltage.
  Status: 0x6041, bit 4 = TRUE
- The i700 servo inverter must not be in the **STO** state. The safety functions must be parameterised.
  Status: 0x6041, bit 15 = FALSE
  ▸ STO ("Safe Torque Off") (📖 240)
- No error message must be active.
  Status: 0x6041, bit 3 = FALSE
  If an error message is active, eliminate these errors first and reset the error message.

How to set the Cable Check function to be executed automatically

1. Open the »EASY Starter« engineering tool.
2. Establish an online connection to the i700 servo inverter.
3. Call the **Motor commissioning**tab in the workspace of the »EASY Starter«.
4. Call the **Feedback** tab there.
5. In the jalousie **Cable check: Motor and motor encoder** , set the behaviour after switch-on to
   - Check only at initial switch-on or after motor encoder error (0x2C64:1 = **1**) or
   - Check after every switch-on (0x2C64:1 = **2**).
6. In a last step, changed settings must be saved with mains failure protection

   ▶ Saving changed parameters safe against mains failure (📖 70)

Note!

When the pole position identification is set after every switch-on (0x2C63:1 = **2**) and **Cable Check** after every switch-on (0x2C64:1 = **2**), the pole position identification is carried out first, followed by the **Cable Check**.

Depending on the parameter setting in 0x2C64:1, the **Cable Check** starts automatically if the controller activates operation of the i700 servo inverter via the CiA402 control word (0x6040 or 0x6840 for axis B).

- The check takes approx. one second.
  During this time, 0x2C64, bit 1 is set to TRUE ("Cable Check running").

A possibly available motor holding brake remains closed. In this case, the rotor is moved against the closed motor holding brake.

- If no motor holding brake is available, the rotor of the motor is moved electrically by approx. 20.

The inverter remains in the **switched-on** state (see CiA402 status word 0x6041 (or 0x6841 for axis B) until the cable check has been completed.

Tip!

The **Cable Check** function can always be started via the **Execute Cable Check** button if the preconditions for this are met.

▶ Preconditions for the manual execution of the Cable Check function (📖 74)

Result of the Cable Check function

Status bit	Result of the Cable Check		INFO
	No error	Error	
0x2C64, Cable Check			
Bit 00	FALSE		Cable Check enabled
Bit 01	FALSE		Cable check running
Bit 02	TRUE	TRUE	Cable check completed The bit remains TRUE until the 24V supply is switched off (blue and red LED = OFF) or the cable check function is executed again.
Bit 03	FALSE	TRUE	Cable check failed In the event of an error, the bit remains TRUE until the function is executed again or the 24V supply is switched off (blue and red LED = OFF).
0x2832, identification			
Bit 00	FALSE		Identification enabled
Bit 01	FALSE		Identification running
Bit 02	TRUE	TRUE	Identification completed
Bit 03	FALSE		Identification failed

No error detected by the Cable Check function

The i700 servo inverter then goes to the **Operation enabled** state.

▸ Operation enabled (📖 236)

Error detected by the Cable Check function

If the **Cable Check** function detects an error, the i700 servo inverter automatically changes to the **Fault** status.

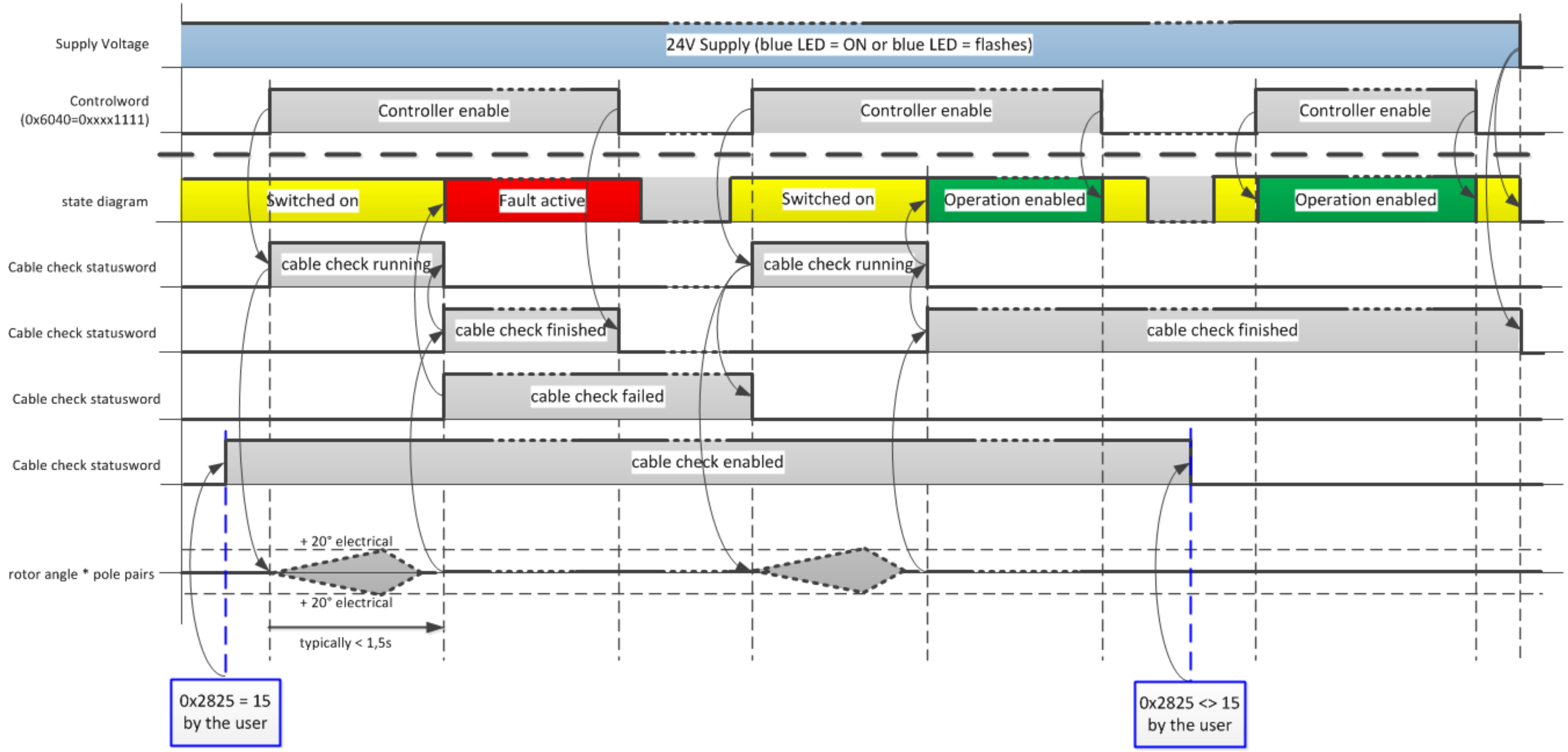
▸ Fault (📖 239)

Possible error sources

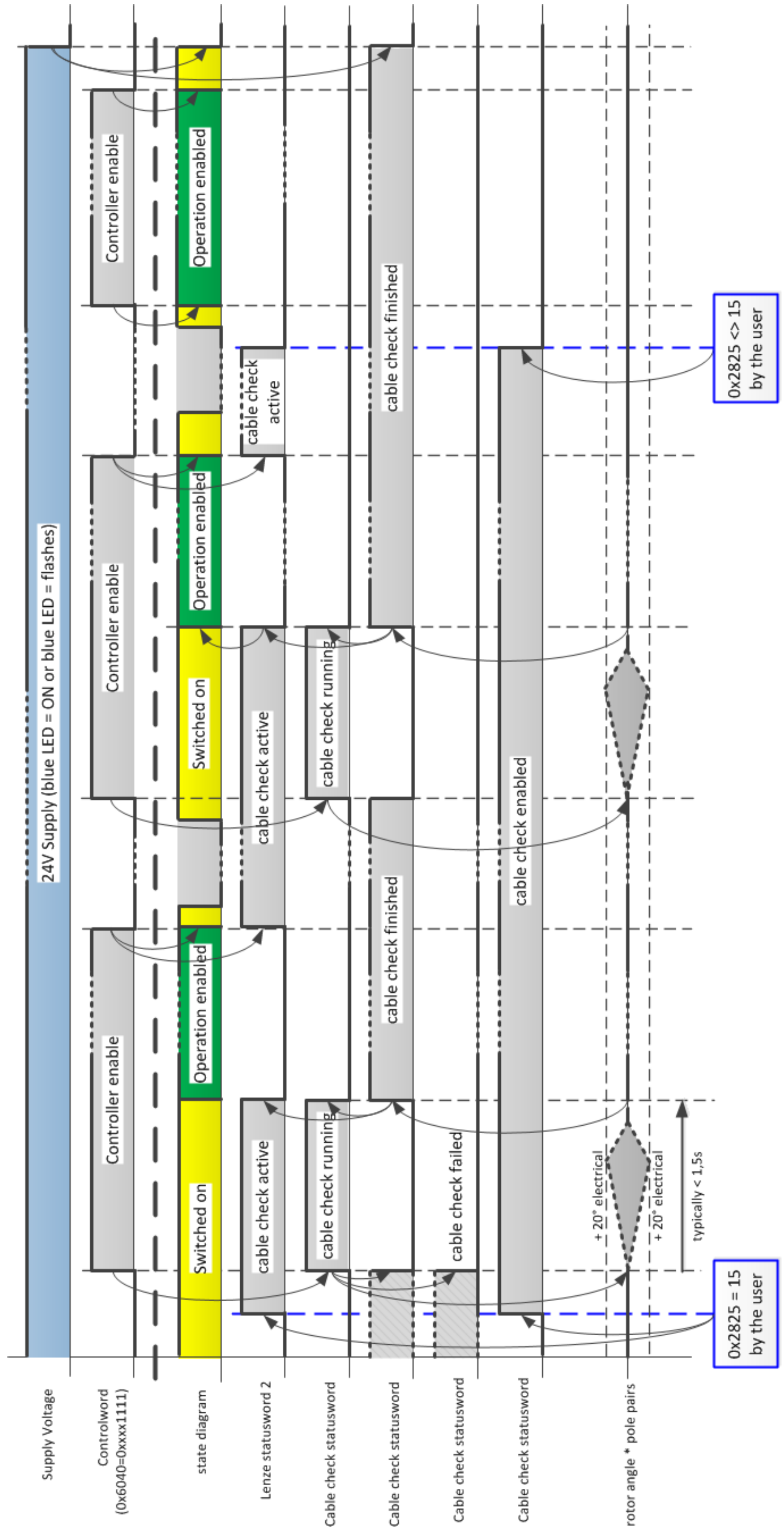
- Failure of a motor phase: 0xFF09 ... 0xFF0C
- Incorrect direction of rotation detected: 0xFF27

Line diagrams

The following line diagram shows the status sequence occurring if an error is detected in the first cycle:



The following line diagram indicates the error-free status sequence with an automatic activation of the Cable Check:


Supply Voltage
24V Supply (blue LED = ON or blue LED = flashes)
Controlword (0x6040=0xxxx1111)
Controller enable
Controller enable
Controller enable
state diagram
Switched on
Operation enabled
Switched on
Operation enabled
Operation enabled
Lenze statusword 2
cable check active
cable check active
cable check active
Cable check statusword
cable check running
cable check running
Cable check statusword
cable check finished
cable check finished
Cable check statusword
cable check failed
Cable check statusword
cable check enabled
rotor angle * pole pairs
+ 20° electrical
+ 20° electrical
typically < 1,5s
0x2825 = 15 by the user
0x2825 <> 15 by the user

5.4 Wiring check by means of manual test modes

Tip!

The wiring check by means of manual test modes is mainly replaced by the **Cable Check** function which is available from software version V02.13.xx onwards!

▸ Wiring test by means of the "Cable Check" function (📖 71)

Before the parameterisation of the actual control is started, the wiring of the motor (power and encoder connection) should be checked for errors and function and, if required, should be corrected.

1. Provided that there is a in-phase motor connection and a positive field frequency (0x2DDD / 0x35DD), the motor shaft rotates clockwise.
2. An existing speed feedback (motor encoder) in the rotor position (0x2DDD / 0x35DD) generates a numerical value with positive counting direction.
   If necessary, take the following measures:

Field frequency 0x2DDD / 0x35DD	Display 0x2DDE / 0x35DE	Measure
CW	0 ... 2047	None
	2047 ... 0	Correct motor cable / encoder cable
CCW	2047 ... 0	None
	0 ... 2047	Correct motor cable / encoder cable

3. After controller inhibit via the 0x2825 parameter (or 0x3025 for axis B), the following test modes are available for activation:
   - Manual test mode "voltage/frequency"
   - Manual test mode "current/frequency"

   The parameters for the test modes can be adapted via parameter 0x2835 (or 0x3035 for axis B). For this, observe the notes in the description of the respective test mode.

0x2835 | 0x3035 - Manual test mode: Settings

Sub.	Name	Lenze setting	Data type
1	Manual test mode: Current setpoint	0 %	INTEGER_16
2	Manual test mode: Frequency	0.0 Hz	INTEGER_16
3	Manual test mode: Starting angle	0.0 °	INTEGER_16
4	Manual test mode: Frequency 2	0.0 Hz	INTEGER_16

Subindex 1: **Manual test mode: Setpoint current**				
Selection of the r.m.s. value of a phase current for test mode • 100 % ≡ rated motor current (0x6075 or 0x6875 for axis B)				
Setting range (min. value | unit | max. value)			Lenze setting	
0	%	1000	0 %	
☑ Write access ☐ CINH ☑ OSC ☐ P ☑ RX ☐ TX				INTEGER_16

Subindex 2: **Manual test mode: Frequency**				
Selection of the frequency for test mode Please observe the notes regarding the limitation of the output frequency in the chapters • Manual jog • Limitation of the output frequency				
Setting range (min. value | unit | max. value)			Lenze setting	
-1000.0	Hz	1000.0	0.0 Hz	
☑ Write access ☐ CINH ☑ OSC ☐ P ☑ RX ☐ TX			Scaling: 1/10	INTEGER_16

Subindex 3: **Manual test mode: Starting angle**				
Selection of the starting angle for test mode **Note!** In the case of the synchronous motor, a jerky compensating movement occurs after controller enable if the pole position of this movement does not correspond to the starting angle.				
Setting range (min. value | unit | max. value)			Lenze setting	
-1000.0	°	1000.0	0.0 °	
☑ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			Scaling: 1/10	INTEGER_16

Subindex 4: **Manual test mode: Frequency 2**				
Setting range (min. value | unit | max. value)			Lenze setting	
0.0	Hz	100.0	0.0 Hz	
☑ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			Scaling: 1/10	INTEGER_16

5.4.1 Manual test mode "voltage/frequency"

Functional description

Note!

In case of devices that correspond to the "Dual Use Regulation" (EC 428/2009), values of + 599 Hz to + 1000 Hz or - 599 Hz to - 1000 Hz do not cause an increase of the output frequency. Please observe the dead band that occurs in this case.

More information can be found in the section entitled Frequency and speed limitations (📖 204).

In this test mode, a rotating field voltage with the set output frequency f_{out} is output at the motor terminals after controller enable.

- If the frequency selection is positive, the motor should rotate clockwise when one is looking at the A-side of the motor. If this is not the case, there is a simple rotating movement in the motor phases.
- The level of the output voltage is determined via the following equation:

Equation for calculating the output voltage

$$U_{OUT} = f_{OUT} \cdot \frac{U_{rated}}{f_{rated}}$$

Axis A	Axis B	Symbol	Description	Dimension unit
0x2D82	0x3582	V_{out}	Current output voltage	V
0x2835:2	0x3035:2	f_{out}	Output frequency for test mode Please observe the notes in the "Output frequency" section of the chapter entitled ▸ Frequency and speed limitations	Hz
0x2B01:1	0x3301:1	V_{rated}	V/f rated voltage	V
0x2B01:2	0x3301:2	f_{rated}	V/f base frequency	Hz
Greyed out = read access only				

By means of the manual "Voltage/frequency" test mode, the wiring of the feedback system can also be checked.

- If the feedback system is set correctly, the following actual speed should be shown:

Equation for calculating the actual speed

$$n_{act} = \frac{f_{OUT}}{zp_{motor}} \cdot 60$$

Axis A	Axis B	Symbol	Description	Dimension unit
0x606C	0x686C	n_{act}	Actual speed	rpm
0x2835:3	0x3035:3	f_{out}	Output frequency for test mode	Hz
0x2C01:1	0x3401:1	zp_{Motor}	Motor - number of pole pairs	
Greyed out = read access only				

Preconditions for the execution

- The motor must be able to rotate freely.
- The i700 servo inverters is free of errors and is in the "Switched on" device status.

Response of the motor during the execution

The motor rotates as a function of the set output frequency.

How to activate the manual test mode "voltage/frequency":

1. If the i700 servo inverters is enabled, disable the i700 servo inverters.
   ▸ Enable/inhibit operation via control word (📖 69)
2. Set parameter 0x2825 (or 0x3025 for axis B) to "1" to change to the "Voltage/frequency" test mode.
3. Enable the i700 servo inverters to start the test mode.
4. To stop the test mode:
   - Disable the i700 servo inverters.
   - Set parameter 0x2825 (or 0x3025 for axis B) to "0" to change back to the CiA402 mode.

5.4.2 Manual test mode "current/frequency"

Functional description

In this test mode, three phase currents are injected in the connected motor after controller enable.

- Via the following parameters, the test mode can be adapted:

Object		Info	Data type
Axis A	Axis B		
0x2835:1	0x3035:1	r.m.s. value of a phase current • Selection in [%] relative to the rated motor current.	INTEGER_16
0x2835:2	0x3035:2	Frequency	INTEGER_16
0x2835:3	0x3035:3	Starting angle	INTEGER_16

- The current phase currents can be read via the following parameters:

Object		Info	Data type
Axis A	Axis B		
0x2D83:1	0x3583:1	Motor current zero system	INTEGER_32
0x2D83:2	0x3583:2	Motor current phase U	INTEGER_32
0x2D83:3	0x3583:3	Motor current phase V	INTEGER_32
0x2D83:4	0x3583:4	Motor current phase W	INTEGER_32
Greyed out = read access only			

Advantages compared to the manual test mode "voltage/frequency":

- The current does not set freely, but is controlled to a defined value.
- If a synchronous motor is connected, the generated torque can be predicted.

Preconditions for the execution

- The motor must be able to rotate freely.
- The i700 servo inverters is free of errors and is in the "Switched on" device status.

Response of the motor during the execution

The motor rotates as a function of the set output frequency.

Stop!

In the case of the synchronous motor, a jerky compensating movement occurs after controller enable if the pole position of this movement does not correspond to the starting angle.

How to activate the manual test mode "current/frequency":

1. If the i700 servo inverters is enabled, disable the i700 servo inverters.
   ▸ Enable/inhibit operation via control word (📖 69)
2. Set parameter 0x2825 (or 0x3025 for axis B) to "2" to change to the "Current/frequency" test mode.
3. Enable the i700 servo inverters to start the test mode.
4. To stop the test mode:
   - Disable the i700 servo inverters.
   - Set parameter 0x2825 (or 0x3025 for axis B) to "0" to change back to the CiA402 mode.

5.5 Manual control

As an alternative wiring check, the "Manual control" mode can be activated.

Functional description

The "manual control" modes enables manual operation of the i700 servo inverters via the Engineering tools »EASY Starter« and »PLC Designer«.

- In the manual control mode both the current and the frequency are run to the set final value via a parameterisable ramp time. The ramp times constitute the time span during which the respective parameter is run from zero to the final value.
- In the manual control mode holding brakes, if any, are automatically released. ▶ Holding brake control (📖 205)

Note!

Controlled synchronous motors (SM) can only be traversed under various restrictions:

Restriction 1: The speed variation per control cycle may only be very minor.

- Either make minor changes/leaps of the "Manual jog: Frequency" parameter or select the "Manual jog: Ramp time - frequency" very high.

Restriction 2: The "Manual jog: Frequency" parameter must be so low that the voltage range is not left or that the field weakening range is not reached.

- Provide a reserve of approximately 20 % towards the voltage limit. The voltage requirement is proportional to the speed and thus proportional to the frequency; the voltage limit is determined by the DC-bus voltage.
- For 400-V motors in combination with a 400-V mains, "voltage range" roughly describes the range below the rated motor speed.

Restriction 3: The "Manual jog: Setpoint current" parameter must be selected so high that the friction and load torques of the machine at final speed (= proportional to the parameter "Manual jog: Frequency") can be overcome.

In case of devices that correspond to the "Dual Use Regulation" (EC 428/2009), values of + 599 Hz to + 1000 Hz or - 599 Hz to - 1000 Hz do not cause an increase of the output frequency. Please observe the dead band that occurs in this case.

More information can be found in the section entitled Frequency and speed limitations (📖 204).

Comparison of the test mode "current/frequency" and the "manual control" mode

The following table shows the differences between the two modes:

<table>
<tr><th colspan="3">Test mode "Current/frequency"</th><th colspan="3">Mode "Manual control"</th></tr>
<tr><td colspan="3">Current setpoint is pending immediately after controller enable.
• 100 % ≡ rated motor current (0x6075 or 0x6875 for axis B)</td><td colspan="3">Current setpoint is run to the final value via a parameterisable ramp time after controller enable.
• 100 % ≡ rated axis current (0x2DDF:1 or 0x35DF:1 for axis B)</td></tr>
<tr><td>Axis A</td><td>Axis B</td><td>Parameters</td><td>Axis A</td><td>Axis B</td><td>Parameters</td></tr>
<tr><td>0x2835:1</td><td>0x3035:1</td><td>Current setpoint</td><td>0x2836:1</td><td>0x3036:1</td><td>Current setpoint (final value)</td></tr>
<tr><td>-</td><td>-</td><td>(No ramp can be set)</td><td>0x2836:3</td><td>0x3036:3</td><td>Ramp time</td></tr>
<tr><td colspan="3">Test frequency is pending immediately after controller enable.</td><td colspan="3">Test frequency is run up to the final value via a parameterisable ramp time after controller enable (and is also run down again). The frequency ramp starts when the current setpoint has reached its final value.</td></tr>
<tr><td>Axis A</td><td>Axis B</td><td>Parameters</td><td>Axis A</td><td>Axis B</td><td>Parameters</td></tr>
<tr><td>0x2835:2</td><td>0x3035:2</td><td>Test frequency</td><td>0x2836:2</td><td>0x3036:2</td><td>Test frequency (final value)</td></tr>
<tr><td>-</td><td>-</td><td>(No ramp can be set)</td><td>0x2836:4</td><td>0x3036:4</td><td>Ramp time</td></tr>
<tr><td colspan="3">Starting angle is pending immediately after controller enable or the rotating field starts with the starting angle.
• In the case of the synchronous motor, a jerky compensating movement occurs once if the starting angle does not correspond to the current pole position of the synchronous motor.</td><td colspan="3" rowspan="3">Starting angle is the current commutation angle.
• For the synchronous motor, therefore no compensating movement should be effected.</td></tr>
<tr><td>Axis A</td><td>Axis B</td><td>Parameters</td></tr>
<tr><td>0x2835:3</td><td>0x3035:3</td><td>Starting angle</td></tr>
<tr><td colspan="3" rowspan="3">Duration unlimited.
(If permitted by the device utilisation and motor temperature.)</td><td colspan="3">Duration adjustable.
The time starts with the start of the frequency ramp (when the current setpoint has run up to its final value).
• If the test frequency is not rewritten within the time set for the time monitoring, the frequency is decreased to zero via the parameterised ramp.
• When 0 Hz have been reached, the i700 servo inverters changes to the error status and is disabled. After the error has been acknowledged, the "Switch on disabled" device status must be changed to "Operation enabled" again before manual control can be continued. ▶ Enable/inhibit operation via control word</td></tr>
<tr><td>Axis A</td><td>Axis B</td><td>Parameters</td></tr>
<tr><td>0x2836:5</td><td>0x3036:5</td><td>Time span for time monitoring</td></tr>
</table>

Preconditions for the execution

- The motor must be able to rotate freely.
- The i700 servo inverters is free of errors and is in the "Switched on" device status.

Response of the motor during the execution

The motor rotates according to the manual jog commands.

How to activate manual control

1. If the i700 servo inverters is enabled, disable the i700 servo inverters.
   ▸ Enable/inhibit operation via control word (📖 69)
2. Set object 0x2825 (or 0x3025 for axis B) to "4" to change to the "Manual control" mode.
3. Enable the i700 servo inverters to start the manual control.
   - In the »EASY Starter«, the i700 servo inverters can be enabled via the operator dialog for manual control. The operator dialog also takes priority over the control word (0x6040 or 0x6840 for axis B).
   - The possible settings for the manual control mode are described in the following.
4. To stop the manual control:
   - Disable the i700 servo inverters.
   - Set object 0x2825 (or 0x3025 for axis B) to "0" to change back to the CiA402 mode.

0x2836 | 0x3036 - Manual jog: Settings

Sub.	Name	Lenze setting	Data type
1	Manual jog: Current setpoint	30 %	UNSIGNED_16
2	Manual jog: Frequency	0.0 Hz	INTEGER_16
3	Manual jog: Ramp time - current	0 ms	UNSIGNED_16
4	Manual jog: Ramp time - frequency	500 ms	UNSIGNED_16
5	Manual jog: Time monitoring	2500 ms	UNSIGNED_32
6	Manual jog: Current controller - gain	20.00 V/A	UNSIGNED_32
7	Manual jog: Current controller - reset time	20.00 ms	UNSIGNED_32

Subindex 1: **Manual jog: Setpoint current**				
Selection of the current setpoint for the manual control • 100 % ≡ rated axis current (0x2DDF:1 or 0x35DF:1 for axis B)				
Setting range (min. value | unit | max. value)			Lenze setting	
0	%	200	30 %	
☑ Write access ☐ CINH ☐ OSC ☐ P ☑ RX ☐ TX				UNSIGNED_16

Subindex 2: **Manual jog: Frequency**				
Selection of the frequency for manual control Please observe the notes regarding the limitation of the output frequency in the chapters • Manual jog • Limitation of the output frequency				
Setting range (min. value | unit | max. value)			Lenze setting	
-1000.0	Hz	1000.0	0.0 Hz	
☑ Write access ☐ CINH ☑ OSC ☐ P ☑ RX ☐ TX			Scaling: 1/10	INTEGER_16

Subindex 3: **Manual jog: Ramp time - current**				
Time span during which the current setpoint is run from zero to the final value set.				
Setting range (min. value | unit | max. value)			Lenze setting	
0	ms	1000	0 ms	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 4: **Manual jog: Ramp time - frequency**				
Time span during which the frequency is run from zero to the final value set.				
Setting range (min. value | unit | max. value)			Lenze setting	
0	ms	10000	500 ms	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 5: **Manual jog: Time monitoring**				
Time span for time monitoring • The manual control features a time monitoring function which is coupled to a write access to subindex 2 (Manual jog: Frequency). • If no write access to the subindex 2 takes place within the time period set here, the frequency is lead to zero via the parameterised ramp. When the 0 Hz have been reached, the servo inverter changes to the error status and is inhibited. After acknowledging the error, switch the CiA402 state machine from the "switch on inhibited" back to the "operation enabled" status before proceeding with manual jog.				
Setting range (min. value | unit | max. value)			Lenze setting	
0	ms	100000	2500 ms	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_32

Subindex 6: **Manual jog: Current controller gain**				
Setting range (min. value | unit | max. value)			Lenze setting	
0.00	V/A	750.00	20.00 V/A	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/100	UNSIGNED_32

Subindex 7: **Manual jog: Current controller reset time**				
Setting range (min. value | unit | max. value)			Lenze setting	
0.01	ms	2000.00	20.00 ms	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/100	UNSIGNED_32

5.6 Setting the control mode

The i700 servo inverters supports different modes for open-loop/closed-loop motor control:

- Servo control for synchronous motor (SM)
- Servo control for asynchronous motor (ASM)
- V/f characteristic control for asynchronous motor (ASM)

Servo control

The field-oriented servo control is based on a decoupled, separated control of the torque-producing and field-producing current component. The motor control is based on a fed back, field-oriented and cascaded controller structure and enables dynamic and stable operation in all four quadrants. It can be used for synchronous motors (SM) and asynchronous motors (ASM).

V/f characteristic control

The V/f characteristic control is a motor control mode for typical frequency inverter applications on the basis of a simple and robust control mode for the operation of asynchronous motors with linear or square-law load torque characteristics (e.g. fans). This motor control mode is also suitable for group drives and special motors.

0x2C00 | 0x3400 - Motor control

Setting of the mode for open-loop/closed loop motor control

Selection list(Lenze setting printed in bold)		Info	
1	**Servo control - synchronous motor (SM)**	Servo control for synchronous motor (SM)	
2	Servo control - asynchronous motor (ASM)	Servo control for asynchronous motor (ASM)	
6	VFC: V/f characteristic control	V/f characteristic control for asynchronous motor (ASM)	
☑ Write access ☑ CINH ☐ OSC ☑ P ☐ RX ☐ TX			UNSIGNED_8

5.7 Accepting/adapting plant parameters

The "plant parameters" summarise all parameters which result from the combination of motor and load. These characterise the transfer behaviour of the entire controlled system including the required monitoring modes.

- The plant parameters depend on the application in which the i700 servo inverters and motor are used.
- When a Lenze motor is selected, respective plant parameters are suggested for the load-free operation. This enables the motor to be moved without any load. During operation with a load it is required to change the plant parameters according to the load. Details on how to determine the adapted plant parameters are described in the corresponding objects.

Overview of plant parameters

Object		Info	Lenze setting		Control	
Axis A	Axis B		Value	Unit	Servo	V/f
0x6080	0x6880	Max motor speed	6075	r/min	●	●
0x6073	0x6873	Max current	150	%	●	●
0x2900:1	0x3100:1	Speed controller: Gain	0.00038	Nm/rpm	●	
0x2900:2	0x3100:2	Speed controller: Reset time	15.0	ms	●	
0x2900:3	0x3100:3	Speed controller: Rate time	0.00	ms	●	
0x2904	0x3104	Speed: Actual speed - filter time	0.6	ms	●	●
0x2D44:1	0x3544:1	Motor speed monitoring: Threshold	8000	r/min	●	●

5.8 Compensating for inverter influence on output voltage

Note!

In the V/f characteristic control mode the procedure described below is recommended because a well-adapted inverter characteristic leads to a significantly improved drive behaviour during V/f operation.

Functional description

An inverter generates a three-phase voltage system with pulse width modulation. Inherent to its functional principle, the inverter also generates current-dependent and switching frequency-dependent losses which influence the output voltage. The motor voltage actually provided at the output terminals is not measured in order to compensate for deviations. An adjustment is made by means of the inverter characteristic to compensate for deviations.

Among other things, the inverter characteristic depends on the length of the motor cable and at least has to be individually determined once for the connected motor by means of the device command "Calculate inv. characteristic". For an automatic determination of the motor parameters, this ensures that the current has a sinusoidal form.

Danger!

This procedure may only be carried out during commissioning, not during operation!

During the procedure the motor is energised so that:

- it cannot be excluded that the connected mechanical components may move!
- the windings heat up.
  If you repeat the procedure, ensure that the motor is not thermally overloaded (particularly if no temperature feedback is used).

Identification of the inverter characteristic - procedure

If no error is pending, the motor is energised during the procedure with a maximum DC current corresponding to the lower of the two following values:

$$\sqrt{2} \cdot \text{Rated device current}$$

or

$$\sqrt{2} \cdot 1.8 \cdot \text{Rated motor current}$$

- Ideally, the first value should be reached, the second value is to ensure that the load on the motor is not too high during this test.
- During the procedure, the motor current increases to the maximum value specified and falls back to "0" to repeat the cycle with a negative current sign. Altogether, the maximum value is reached four times.
- The switching frequency of the inverter is set to rated switching frequency and after the procedure, it is reset to the original value.
  - If the switching frequency should be changed later during operation, the characteristic will automatically be adapted to the current switching frequency.

Preconditions for the execution

- The motor may be firmly braked.
- The i700 servo inverters is free of errors and is in the "Switched on" device status.

Response of the motor during the execution

If the motor is not firmly braked, it will move slightly.

How to determine the inverter characteristic:

1. If the i700 servo inverters is enabled, disable the i700 servo inverters.
   ▶ Enable/inhibit operation via control word (📖 69)
2. Set object 0x2825 (or 0x3025 for axis B) to "8" to change to the "Inverter characteristic: Identification" operating mode.
3. Enable the i700 servo inverters to start the procedure.

   Note: By means of controller inhibit, the started procedure can be cancelled any time. Characteristic values that have already been determined are rejected in this case.

After successful completion...

...the controller will be inhibited automatically and the points of the determined inverter characteristic will be set in object 0x2947 (or 0x3147 for axis B).

- For permanent storage, the changed settings must be uploaded to the controller from the i700 servo inverters.

  The »EASY Starter« serves to save the parameter settings of the i700 servo inverters as parameter file (*.gdc). In the »PLC Designer«, this file can then be imported in the corresponding axis. ▶ Saving changed parameters safe against mains failure (📖 70)
- The inverter characteristic must only be detected again if the i700 servo inverters, motor or motor cable has changed e.g. due to an exchange.
- The controller inhibit automatically set by the procedure can be deactivated via the Controlword (0x6040 or 0x6840 for axis B) (setting = 7, 15).

In the event of an error

If an error occurs during the procedure or the pulse inhibit gets active (e.g. due to short-time undervoltage), the procedure is terminated with controller inhibit without a change in settings.

Advanced settings

For characteristic detection, the current controller is parameterised automatically at the beginning of the identification process. For motors with a very low stator leakage inductance (< 1 mH), an automatic parameter setting may fail and the actual identification process is aborted with an error message like e.g. "short circuit".

- For such a case, it is possible to set the current controller manually via the object 0x2942 (or 0x3142 for axis B).
- Whether the current controller is to be selected automatically or the values below 0x2942 (or 0x3142 for axis B) are active, is selected via the object 0x2DE0:1 (or 0x35E0:1 for axis B).

0x2DE0 | 0x35E0 - Advanced settings

Sub.	Name	Lenze setting	Data type
1	Current controller: Setting for identification	0: Automatic setting	UNSIGNED_8
2	Sensorless synchronous control: Signal for test mode	0: Off	UNSIGNED_8
3	Resolver: Position detection - dynamics	100 %	UNSIGNED_16
4	Resolver: 8 kHz safety signal	0: Automatically by device type	UNSIGNED_8
5	Torque: Internal interpolation	1: On	UNSIGNED_8
6	OEM service data: Serial number	0	UNSIGNED_32
14	Overwrite CiA control word bit 4		UNSIGNED_8

Subindex 1: **Current controller: Setting for identification**			
Selection list(Lenze setting printed in bold)			
0	**Automatic setting**		
1	Manual setting (0x2942/0x3142)		
☑ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			UNSIGNED_8

Subindex 2: **Sensorless synchronous control: Signal for the test mode**			
Selection list(Lenze setting printed in bold)			
0	**Off**		
1	On		
☑ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			UNSIGNED_8

Subindex 3: **Resolver: Position detection - dynamics**				
Setting range (min. value | unit | max. value)			Lenze setting	
20	%	1000	100 %	
☑ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 4: **Resolver: 8 kHz safety signal**			
Selection list(Lenze setting printed in bold)			
0	**Automatically by device type**		
1	On		
2	Off		
☑ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			UNSIGNED_8

Subindex 5: **Torque: Internal interpolation**			
Selection list(Lenze setting printed in bold)			
0	Off		
1	**On**		
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			UNSIGNED_8

Subindex 6: **OEM service data: Serial number**				
Display range (min. value | unit | max. value)			Initialisation	
0		4294967295	0	
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX				UNSIGNED_32

Subindex 14: **Overwrite CiA control word bit 4**			
Selection list(Lenze setting printed in bold)			
0	Do not overwrite		
1	Overwrite with FALSE		
2	Overwrite with TRUE		
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			UNSIGNED_8

Loading the standard inverter characteristic

If an incorrect inverter characteristic has been determined or none at all, it is possible to load a device-typical standard inverter characteristic.

How to load the standard inverter characteristic:

1. Set object 0x2822 (or 0x3022 for axis B) to "1".
   - The progress of the procedure is shown in object 0x2823 (or 0x3023 for axis B).
2. For permanent storage: After completion of the procedure, upload the inverter characteristic set in 0x2947 (or 0x3147 for axis B) to the controller from the i700 servo inverters.

   The »EASY Starter« serves to save the parameter settings of the i700 servo inverters as parameter file (*.gdc). In the »PLC Designer«, this file can then be imported in the corresponding axis. ▶ Saving changed parameters safe against mains failure (📖 70)

0x2947 | 0x3147 - Inverter characteristic: Voltage grid points (y)

y values of the 17 grid points of the inverter characteristic

Sub.	Name	Lenze setting	Data type
1	IC: y1 = U01 (x = 0.00 %)	0.00 V	UNSIGNED_16
2	IC: y2 = U02 (x = 6.25 %)	0.00 V	UNSIGNED_16
3	IC: y3 = U03 (x = 12.50 %)	0.00 V	UNSIGNED_16
4	IC: y4 = U04 (x = 18.75 %)	0.00 V	UNSIGNED_16
5	IC: y5 = U05 (x = 25.00 %)	0.00 V	UNSIGNED_16
6	IC: y6 = U06 (x = 31.25 %)	0.00 V	UNSIGNED_16
7	IC: y7 = U07 (x = 37.50 %)	0.00 V	UNSIGNED_16
8	IC: y8 = U08 (x = 42.75 %)	0.00 V	UNSIGNED_16
9	IC: y9 = U09 (x = 50.00 %)	0.00 V	UNSIGNED_16
10	IC: y10 = U10 (x = 56.25 %)	0.00 V	UNSIGNED_16
11	IC: y11 = U11 (x = 62.50 %)	0.00 V	UNSIGNED_16
12	IC: y12 = U12 (x = 68.75 %)	0.00 V	UNSIGNED_16
13	IC: y13 = U13 (x = 75.00 %)	0.00 V	UNSIGNED_16
14	IC: y14 = U14 (x = 81.25 %)	0.00 V	UNSIGNED_16
15	IC: y15 = U15 (x = 87.50 %)	0.00 V	UNSIGNED_16
16	IC: y16 = U16 (x = 93.25 %)	0.00 V	UNSIGNED_16
17	IC: y17 = U17 (x = 100.00 %)	0.00 V	UNSIGNED_16
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			

5.9 Setting the motor parameters for the servo control

If the servo control (SM, ASM) is applied, it is required to parameterise the so-called motor model first. These are electrical variables which are provided in the i700 servo inverters as parameters.

5.9.1 Lenze motor: Easy loading of motor data from the catalogue via the »PLC-Designer«

For Lenze motors with a standard feedback system installed by Lenze, the required motor parameters are provided by the »PLC Designer« during the motor selection by means of catalogues and automatically copied to the i700 servo inverters in the default setting.

In this case, you can directly go to the next commissioning step:
▸ Setting the feedback system for the servo control

5.9.2 Motors of other manufacturers

For motors of other manufacturers or if there are no catalogue data available for the motor, three alternative possibilities for setting the motor parameters are provided:

A. Enter motor nameplate data
(and have motor model parameters determined automatically)

B. Set motor parameters manually

C. Determine motor parameters automatically via "motor parameter identification"

The following subchapters provide detailed information on the three alternative methods. The Engineering tool saves the used method to the following object so that the suitable parameterisation dialogs are available the next time an online connection is present.

0x2C08 | 0x3408 - motor: Setting method - motor parameters

Selection list(Lenze setting printed in bold)		Info
1	**Select from catalogue (Lenze motor)**	For Lenze motors with a standard feedback system installed by Lenze, the required motor parameters are provided by the »PLC Designer« during the motor selection by means of catalogues and automatically copied to the i700 servo inverter in the default setting.
2	Enter motor nameplate data (other motors)	▸ Enter motor nameplate data (and have motor model parameters determined automatically)
3	Manually (other motors)	▸ Set motor parameters manually
4	Identification in progress (all motors)	▸ Determine motor parameters automatically via "motor parameter identification"
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX		UNSIGNED_8

5.9.2.1 Enter motor nameplate data

If the equivalent circuit data of the motor required for the motor model are not known, they can automatically be determined by approximation by the i700 servo inverters by means of the motor nameplate data before they are set.

How to have the equivalent circuit data determined by the i700 servo inverters:

1. Set the complete motor nameplate data in object 0x2C01 (or 0x3401 for axis).
2. Set the rated motor current in object 0x6075 (or 0x6875 for axis B).
3. Set object 0x2822 (or 0x3022 for axis B) to "3" in order to have the equivalent circuit data determined by approximation.
   - The progress of the procedure is shown in object 0x2823 (or 0x3023 for axis B)

Note!

This axis command serves to derive the scaled warning and switch-off thresholds of the motor temperature monitoring from the parameterised insulation class (0x2C01:9 or 0x3401:9). There are motor/device combinations, where the warning and switch-off thresholds (0x2D49:3 / 0x3549:3 or 0x2D49:4 / 0x3549:4) are assigned to a lower insulation class (e.g. winding design in insulation class H, switch-off temperature 155 °C).

In this case, a manual parameterisation is required at the end of the motor commissioning.

4. For permanent storage: Upload the set equivalent circuit data to the controller from the i700 servo inverters after the procedure has been completed:
   - For asynchronous motor (ASM): Object 0x2C02 (or 0x3402 for axis B).
   - For synchronous motor (SM): Object 0x2C03 (or 0x3403 for axis B).

   The »EASY Starter« serves to save the parameter settings of the i700 servo inverters as parameter file (*.gdc). In the »PLC Designer«, this file can then be imported in the corresponding axis. ▶ Saving changed parameters safe against mains failure (📖 70)

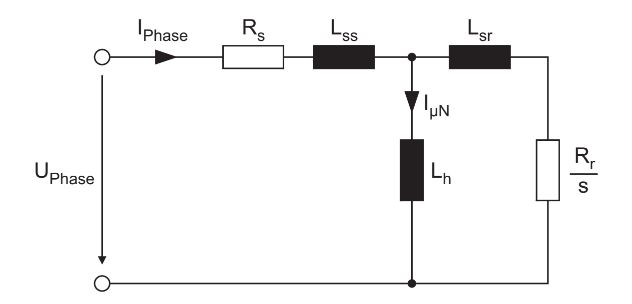
5.9.2.2 Set motor parameters manually

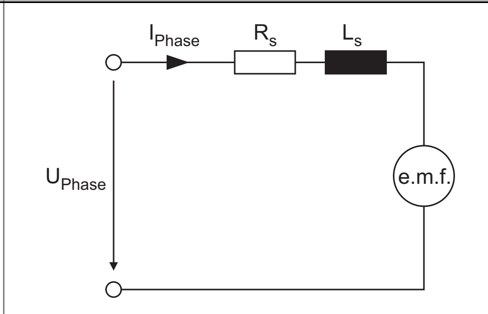
If all required motor data are known (e.g. by means of a data sheet provided by the motor manufacturer), they can be set manually in the following parameters:

Object		Name	Required for	
Axis A	Axis B		SM	ASM
0x2C01:1	0x3401:1	Motor: Number of pole pairs	●	●
0x2C01:2	0x3401:2	Motor: Stator resistance (value at 20°C)	●	●
0x2C01:3	0x3401:3	Motor: Stator leakage inductance	●	●
0x2C01:4	0x3401:4	Motor: Rated speed	●	●
0x2C01:5	0x3401:5	Motor: Rated frequency	●	●
0x2C01:6	0x3401:6	Motor: Rated power	●	●
0x2C01:7	0x3401:7	Motor: Rated voltage	●	●
0x2C01:8	0x3401:8	Motor: Rated cosine phi	●	●
0x2C01:9	0x3401:9	Motor: Insulation class	●	●
0x2C02:1	0x3402:1	Motor (ASM): Rotor resistance (value at 20 °C)		●
0x2C02:2	0x3402:2	Motor (ASM): Mutual inductance		●
0x2C02:3	0x3402:3	Motor (ASM): Magnetising current		●
0x2C03:1	0x3403:1	Motor (SM): e.m.f. constant (KELL, line-to-line, value is 20° C)	●	
0x2C03:2	0x3403:2	Motor (SM): Pole position	●	
0x2C03:3	0x3403:3	Motor (SM): Temperature coefficient - magnets	●	
0x6075	0x6875	Motor rated current	●	●
0x6076	0x6876	Motor rated torque	●	●
Greyed out = read access only				

Motor equivalent circuit (star connection)

Async. motor	Synchronous motor



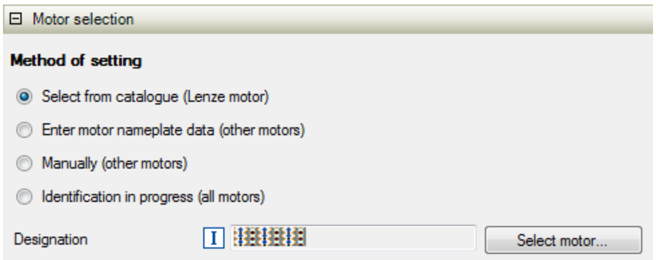


Axis A	Axis B	Symbol	Description	Dimension unit
0x2C01:2	0x3401:2	R_s	Motor stator resistance (value at 20°C)	Ω
0x2C01:3	0x3401:3	L_{ss}	Motor stator leakage inductance (ASM)	H
		L_s	Motor stator inductance (SM)	H
-	-	L_{sr}	Motor rotor leakage inductance (is assumed to be equal to L_{ss})	H
0x2C02:1	0x3402:1	R_r	Motor rotor resistance (value at 20°C)	Ω
0x2C02:2	0x3402:2	L_h	Motor magnetising inductance	H
0x2C02:3	0x3402:3	$I_{\mu N}$	Motor magnetising current	A
-	-	s	Slip	
-	-	e.m.f.	Electromotive force	

Motor with modifiable connection type (star to delta)

In general, a motor has different rated operating points as a function of its connection type. Thus, this dependency must be considered when the associated objects of the i700 servo inverter are entered.

A Lenze motor with a modifiable connection type usually has two nameplates. For parameter setting from the Lenze motor catalogue, the nameplate suitable for the connection type is used with the code C86 given there. All motor data are then automatically entered correctly in the corresponding objects according to the connection type.



If you click "**Select motor** in the shown dialog that is identical in the Lenze Engineering Tool »PLC Designer« or »EASY Starter, a table opens where you can enter the value given on the nameplate under "C86", e.g.:

- C86 = **1651** for the motor in star connection
- C86 = **1652** for the same motor in delta connection.

If it is a motor of another manufacturer, the data must be entered manually.

Usually the motor manufacturer provides the data for a phase, i.e. a winding. Thus, they are independent of the connection type. The data sheet contains at least stator resistance, rotor resistance, leakage inductance, mutual inductance and magnetising current.

As the i700 servo inverter operates with a motor model in ⅄ connection, the winding data must be converted for a motor in △ connection:

Object		**Data in**	**Conversion for the entry into the object**
Axis A	Axis B	**Motor data sheet**	**for a motor in △ connection**
0x2C01:2	0x3401:2	Stator resistance R_s	$R_s / 3$
0x2C01:3	0x3401:3	Stator leakage inductance L_s	$L_s / 3$
0x2C02:1	0x3402:1	Rotor resistance R_r	$R_r / 3$
0x2C02:2	0x3402:2	Mutual inductance L_h	$L_h / 3$
0x2C02:3	0x3402:3	Magnetising current I_μ	$I_\mu \cdot \sqrt{3}$

5.9.2.3 Determine motor parameters automatically via "motor parameter identification"

Danger!

This procedure may only be carried out during commissioning, not during operation!

- During the procedure the motor is energised so that:
  - it cannot be excluded that the connected mechanical components may move!
  - the windings heat up.
    If you repeat the procedure, ensure that the motor is not thermally overloaded (particularly if no temperature feedback is used).

Functional description

The motor parameters listed in the table below can be determined automatically via the motor parameter identification function if they are not known. The resistance values are converted via the actual motor temperature into values that correspond to a temperature of 20°C. If a thermal detector is not connected, a temperature value of 90°C is assumed.

Object		Name	Required for	
Axis A	Axis B		SM	ASM
0x2C01:2	0x3401:2	Motor: Stator resistance (value at 20°C)	●	●
0x2C01:3	0x3401:3	Motor: Stator leakage inductance (L_{ss})	●	●
0x2C02:1	0x3402:1	Motor (ASM): Rotor resistance (value at 20 °C)		●
0x2C02:2	0x3402:2	Motor (ASM): Mutual inductance (L_h)		●
0x2C02:3	0x3402:3	Motor (ASM): Magnetising current		●

Sequence of the motor parameter identification

The apparent resistance of the plant is determined for approx. 30 different frequencies. Then a mathematical procedure is used to extract the electrical parameters of the motor.

- Since the procedure starts with very low frequencies and always considers several complete periods, the whole process takes approx. 3 minutes.
- During the procedure, the motor is energised with a current, the r.m.s. value of which corresponds to the lower of the following two values:

Rated device current

or

$$\frac{1}{2} \cdot \text{Rated motor current}$$

Tip!

If an asynchronous motor is to be identified, the identification should be executed with half the rated motor current in order to achieve an optimal result. For this purpose, the rated device current has to be higher than half the rated motor current. In case of a synchronous motor, this is irrelevant.

After the parameters have been extracted from the impedance, they are checked for consistency with the required rated values. If an inconsistent parameter set is detected, is this an indication of faulty rated values on the nameplate.

Preconditions for the execution

- The synchronous motor must be able to rotate freely.
- The asynchronous motor may be firmly braked.
- The i700 servo inverters is free of errors and is in the "Switched on" device status.
- The motor parameters listed in the following table are excluded from the automatic determination and must therefore be adapted to the motor used (see motor nameplate before the determination.

Object		Name
Axis A	Axis B	
0x6075	0x6875	Motor rated current (The current amount for the procedure is derived from this specification)
0x2C01:2	0x3401:2	Motor: Stator resistance (Default setting is used as starting value for the automatic determination.)
0x2C01:4	0x3401:4	Motor: Rated speed
0x2C01:5	0x3401:5	Motor: Rated frequency
0x2C01:6	0x3401:6	Motor: Rated power
0x2C01:7	0x3401:7	Motor: Rated voltage

Response of the motor during the execution

- A DC current is superimposed over the identification current that keeps the motor idling. After the controller enable, the shaft will adjust once, which is irrelevant to measurement though.
- With asynchronous motors, slight rotations might possibly occur. Their influence on the measurements is, however, not worth mentioning.

Note!

- In case of uncertainties, the measurement should be repeated several times to check if the results for the stator resistance, the leakage inductance of the stator and the rotor resistance differ widely. This should not be the case.
- The mutual inductance and the cos(φ) values are not that important for the diagnostics, because they are strongly non-linear.

How to carry out the motor parameter identification:

1. If the i700 servo inverters is enabled, disable the i700 servo inverters.
   ▸ Enable/inhibit operation via control word (📖 69)
2. Set object 0x2825 (or 0x3025 for axis B) to "9" to change to the "Motor: Parameter identification" operating mode.
3. Enable the i700 servo inverters to start the procedure.

   Note: By means of controller inhibit, the procedure started can be cancelled any time, if required, without a change in settings.

After successful completion...

...the controller is automatically inhibited and the determined motor data are set in the corresponding objects (0x2C01 and 0x2C02 or 0x3401 and 0x3402 for axis B).

- For permanent storage, the changed settings must be uploaded to the controller from the i700 servo inverters.

  The »EASY Starter« serves to save the parameter settings of the i700 servo inverters as parameter file (*.gdc). In the »PLC Designer«, this file can then be imported in the corresponding axis. ▸ Saving changed parameters safe against mains failure (📖 70)
- The controller inhibit automatically set by the procedure can be deactivated via the Controlword (0x6040 or 0x6840 for axis B) (setting = 7, 15).

In the event of an error

If an error occurs during the procedure or the pulse inhibit gets active (e.g. due to short-time undervoltage), the procedure is terminated with controller inhibit without a change in settings.

Advanced settings

For motor parameter detection, the current controller is parameterised automatically at the beginning of the identification process. For motors with a very low stator leakage inductance (< 1 mH), an automatic parameter setting may fail and the actual identification process is aborted with an error message like e.g. "short circuit".

- For such a case, it is possible to set the current controller manually via the object 0x2942 (or 0x3142 for axis B).
- Whether the current controller is to be selected automatically or the values below 0x2942 (or 0x3142 for axis B) are active, is selected via the object 0x2DE0:1 (or 0x35E0:1 for axis B).

5.9.3 Motor parameters (object descriptions)

Observe the notes regarding a motor with modified connection (⅄ → △)

▶ Motor with modifiable connection type (star to delta) (📖 103)

0x2C01 | 0x3401 - Motor: Common parameters

Sub.	Name	Lenze setting	Data type
1	Motor: Number of pole pairs		UNSIGNED_8
2	Motor: Stator resistance (20°C)	13.5000 ohms	UNSIGNED_32
3	Motor: Stator leakage inductance	51.000 mH	UNSIGNED_32
4	Motor: Rated speed	4050 rpm	UNSIGNED_16
5	Motor: Rated frequency	270.0 Hz	UNSIGNED_16
6	Motor: Rated power	0.25 kW	UNSIGNED_16
7	Motor: Rated voltage	225 V	UNSIGNED_16
8	Motor: Rated cosine phi	0.80	UNSIGNED_16
9	Motor: Insulation class	4: F (cut-off temperature = 155 °C)	UNSIGNED_8
10	Motor: Designation		STRING(50)

Subindex 1: **Motor: Number of pole pairs**				
Display range (min. value | unit | max. value)			Initialisation	
0		255		
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX				UNSIGNED_8

Subindex 2: **Motor: Stator resistance (20°C)**				
see also motor equivalent circuit Stator resistance of a motor phase (reference: ϑ = 20 °C)				
Setting range (min. value | unit | max. value)			Lenze setting	
0.0000	Ohm	125.0000	13.5000 ohms	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10000	UNSIGNED_32

Subindex 3: **Motor: Stator leakage inductance**				
see also motor equivalent circuit Stator leakage inductance (ASM) or leakage inductance (SM) of a motor phase				
Setting range (min. value | unit | max. value)			Lenze setting	
0.000	mH	500.000	51.000 mH	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/1000	UNSIGNED_32

Subindex 4: **Motor: Rated speed**				
Setting range (min. value | unit | max. value)			Lenze setting	
0	r/min	50000	4050 rpm	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 5: **Motor: Rated frequency**				
Setting range (min. value | unit | max. value)			Lenze setting	
0.0	Hz	1000.0	270.0 Hz	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	UNSIGNED_16

Subindex 6: **Motor: Rated power**				
Setting range (min. value | unit | max. value)			Lenze setting	
0.00	kW	655.35	0.25 kW	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/100	UNSIGNED_16

Subindex 7: **Motor: Rated voltage**				
Setting range (min. value | unit | max. value)			Lenze setting	
0	V	65535	225 V	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 8: **Motor: Rated cosine phi**				
Setting range (min. value | unit | max. value)			Lenze setting	
0.00		655.35	0.80	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/100	UNSIGNED_16

Subindex 9: **Motor: Insulation class**			
see also motor equivalent circuit The axis command 0x2822:0 / 0x3022:0 = 3 serves to derive the scaled warning and switch-off thresholds of the motor temperature monitoring from the parameterised insulation class. Observe the following for applications with third-party and special motors: There may be constellations where warning and switch-off thresholds (0x2D49:3 / 0x3549:3 or 0x2D49:4 / 0x3549:4) correspond to a low insulation class. A manual parameter setting at the end of the motor commissioning is required here. **Example** The winding design corresponds to the insulation class H, the switch-off temperature, however, is already reached at ϑ = 155° C.			
Selection list(Lenze setting printed in bold)			
0	Y (cut-off temperature = 90°C)		
1	A (cut-off temperature = 105 °C)		
2	E (cut-off temperature = 120°C)		
3	B (cut-off temperature = 130°C)		
4	**F (cut-off temperature = 155°C)**		
5	H (cut-off temperature = 180°C)		
6	G (cut-off temperature > 180°C)		
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			UNSIGNED_8

Subindex 10: **Motor: Designation**		
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX		STRING(50)

0x2C02 | 0x3402 - Motor (ASM): Parameter

Sub.	Name	Lenze setting	Data type
1	Motor (ASM): Rotor resistance (20°C)	0.0000 ohms	UNSIGNED_32
2	Motor (ASM): Mutual inductance	0.0 mH	UNSIGNED_32
3	Motor (ASM): Magnetising current	0.00 A	UNSIGNED_16

Subindex 1: **Motor (ASM): Rotor resistance (20°C)**				
see also motor equivalent circuit Rotor resistance of a motor phase (reference: ϑ = 20 °C).				
Setting range (min. value | unit | max. value)			Lenze setting	
0.0000	Ohm	214748.3647	0.0000 ohms	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10000	UNSIGNED_32

Subindex 2: **Motor (ASM): Mutual inductance**				
see also motor equivalent circuit Mutual inductance of a motor phase (reference: ϑ = 20 °C).				
Setting range (min. value | unit | max. value)			Lenze setting	
0.0	mH	214748364.7	0.0 mH	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	UNSIGNED_32

Subindex 3: **Motor (ASM): Magnetising current**				
Setting range (min. value | unit | max. value)			Lenze setting	
0.00	A	500.00	0.00 A	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/100	UNSIGNED_16

0x2C03 | 0x3403 - Motor (SM): Parameter

Sub.	Name	Lenze setting	Data type
▸ 1	Motor (SM): e.m.f. constant (KE conductor-conductor, 20°C)	41.8 V/1000 rpm	UNSIGNED_32
▸ 2	Motor (SM): Pole position resolver	-90.0 °	INTEGER_16
▸ 3	Motor (SM): Temperature coefficient - magnets (kTN)	-0.110 %/°C	INTEGER_16
▸ 4	Motor (SM): Pole position encoder	0.0 °	INTEGER_16

Subindex 1: **Motor (SM): e.m.f. constant (KE conductor-conductor, 20°C)**				
For permanently excited synchronous motors, the e.m.f. constant describes the r.m.s. value of the line-to-line voltage (phase voltage) induced in idle state by the motor. Reference: 1000 rpm, 20 °C When third-party motors are used, the data from the motor data sheet may have to be adapted according to the values and units mentioned above.				
Setting range (min. value | unit | max. value)			Lenze setting	
0.0	V/1000 rpm	100000.0	41.8 V/1000 rpm	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	UNSIGNED_32

Subindex 2: **Motor (SM): Pole position resolver**				
Setting range (min. value | unit | max. value)			Lenze setting	
-179.9	°	179.9	-90.0 °	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	INTEGER_16

Subindex 3: **Motor (SM): Temperature coefficient - magnets (kTN)**				
Setting range (min. value | unit | max. value)			Lenze setting	
-1.000	%/°C	0.000	-0.110 %/°C	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/1000	INTEGER_16

Subindex 4: **Motor (SM): Pole position encoder**				
From software version V01.03.xx onwards				
Setting range (min. value | unit | max. value)			Lenze setting	
-179.9	°	179.9	0.0 °	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	INTEGER_16

0x6075 | 0x6875 - Motor rated current

The rated motor current set here serves as a reference value for the following objects:
- 0x6073 | 0x6873 - Device: Max. current
- 0x6078 | 0x6878 - Current actual value
- 0x2835 | 0x3035 - Manual test mode: Settings
- 0x2C61 | 0x3461 - Pole position identification PPI (360°)
- 0x2C62 | 0x3462 - Pole position identification PPI (min. movement)
- 0x2D4D | 0x354D - Motor utilisation (I²xt): User-definable characteristic

Setting range (min. value | unit | max. value)			Lenze setting	
0.001	A	500.000	1.300 A	
☑ Write access ☑ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/1000	UNSIGNED_32

0x6076 | 0x6876 - Motor rated torque

The rated motor torque set here serves as a reference value for the following objects:
- 0x6071 | 0x6872 - Target torque
- 0x6072 | 0x6872 - Max. torque
- 0x6074 | 0x6874 - Torque demand
- 0x6077 | 0x6877 - Torque actual value
- 0x60B2 | 0x68B2 - Torque offset
- 0x60E0 | 0x68E0 - Positive torque limit value
- 0x60E1 | 0x68E1 - Negative torque limit value
- 0x2DD4 | 0x35D4 - Speed controller: Output signal
- 0x2DD5 | 0x35D5 - torque: Target torque
- 0x2DD6 | 0x35D6 - Torque: Filter cascade

Setting range (min. value | unit | max. value)			Lenze setting	
0.001	Nm	1000.000	0.600 Nm	
☑ Write access ☑ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/1000	UNSIGNED_32

5.10 Setting the feedback system for the servo control

After setting the motor parameters, the feedback system for the servo control must be set.

Note!

The feedback system has already been preselected by the hardware of the available device version. Either the objects for resolver evaluation or the objects for encoder evaluation are effective. Access to ineffective objects of hardware not available is ignored.

The following table shows which parameters are valid for which feedback system:

Object Axis A	Object Axis B	Name
General settings		
0x2C45	0x3445	Open circuit in feedback system: Response
0x2C46	0x3446	Feedback system: Specifiable number of revolutions
Settings for "resolver" version		
0x2C03:2	0x3403:2	Motor (SM): Pole position resolver
0x2C43	0x3443	Resolver: Number of pole pairs
Settings for "encoder" version		
0x2C03:4	0x3403:4	Motor (SM): Pole position encoder
0x2C40	0x3440	Encoder: Type
0x2C42:1	0x3442:1	Encoder: Increments / revolution
0x2C42:2	0x3442:2	Encoder: Supply voltage
0x2C42:3	0x3442:3	Encoder: Angle drift - Actual angle error
0x2C42:4	0x3442:4	Encoder: Signal quality - Actual amplitude
Additional settings for SinCos absolute value encoders with HIPERFACE® protocol		
0x2C41:1	0x3441:1	Hiperface: Determined type code
0x2C41:2	0x3441:2	Hiperface: User def. encoder - type code
0x2C41:3	0x3441:3	Hiperface: User def. encoder - specifiable revolutions
0x2C41:4	0x3441:4	Hiperface absolute value fault: Response
0x2C41:5	0x3441:5	Hiperface: Serial number
0x2C41:6	0x3441:6	Hiperface: Raw data - Actual position
0x2C41:7	0x3441:7	Hiperface: Detected Increments / revolution
0x2C41:8	0x3441:8	Hiperface: Type code supported by firmware
0x2C41:9	0x3441:9	Hiperface: Encoder type
0x2C41:10	0x3441:10	Hiperface: Period length linear encoder
Greyed out = read access only		

5.10.1 General settings

Encoder open-circuit monitoring

In the Lenze setting the resolver/encoder cable is monitored for open circuit.

Danger!

If the encoder/resolver is used as motor encoder:
Safe operation of the motor is no longer ensured in the event of an error!

- For safety reasons, always select "Fault" (Lenze setting) as a response for the (open-circuit) monitoring of the encoder/resolver!
- To avoid the injection of interference when an encoder is being used, only use shielded motor and encoder cables!

When does the open-circuit monitoring system respond?

Resolver	Multi encoder
• If there is an open circuit in the encoder cable. • If the impedance of the resolver is too high. • In the case of interference injections (EMC interferences).	• If there is an open circuit in the encoder cable.

0x2C45 | 0x3445 - feedback system: Response to open circuit

If there is an open circuit in the encoder cable, the response set here is triggered.

Selection list(Lenze setting printed in bold)		Info	
0	No response		
1	**Interference**	• In the device statuses "not ready to start" and "switch-on inhibited", a warning is output. • In all other device statuses, the error response set here is executed.	
2	Warning		
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			UNSIGNED_8

Display parameter

0x2C46 | 0x3446 - feedback system: Specifiable number of revolutions

From software version V01.03.xx onwards
The parameter shows the number of revolutions to be distinguished by the feedback system. This part of the position information is available again as absolute actual position after mains switching.
The value range depends on the feedback system used:
- "0" if no absolute value encoder (e.g. sin/cos encoder) or resolver with a number of pole pairs > 1 is used.
- "1" of a singleturn absolute value encoder (e.g. Hiperface sin/cos absolute value encoder, singleturn) or resolver with a number of pole pairs = 1 is used.
- ">1" of a multiturn absolute value encoder (e.g. Hiperface sin/cos absolute value encoder, multiturn) is used.

Display range (min. value | unit | max. value)			Initialisation	
0		65535		
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX				UNSIGNED_16

0x2C5F | 0x345F - Feedback system: Parameter CRC

From software version V01.03.xx onwards

Display range (min. value | unit | max. value)			Initialisation	
0		4294967295		
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX				UNSIGNED_32

5.10.2 Settings for "resolver" version

The "resolver" device version means that a resolver must be used in the feedback.

0x2C43 | 0x3443 - Resolver: Number of pole pairs

Setting range (min. value | unit | max. value)			Lenze setting	
1		10	1	
☑ Write access ☑ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_8

Note!

Resolvers with a number of pole pairs > 1 are no absolute value encoders.
- Thus, bit 4 in the Lenze status word 2 (0x2833 or 0x3033 for axis B) remains set to "0".
- The "distinguishable revolutions" specification in 0x2C46 (or 0x3446 for axis B) is also set to "0".

The following applies to synchronous motors:
- In case of integer ratios of the number of pole pairs of the motor (0x2C01:1 or 0x3401:1 for axis B) to the number of pole pairs of the resolver, the pole position identification is only required once.
- In case of non-integer ratios, a pole position identification has to be executed after every 24-V switching operation of the i700 servo inverters.

▶ Synchronous motor (SM): Pole position identification (📖 130)

Resolver error compensation, resolover error identification

The actual position determined via the resolver does not exactly correspond to the actual physical position. There will always be some greater or lesser deviation due to the following causes:

Cause	Remedy
The inductances of the sine and cosine track of the resolver have slightly different values.	Adaptation of the gains for the digital-analog converters supplying the resolver tracks. • In the Lenze setting, the gains for both resolver tracks are preset identically.
Sine and cosine track do not magnetise orthogonally to each other.	Correction of the angle by means of which the two resolver tracks are supplied in a manner relative to one another.

Thanks to an identification run, the i700 servo inverters enables the resolver compensation to be executed in such a way that the adjustment values for compensation the resolver error are generated automatically and entered in an available parameter.

Preconditions for executing the identification run

- Mechanical connection of the motor / i700 servo inverter
  - Motor and resolver must be connected correctly to the i700 servo inverter (see i700 hardware manual).
  - The motor must be able to mechanically rotate freely.
- Voltage supply of the i700 servo inverter
  - The i700 servo inverter must be supplied with mains voltage. Check: 0x6041 , bit 4 = TRUE.
  - The control electronics must be supplied by an external voltage source.
- Correct setting of the following data in the engineering tool (e.g. »EASY Starter«):
  - Motor data
  - Speed-controlled or position-controlled motor in servo control
  - Number of the resolver pole pairs (0x2C43)
- The Servo-Inverter i700 must be connected "online" with the engineering tool.

Tip!

The Servo Inverter i700 is in the "**Operation enabled**" device status
▸ Operation enabled

If possible, execute the identification run before the motor is installed in the machine. Bigger load changes at the motor can influence the identification result in a negative way.

Possible reactions during the execution

- The motor may be running rough during identification due to the identification method.
- The direction of rotation may change.
  - This has no negative effect on the quality of the identification. In this case, the inverter automatically interrupts the identification run and continues it automatically as soon as a constant speed has been reached again.
- If the motor already installed in the machine does not provide enough range in one direction for executing the identification run, you can reverse the driving direction during the active identification. In this case, the identification automatically switches to the "Identification temporarily interrupted" status. The status is cancelled as soon as a constant speed has been reached again.

Note!

The execution of the identification run requires a software version ≥ V 01.10.

- Servo Inverter i700 with a software version < V 01.10: The identification run is stopped in case of an interruption. An error message is output.

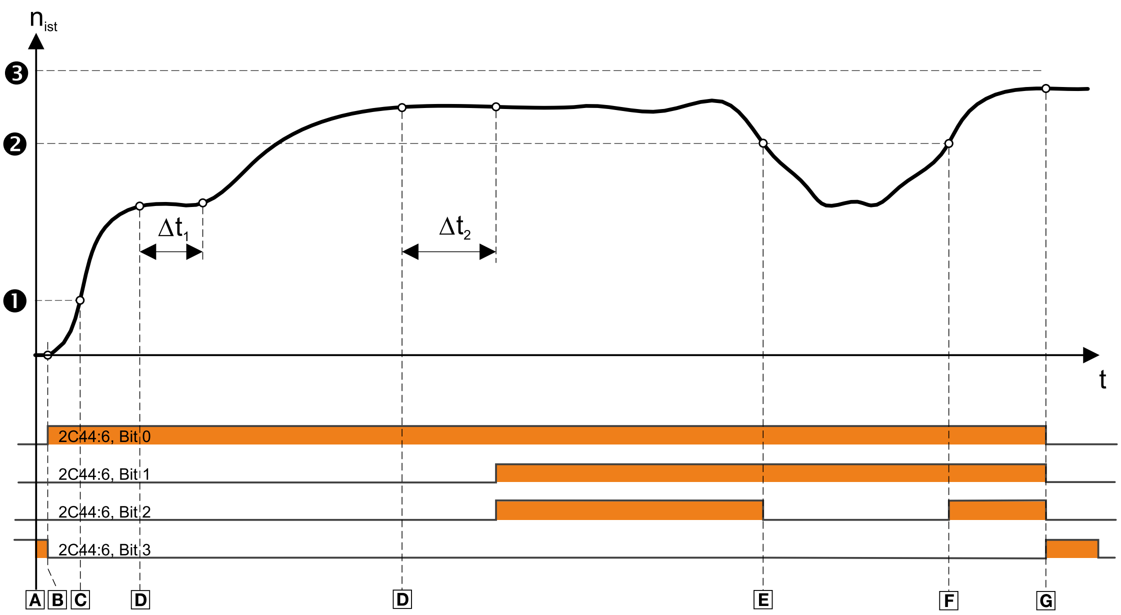
The detected gain can take values between 0 ...100 %.

- With a setting of 0 %, the gain of the corresponding resolver track is only 95 % of the Lenze setting.
- With a sensible resolver error compensation only one of the two gains is adapted. The other remains at 100 %.

How to carry out the identification run

1. Set object 0x2822 (or 0x3022 for axis B) to "9", see graphics ▶ Identification run for resolver compensation.
2. Enable i700 servo inverter.
   - The identification run is in standby mode.
3. Enter a constant speed between n = 500 rpm und n = 3000 rpm.
   - The identification run is automatically started after the drive has reached a constant speed and maintains it over the time defined in 0x2C44:4.
   - This speed is saved for the identification run. In order that the identification run can be continued again, e.g. after an interruption, the drive must be operated again with this speed.
4. During the Identification in progress, the progress of the procedure is shown in the object 0x2823 (or 0x3023 for axis B).
   - After the resolver error identification has been executed successfully, objects 0x2C44:1 ... 3 are automatically written. The resolver now operates with these settings.
   - A short-time interruption, e.g. by deactivating the controller enable, does not stop the measurement. It is continued after a renewed enable. As status message for the duration of the interruption, the message: "Identification temporarily interrupted" appears).
   - The measurement is aborted in case of a longer controller inhibit or after the timeout has elapsed (0x2C44:5). A timeout error is output for the identification run, see error messages in the logbook.
5. In case of a successful measurement, the motor can be stopped.
6. For permanent storage: After the procedure has been completed, upload the parameters changed in 0x2C44:1 ... 3 (or 0x3444:1 ... 3 for axis B) from the i700 servo inverters into the Controller.

   The »EASY Starter« serves to save the parameter settings of the i700 servo inverters as parameter file (*.gdc). In the »PLC Designer«, this file can then be imported in the corresponding axis. ▶ Saving changed parameters safe against mains failure (📖 70)



A Activation of the identification run by 0x2822 = 9
B Controller enable
C Acceleration monitoring starts
D Acceleration = 0 is recognised
E Speed falls below the lower threshold
F Identification is continued
G Successful end of identification
❶ Minimum speed
❷ Lower threshold: 0.8 x search speed
❸ Upper threshold: 1.2 x search speed

Δt_1 < then the time set in 0x2C44:4, i.e.
the monitoring period is aborted and the identification run does not start yet.

Δt_2 > than the time set in 0x2C44:4, i.e.
the search speed is accepted, identification run and progress bar start

[5-1] Identification run for resolver compensation

0x2C44 | 0x3444 - Resolver error compensation: Parameter

Sub.	Name	Lenze setting	Data type
1	Resolver error compensation: Angle	0	INTEGER_16
2	Resolver error compensation: Cosine track gain	100 %	UNSIGNED_16
3	Resolver error compensation: Sine track gain	100 %	UNSIGNED_16
4	Identification: Search time for v = const.	100 ms	UNSIGNED_16
5	Identification: Time-out	300 s	UNSIGNED_16
6	Identification: Actual status		UNSIGNED_16

Subindex 1: **Resolver error compensation: Angle**			
[´] = angular minutes			
Setting range (min. value | unit | max. value)			Lenze setting
-100		100	0
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			INTEGER_16

Subindex 2: **Resolver error compensation: Cosine track gain**			
Setting range (min. value | unit | max. value)			Lenze setting
0	%	100	100 %
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			UNSIGNED_16

Subindex 3: **Resolver error compensation: Sine track gain**			
Setting range (min. value | unit | max. value)			Lenze setting
0	%	100	100 %
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			UNSIGNED_16

Subindex 4: **Identification: Search time for v = const.**			
Setting range (min. value | unit | max. value)			Lenze setting
20	ms	1000	100 ms
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			UNSIGNED_16

Subindex 5: **Identification: Time-out**			
Setting range (min. value | unit | max. value)			Lenze setting
20	s	3600	300 s
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			UNSIGNED_16

Subindex 6: **Identification: Actual status**			
From software version V02.12.xx onwards			
Display range (min. value | unit | max. value)			Initialisation
0		65535	
Value is bit-coded:			Info
Bit 0	Enable identification		
Bit 1	Constant speed detected		
Bit 2	Identification in progress		
Bit 3	Identification complete		
Bit 4	Identification failed		
Bit 5-15	Reserved		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☐ TX			UNSIGNED_16

Deactivating the resolver error compensation

For the deactivation of the resolver error compensation, the corresponding parameters must be reset to the Lenze setting again.

5.10.3 Settings for "encoder" version

The use of an incremental sin/cos encoder is preset. If a sin/cos absolute value encoder with HIPERFACE® protocol is connected instead, select "2" in 0x2C40 (or 0x3440 for axis B) and adapt the encoder parameters (e.g. supply voltage) accordingly.

0x2C40 | 0x3440 - Encoder: Type

Selection list(Lenze setting printed in bold)			
1	**Sin/cos encoder**		
2	Hiperface absolute value encoder		
4	Internal		
☑ Write access ☑ CINH ☐ OSC ☑ P ☐ RX ☐ TX			UNSIGNED_8

0x2C42 | 0x3442 - Encoder: Parameter

Sub.	Name	Lenze setting	Data type
1	Encoder: Increments / revolution	1024	UNSIGNED_32
2	Encoder: Supply voltage	5.0 V	UNSIGNED_8
3	Encoder: Angle drift - Actual angle error		INTEGER_16
4	Encoder: Signal quality - Actual amplitude		UNSIGNED_8

Subindex 1: **Encoder: Increments / revolution**				
Setting range (min. value | unit | max. value)			Lenze setting	
1		131072	1024	
☑ Write access ☑ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_32

Subindex 2: **Encoder: Supply voltage**				
Setting range (min. value | unit | max. value)			Lenze setting	
5.0	V	12.0	5.0 V	
☑ Write access ☑ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	UNSIGNED_8

Subindex 3: **Encoder: Angle drift - Actual angle error**				
From software version V01.03.xx onwards				
Display range (min. value | unit | max. value)			Initialisation	
-3276.8	°	3276.7		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: 1/10	INTEGER_16

Subindex 4: **Encoder: Signal quality - Actual amplitude**				
From software version V01.03.xx onwards The signal quality indicates the actual amplitude of the SinCos analog signals with regard to 1 Vss = 100 %. • In case of higher-order drives, the signal quality should be between 95 % and 105 %. • There is no need for optimisation if the signal quality is within the tolerance zone for the analog encoder signals given in the data sheet of the encoder manufacturer.				
Display range (min. value | unit | max. value)			Initialisation	
0	%	255		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☐ TX				UNSIGNED_8

5.10.4 Evaluation of the signal quality

Determination and display of the current angular drift

The value displayed in the 0x2C42:3 object (or 0x3422:3 for axis B) indicates whether too much or too less pulses, caused by EMC, have been detected by the device-internal counter unit. This value is detected differently, depending on the encoder type:

In case of an incremental sin/cos encoder, the pulses are counted between two zero pulse events of the Z track. In an error-free status, this value corresponds to the set number of increments. The accuracy of this procedure corresponds to a line graduation of the encoder of ± 1. The difference between set number of increments and counted pulses is converted to an angle with an accuracy of ±0.1°. A disadvantage is that only after one complete encoder revolution, an updated value of the angular drift is available and thus the update rate depends on the speed.

In case of a sin/cos absolute value encoder with HIPERFACE® protocol, no Z track is available. Here, instead, the position is regularly read out of the encoder. When the encoder is read out for the first time (after power-up or removal of an open circuit), the encoder position is used to initialise the device-internal counter unit and set a device-internal position. All other read-out processes from the encoder are used to generate a difference between the device-internal position and the encoder position. In an error-free status, the difference is zero. The accuracy of the process, however, is, speed-dependent due to the dead time of the communication with the encoder and thus restricted towards the zero pulse procedure. An advantage here, however, is that the update rate does not depend on the speed but on the communication rate only. The update rate is encoder-specific and generally is within the range between 30 ... 50 ms.

An evaluation of the angular drift regarding an error response is not provided in the i700 servo inverters. This has to be carried out in the control system.

From software version V01.06.05 onwards, communication with the encoder during the angular drift determination is not monitored anymore. If a transmission error occurs

- the current angular drift is marked as invalid by the 0x2833, bit 7 status flag,
- the i700 servo inverter keeps its operating status.

The communication error monitoring (0x2C41:4) from software version V01.06.05 onwards is only used for monitoring the initial reading out and setting of the position.

The 0x2C42:4 object displays the signal amplitude of the encoder as a nominal value in percent. Standard encoders provide 1 Vss (peak value 0.5 V) on the signal cables. In such a case, 100 % are displayed in the object.

Open-circuit monitoring

The open-circuit monitoring of the i700 detects an error if the amplitude value has decreased several times in a row to less than 73 % or increased to more than 126 %. For a reliable operation, we recommend a signal amplitude in a range of 90 % ... 110 %.

If open circuit messages occur sporadically (e.g. due to defective wiring), the signal in the oscilloscope can be used for recording and analysis. Here, not every fault is displayed since the update rate of the internal monitoring is considerably higher compared to the 0x2C42:4 object or the oscilloscope:

- 20 ms ... 30 ms for the object 0x2C42:4 / the oscilloscope
- 62.5 µs for the internal monitoring

From software version V01.06.05 onwards, the sensitivity of open-circuit monitoring can be set in percent by the 0x2C47 object. The reduction of the monitoring sensitivity with an EMC-polluted environment has proven advantageous.

Note!

A reduced sensitivity causes a delayed response to real open circuits!

If the sensitivity is not reduced (100 %), the software response time of the monitoring is
- approx. 3.5 ms for an encoder
- 0.3 ms for a resolver.

Halving the sensitivity results in a doubling of the response time.

5.10.5 Additional settings for SinCos absolute value encoders with HIPERFACE® protocol

Absolute value encoders are especially suitable for:

- Synchronous motors operated in the "servo control" mode. The synchronous motor (SM) servo control requires a pole position angle. This has to be detected only once during commissioning and saved as offset towards the absolute position in the axis data.
- Positioning modes in which homing is to be carried out only once.

The analog evaluation of the sin/cos tracks causes a high resolution. With regard to the storage of the position information, we distinguish between singleturn and multiturn encoders:

- Singleturn: Storage within one revolution
- Multiturn: Storage within a number of revolutions

Supported encoder types with HIPERFACE® protocol

The following encoder types are supported by the i700 servo inverters:

Type	Increments/revolution	Absolute revolutions	Type code (0x2C41:1 | 0x3441:1)
AM1024-8V-H (SRM50)	1024	4096 (Multiturn)	39
AM1024-8V-H (SFM60)	1024		39
AM1024-8V-K2 (SRM50S)	1024		39
AM128-8V-H (SKM36)	128		55
AM16-8V-H (SEL37)	16		71
AM16-8V-H (SEL52)	16		71
AM512-8V-H (SCM70)	512		7
AS1024-8V-H (SRS50)	1024	1 (Singleturn)	34
AS1024-8V-K2 (SRS50S)	1024		34
AS16-8V-H (SEK37)	16		66
AS16-8V-H (SEK52)	16		66
AS512-8V-H (SCS70)	512		2

Supported SinCos encoder types without HIPERFACE® protocol

The following encoder types are supported by the i700 servo inverters:

Type	Increments/revolution	Absolute revolutions
IG1024-5V-V3 (RVS58S)	1024	0
IG2048-5V-S (ITD22)	2048	0
IG2048-5V-S	2048	0

Use of non-supported encoder types

If an encoder is to be used, the type code of which is not listed in the table of the supported encoder types, this encoder can be introduced to the i700 servo inverters via the subindices 2 and 3 of the hiperface parameters described in the following. Please also observe the notes in the description of subindex 8.

Reading data out of the encoder

The "Determine data of the Hiperface encoder" function in parameter 0x2822 (or 0x3022 for axis B) serves to read the type code, number of increments and number of distinguishable revolutions out of the encoder and automatically enter them into the corresponding Hiperface parameters.

0x2C41 | 0x3441 - Hiperface: Parameter

Sub.	Name	Lenze setting	Data type
1	Hiperface: Determined type code	0	UNSIGNED_8
2	Hiperface: User def. encoder - type code	0	UNSIGNED_8
3	Hiperface: User def. encoder - specifiable revolutions	1	UNSIGNED_16
4	Hiperface absolute value fault: Response	1: Trouble	UNSIGNED_8
5	Hiperface: Serial number		STRING(50)
6	Hiperface: Raw data - Actual position		UNSIGNED_32
7	Hiperface: Detected Increments / revolution		UNSIGNED_16
8	Hiperface: Type code supported by firmware	0: Not supported	UNSIGNED_8
9	Hiperface: Encoder type	0: Rotary transducer	UNSIGNED_8
10	Hiperface: Period length linear encoder		UNSIGNED_32
11	Hiperface: Data block address	0	UNSIGNED_16
12	Hiperface: Data block length	1	UNSIGNED_8
13	Hiperface: Data block command	0: Completed/no action	UNSIGNED_8
14	Hiperface: Data block status	1	UNSIGNED_16
15	Hiperface: Data block		BYTE

Subindex 1: **Hiperface: Determined type code**				
Type code read out of the encoder If a sin/cos encoder is set in 0x2C40 (or 0x3440 for axis B) or a communication error has occurred, this value is zero.				
Display range (min. value | unit | max. value)			Initialisation	
0		255	0	
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX				UNSIGNED_8

Subindex 2: **Hiperface: User-defined encoder - type code**				
If the encoder is not supported by the firmware (see subindex 8): → Here, manually set the type code displayed in the subindex 1.				
Setting range (min. value | unit | max. value)			Lenze setting	
0		255	0	
☑ Write access ☑ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_8

Subindex 3: **Hiperface: User-defined encoder - number of revolutions**				
If the encoder is not supported by the firmware (see subindex 8): → Here, manually set the number of distinguishable revolutions. **Stop!** A wrong setting of the number of distinguishable revolutions may cause a breakdown!				
Setting range (min. value | unit | max. value)			Lenze setting	
1		65535	1	
☑ Write access ☑ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 4: **Hiperface absolute value fault: Response**
Selection of the response to communication problems or unknown encoder

Subindex 4: **Hiperface absolute value fault: Response**			
Selection list(Lenze setting printed in bold)			
0	No response		
1	**Interference**		
2	Warning		
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			UNSIGNED_8

Subindex 5: **Hiperface: Serial number**		
From software version V01.03.xx onwards The displayed serial number can be used for detecting an encoder exchange		
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX		STRING(50)

Subindex 6: **Hiperface: Raw data - Actual position**				
From software version V01.03.xx onwards The encoder-internal position value is output without being converted				
Display range (min. value | unit | max. value)			Initialisation	
0		4294967295		
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☑ TX				UNSIGNED_32

Subindex 7: **Hiperface: Detected Increments / revolution**				
From software version V01.03.xx onwards Encoder increment according to type code or encoder nameplate				
Display range (min. value | unit | max. value)			Initialisation	
0		65535		
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☑ TX				UNSIGNED_16

Subindex 8: **Hiperface: Type code supported by firmware**			
From software version V01.03.xx onwards If an encoder is connected that is not supported by the firmware, it will be displayed here. In this case, the same response takes place as in case of a communication error. The error can be removed by manually setting the type code displayed in subindex 1 in subindex 2. This serves to signalise to the firmware that the number of distinguishable revolutions is as well set correctly in the subindex 3 by the user.			
Selection list (read only)			
0	**Not supported**		
1	Supported by firmware		
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☑ TX			UNSIGNED_8

Subindex 9: **Hiperface: Encoder type**			
From software version V01.05.xx onwards Detected encoder type (rotary/linear)			
Selection list (read only)			
0	**Rotary encoder**		
1	Linear encoder		
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☑ TX			UNSIGNED_8

Subindex 10: **Hiperface: Period length of linear encoders**				
From software version V01.05.xx onwards In case of a linear encoder, here the period length or scaling is displayed in [nm]. In case of a rotary encoder, the value "0" is displayed.				
Display range (min. value | unit | max. value)			Initialisation	
0	nm	4294967295		
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☑ TX				UNSIGNED_32

Subindex 11: **Hiperface: Data block address**	
From software version V02.12.xx onwards Address in the non-volatile memory of the Hiperface encoder.	
Setting range (min. value | unit | max. value)	Lenze setting

Subindex 11: **Hiperface: Data block address**			
0		65535	0
Value is bit-coded: (☑ = bit set)			Info
Bits 0-7 ☐	Offset		
Bits 8-15 ☐	Field ID		
☑ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			UNSIGNED_16

Subindex 12: **Hiperface: Data block length**			
From software version V02.12.xx onwards Length of the range to be read out starting from the address from 0x2C41:011.			
Setting range (min. value | unit | max. value)			Lenze setting
1		32	1
☑ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			UNSIGNED_8

Subindex 13: **Hiperface: Data block command**		
From software version V02.12.xx onwards With the edge change from 0 to 1, the data is read out from 0x2C41:011 and 0x2C41:012.		
Selection list(Lenze setting printed in bold)		
0	**Completed/No action**	
1	Read	
☑ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX		UNSIGNED_8

Subindex 14: **Hiperface: Data block status**			
From software version V02.12.xx onwards Status of the read operation:			
Display range (min. value | unit | max. value)			Initialisation
0		65535	1
Value is bit-coded:			Info
Bit 0	Done		The bit is always set when a request is completed. As long as the bit has the value "0", the request is in progress.
Bit 1	Error		The bit is set in addition to bit 0 when an error has occurred. The Hiperface-specific error code is shown in the bits 8-15.
Bit 2	Valid		The bit is set with bit 0 when • the request has been completed successfully and • the data in 0x2C41:015 is valid and can be read out.
Bit 3-7	Reserved		
Bit 8-15	Error code		
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			UNSIGNED_16

Subindex 15: **Hiperface: Data block**			
From software version V02.12.xx onwards Read-out motor serial number			
Display range (min. value | unit | max. value)			Initialisation
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			BYTE

Communication error monitoring

Communication with the encoder is monitored by the protocol and by generating a checksum. If a violation of the communication protocol or a defect frame is detected, a response occurs as a function of the CiA device status and the setting in subindex 4 (Hiperface communication error: Response):

CiA device state	Response in the event of an error (depending on the error responses set in the Subindex 4)		
	0: No Response	1: Trouble	2: Warning
Not ready to switch on	Warning		
Switch on disabled	Warning		
Ready to switch on	-	Interference	Warning
Switched on	-	Interference	Warning
Operation enabled	-	Interference	Warning
Quick stop active	-	Interference	Warning
Fault reaction active	-	Warning	
Fault	-	Warning	

5.10.6 Detection of changed settings of the feedback system

Bit 0 of the Lenze status word 2 (0x2833 or 0x3033 for axis B) displays whether the settings of the feedback system have been changed since the Not ready to switch on has been left. In case of a change, bit 0 is set to "1".

A transition to the Operation enabled state causes bit 0 to be reset again to "0". In all device states, the monitoring of changes at the following objects remains active:

Object		Name
Axis A	Axis B	
0x2C40	0x3440	Encoder: Type
0x2C41:2	0x3441:2	Hiperface: User defined - Type code
0x2C41:3	0x3441:3	Hiperface: User defined - Number of revolutions
0x2C41:5	0x3441:5	Hiperface: Serial number
0x2C42:1	0x3442:1	Encoder: Increments / revolution
0x608F	0x688F	Position encoder resolution

After a controller enable, bit 0 of the Lenze status word 2 is always reset to "0".

5.11 Synchronous motor (SM): Pole position identification

Note!

An identification of pole position (PLI) is only required:
- For servo control with synchronous motor of a third-party manufacturer.
- For servo control with synchronous motor and use of incremental encoders (TTL or sin/cos encoders).
- After changes of the motor feedback system, e.g. encoder exchange.

For the control of a permanent-magnet synchronous motor, the pole position – the angle between motor phase U and the field axis of the rotor – must be known.

- For Lenze motors with an absolute encoder or resolver, the pole position has already been set correctly.
- When incremental encoders (TTL or sin/cos encoders without absolute position information) are used, a pole position identification (PPI) is always required after switching on the mains (initialisation), even with Lenze motors.

Selection criteria for using the suitable pole position identification

For identifying the pole position for the currently activated motor encoder, the three functions are available that provide nearly the same result. Due to e.g. friction, bearing reactions and a trapezoidal field characteristic, the results may differ.

▶ Pole position identification PPI (360°)
- The motor must not be braked, blocked or driven mechanically during the pole position identification! For this reason, this function is not permitted for hanging loads!
- Especially in case of drives in idle state or with a low load (inertia / friction), this function provides the most exact results of all three PLIs.

▶ Pole position identification PPI (min. movement)
- The motor must not be braked, blocked or driven during the pole position identification! For this reason, this function is not permitted for hanging loads!
- This function is in the medium range regarding the accuracy. When the current amplitude is increased in percent, the accuracy of the result may be increased.

▶ Pole position identification PPI (without movement)
- In case of firmly braked motors (e.g. for hanging loads), only this function may be used!
- This function has been developed for a great bandwidth of motor characteristics. In case of some motor types, the identified pole position angle can considerably differ from the real one which may result in a significant torque loss and higher motor losses. Hence, especially for third-party motors, we recommend the execution of a reference identification using the Pole position identification PPI (360°) with an idling motor.
  If the identified values of both processes differ from each other by more than 20°, please contact Lenze.
- When the current amplitude is increased in percent, the accuracy of the result may be increased.

You can find detailed information on the respective function in the following subchapters.

5.11.1 Monitoring of the pole position identification

If an error occurs during the pole position identification or the pulse inhibit gets active (e.g. due to short-time undervoltage), the procedure is terminated with controller inhibit without a change in settings.

If the motor was braked or blocked during the procedure, this will be recognised at the end of the measurement and no change is made either (exception: "Pole position identification PLI (without motion").

The error response can be parameterised:

0x2C60 | 0x3460 - Monitoring of pole position identification: Response

If the pole position identification is aborted, the response set here is triggered.
- If this behaviour is not wanted, deactivate the monitoring by selecting "0: No response".

Selection list(Lenze setting printed in bold)			
0	No response		
1	**Interference**		
2	Warning		
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			UNSIGNED_8

5.11.2 Pole position identification PPI (360°)

Stop!

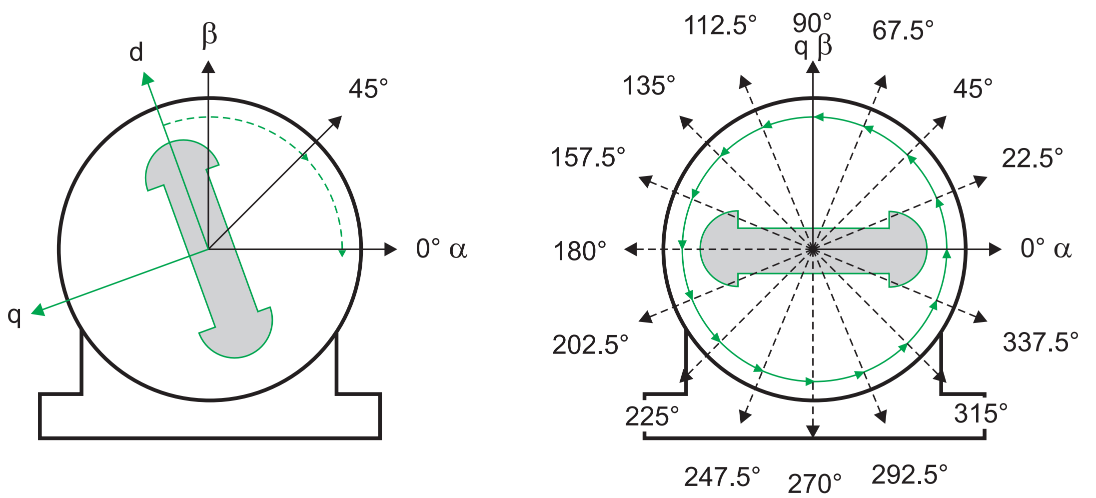
Check the correct parameterisation of the followng monitoring modes before carrying out the pole position identification to prevent the motor from being permanently damaged in the event of an error:
- Monitoring of the motor utilisation (I²xt) (📖 296)
- Monitoring of the ultimate motor current (📖 313)

Note!

Please observe the following:

▶ Selection criteria for using the suitable pole position identification

Functional description



If the servo control for synchronous motor is set and if no error is pending, the current is increased to 141 % of the rated motor current along a ramp first after controller enable.

Left image

First, the rotor is moved from its optional position of rest to the 0° angle.

- For this purpose, the amplitude of the d current vector in the stator coordinate system is built up at a starting angle of 45° and then turned to 0°.
- A sufficiently high amplitude of the d current vector and its rotary motion result in a magnetic force that moves the rotor to the angle 0°.

Right image

Afterwards, the d current vector is turned further in 15 steps by 22.5° each starting at the angle 0°.

- The rotor adjusts itself to the corresponding angle due to the magnetic forces.
- After 16 steps, the rotor has moved electrically by absolute 360°.

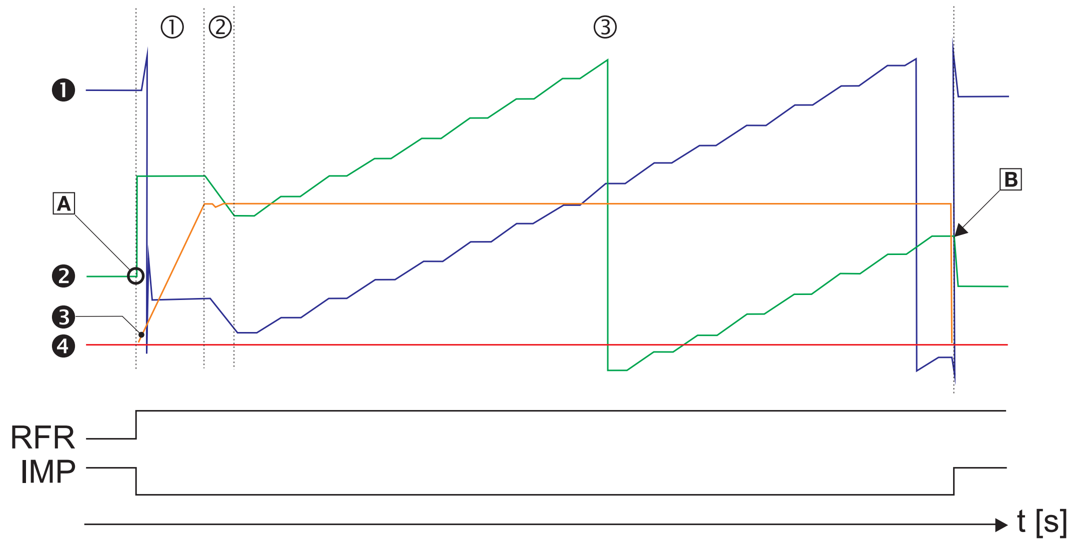
Result

For determining the pole position, a mean value is calculated from all 16 messages.
The rotor displacement angle can be recorded via the 0x2DDE object (actual motor rotor angle position).
The detected pole position is stored in the i700 object 0x2C03:2, 0x2C03:4 and then has to be saved in the PLC project.

Abort of the pole position identification

The pole position identification is aborted if the deviations between the rotary motion of the current vector and the rotor exceed the error tolerance set in 0x2C41:4.

[5-2] PLI function (360°)



A Start of the pole position identification
B End of the pole position identification (IMP = High)
❶ Current position of the rotor (0x6064:0, actual position or 0x2DDE:0, actual position of rotor angle)
❷ Commutation angle of current vector, variable of the motor control (cannot be displayed in case of i700)
❸ Amplitude of the d current vector (0x2DD1:1 actual d current)
❹ Amplitude of the q current vector (0x2DD1:2 actual q current), is not used for PPI_360°
① Structure of the d current vector at electrical 45° (starting angle)
② Angular change of the d current vector to electrical 0°; alignment of the rotor
③ 15 angular changes by 22.5° each and subsequent plausibility check

[5-3] Chronological sequence of the pole position identification (360°)

Preconditions for the execution

- The motor must not be braked or blocked during the pole position identification.
- The i700 servo inverters is free of errors and is in the "Switched on" device status.

Response of the motor during the execution

During the pole position identification the rotor aligns itself. The motor shaft moves by max. one electrical revolution which causes the corresponding movement of the connected mechanical components!

How to execute the pole position identification PPI (360°):

1. If the i700 servo inverters is enabled, disable the i700 servo inverters.
   ▶ Enable/inhibit operation via control word (📖 69)
2. Set object 0x2825 (or 0x3025 for axis B) to "5" to change to the "Pole position identification PPI (360°)" operating mode.
3. Before the PLI can be started, the works mentioned in the following have to be completed.
   All setting values are examples and apply to the motor 115UDD305BBWBA-SKTY

Activate/parameterise motor overload monitoring (I^2xt)

- 0x2D4C:0 = 4
- 0x2D4C:1 = 175 s
- 0x2D4C:2 = 2300 s
- 0x2D4C:3 = 27 %
- 0x2D4C:4 = 0
- 0x2D4D:0 = 8
- 0x2D4D:1 = 0
- 0x2D4D:2 = 72 %
- 0x2D4D:3 = 1 %
- 0x2D4D:4 = 137 %
- 0x2D4D:5 = 100 %
- 0x2D4D:6 = 100 %
- 0x2D4D:7 = 133 %
- 0x2D4D:8 = 50 %
- 0x2D4E:0 = 100 %
- 0x2D4F:0 = 0
- 0x2D50:0 = 2
- 0x2D50:0 = Störung [1]
- 0x2D50:0 = 105

Calculate or metrologically determine basic current controller parameters and set them (operating point of rated current is initially sufficient)

- 0x2822:0 = no command [0]
- 0x2939:0 = 8 kHz [1]
- 0x2941:0 = activate [1]
- 0x2942:1 = 9.25 V/A
- 0x2942:2 = 12.90 ms

Parameterise filter time constant for actual speed value

- 0x2904:0 = 0.6 ms

Parameterise monitoring of the maximum motor current

- 0x2D46:0 = 2
- 0x2D46:1 = 50 A
- 0x2D46:2 = fault [1]

4. Enable the i700 servo inverters to start the pole position identification (360°).

   Note: By means of controller inhibit, the procedure started can be cancelled any time, if required, without a change in settings.

After successful completion of the pole position identification...

...controller inhibit is set automatically and the pole position specified in object 0x2C03:2 (or 0x3403:2 for axis B) for the activated feedback system is set.

- For permanent storage, the changed settings must be uploaded to the controller from the i700 servo inverters.
  The »EASY Starter« serves to save the parameter settings of the i700 servo inverters as parameter file (*.gdc). In the »PLC Designer«, this file can then be imported in the corresponding axis. ▶ Saving changed parameters safe against mains failure (📖 70)

- The controller inhibit automatically set by the procedure can be deactivated via the Controlword (0x6040 or 0x6840 for axis B). ▶ Enable/inhibit operation via control word (📖 69)

5.11.2.1 Adapt pole position identification PPI (360°)

For drives with a high static friction, mass inertia or alternating load, an optimisation may be required:

- The amplitude of the current vector has to be set so high that the motor can be accelerated with a high mass inertia.
- The cyclic continued rotation of the current vector by 22.5° has to cause an equivalent angular rotation of the motor shaft (rotor). A step function has to be achieved as displayed in the figure [5-3]. Here, actual positions with very low overshoots are visible.

Stop!

If there is no temperature monitoring in the motor and/or the I²xt motor monitoring and the maximum current monitoring are not parameterised correctly, the motor might be damaged permanently when the current amplitude is set too high!

▶ Monitoring of the motor utilisation (I²xt) (📖 296)
▶ Monitoring of the ultimate motor current (📖 313)

0x2C61 | 0x3461 - Pole position identification PPI (360°)

Sub.	Name	Lenze setting	Data type
▸ 1	PPI (360°): Current amplitude	100 %	UNSIGNED_16
▸ 2	PPI (360°): Ramp time	40 s	UNSIGNED_16
▸ 3	PPI (360°): Direction of rotation	0: Field: CW	UNSIGNED_8
▸ 4	PPI (360°): Error tolerance	20 °	UNSIGNED_8
▸ 5	PPI (360°): Absolute current amplitude		UNSIGNED_32

Subindex 1: **PPI (360°): Current amplitude**				
Adjustment of the current amplitude in percent • For large machines and high mass inertia values or for linear direct drives, the current amplitude usually has to be increased. • A Lenze setting of "100 %" corresponds to 141 % of the rated motor current (0x6075 or 0x6875 for axis B). **Note!** If the current amplitude is set to > 100 %, the device utilisation (Ixt) monitoring and/or one of the motor monitoring functions may respond and cause the abort of the pole position identification.				
Setting range (min. value | unit | max. value)			Lenze setting	
1	%	1000	100 %	
☑ Write access ☑ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 2: **PPI (360°): Ramp time**				
Adjustment of the ramp time in percent • For large machines and high mass inertia values, the ramp time usually has to be increased. • For small machines, a reduction of the ramp time can speed up the pole position identification process.				
Setting range (min. value | unit | max. value)			Lenze setting	
1	s	600	40 s	
☑ Write access ☑ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 3: **PPI (360°): Direction of rotation**			
Selection of the travel direction • In some situations it may be helpful to reverse the travel direction for the pole position identification (e.g. for linear motor at the end stop).			
Selection list(Lenze setting printed in bold)			
0	**Field: CW**		
1	Field: CCW		
☑ Write access ☑ CINH ☐ OSC ☑ P ☐ RX ☐ TX			UNSIGNED_8

Subindex 4: **PPI (360°): Error tolerance**				
Error tolerance for the plausibility check • If the rotor position detected via the encoder system is not within the tolerance zone around the position that is output in a controlled manner, the pole position identification is aborted and the error response parameterised is tripped.				
Setting range (min. value | unit | max. value)			Lenze setting	
15	°	50	20 °	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_8

Subindex 5: **PPI (360°): Absolute current amplitude**	
Display of the absolute current amplitude	
Display range (min. value | unit | max. value)	Initialisation

Subindex 5: PPI (360°): Absolute current amplitude				
0.00	A	42949672.95		
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			Scaling: 1/100	UNSIGNED_32

Overview of more objects available for

- Identification
- Control
- Diagnostics

Tip!
The optimisation can be carried out using an oscilloscope.

Index	Subindex	Value / unit	INFO
0x2825	0	CiA402 mode active [0]	Operating mode [5] for PLI 360°
0x2824	0	Activation [1]	
0x6040	0	0x0000	Simulation of the CiA state machine
0x2823	0	100	Progress bar
0x2832	0	0x0000	Diagnostics of oscilloscope variable
0x2C61	1	100 %	PLI(360°) current amplitude
0x2C61	2	40 s	PLI(360°) ramp time
0x2C61	3	Field: Clockwise [0]	PLI(360°) direction of rotation
0x2C61	4	20°	PLI(360°) error tolerance
0x2C61	5	4.81 A	Display
0x2C03	2	-90.0°	Detected pole position values
0x2C03	4	0.0°	
0x2DDE	0	1850	Diagnostics of oscilloscope variable
0x2D83	2	0.03 A	Diagnostics
0x2D83	3	0.04 A	Diagnostics
0x2D83	4	-0.01 A	Diagnostics
0x2DD1	3	0.00 A	Diagnostics of oscilloscope variable
0x2DD1	1	0.01 A	Diagnostics of oscilloscope variable
0x6073	0	150.0 %	Motor control parameters
0x6075	0	3.400 A	Reference for 0x2C61:1
0x2D46	1	16.5 A	Motor protection parameters
0x2DDF	1	5.00 A	User information regarding max. peak axis current
0x2DDF	2	10.00 A	User information regarding max. RMS axis current

5.11.3 Pole position identification PPI (min. movement)

Danger!

The motor must not be braked or blocked during the pole position identification! For this reason, this function is not permitted for hanging loads!

Stop!

Check the correct parameterisation of the followng monitoring modes before carrying out the pole position identification to prevent the motor from being permanently damaged in the event of an error:

- Monitoring of the motor utilisation (I²xt) (📖 296)
- Monitoring of the ultimate motor current (📖 313)

Functional description

If servo control for synchronous motor is set and if no error is pending, the current position is memorised after controller enable, and the current is increased along a ramp for 10 s to 35 % of the rated motor current. This will cause the rotor to align, which, however, is compensated by a position control. If the rotor makes an electrical movement of more than 20°, an error message is output, and the value measured is discarded. This might occur in the case of motors with considerable detent torques.

If the current has reached its final value, a plausibility check is executed after a short interval: In order to detect a non-permissible blocking of the motor, a positive and negative test angle (± 20°) relative to the current position are defined after the identification. The motor must align itself to these two test angles within a tolerance of 25 %.

Preconditions for the execution

- The motor must not be braked or blocked during the pole position identification.
- The i700 servo inverters is free of errors and is in the "Switched on" device status.

Response of the motor during the execution

The motion of the motor will maximally correspond to the set "Max. permissible motion" (Lenze setting: 20°). If a greater motion is detected via the encoder system, the pole position identification is cancelled and the parameterised error response (Lenze setting: Fault) is triggered.

How to execute the pole position identification PPI (min. movement):

1. If the i700 servo inverters is enabled, disable the i700 servo inverters.
   ▶ Enable/inhibit operation via control word (📖 69)
2. Set object 0x2825 (or 0x3025 for axis B) to "6" to change to the "Pole position identification (min. movement)" operating mode.
3. Enable the i700 servo inverters to start the procedure.

   Note: By means of controller inhibit, the procedure started can be cancelled any time, if required, without a change in settings.

After successful completion of the pole position identification...

...controller inhibit is set automatically and the pole position specified in object 0x2C03:2 (or 0x3403:2 for axis B) for the activated feedback system is set.

- For permanent storage, the changed settings must be uploaded to the controller from the i700 servo inverters.
  The »EASY Starter« serves to save the parameter settings of the i700 servo inverters as parameter file (*.gdc). In the »PLC Designer«, this file can then be imported in the corresponding axis. ▶ Saving changed parameters safe against mains failure (📖 70)
- The controller inhibit automatically set by the procedure can be deactivated via the Controlword (0x6040 or 0x6840 for axis B). ▶ Enable/inhibit operation via control word (📖 69)

Adapt pole position identification PPI (min. movement)

The above-described procedure for pole position identification can be adjusted to the respective machine and the prevailing moments of inertia by means of the parameters described in the following.

STOP

Stop!

If there is no temperature monitoring in the motor and/or the I²xt motor monitoring and the maximum current monitoring are not parameterised correctly, the motor might be damaged permanently when the current amplitude is set too high (e.g. to the maximum value!

▶ Monitoring of the motor utilisation (I²xt) (📖 296)

▶ Monitoring of the ultimate motor current (📖 313)

0x2C62 | 0x3462 - Pole position identification PPI (min. movement)

Sub.	Name	Lenze setting	Data type
1	PPI (min. movement): Current amplitude	25 %	UNSIGNED_16
2	PPI (min. movement): Ramp time - current	10 s	UNSIGNED_16
3	PPI (min. movement): Gain	0 %	UNSIGNED_16
4	PPI (min. movement): Reset time	62.5 ms	UNSIGNED_16
5	PPI (min. movement): Max. move permitted	20 °	UNSIGNED_8
6	PPI (min. movement): Absolute current amplitude		UNSIGNED_32

Subindex 1: **PPI (min. movement): Current amplitude**				
Adjustment of the current amplitude in percent • For large machines and high mass inertia values or for linear direct drives, the current amplitude usually has to be increased. • A Lenze setting of "100 %" corresponds to 35 % of the rated motor current (0x6075 or 0x6875 for axis B). **Note!** If the current amplitude is set to > 400 %, the device utilisation (Ixt) monitoring and/or one of the motor monitoring functions may respond and cause the abort of the pole position identification.				
Setting range (min. value | unit | max. value)			Lenze setting	
1	%	1000	25 %	
☑ Write access ☑ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 2: **PPI (min. movement): Ramp time - current**				
Adjustment of the rate of rise of the current in percent				
Setting range (min. value | unit | max. value)			Lenze setting	
1	s	600	10 s	
☑ Write access ☑ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 3: **PPI (min. movement): Gain**				
Adjustment of the proportional gain of the PI controller • With the Lenze setting "0", the PI controller works as an I-controller.				
Setting range (min. value | unit | max. value)			Lenze setting	
0	%	1000	0 %	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 4: **PPI (min. movement): Reset time**				

Adjustment of the PI controller reset time
• For this, observe the following setting details:
  • In order to be able to compensate a positional variation faster, first the reset time should be reduced. If this does not result in the desired behaviour, the proportional gain can be increased.
  • Ensure that the position control does not get unstable. We therefore recommend to use an I controller.				

Setting range (min. value | unit | max. value)			Lenze setting	
0.1	ms	6000.0	62.5 ms	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	UNSIGNED_16

Subindex 5: **PPI (min. movement): Max. move permitted**				
Adjustment of the permitted movement • The pole position identification comprises a monitoring function for the follow-up control. If a movement greater than the permissible movement set is detected by the encoder system, the pole position identification is aborted and the error response parameterised is tripped: • In order to detect a non-permissible blocking of the machine, a positive and negative test angle relative to the current position are defined after the identification. The machine must align itself to these two test angles within a tolerance of 25 %. The size of the test angle corresponds to the max. move permitted set here.				
Setting range (min. value | unit | max. value)			Lenze setting	
1	°	90	20 °	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_8

Subindex 6: **PPI (min. movement): Absolute current amplitude**				
Display of the absolute current amplitude				
Display range (min. value | unit | max. value)			Initialisation	
0.00	A	42949672.95		
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			Scaling: 1/100	UNSIGNED_32

5.11.4 Pole position identification PPI (without movement)

This function extension is available from software version V01.03.xx onwards!

Note!

During the pole position identification, the error 0xFF13 ("identification cancelled") may occur. This may be an indication that the motor features are not suitable for this PLI process.

Functional description

After controller enable, a defined pulse pattern is output which provides currents up to approximately maximum motor current. The respective currents are measured. Based on these currents, the field distribution can be detected so that the pole position can be calculated. Afterwards, the controller is inhibited automatically.

The pole position identification (without movement) does not require any further parameterisation.

Preconditions for the execution

- The three motor phases and the motor encoder have to be wired according to the specifications given in the i700 hardware manual.
- The motor may be firmly braked.
- The i700 servo inverters is free of errors and is in the "Switched on" device status.
- Please observe the notes from the Selection criteria for using the suitable pole position identification section.

Response of the motor during the execution

The current test pulses cause audible engine noises that can be increased by the machine mechanics depending on the mechanical coupling!

How to execute the pole position identification (without movement):

1. If the i700 servo inverters is enabled, disable the i700 servo inverters.
   ▸ Enable/inhibit operation via control word (📖 69)
2. Set object 0x2825 (or 0x3025 for axis B) to "7" to change to the "Pole position identification PPI (without movement)" operating mode.
3. Enable the i700 servo inverters to start the procedure.

   Note: By means of controller inhibit, the procedure started can be cancelled any time, if required, without a change in settings.

After successful completion of the pole position identification...

...controller inhibit is set automatically and the pole position specified in object 0x2C03:2 (or 0x3403:2 for axis B) for the activated feedback system is set.

- For permanent storage, the changed settings must be uploaded to the controller from the i700 servo inverters.
  The »EASY Starter« serves to save the parameter settings of the i700 servo inverters as parameter file (*.gdc). In the »PLC Designer«, this file can then be imported in the corresponding axis. ▸ Saving changed parameters safe against mains failure (📖 70)
- The controller inhibit automatically set by the procedure can be deactivated via the Controlword (0x6040 or 0x6840 for axis B). ▸ Enable/inhibit operation via control word (📖 69)

Optional settings (starting performance)

Optionally, a PLI without motion can be activated after switching on the i700 servo inverters.

> **Danger!**
>
> Each pole position identification causes an update of the pole position set in the device! Therefore, ensure that the response to open circuit in the feedback system is set to Lenze setting "1: Fault" in 0x2C45 (or 0x3445 for axis B)!
> Otherwise, the status of the feedback system in case of open circuit is undefined and the pole position can assume any values. There is a danger that the machine accelerates in an uncontrolled way after pole position identification!

> **Note!**
>
> The process of the pole position identification only lasts some milliseconds. During the pole position identification, the device status does not change. Only after the pole position identification, the "Switched on" device status changes to the "Operation enabled" device status.

0x2C63 | 0x3463 - pole position identification PLI (without movement)

From software version V01.05.xx onwards

Sub.	Name	Lenze setting	Data type
▸ 1	Behaviour after switch-on	0: No operation	UNSIGNED_8

Subindex 1: **Behaviour after switch-on**		
Starting performance (without or with PLI before starting) in case of synchronous motor (SM) servo control • In case of all other control modes, the setting has no influence.		
Selection list(Lenze setting printed in bold)		Info
0	**No Operation**	In the Lenze setting, the behaviour is the same as before.
1	PPI at initial switch-on and after encoder error only	After the first controller enable and after each encoder wire breakage, a PLI without movement takes place. **Note:** In order that the PLI is executed after each encoder open circuit, the response to open circuit in the feedback system has to be set to the Lenze setting "1: Fault" in 0x2C45 (or 0x3445 for axis B). Otherwise, the status of the feedback system in case of open circuit is undefined and the pole position can assume any values. There is a danger that the machine accelerates in an uncontrolled way after pole position identification!
2	PPI after every switch-on	After every controller enable, a PLI without movement is carried out in advance.
☑ Write access ☑ CINH ☐ OSC ☑ P ☐ RX ☐ TX		UNSIGNED_8

5.12 Setting control loops

Subsequent to the motor parameterisation, the different control loops must be set. For quick commissioning, the calculations and settings can be carried out automatically using the commissioning functions of the i700 servo inverters. For manual setting, applicable equations are offered in the following subchapters, too.

The following table shows the corresponding commissioning steps required for the different control types and motors:

Commissioning step	Servo control SM	Servo control ASM	V/f characteristic control
Setting and optimising the current controller	(●)	(●)	●[1]
Determining the total moment of inertia	●	●	
Setting the speed controller	●	●	
Setting the position controller	●	●	
Setting the field controller (ASM)		(●)	
Setting the field weakening controller (ASM)		(●)	
Information on the following control loops/commissioning steps for the V/f characteristic control can be found in the chapter "Parameterising the V/f characteristic control"			
Set Imin controller ▸ Activating the voltage vector control (Imin controller)			●
Set Imax controller ▸ Defining the behaviour at the current limit (Imax controller)			●
Set "restart on the fly" controller ▸ "Flying restart" function			●

● Required

(●) Only required for other manufacturers' motors

[1] Only required if voltage vector control, DC-injection braking, or flying restart process is activated.

5.12.1 Setting and optimising the current controller

Note!

For a servo control, the current controller should always be optimised if a motor of another manufacturer with unknown motor data is used!

For a V/f characteristic control, the current controller only has to be optimised if voltage vector control is used, or if DC-injection braking or the flying restart process is activated.

The control system includes two current controllers, a direct-axis current controller and a cross current controller, whose parameterisation is carried out identically. The direct-axis current controller controls the field-producing current (D current). The cross current controller controls the torque-producing current (Q current). There is a coupling between the two control loops which makes every actuation of every one of the controllers occur as fault in the control loop of the other controller. This coupling can be compensated for by activating the current controller feedforward control via object 0x2941 (or 0x3141 for axis B).

For the automatic calculation of the two controller parameters, the "Current controller: Calculate controller parameters" function is provided via object 0x2822 (or 0x3022 for axis B).

- The calculating function is based on the stator resistance (0x2C01:2 or 0x3401:2 for axis B) and the stator leakage inductance (0x2C01:3 or 0x3401:3 for axis B). Thus, these motor parameters have to be parameterised before. This can be done either by entering the data sheet values manually or by a motor parameter identification run.

Note!

If one of the values calculated exceeds the upper object limit, the value is limited to the limit value.

The calculated controller parameters can be optimised by means of an experimental adjustment subsequently. The procedure is described in the following subchapter, Manual test mode "Current pulse".

Equations for calculating the gain and the reset time for the synchronous motor

$$V_p = \frac{L_{ss}}{T_{\text{Dead time}}} \qquad T_n = \frac{L_{ss}}{R_s}$$

Axis A	Axis B	Symbol	Description	Dimension unit
0x2942:1	0x3142:1	V_p	Current controller gain	V/A
0x2C01:3	0x3401:3	L_{ss}	Motor stator inductance	H
-	-	$T_{\text{dead time}}$	Equivalent time constant for analog detection and scanning = 0.00034 s (340 µs)	s
0x2942:2	0x3142:2	T_n	Current contr. reset time	s
0x2C01:2	0x3401:2	R_s	Motor stator resistance (value at 20°C)	Ω

Equations for calculating the gain and the reset time for the asynchronous motor

$$V_p = \frac{\sigma \cdot L_s}{T_{\text{Dead time}}} \approx \frac{2 \cdot L_{ss}}{T_{\text{Dead time}}} \qquad T_n = \frac{\sigma \cdot L_s}{R_s} \approx \frac{2 \cdot L_{ss}}{R_s}$$

Axis A	Axis B	Symbol	Description	Dimension unit
0x2942:1	0x3142:1	V_p	Current controller gain	V/A
-	-	σ	Leakage	
-	-	L_s	Motor stator inductance	H
0x2C01:3	0x3401:3	L_{ss}	Motor stator leakage inductance	H
-	-	$T_{\text{dead time}}$	Equivalent time constant for analog detection and scanning = 0.00034 s (340 µs)	s
0x2942:2	0x3142:2	T_n	Current contr. reset time	s
0x2C01:2	0x3401:2	R_s	Motor stator resistance (value at 20°C)	Ω

Tip!

To ensure a smooth transition to the field weakening range, it is recommended to apply the equations for the synchronous motor for the asynchronous motor as well.

0x2941 | 0x3141 - Current controller: Feedforward control

Activate/deactivate the current controller feedforward control
- Since the positioning movements are known, they can be precontrolled in order to slightly increase the achievable dynamic performance of the control loop.

Note!
A successful feedforward control requires knowledge of the equivalent circuit data of the motor. If only estimated values are available, the feedforward control should not be activated.

Selection list(Lenze setting printed in bold)			
0	**Off**		
1	Activate		
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			UNSIGNED_8

0x2942 | 0x3142 - Current controller: Parameter

Sub.	Name	Lenze setting	Data type
1	Current controller: Gain	148.21 V/A	UNSIGNED_32
2	Current controller: Reset time	3.77 ms	UNSIGNED_32

Subindex 1: **Current controller: Gain**				
Setting range (min. value | unit | max. value)			Lenze setting	
0.00	V/A	750.00	148.21 V/A	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/100	UNSIGNED_32

Subindex 2: **Current controller: Reset time**				
Setting range (min. value | unit | max. value)			Lenze setting	
0.01	ms	2000.00	3.77 ms	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/100	UNSIGNED_32

0x2943 | 0x3143 - Motor: Current setpoint - filter time

Setting range (min. value | unit | max. value)			Lenze setting	
0.00	ms	10.00	0.00 ms	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/100	UNSIGNED_16

5.12.1.1 Manual test mode "Current pulse"

The current controller must be adapted to the electrical characteristics of the motor – stator resistance and stator inductance. For an experimental adjustment, the manual test mode "Current pulse" can be used.

Note!

This test mode is intended for the adjustment of the current controller in the "servo control for synchronous motor/asynchronous motor" mode and not for the adjustment of the Imax controller in the "V/f characteristic control (VFC)" mode!

Functional description

In the manual "Current pulse" test mode, setpoint step-changes are applied to the current controller input after controller enable. The step responses then either have to be recorded using the oscilloscope and clamp-on ammeter, or by means of the i700 servo inverters oscilloscope function. By evaluating the step responses, it is the objective to optimise the two current controller parameters "gain" and "reset time" so that a quick current characteristic free of harmonics is obtained. (See the following instructions.)

Motors with individual pole windings, which can feature very distinct saturation phenomena in the stator leakage inductance, satisfactory results are possibly only achieved with a current-dependent correction of the current controller parameters. For this purpose, a characteristic is stored in the i700 servo inverters, describing the current dependence of the stator leakage inductance and correcting the current controller gain. ▶ Correction of the stator leakage inductance (Lss)...

Stop!

In the case of the synchronous motor, a jerky compensating movement occurs after controller enable if the pole position of this movement does not correspond to the starting angle.

Motor phase U is supplied with a DC current, the level of which is determined via the following equation on the left. Motor phases V and W then carry half of this DC current, respectively (negative; from the motor).

Equations for calculating the DC currents in motor phases U, V, W				
$I_{phase\_U} = \sqrt{2} \cdot I_{test}[\%] \cdot \frac{I_{rated}}{100\,\%}$		$I_{phase\_V,\_W} = -0.5 \cdot \sqrt{2} \cdot I_{test}[\%] \cdot \frac{I_{rated}}{100\,\%}$		
Axis A	Axis B	Symbol	Description	Dimension unit
0x2D83:2	0x3583:2	I_{phase_U}	Present current in motor phase U	A
0x2D83:3	0x3583:3	I_{phase_V}	Present current in motor phase V	A
0x2D83:4	0x3583:4	I_{phase_W}	Present current in motor phase W	A
0x2835:1	0x3035:1	I_{test}	Setpoint current for manual test mode	%
0x6075	0x6875	I_{rated}	Rated motor current	A
Greyed out = read access only				

Preconditions for the execution

- The motor has to be completely parameterised.
- The monitoring of the motor utilisation (I²xt) has to be parameterised and switched actively.
  ▸ Monitoring of the motor utilisation (I²xt) (📖 296)
- The motor must be able to rotate freely.
- The i700 servo inverters is free of errors and is in the "Switched on" device status.
- For the test, the rotor of synchronous motors has to be positioned in the pole centre. Some synchronous motors may require an alignment and blocking in the pole centre.
  - The use of the manual test mode "current/frequency" for one-time alignment of the rotor is reasonable with the following settings:
    R.m.s. value = 70 ... 100 %; frequency = 0 Hz; Starting angle = 0°
    ▸ Manual test mode "current/frequency" (📖 86)
  - Afterwards fixing by applying the holding brake or using external fixing tools.

Stop!

After the current controller has been adjusted, remove the mechanical fixing again!

Response of the motor during the execution

The motor will align during the first controller enable, then usually it won't align anymore.

How to adjust the current controller by means of the manual test mode "current pulse":

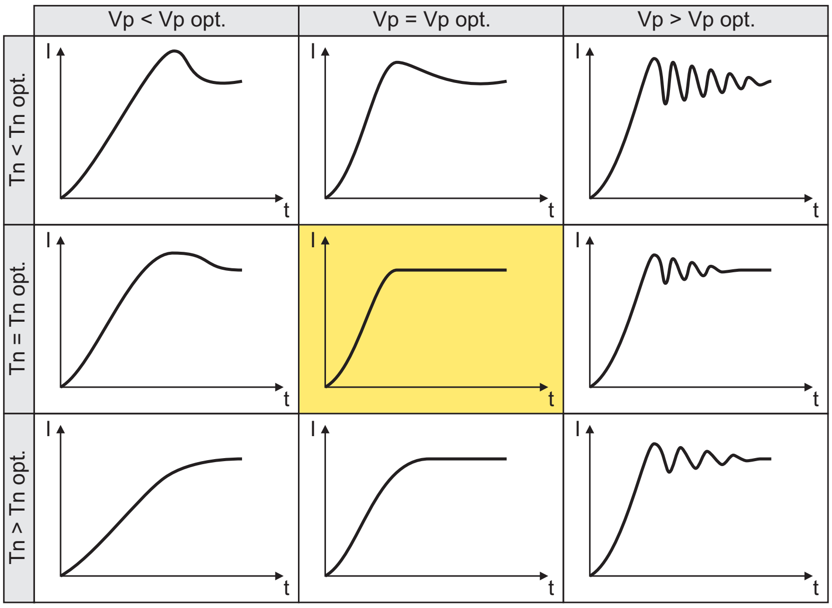
1. If the i700 servo inverters is enabled, disable the i700 servo inverters.
   ▸ Enable/inhibit operation via control word (📖 69)
2. Calculate the starting parameters for the current controller based on the parameterised motor data.
   - For the automatic calculation, the "Current controller: Calculate controller parameters" function is available via the object 0x2822 (or 0x3022 for axis B). ▸ Setting and optimising the current controller (📖 146)
   - Optionally, a manual determination is possible as well.
3. Set object 0x2825 (or 0x3025 for axis B) to "3" to change to the "Current pulse" test mode.
4. Set setpoint current for manual test mode in the object 0x2835:1 (or 0x3035:1 for axis B).

   The number of test points for the setpoint current results from the permissible current range of the motor and its characteristic (e.g. the Lss saturation for PM servo motors).
5. Enable the i700 servo inverters for a short time to start the test mode.

   After 100 ms, controller enable is cancelled automatically by the i700 servo inverters. Thus the danger of an undesired motor heat-up is avoided.

6. Record the step response of the motor current with the oscilloscope function of the »PLC Designer«/»EASY Starter«.
   - Parameters to be recorded:
     Actual D current (0x2DD1:1 or 0x35D1:1 for axis B)
     Setpoint D current (0x2DD1:3 or 0x35D1:3 for axis B)
     D voltage (0x2DD7:4 or 0x35D7:4 for axis B)
   - Oscilloscope settings: Sampling rate = 0.0625 ms; time base = 2 ... 5 ms

   Alternatively, the step response of the motor current in the motor phase U can be measured by means of an oscilloscope and clamp-on ammeter.
7. Evaluate the step response:



8. Adjust the gain and reset time of the current controller.
9. Repeat steps 3 ... 6 iteratively until the optimum step response of the motor current is reached.
   - In the optimised state the current rise time typically is 0.5 ... 1 ms.
   - If the adjustment does not provide any satisfactory results, the current controller feedforward control via object 0x2941 (or 0x3141 for axis B) can be activated additionally. Then steps 3 ... 6 are to be repeated.
10. To stop the test mode:
    - Disable the i700 servo inverters.
    - Set object 0x2825 (or 0x3025 for axis B) to "0" to change back to the CiA402 mode.
11. For permanent storage: Upload changed current controller parameters (0x2942 or 0x3142 for axis B) from the i700 servo inverters into the Controller.

    The »EASY Starter« serves to save the parameter settings of the i700 servo inverters as parameter file (*.gdc). In the »PLC Designer«, this file can then be imported in the corresponding axis. ▶ Saving changed parameters safe against mains failure (📖 70)

5.12.2 Determining the total moment of inertia

The total moment of inertia is composed as follows:

Equation for calculating the total moment of inertia				
$J_{Total} = J_{Motor} + J_{Holding\ brake} + J_{Gearboxes}{}^{*} + J_{Load}{}^{*}$				
Axis A	Axis B	Symbol	Description	Dimension unit
-	-	J_{total}	Total moment of inertia	kg m^2
0x2910:1	0x3110:1	J_{Motor}	When a Lenze motor is selected, this value is automatically set.	kg cm^2
-	-	$J_{Holding\ brake}$	This value can be determined from the catalog data and added to the values mentioned above.	kg m^2
0x2910:2	0x3110:2	$J_{Gearbox}{}^{*}$ $J_{Load}{}^{*}$	Sum of all other moments of inertia of the system. Here, it is to be considered that these are values that have to be transformed to the motor shaft (marked by *).	kg cm^2

For determining the total moment of inertia, the i700 servo inverters is triggered by the higher-level Controller and a test path (motion profile) is executed. After a mathematical procedure, the total mass moment of inertia is determined from the detected speed and torque characteristics. The load moment of inertia results from the difference between the total moment of inertia and the moment of inertia of the motor.

The mass inertia of the motor can be obtained from the motor data sheet or it is entered automatically when a Lenze motor is used (0x2910:1). In addition, further moments of inertia must be taken into account.

Determining further moments of inertia

For determining further moments of inertia ($J_{Brake} + J_{Gearbox} + J_{Load}$), there are the following options:

- Extraction from the rated data of the drive dimensioning
- Determination based on data sheets of the components
- Calculation/estimation (for simple kinematic arrangements)
- Empirical determination by test mode with trapezoidal motion profile

You can use, for instance, the "Lenze Smart Formulas App" for the calculation.

Tip!

Lenze apps can be found on the Internet:

http://www.lenze.com → Download

The sum of the set moments of inertia ($J_{Motor} + J_{Gearbox}{}^{*} + J_{Load}{}^{*}$) forms the basis for an initial calculation of the speed controller gain with the "Speed controller: Calculate controller parameters" function (0x2822 or 0x3022 = 5).

Note!

The reduced moment of inertia of the load (0x2910:2 or 0x3110:2 for axis B) is included
- directly in the calculation of the speed controller (0x2822 or 0x3022 = 5) and
- indirectly in the calculation of the position controller (0x2822 or 0x3022 = 6).

Thus, give the data with all due consideration.

0x2910 | 0x3110 - Moments of inertia

Sub.	Name	Lenze setting	Data type
1	Moment of inertia: Motor	0.14 kg cm²	UNSIGNED_32
2	Moment of inertia: Load	0.00 kg cm²	UNSIGNED_32
3	Moment of inertia: Motor-load coupling	0: Rigid system	UNSIGNED_8

Subindex 1: **Moment of inertia: Motor**				
Setting range (min. value | unit | max. value)			Lenze setting	
0.00	kg cm²	20000000.00	0.14 kg cm²	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/100	UNSIGNED_32

Subindex 2: **Moment of inertia: Load**				
Setting range (min. value | unit | max. value)			Lenze setting	
0.00	kg cm²	20000000.00	0.00 kg cm²	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/100	UNSIGNED_32

Subindex 3: **Moment of inertia: Motor-load coupling**			
Selection list(Lenze setting printed in bold)			
0	**Rigid system**		
1	Elastic system		
2	System with backlash		
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			UNSIGNED_8

5.12.3 Setting the speed controller

When you select **5** ("speed controller: Calculate controller parameter"), the 0x2822 object (or 0x3022 for axis B) provides the automatic calculation of gain and reset time. For an optimal operation, we recommend a manual post-optimisation.

The following equations apply to a rigid system. For elastic systems and systems with backlash, the gain is reduced. The moment of inertia required for the calculation usually consists of the moment of inertia of the motor and the load moments of inertia transformed to the motor side.

Equations for calculating the gain and the reset time (applicable for rigid systems)

$$V_p = \frac{J}{a \cdot (T_{Filter} + T_{Current\ controller})} \cdot \frac{2\pi}{60}$$

$$T_n = a^2 \cdot (T_{Filter} + T_{Current\ controller})$$

Axis A	Axis B	Symbol	Description	Dimension unit
0x2900:1	0x3100:1	V_p	Speed controller gain	Nm/rpm
-	-	J	Moment of inertia = J_{Motor} + sum (J_{Load}) ▸ Determining further moments of inertia	kg m^2
-	-	a	Unit for the phase reserve (recommendation: a = 4 ≡ phase reserve of 60°)	
0x2904	0x3104	T_{filter}	Actual speed value filter time constant	s
-	-	$T_{current\ controller}$	Equivalent time constant of the current control loop = 0.0005 s (500 µs)	s
0x2900:2	0x3100:2	T_n	Speed controller reset time	s

Special case of the linear motor

Here a re-calculation from a linear system to a rotary system must be made. Therefore, via the feedback system a degree of freedom results for the determination of the number of pole pairs.

- For a rotary system, the number of pole pairs specifies the ratio of electrical and mechanical revolution, the number of encoder increments being defined via one mechanical revolution.
- In the case of a linear system, the user is free to decide for which length he or she wants to specify the number of encoder increments. Usually, the number of increments is given for a pole distance or for the total length of the linear scale. If the number of increments = "number of increments for one pole distance" is selected, a motor with the number of pole pairs zp = 1 is created.

The effective moment of inertia for a linear motor can be calculated according to the following equations. With this J value, the equations shown above can be used to calculate the speed controller gain and reset time.

Equations for calculating the effective moment of inertia for the linear motor

$$zp = \mathrm{Integer}\left(\frac{s}{2\tau_{Pole\ pair}}\right)$$

$$J = m \cdot \left(\frac{zp \cdot 2\tau_{Pole\ pair}}{2\pi}\right)^2$$

Axis A	Axis B	Symbol	Description	Dimension unit
-	-	s	Length which is the basis of the number of encoder increments. (e.g. pole spacing or total length)	m
-	-	$2\tau_{pole\ pair}$	Pole spacing of the permanent magnets, pole pair width	m
-	-	J	Moment of inertia = $J_{Forcer} + J_{Slide} + J_{Load}$	kg m^2
-	-	m	Moving mass = $m_{Forcer} + m_{Slide} + m_{Load}$	kg

0x2900 - speed controller: Parameter

Sub.	Name	Lenze setting	Data type
1	Speed controller: Gain	0.00033 Nm/rpm	UNSIGNED_32
2	Speed controller: Reset time	17.6 ms	UNSIGNED_16
3	Speed controller: Rate time	0.00 ms	UNSIGNED_16

Subindex 1: **Speed controller: Gain**				
Setting range (min. value | unit | max. value)			Lenze setting	
0.00000	Nm/rpm	20000.00000	0.00033 Nm/rpm	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/100000	UNSIGNED_32

Subindex 2: **Speed controller: Reset time**				
When the maximum value is set, the I component of the speed controller is deactivated. A pure P controller or PD controller is used.				
Setting range (min. value | unit | max. value)			Lenze setting	
1.0	ms	6000.0	17.6 ms	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	UNSIGNED_16

Subindex 3: **Speed controller: Rate time**				
Setting range (min. value | unit | max. value)			Lenze setting	
0.00	ms	3.00	0.00 ms	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/100	UNSIGNED_16

0x2901 | 0x3101 - Speed controller: Gain - adaptation

Setting range (min. value | unit | max. value)			Lenze setting	
0.00	%	200.00	100.00 %	
☑ Write access ☐ CINH ☑ OSC ☑ P ☑ RX ☐ TX			Scaling: 1/100	UNSIGNED_16

0x2902 | 0x3102 - Speed controller: Load value I component

Setting range (min. value | unit | max. value)			Lenze setting	
-1000.0	%	1000.0	0.0 %	
☑ Write access ☐ CINH ☑ OSC ☑ P ☑ RX ☐ TX			Scaling: 1/10	INTEGER_16

0x2903 | 0x3103 - Speed: Speed setpoint - filter time

The selected parameter setting of the speed controller can cause great overshoots in case of setpoint step-changes in the speed control loop. This can be dampened by this object. For optimising the control mode, the position controller must also be corrected if required, see Set position controller.

Setting range (min. value | unit | max. value)			Lenze setting	
0.0	ms	50.0	0.0 ms	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	UNSIGNED_16

0x2904 | 0x3104 - Speed: Actual speed - filter time

Time constant for actual speed filter

- In order to maximise the dynamics of the speed control loop, the actual speed filter should be operated with a time constant as low as possible. The lower the time constant the higher the gain of the speed controller. Since actual value filters have the task to dampen measuring errors or interference components, it must be found a compromise between filter task and the resulting delay.
- When a Lenze motor is selected, a starting value is automatically preset for the time constant in order that most of the applications will be feasible. Modifications required for individual cases have to be determined empirically, e.g. due to
  - higher requirements in the drive dynamics,
  - lower quality of the encoder signals,
  - very long encoder cables.

Setting range (min. value | unit | max. value)			Lenze setting	
0.0	ms	50.0	0.6 ms	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	UNSIGNED_16

5.12.4 Setting the position controller

For the automatic calculation of the gain, the "Position controller: Calculate controller parameters" function is provided via object 0x2822 (or 0x3022 for axis B).

Equations for calculating the gain				
$V_p = \frac{1}{32 \cdot T_{Sum}}$			$T_{Sum} = T_{Filter} + T_{Current\ controller}$	
Axis A	Axis B	Symbol	Description	Dimension unit
0x2980	0x3180	V_p	Position controller: Gain	Hz
0x2985	0x3185	$V_p(n)$	Speed-dependent V_p adaptation	
0x2904	0x3104	T_{filter}	Actual speed value filter time constant	s
-	-	$T_{current\ controller}$	Equivalent time constant of the current control loop = 0.0005 s (500 µs)	s

Instability of the position control loop due to too high dynamic performance of the speed controller

In case of high dynamic performance of the speed controller, please also observe the notes regarding the object 0x2903 or 0x3103.

The following countermeasure must be taken if the following error cannot be reduced to acceptable values while setting the position controller:

1. Reduce speed controller by the factor 2 and slowly increase the position controller until it gets slightly unstable again.
2. Reduce the position controller slightly and increase the speed controller until the position control loop gets slightly unstable again.
3. Repeat these steps until the following error is reduced to acceptable values.

0x2980 | 0x3180 - Position controller: Gain

Setting range (min. value | unit | max. value)			Lenze setting	
0.00	1/s	1000.00	28.40 1/s	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/100	UNSIGNED_32

0x2981 | 0x3181 - Position controller: Gain - adaptation

Setting range (min. value | unit | max. value)			Lenze setting	
0.00	%	200.00	100.00 %	
☑ Write access ☐ CINH ☑ OSC ☑ P ☑ RX ☐ TX			Scaling: 1/100	UNSIGNED_16

0x2982 | 0x3182 - Position controller: Output signal limitation

Setting range (min. value | unit | max. value)			Lenze setting	
0	[n unit]	480000	480000 [n unit]	
☑ Write access ☐ CINH ☑ OSC ☑ P ☑ RX ☐ TX			Scaling: $480000/2^{31}$	UNSIGNED_32

0x2983 | 0x3183 - set actual position: Starting value

Setting range (min. value | unit | max. value)			Lenze setting	
-2147483647	[Pos unit]	2147483647	0 [Pos unit]	
☑ Write access ☐ CINH ☑ OSC ☑ P ☑ RX ☐ TX				INTEGER_32

0x2984 | 0x3184 - set actual position: Mode

Selection list(Lenze setting printed in bold)			
0	**Absolute: (actual position = starting value)**		
1	Relative: (actual position = actual position + starting value)		
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			UNSIGNED_8

0x2985 | 0x3185 - position controller: Internal adaptation - characteristics

Sub.	Name	Lenze setting	Data type
1	Speed 1	200 rpm	UNSIGNED_16
2	Vp 1	150 %	UNSIGNED_16
3	Speed 2	400 rpm	UNSIGNED_16
4	Vp 2	130 %	UNSIGNED_16
5	Speed 3	600 rpm	UNSIGNED_16
6	Vp 3	110 %	UNSIGNED_16
7	Speed 4	1000 rpm	UNSIGNED_16
8	Vp 4	100 %	UNSIGNED_16
9	Speed 5	5000 rpm	UNSIGNED_16
10	Vp 5	100 %	UNSIGNED_16
11	Activation of adapted curve	0: Off	UNSIGNED_8

Subindex 1: **Speed 1**				
Setting range (min. value | unit | max. value)			Lenze setting	
1	r/min	50000	200 rpm	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 2: **Vp 1**				
Setting range (min. value | unit | max. value)			Lenze setting	
1	%	400	150 %	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 3: **Speed 2**				
Setting range (min. value | unit | max. value)			Lenze setting	
1	r/min	50000	400 rpm	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 4: **Vp 2**				
Setting range (min. value | unit | max. value)			Lenze setting	
1	%	400	130 %	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 5: **Speed 3**				
Setting range (min. value | unit | max. value)			Lenze setting	
1	r/min	50000	600 rpm	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 6: **Vp 3**				
Setting range (min. value | unit | max. value)			Lenze setting	
1	%	400	110 %	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 7: **Speed 4**	
Setting range (min. value | unit | max. value)	Lenze setting

Subindex 7: **Speed 4**				
1	r/min	50000	1000 rpm	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 8: **Vp 4**				
Setting range (min. value | unit | max. value)			Lenze setting	
1	%	400	100 %	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 9: **Speed 5**				
Setting range (min. value | unit | max. value)			Lenze setting	
1	r/min	50000	5000 rpm	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 10: **Vp 5**				
Setting range (min. value | unit | max. value)			Lenze setting	
1	%	400	100 %	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 11: **Activation of adapted curve**			
Selection list(Lenze setting printed in bold)			
0	**Off**		
1	On		
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			UNSIGNED_8

0x2986 | 0x3186 - Position controller:Resulting gain adaptation

Display range (min. value | unit | max. value)			Initialisation	
0.00	%	655.35		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☐ TX			Scaling: 1/100	UNSIGNED_16

5.12.5 Setting the field controller (ASM)

For motors with great rotor time constants or small rotor resistances, very high gain factors are calculated. Since the setting range of the field controller is limited to the double rated magnetising current, the field control loop in the case of these motors tends to a two-point response when the values calculated are entered.

Tip!

Starting from a calculated gain factor of approx. 1000 A/Vs, do not set the full value anymore.

Example: Calculated value = 10000 A/Vs → setting = 3000 A/Vs

For the automatic calculation of the controller parameters, the "Field controller: Calculate controller parameters" function is provided via object 0x2822 (or 0x3022 for axis B).

Equations for calculating the gain and the reset time

$$V_p \approx \frac{1}{4 \cdot R_r \cdot t_{\text{Current controller}}} \qquad \left(V_p = \frac{T_n}{4 \cdot K_{\text{Distance}} \cdot t_{\text{Current controller}}} = \frac{\frac{L_r}{R_r}}{4 \cdot L_H \cdot t_{\text{Current controller}}} \right)$$

$$T_n = T_r = \frac{L_r}{R_r}$$

Axis A	Axis B	Symbol	Description	Dimension unit
0x29C0:1	0x31C0:1	V_p	Field controller gain	A/Vs
0x29C0:2	0x31C0:2	T_n	Field contr. reset time	s
0x2C02:2	0x3402:2	L_h	Mutual motor inductance (ASM)	H
0x2C02:1	0x3402:1	R_r	Motor rotor resistance (ASM)	Ω
-	-	$T_{\text{current controller}}$	Equivalent time constant of the current control loop = 0.0005 s (500 µs)	s
-	-	T_r	Motor rotor time constant	
-	-	K_{path}	Gain of the controlled path	
-	-	L_r	Motor rotor inductance (ASM)	H

0x29C0 | 0x31C0 - Field controller: Parameter

Sub.	Name	Lenze setting	Data type
1	Field controller: Gain	165.84 A/Vs	UNSIGNED_32
2	Field controller: Reset time	15.1 ms	UNSIGNED_16

Subindex 1: **Field controller: Gain**				
Setting range (min. value | unit | max. value)			Lenze setting	
0.00	A/Vs	50000.00	165.84 A/Vs	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/100	UNSIGNED_32

Subindex 2: **Field controller: Reset time**				
Setting range (min. value | unit | max. value)			Lenze setting	
1.0	ms	6000.0	15.1 ms	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	UNSIGNED_16

5.12.6 Setting the field weakening controller (ASM)

Since the controlled system gain changes with the speed, the field weakening controller is corrected via the speed.

For the automatic calculation of the controller parameters, the "Field weakening controller: Calculate controller parameters" function is provided via object 0x2822 (or 0x3022 for axis B).

Equations for calculating the gain and the reset time

$$V_p = 0$$

$$V_{Distance\_Fs} = p \cdot n_{Transition} \cdot \frac{2\pi}{60}$$

$$T_n = 4 \cdot \frac{V_{Distance\_Fs}}{60} \cdot (T_{EF} + T_{Filter})$$

$$T_{EF} = T_r = \frac{L_r}{R_r} \approx \frac{L_h + L_{ss}}{R_r}$$

Axis A	Axis B	Symbol	Description	Dimension unit
0x29E0:1	0x31E0:1	V_p	Field weakening controller gain	Vs/V
-	-	V_{path_Fs}	Gain of the controlled path	
-	-	p	Number of pole pairs	
-	-	$n_{transition}$	Speed at which the field weakening is approximately initiated.	rpm
0x29E0:2	0x31E0:2	T_n	Field weak. contr. reset time	s
-	-	T_{EF}	Equivalent time constant of the field control loop	
0x29E3	0x31E3	T_{filter}	Filter time constant for the required voltage	s
-	-	T_r	Motor rotor time constant	
-	-	L_r	Motor rotor inductance	
0x2C02:2	0x3402:2	L_h	Mutual motor inductance (ASM)	H
0x2C01:3	0x3401:3	L_{ss}	Motor stator leakage inductance (ASM) or motor stator inductance (SM)	H
0x2C02:1	0x3402:1	R_r	Motor rotor resistance (ASM)	Ω

0x29E0 - field weakening controller: Parameter

Sub.	Name	Lenze setting	Data type
1	Field weakening controller: Gain	0.000 Vs/V	UNSIGNED_32
2	Field weakening controller: Reset time	2000.0 ms	UNSIGNED_32

Subindex 1: **Field weakening controller: Gain**				
When the minimum value is set, the P component of the field weakening controller is deactivated. A pure I controller is used.				
Setting range (min. value | unit | max. value)			Lenze setting	
0.000	Vs/V	2147483.647	0.000 Vs/V	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/1000	UNSIGNED_32

Subindex 2: **Field weakening controller: Reset time**				
Setting range (min. value | unit | max. value)			Lenze setting	
1.0	ms	240000.0	2000.0 ms	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	UNSIGNED_32

0x29E1 | 0x31E1 - field: Field set value limitation

Setting range (min. value | unit | max. value)			Lenze setting	
5.00	%	100.00	100.00 %	
☑ Write access ☐ CINH ☑ OSC ☑ P ☑ RX ☐ TX			Scaling: 1/100	UNSIGNED_16

0x29E2 | 0x31E2 - DC link circuit voltage: Filter time

Setting range (min. value | unit | max. value)			Lenze setting	
1.0	ms	1000.0	25.0 ms	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	UNSIGNED_16

0x29E3 | 0x31E3 - motor: Actual voltage - filter time

Setting range (min. value | unit | max. value)			Lenze setting	
1.0	ms	1000.0	25.0 ms	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	UNSIGNED_16

0x29E4 | 0x31E4 - Voltage reserve range

Voltage reserve at the transition point to field weakening
From software version V01.03.xx onwards
- Only relevant for servo control with asynchronous motor (selection "2" in 0x2C00 or 0x3400 for axis B).

From software version V01.10.xx or V02.10.xx onwards
- Relevant for servo control with synchronous motor (selection "1" in 0x2C00 or 0x3400 for axis B).
  - For more information, see Delaying the build-up of field weakening.
- Relevant for servo control with asynchronous motor (selection "2" in 0x2C00 or 0x3400 for axis B).

Setting range (min. value | unit | max. value)			Lenze setting	
1	%	20	5 %	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_8

5.12.7 Field weakening operation - synchronous motor (SM)

The control of the i700 servo inverters permits the operation of a synchronous motor outside the voltage range. If a motor is selected in the »Easy Starter«, the control is automatically parameterised.

In order to improve the transition from the base speed range to field weakening, we recommend the activation of the current controller feedforward control via the object 0x2941 (or 0x3141 for axis B).

- The current controller feedforward control is determined via the following parameters:
  - Motor stator resistance (0x2C01:2 or 0x3401:2 for axis B)
  - Motor stator leakage inductance (0x2C01:3 or 0x3401:3 for axis B)
  - e.m.f constant (0x2C03:1 or 0x3403:1 for axis B)
- If a third-party motor is to be operated in the field weakening range, the parameters mentioned before have to be selected carefully.

Stop!

Operation of synchronous motors outside the voltage range:

If a pulse inhibit is set in the i700 servo inverters, e.g. in the case of a controller inhibit or an error, the DC bus will load up to the terminal voltage according to the current speed (see equation below).

- At high speeds outside the voltage range, the terminal voltage may be higher than the mains voltage!
- Connect a brake chopper to the DC bus to prevent the DC bus from reaching impermissibly high voltages!

The terminal voltage approximately corresponds to the following equation:

Equations for calculating the terminal voltage

$$V_{\text{Terminal}} = \text{Actual speed} \cdot \frac{\text{Rated mains voltage}}{\text{Rated motor speed}}$$

Axis A	Axis B	Description	Dimension unit
0x2540:1		Rated mains voltage	V
0x2C01:4	0x3401:4	Rated motor speed	r/min

Delaying the build-up of field weakening

In case of a factory-set setting (5 %), field weakening is started just before the voltage limit is reached for a timely build-up of the field weakening current.

From software version V01.10.xx / V02.10.xx onwards

In case of synchronous motors, the setting of the 0x29E4 object can delay the field weakening for synchronous machines to e.g. slightly reduce the thermal load of the motor.

In contrast to the asynchronous machine, the voltage reserve with the settings 1 ... 5 results in significant differences:

setting	Voltage reserve ASM [%]	Voltage reserve SM [%]
1	1	0
2	2	3
3	3	6
4	4	9
5 ... 20	5 ... 20	12

From software version V01.11.xx / V02.11.xx onwards

setting	Voltage reserve ASM / SM [%]
1 ... 20	1 ... 20

5.13 Fine adjustment des motor model

The further commissioning steps are only required for servo controls if more stringent requirements with regard to the torque linearity have to be met. During the commissioning process of Lenze motors, typical values for the relevant parameters are provided. For motors of other manufacturers, these values are to be requested from the motor manufacturer, or they have to be estimated.

5.13.1 Correction of the stator leakage inductance (Lss)...

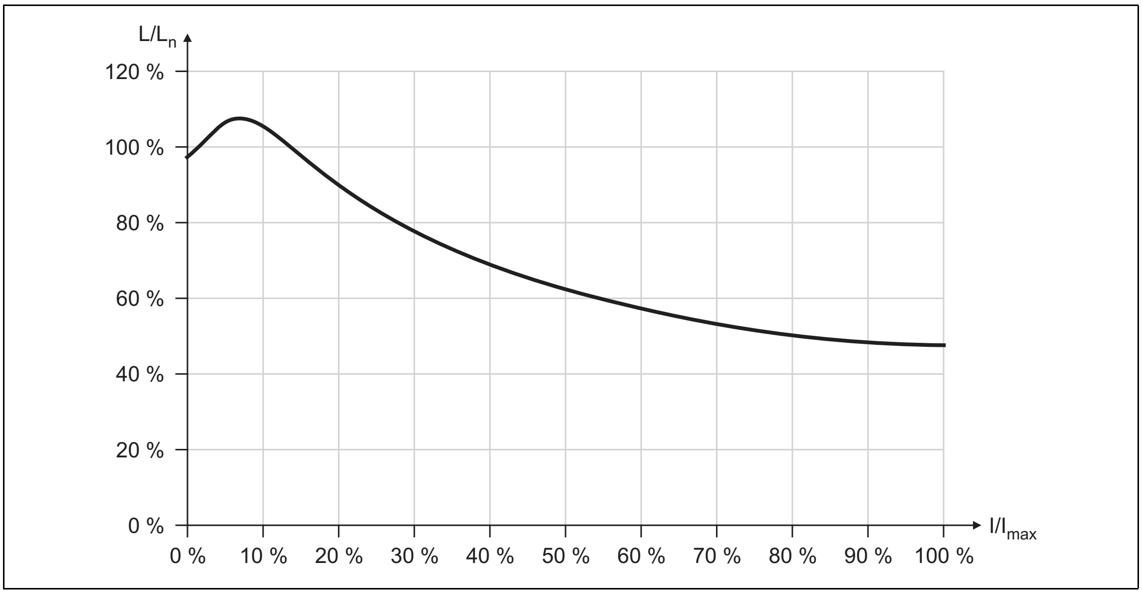
...and the current controller parameters by means of the saturation characteristic

For the most part, the electrical characteristics of the motor are the relevant factors for an optimal current controller setting (Vp, Ti), especially the stator resistance and the stator leakage inductance (Lss). However, modern motors have their stator leakage inductance changed along with the current level so that it is impossible to have an optimal current controller setting for all working points at all times.

For applications with operating phases that involve very different current and torque requirements and, at the same time, high requirements on dynamic drive behaviour, the i700 servo inverters provides the possibility of the correction of the stator leakage inductance and the current controller settings by means of the adjustable saturation characteristic.

The saturation characteristic is a typical characteristic of motors of one type/size. It does not depend on the maximum process current of the motor in the prevailing application. Thus the defined values should be based on the key data of the motors. These are rated motor current, peak motor current for a limited time and the ultimate motor current.

The following picture shows a typical saturation characteristic of an MCS motor:



[5-4] Saturation characteristic: Inductance referring to the inductance for rated current

The saturation characteristic represents the change in inductance (L/Ln) as a function of the motor current (I/Imax). The variables of both axes which were scaled to a reference value are represented as percentages.

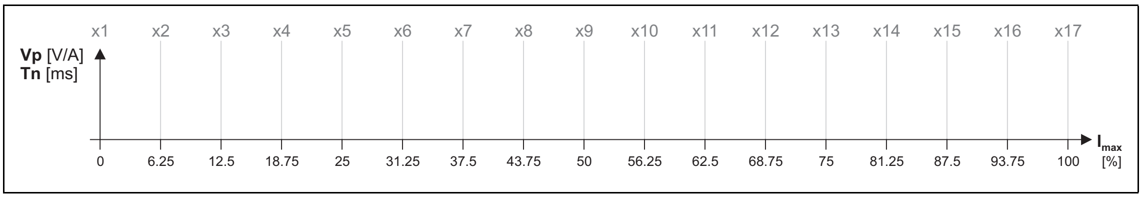
- When a Lenze motor is selected, the saturation characteristic is already filled with values typical of the series.

Note!

The saturation characteristic is not only used to correct the current controller, but it also influences the current controller feedforward control (can be activated via object 0x2941 or 0x3141 for axis B).

Distribution of the grid points

- The saturation characteristic is represented by means of 17 grid points.
- The 17 grid points are spaced on the X axis at equal intervals (equidistantly) in a range of 0 ... 100 %. The 100% value of the X axis refers to the current value (max. motor current in the process) set in object 0x2C05 (or 0x3405 for axis B).
- The Y values for the grid points can be accessed via the subindices of object 0x2C04 (or 0x3404 for axis B).



[5-5] Saturation characteristic: Distribution of the grid points

- The 100 % value of a grid point refers to
  - the set motor stator leakage inductance (0x2C01:3 or 0x3401:3 for axis B) and
  - the set current controller gain Vp (0x2942:1 or 0x3142:1 for axis B).
- Preferably select a display area of the grid points which includes at least the ultimate motor current. The current controller step response is then recorded actively only until the grid point with peak motor current. In order to prevent the motor winding from being overloaded, the manual test mode "current pulse" should be used for recording. The grid points with current setpoints above the peak motor current are determined through interpolation.
- When the saturation characteristics for motor types are determined, it makes sense in some cases to select a scaled representation of the grid point distribution. This requires to know the highest value of the quotient from "ultimate motor current / rated motor current" of the motor series.

_ _

0x2C04 | 0x3404 - Motor: Lss saturation characteristic - inductance grid points (y)

From software version V01.05.xx onwards, the correction by means of saturation characteristic via subindex 18 can also be switched off.

Sub.	Name	Lenze setting	Data type
1	Lss: y1 = L01 (x = 0.00 %)	165 %	UNSIGNED_16
2	Lss: y2 = L02 (x = 6.25 %)	200 %	UNSIGNED_16
3	Lss: y3 = L03 (x = 12.50 %)	146 %	UNSIGNED_16
4	Lss: y4 = L04 (x = 18.75 %)	117 %	UNSIGNED_16
5	Lss: y5 = L05 (x = 25.00 %)	97 %	UNSIGNED_16
6	Lss: y6 = L06 (x = 31.25 %)	82 %	UNSIGNED_16
7	Lss: y7 = L07 (x = 37.50 %)	71 %	UNSIGNED_16
8	Lss: y8 = L08 (x = 42.75 %)	62 %	UNSIGNED_16
9	Lss: y9 = L09 (x = 50.00 %)	55 %	UNSIGNED_16
10	Lss: y10 = L10 (x = 56.25 %)	50 %	UNSIGNED_16
11	Lss: y11 = L11 (x = 62.50 %)	46 %	UNSIGNED_16
12	Lss: y12 = L12 (x = 68.75 %)	43 %	UNSIGNED_16
13	Lss: y13 = L13 (x = 75.00 %)	42 %	UNSIGNED_16
14	Lss: y14 = L14 (x = 81.25 %)	41 %	UNSIGNED_16
15	Lss: y15 = L15 (x = 87.50 %)	41 %	UNSIGNED_16
16	Lss: y16 = L16 (x = 93.25 %)	41 %	UNSIGNED_16
17	Lss: y17 = L17 (x = 100.00 %)	41 %	UNSIGNED_16
18	Motor: Lss saturation characteristic - Activation	1: Active	UNSIGNED_16
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			

0x2C05 | 0x3405 - Motor: Lss saturation charcteristic - reference for current grid points (x)

Maximum motor current
- Serves as a reference value for the scaled current values of the X axis of the saturation characteristic.

Setting range (min. value | unit | max. value)			Lenze setting	
0.0	A	500.0	5.4 A	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	UNSIGNED_16

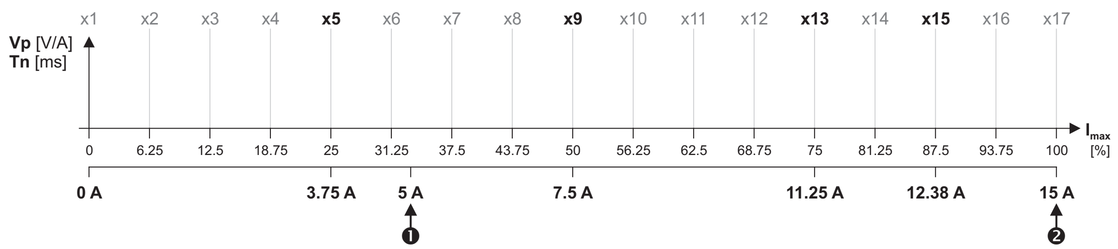
5.13.1.1 Example for determining the saturation characteristic

Given values:

- Rated motor current: 5 A
- Maximum motor current: 20 A
- Maximum process current: 15 A

Procedure:

1. Deactivate correction: Set all subindices of object 0x2C04 (or 0x3404 for axis B) to 100 %.
2. Use object 0x2C05 (or 0x3405 for axis B) to set the maximum current up to which the motor is to be actuated in the process (in this example "15 A").
3. Adjust the current controller with different current setpoints by means of the manual test mode "current pulse" and take down the corresponding settings for Vp and Tn.
   - The procedure is described in the Manual test mode "Current pulse" chapter. (📖 149)
   - The current setpoints to be set for the corresponding adjustment in object 0x2835:1 (or 0x3035:1 for axis B) result from the scaling of the maximum process current to the X axis of the saturation characteristic.
   - The grid points which are required to define the saturation characteristic with a sufficient quality varies from motor to motor and thus has to be determined individually.
   - For this example, currents that are part of the grid points 5, 9, 13, and 15 have been selected, and a measurement at rated motor current was carried out additionally:

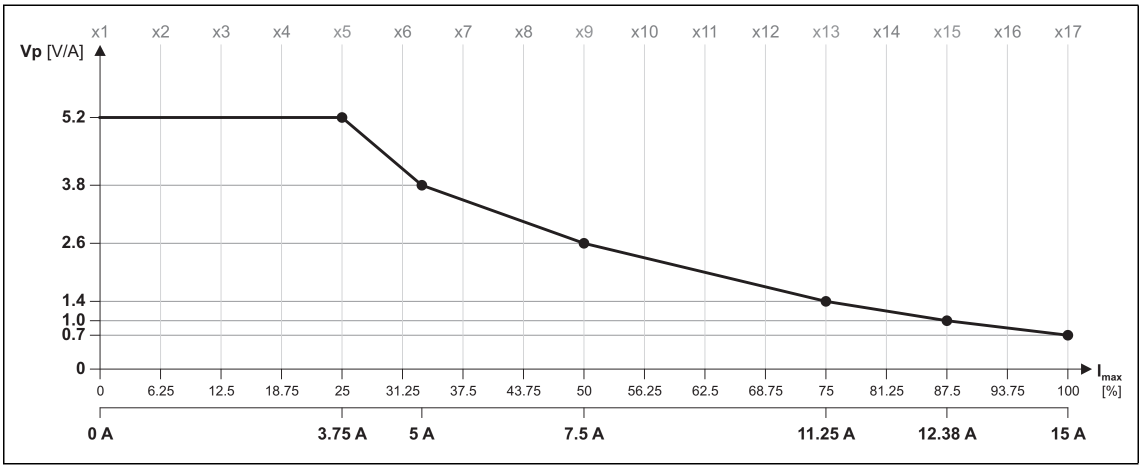


❶ Rated motor current
❷ Maximum process current (≡ 100 %)

[5-6] Saturation characteristic: Distribution of the grid points

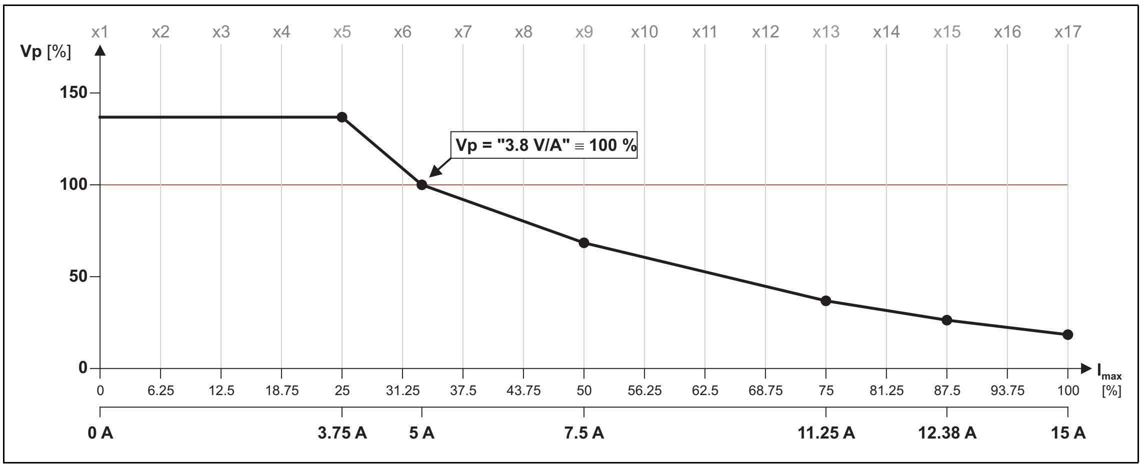
Specifications for adjustment			Measured values	
Grid point	**Standardisation**	**Current setpoint**	**Vp [V/A]**	**Tn [ms]**
5	0.25 * 15 A =	3.75 A	5.2	6.5
9	0.5 * 15 A =	7.5 A	2.6	4
13	0.75 * 15 A =	11.25 A	1.4	2.5
15	0.875 * 15 A =	12.38 A	1.0	2
17	1.0 * 15 A =	15 A	0.7	1.7
	Rated motor current =	5 A	3.8	5

4. Set the characteristic by means of the determined values for Vp (but do not enter any values in 0x2C04 or 0x3404 for axis B yet).
   - Here, the values of the grid points which have not been adjusted must be determined by interpolation between two values.
   - **Note:** In this example it was assumed that the inductance does not change considerably below 3.75 A. For this reason the same Vp value resulting from a measurement with a motor current of 3.75 A was used for all grid points below 3.75 A.



[5-7] Determined saturation characteristic

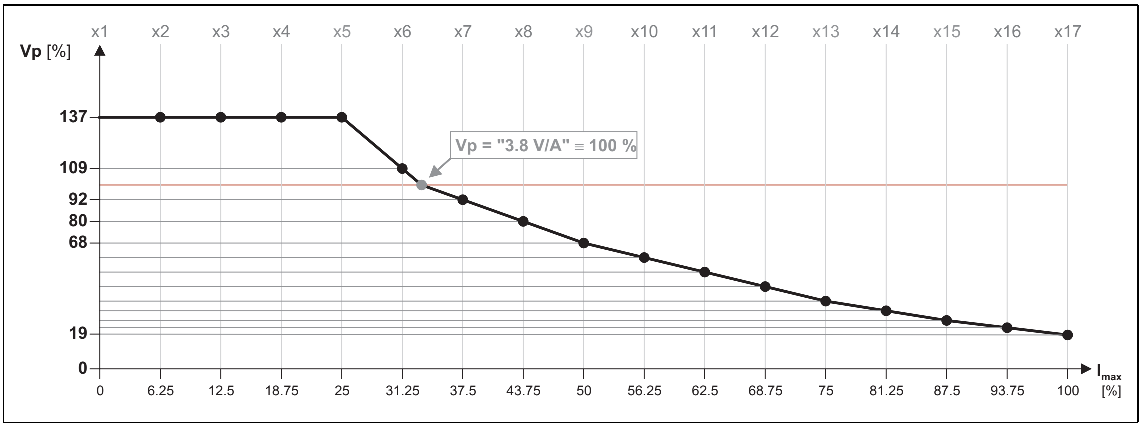
5. Set the gain Vp and the reset time Tn to the values that have been determined during the adjustment with rated motor current (in this example "5 A"):
   - Set 0x2942:1 (or 0x3142:1 for axis B) = "3.8 V/A".
   - Set 0x2942:2 (or 0x3142:2 for axis B) = "5 ms".
6. Scale the Vp values on the Y axis of the characteristic to a Vp setting of "3.8 V/A":



[5-8] Scaling of the determined saturation characteristic to "100 % Vp"

7. Enter the Vp values in per cent from the grid points into the subindices of object 0x2C04 (or 0x3404 for axis B):

Setting for grid points 1 ... 17 in [%]																
y1	y2	y3	y4	y5	y6	y7	y8	y9	y10	y11	y12	y13	y14	y15	y16	y17
137	137	137	137	137	109	92	80	68	61	53	45	37	32	26	22	19



[5-9] Grid point values of the saturation characteristic determined

8. Enter the maximum process current ("15 A") as maximum current in object 0x6073 (or 0x6873 for axis B).
   - With these settings, the same current characteristic should occur, irrespective of the current magnitude.
   - Since the current controller gain is now corrected actively, the step responses may differ slightly compared to the previous measurements. In this case, the current controller parameters must be optimised one last time.
9. For permanent storage: Upload the detected characteristic from the i700 servo inverters into the Controller.

   The »EASY Starter« serves to save the parameter settings of the i700 servo inverters as parameter file (*.gdc). In the »PLC Designer«, this file can then be imported in the corresponding axis. ▶ Saving changed parameters safe against mains failure (📖 70)

5.13.2 Synchronous motor (SM): Compensating for temperature and current influences

The properties of the permanent magnets of permanently excited synchronous motors (SM, PSM) depend on the temperature and the amperage. The relationship between motor current and resulting torque changes correspondingly.

The influences of the temperature and the amperage on the magnetisation can be taken into account by the motor control and hence be compensated for.

- To compensate for the temperature dependence of the magnets, the temperature coefficient (kT) of the permanent magnet must be entered in object 0x2C03:3 (or 0x3403:3 for axis B) (linear characteristic).
- To compensate for the current dependence of the magnets, multiple grid points of a characteristic must be entered in the following object (non-linear characteristic):

0x2C06 | 0x3406 - Motor (SM): Magnet characteristic (current) - grid points

Sub.	Name	Lenze setting	Data type
1	Magnet characteristic: x1 = i01/iN	0 %	UNSIGNED_16
2	Magnet characteristic: y1 = kT01/kTN	100 %	UNSIGNED_16
3	Magnet characteristic: x2 = i02/iN	100 %	UNSIGNED_16
4	Magnet characteristic: y2 = kT02/kTN	100 %	UNSIGNED_16
5	Magnet characteristic: x3 = i03/iN	200 %	UNSIGNED_16
6	Magnet characteristic: y3 = kT03/kTN	100 %	UNSIGNED_16
7	Magnet characteristic: x4 = i04/iN	415 %	UNSIGNED_16
8	Magnet characteristic: y4 = kT04/kTN	72 %	UNSIGNED_16
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			

5.13.3 Asynchronous motor (ASM): Identifying the Lh saturation characteristic

In case of an asynchronous motor, the relationship between current and torque is basically determined by the saturation behaviour of the mutual inductance. If the achieved torque accuracy, especially in the field weakening range should not be sufficient, the accuracy can be increased by the individual identification of the saturation characteristic. This behaviour can be measured by the i700 servo inverters.

Preconditions for the execution

- Before this commissioning function is executed, the inverter characteristic and the motor parameters have to be identified.
  ▶ Compensating for inverter influence on output voltage (📖 95)
  ▶ Determine motor parameters automatically via "motor parameter identification" (📖 104)
- The motor may be firmly braked.
- The i700 servo inverters is free of errors and is in the "Switched on" device status.

Response of the motor during the execution

Standstill

How to identify the LH saturation characteristic:

1. If the i700 servo inverters is enabled, disable the i700 servo inverters.
   ▶ Enable/inhibit operation via control word (📖 69)
2. Set object 0x2825 (or 0x3025 for axis B) to "10" to change to the "Determine Lh saturation characteristicn" operating mode.
3. Enable the i700 servo inverters to start the procedure.

   Notes:
   - The identification of the Lh saturation characteristic can take up to 11 minutes. The progress can be checked in the object 0x2823 (or 0x3023 for axis B).
   - By means of controller inhibit, the started procedure can be cancelled any time, if required. Characteristic values that have already been determined are rejected in this case.

After successful completion...

...the controller will be inhibited automatically and the points of the determined LH characteristic will be set in object 0x2C07 (or 0x3407 for axis B).

- For permanent storage, the changed settings must be uploaded to the controller from the i700 servo inverters.
  The »EASY Starter« serves to save the parameter settings of the i700 servo inverters as parameter file (*.gdc). In the »PLC Designer«, this file can then be imported in the corresponding axis. ▶ Saving changed parameters safe against mains failure (📖 70)
- The controller inhibit automatically set by the procedure can be deactivated via the Controlword (0x6040 or 0x6840 for axis B) (setting = 7, 15).

In the event of an error

If an error occurs during the procedure or the pulse inhibit gets active (e.g. due to short-time undervoltage), the procedure is terminated with controller inhibit without a change in settings.

Loading the standard Lh saturation characteristic

If an incorrect Lh saturation characteristic has been determined or none at all, it is possible to load a device-typical standard Lh characteristic.

How to load the standard Lh saturation characteristic:

1. Set object 0x2822 (or 0x3022 for axis B) to "13".
   - The progress of the procedure is shown in object 0x2823 (or 0x3023 for axis B).
2. For permanent storage: After completion of the procedure, upload the Lh saturation characteristic set in 0x2C07 (or 0x3407 for axis B) to the controller from the i700 servo inverters.

   The »EASY Starter« serves to save the parameter settings of the i700 servo inverters as parameter file (*.gdc). In the »PLC Designer«, this file can then be imported in the corresponding axis. ▶ Saving changed parameters safe against mains failure (📖 70)

0x2C07 | 0x3407 - Motor (ASM): Lh saturation characteristic - inductance grid points (y)

Sub.	Name	Lenze setting	Data type
1	Lh: y1 = L01 (x = 0.00 %)	118 %	UNSIGNED_16
2	Lh: y2 = L02 (x = 6.25 %)	118 %	UNSIGNED_16
3	Lh: y3 = L03 (x = 12.50 %)	118 %	UNSIGNED_16
4	Lh: y4 = L04 (x = 18.75 %)	117 %	UNSIGNED_16
5	Lh: y5 = L05 (x = 25.00 %)	116 %	UNSIGNED_16
6	Lh: y6 = L06 (x = 31.25 %)	114 %	UNSIGNED_16
7	Lh: y7 = L07 (x = 37.50 %)	111 %	UNSIGNED_16
8	Lh: y8 = L08 (x = 43.75 %)	107 %	UNSIGNED_16
9	Lh: y9 = L09 (x = 50.00 %)	100 %	UNSIGNED_16
10	Lh: y10 = L10 (x = 56.25 %)	93 %	UNSIGNED_16
11	Lh: y11 = L11 (x = 62.50 %)	86 %	UNSIGNED_16
12	Lh: y12 = L12 (x = 68.75 %)	78 %	UNSIGNED_16
13	Lh: y13 = L13 (x = 75.00 %)	71 %	UNSIGNED_16
14	Lh: y14 = L14 (x = 81.25 %)	64 %	UNSIGNED_16
15	Lh: y15 = L15 (x = 87.50 %)	57 %	UNSIGNED_16
16	Lh: y16 = L16 (x = 93.75 %)	50 %	UNSIGNED_16
17	Lh: y17 = L17 (x = 100.00 %)	42 %	UNSIGNED_16
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			

5.13.4 Estimating the optimal magnetising current

In case of the given Lh saturation behaviour, there is (usually) a magnetising current where the torque efficiency is highest. This magnetising current can be determined by the i700 servo inverters.

- The execution of this function also changes the Lh saturation characteristic (0x2C07 or 0x3407 for axis B) (compressed, extended).
- After executing the function, the magnetising current determined is entered in object 0x2C02:3 (or 0x3402:3 for axis B).

Preconditions for the execution

- Before this commissioning function is executed, the motor parameters and the Lh saturation characteristic have to be identified.
  ▶ Determine motor parameters automatically via "motor parameter identification" (📖 104)
  ▶ Asynchronous motor (ASM): Identifying the Lh saturation characteristic (📖 174)
- The motor may be firmly braked.

Response of the motor during the execution

Standstill

How to estimate the optimal magnetising current:

1. Set object 0x2822 (or 0x3022 for axis B) to "2".
   - The progress of the procedure is shown in object 0x2823 (or 0x3023 for axis B).
2. For permanent storage: Upload the changed controller parameters to the controller from the i700 servo inverters after the procedure has been completed:
   - Lh saturation characteristic (0x2C07 or 0x3407 for axis B)
   - Magnetising current (0x2C02:3 or 0x3402:3 for axis B)

   The »EASY Starter« serves to save the parameter settings of the i700 servo inverters as parameter file (*.gdc). In the »PLC Designer«, this file can then be imported in the corresponding axis. ▶ Saving changed parameters safe against mains failure (📖 70)

5.14 Parameterising filter elements in the setpoint path

5.14.1 Jerk limitation

Max. acceleration change

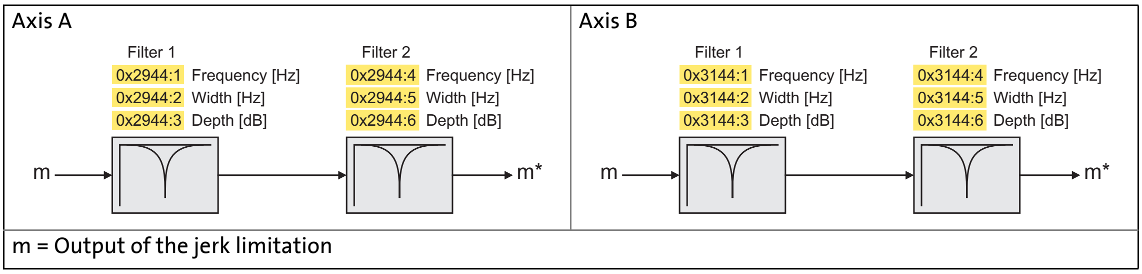
0x2945 | 0x3145 - Torque: Setpoint jerk limitation

Setting range (min. value | unit | max. value)			Lenze setting	
0.1	%	400.0	400.0 %	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	UNSIGNED_16

5.14.2 Notch filters (band-stop filters)

Due to the high dynamic performance or the high limit frequency of the closed current control loop, mechanical natural frequencies can be excited, which can result in resonance and thus cause the speed control loop to become unstable.

In order to suppress or damp these resonant frequencies, two notch filters are integrated in the speed control loop of the controller, which can be parameterised. In the Lenze setting, these filters are switched off:



[5-10] Optional notch filters (filter cascade) in the speed control loop

Use of the notch filters depending on the resonant frequency

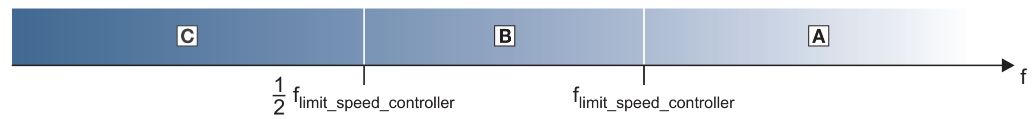
Stop!

Improperly set notch filters have a negative effect on the response and disturbance behaviour of the speed control:

- Increased overshoot of the motor speed in case of response behaviour
- Higher speed deviations (extreme case: Complete instability of the drive) in case of disturbance behaviour

In the case of impairment,

- the drive that is still running must either be coasted down by activating the controller inhibit or immediately be brought to a standstill via a brake.
- the speed controller must be optimised again afterwards.
- the test procedure must be repeated. ▶ Setting the notch filters



A Use of notch filters possible without any interactions
B Use of notch filters only possible with restrictions
C The notch filter should not be used

[5-11] Use of the notch filter depending on the resonant frequency

- The notch filters are suitable for use with resonant frequencies equal to or higher than the limit frequency of the speed controller.
  - Resonant frequencies ≥ $f_{limit\_speed\_controller}$ = 70 Hz ... 110 Hz
- For resonant frequencies lower than the limit frequency of the speed controller, the use of suitable speed profiles with an S-shaped ramp is recommended.

Setting the notch filters

Since the exact frequency response of the speed control path in most cases is not known beforehand, an experimental procedure for setting the notch filters is described in the following.

How to set the notch filters:

1. Setting and optimising the current controller.
2. Adapt the speed controller reset time to the filter time constant of the speed filter time and the equivalent time constant of the current control loop:
   - The following applies to axis A: 0x2900:2 = 16 * (0x2904 + 500 µs)
   - The following applies to axis B: 0x3100:2 = 16 * (0x3104 + 500 µs)

   Note: The setting of the reset time includes the equivalent time constant of the current control loop. The 500 µs indicated are typical in a power range of up to 50 kW. Above this value, greater time constants may occur.
3. Slowly increase the proportional gain of the speed controller in 0x2900:1 (or 0x3100:1 for axis B) until the speed control loop starts to be unstable (acoustic determination, measurement of the motor current or recording of the speed output signal).
4. Measure the oscillation frequency using an oscilloscope:
   - Assessing the motor current via 0x2DD1:4
   - Assessing the motor speed via 0x6044.
5. Set the oscillation frequency determined as filter frequency in 0x2944:1 (or 0x3144:1 for axis B).
6. Set the filter width to 40 % of the filter frequency in 0x2944:2 (or 0x3144:40 for axis B).
   - Example: Filter frequency = 250 Hz → filter width = 100 Hz.
7. Set the filter depth to 40 dB in 0x2944:3 (or 0x3144:3 for axis B).
   - If "0 dB" are set (default setting), the filter is not effective.

8. Further increase proportional gain of the speed controller in 0x2900:1 (or 0x3100:1 for B) until the speed control loop starts to be unstable again.
   - If the oscillation frequency has changed now, readjust the filter frequency by trimming. The use of a second filter is ineffective here.
   - If the oscillation frequency remains the same, readjust the filter depth and/or the filter width by trimming (the first reduces the amplitude, the second lets the phase rotate faster).
   - Repeat step 8 until the desired behaviour or the limit of a sensible speed controller gain has been reached.
9. Check the drive behaviour in case of quick stop (QSP)
   - Accelerate drive
   - Then, brake with quick stop (QSP) and check whether a reduced drive dynamics can be detected.
   - If so, reduce the influence of the filters until the reachable dynamics corresponds to the requirements.

Note!

Readjust the speed controller after setting the notch filters. ▶ Setting the speed controller. (📖 154)

For permanent storage, the changed settings must be uploaded to the controller from the i700 servo inverters.

The »EASY Starter« serves to save the parameter settings of the i700 servo inverters as parameter file (*.gdc). In the »PLC Designer«, this file can then be imported in the corresponding axis. ▶ Saving changed parameters safe against mains failure (📖 70)

0x2944 | 0x3144 - Torque: Notch filter setpoint torque

Sub.	Name	Lenze setting	Data type
▸ 1	Notch filter 1: Frequency	200.0 Hz	UNSIGNED_16
▸ 2	Notch filter 1: Bandwidth	20.0 Hz	UNSIGNED_16
▸ 3	Notch filter 1: Damping	0 db	UNSIGNED_8
▸ 4	Notch filter 2: Frequency	400.0 Hz	UNSIGNED_16
▸ 5	Notch filter 2: Bandwidth	40.0 Hz	UNSIGNED_16
▸ 6	Notch filter 2: Damping	0 db	UNSIGNED_8

Subindex 1: **Notch filter 1: Frequency**				
Setting range (min. value | unit | max. value)			Lenze setting	
1.0	Hz	1000.0	200.0 Hz	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	UNSIGNED_16

Subindex 2: **Notch filter 1: Bandwidth**				
Setting range (min. value | unit | max. value)			Lenze setting	
0.0	Hz	500.0	20.0 Hz	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	UNSIGNED_16

Subindex 3: **Notch filter 1: Damping**				
Setting range (min. value | unit | max. value)			Lenze setting	
0	dB	100	0 db	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_8

Subindex 4: **Notch filter 2: Frequency**				
Setting range (min. value | unit | max. value)			Lenze setting	
1.0	Hz	1000.0	400.0 Hz	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	UNSIGNED_16

Subindex 5: **Notch filter 2: Bandwidth**				
Setting range (min. value | unit | max. value)			Lenze setting	
0.0	Hz	500.0	40.0 Hz	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	UNSIGNED_16

Subindex 6: **Notch filter 2: Damping**				
Setting range (min. value | unit | max. value)			Lenze setting	
0	dB	100	0 db	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_8

5.15 Parameterising the V/f characteristic control

With the V/f characteristic control (VFCplus), the motor voltage of the inverter is determined by means of a linear or quadratic characteristic depending on the field frequency or motor speed to be generated. The voltage follows a preselected characteristic.

Stop!

- The V/f characteristic control is only suitable for asynchronous motors.
- Observe the following when you actuate drives with a square-law V/f characteristic:
  - Please always check whether the corresponding drive is suitable for operation with a quadratic V/f characteristic!
  - If your pump drive or fan drive is not suitable for operation with a square-law V/f characteristic, use the linear V/f characteristic instead, or select the servo control.
- For adjustment, observe the thermal performance of the connected asynchronous motor at low output frequencies.
  - Usually, standard asynchronous motors with insulation class B can be operated for a short time with their rated current in the frequency range 0 Hz ... 25 Hz.
  - Contact the motor manufacturer to get the exact setting values for the max. permissible motor current of self-ventilated motors in the lower speed range.
  - For square-law V/f characteristics we recommend setting a smaller V_{min}.
- As regards the motor nameplate data, at least the rated speed (0x2C01:4 or 0x3401:4 for axis B) and the rated frequency (0x2C01:5 or 0x3401:5 for axis B) must be entered for the i700 servo inverters to calculate the correct number of pole pairs.

Initial commissioning steps

After the motor and i700 servo inverters have been optimally adjusted to each other, the following "initial commissioning steps" are sufficient for a simple V/f characteristic control:

1. Defining the V/f characteristic shape
2. Activating the voltage vector control (Imin controller)
   – or alternatively –
   Setting the voltage boost
3. Setting the load adjustment
4. Defining the behaviour at the current limit (Imax controller)
5. Deactivating or parameterising feedbacks

Optimising the control mode

The following "optimisation steps" can be used to further optimise the control mode of the V/f characteristic control and adapt it to the practical application:

1. Setting the slip compensation
2. Setting the oscillation damping
3. Manual test mode "Current pulse"
   - Only required if the voltage vector control is applied, or if DC-injection braking or the flying restart process is activated.

Parameterisable functions

Optionally the following functions can be activated for the V/f characteristic control:

- "DC-injection braking" function
- "Flying restart" function

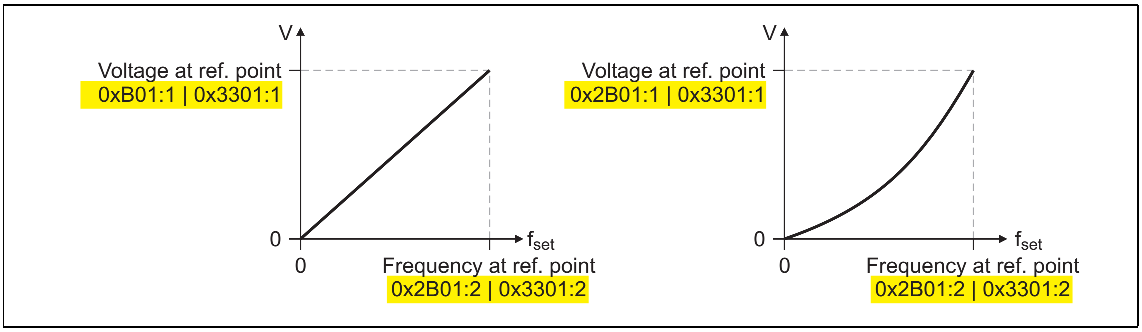
In the Lenze setting these two functions are deactivated.

5.15.1 Defining the V/f characteristic shape

For purposes of adaption to different load profiles, the shape of the characteristic can be selected:

0x2B00 | 0x3300 - VFC: V/f characteristic - shape

Selection list(Lenze setting printed in bold)		Info
0	**Linear (standard)**	Linear characteristic for drives with constant load torque over the speed.
1	Square-law (pumps and fans)	Square-law characteristic for drives with a linear or square-law load torque over the speed. • Square-law V/f characteristics are preferably used for centrifugal pumps and fan drives. • Please always check whether the corresponding drive is suitable for operation with a quadratic V/f characteristic! • If your pump drive or fan drive is not suitable for operation with a square-law V/f characteristic, use the linear V/f characteristic instead, or select the servo control.
2	User-defined	User-definable V/f characteristic for drives requiring an adaptation of the magnetising current via the output speed. • The user-definable V/f characteristic can for instance be used for operation on special machines like the reluctance motor, for the suppression of oscillations on the machine, or for energy optimisation.
☑ Write access ☑ CINH ☐ OSC ☑ P ☐ RX ☐ TX		UNSIGNED_8



[5-12] Principle of a linear V/f characteristic (on the left) and a quadratic V/f characteristic (on the right)

0x2B01 | 0x3301 - VFC: V/f characteristic - define reference point

Sub.	Name	Lenze setting	Data type
1	VFC: V/f characteristic - voltage at reference point	225 V	UNSIGNED_16
2	VFC: V/f characteristic - frequency at reference point	270 Hz	UNSIGNED_16

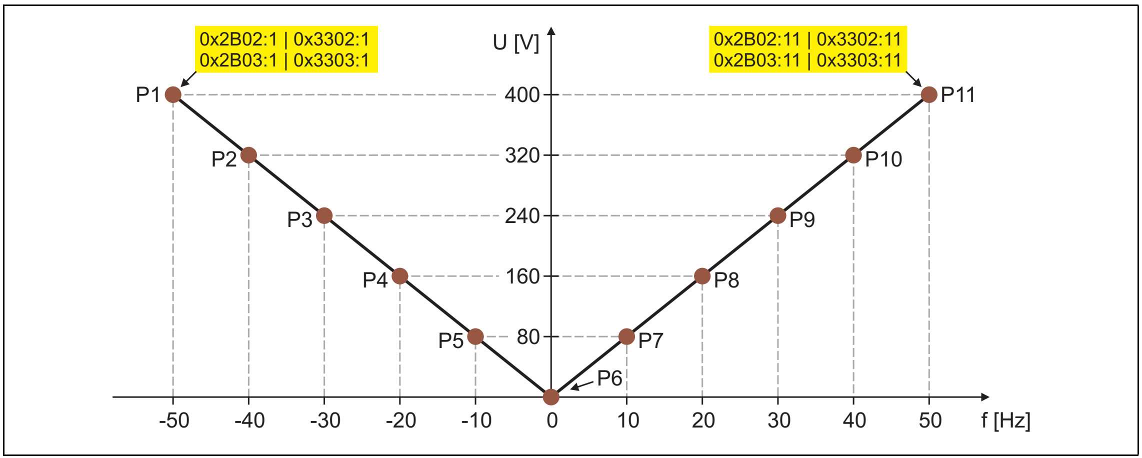
Subindex 1: **VFC: V/f characteristic - voltage at reference point**				
Setting range (min. value | unit | max. value)			Lenze setting	
0	V	5000	225 V	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 2: **VFC: V/f characteristic - frequency at reference point**				
Setting range (min. value | unit | max. value)			Lenze setting	
0	Hz	5000	270 Hz	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Defining a user-defined V/f characteristic

The "User-definable V/f characteristic" is provided for the individual adjustment of the motor magnetisation to the actual application if linear and square-law characteristics are not suitable.

- The characteristic is defined by means of 11 parameterisable grid points (voltage/frequency values).
- In the Lenze setting the 11 grid points represent a linear characteristic:



	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11
V	400 V	320 V	240 V	160 V	80 V	0 V	80 V	160 V	240 V	320 V	400 V
f	-50 Hz	-40 Hz	-30 Hz	-20 Hz	-10 Hz	0 Hz	10 Hz	20 Hz	30 Hz	40 Hz	50 Hz

[5-13] User-definable V/f characteristic (Lenze setting)

0x2B02 | 0x3302 - VFC: User-definable V/f characteristic - frequency grid points (x)

Sub.	Name	Lenze setting	Data type
1	V/f: x1 = f01	-50 Hz	INTEGER_16
2	V/f: x2 = f02	-40 Hz	INTEGER_16
3	V/f: x3 = f03	-30 Hz	INTEGER_16
4	V/f: x4 = f04	-20 Hz	INTEGER_16
5	V/f: x5 = f05	-10 Hz	INTEGER_16
6	V/f: x6 = f06	0 Hz	INTEGER_16
7	V/f: x7 = f07	10 Hz	INTEGER_16
8	V/f: x8 = f08	20 Hz	INTEGER_16
9	V/f: x9 = f09	30 Hz	INTEGER_16
10	V/f: x10 = f10	40 Hz	INTEGER_16
11	V/f: x11 = f11	50 Hz	INTEGER_16
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			

0x2B03 | 0x3303 - VFC: User-definable V/f characteristic - voltage grid points (y)

Sub.	Name	Lenze setting	Data type
1	V/f: y1 = U01 (x = f01)	400.00 V	UNSIGNED_32
2	V/f: y2 = U02 (x = f02)	320.00 V	UNSIGNED_32
3	V/f: y3 = U03 (x = f03)	240.00 V	UNSIGNED_32
4	V/f: y4 = U04 (x = f04)	160.00 V	UNSIGNED_32
5	V/f: y5 = U05 (x = f05)	80.00 V	UNSIGNED_32
6	V/f: y6 = U06 (x = f06)	0.00 V	UNSIGNED_32
7	V/f: y7 = U07 (x = f07)	80.00 V	UNSIGNED_32
8	V/f: y8 = U08 (x = f08)	160.00 V	UNSIGNED_32
9	V/f: y9 = U09 (x = f09)	240.00 V	UNSIGNED_32
10	V/f: y10 = U10 (x = f10)	320.00 V	UNSIGNED_32
11	V/f: y11 = U11 (x = f11)	400.00 V	UNSIGNED_32
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			

5.15.2 Activating the voltage vector control (Imin controller)

This function that can be activated is used if a comparatively high starting torque has to be provided for. This function ensures that the required motor current is maintained in the lower speed range.

Stop!

Up to and including software version V02.12.xx

Make sure that the maximum motor current is not exceeded!

Note!

The function described here adds to the voltage boost.

Only use one of the two functions.

- Recommendation: Torque increase in the lower speed range
- Take into consideration that the increased current at low speeds also entails higher heat losses of the motor.

- The voltage vector control is activated by the selection of a current setpoint.
- For the automatic calculation of the control parameters, the "Calculate Imin controller" function is provided via object 0x2822 (or 0x3022 for axis B).

0x2B04 | 0x3304 - VFC: Voltage vector control - setpoint current

Current setpoint for voltage vector control
- If "0.00 A" are set, the voltage vector control is deactivated.
- When the current setpoint is defined, provide a reserve of 20 % to prevent a motor stalling caused by sudden additional loads.
- Example for starting torque = rated motor torque:
  The current setpoint must be set to approx. 120 % of the load current.

Setting range (min. value | unit | max. value)			Lenze setting	
0.00	A	500.00	0.00 A	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/100	UNSIGNED_32

0x2B05 | 0x3305 - VFC: Voltage vector control parameter

Sub.	Name	Lenze setting	Data type
▸ 1	VFC: Voltage vector controller - gain	148.21 V/A	UNSIGNED_32
▸ 2	VFC: Voltage vector controller - reset time	3.77 ms	UNSIGNED_32

Subindex 1: **VFC: Voltage vector controller - gain**				
Gain for voltage vector control				
Setting range (min. value | unit | max. value)			Lenze setting	
0.00	V/A	750.00	148.21 V/A	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/100	UNSIGNED_32

Subindex 2: **VFC: Voltage vector controller - reset time**				
Reset time for voltage vector control				
Setting range (min. value | unit | max. value)			Lenze setting	
0.01	ms	2000.00	3.77 ms	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/100	UNSIGNED_32

5.15.3 Setting the voltage boost

As an alternative for the voltage vector control, a constant, load-independent voltage boost can be specified for low speeds (below the V/f rated frequency) or for a motor standstill in order to optimise the starting performance.

Stop!

Up to and including software version V02.12.xx

Make sure that the maximum motor current (0x6073 or 0x6873 for axis B) is not exceeded!

From software version V02.13.xx onwards, it is ensured that the maximum motor current is not exceeded if the voltage boost selected is not suitable.

If the motor is operated at standstill for a longer time - especially in case of smaller motors - the motor can be destroyed by overtemperature!

- Connect the KTY of the motor. Parameterise and activate the Motor temperature monitoring. (🕮 305)
- Parameterise and activate the Monitoring of the motor utilisation (I²xt). (🕮 296)
- Operate self-ventilated motors with a blower, if required.

Note!

The voltage boost acts additively to the voltage vector control.

- Only use one of the two "Boost" functions.
  Recommendation: voltage vector control

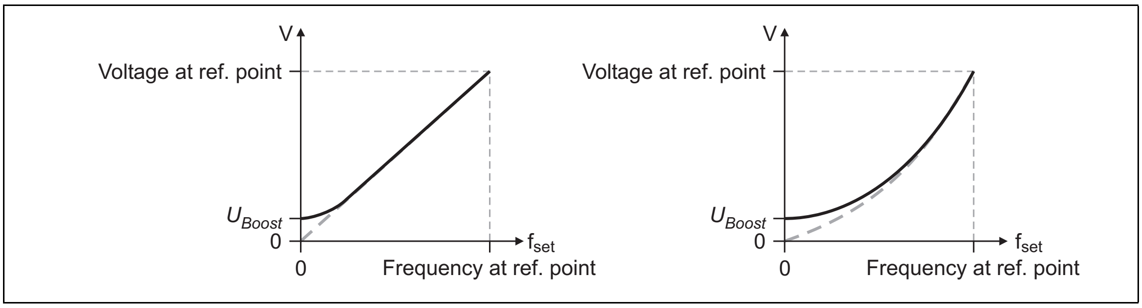
Depending on the required starting torque, the voltage boost must be set so that the required motor current will be available after controller enable.

- The voltage boost can be calculated by multiplying the stator resistance by the rated magnetising current:

$$\text{Starting current} \sim V_{\text{Boost}} = R_S \cdot I_{\text{mN}}$$

- Optionally, the voltage boost can be determined empirically by increasing the setting until the rated magnetising current flows.
- The voltage boost is added geometrically to the voltage of the characteristic:

$$U = \sqrt{U^2_{\text{Characteristic}} + U^2_{\text{Boost}}}$$



[5-14] Voltage boost for a linear V/f characteristic (left) and square-law V/f characteristic (right)

Tip!

For magnetising the motor, consider a sufficient time from the controller enable to the start of the speed ramp function generator.

- The bigger the motor the longer the time required for magnetisation. A motor with a power of 90 kW requires up to 2 seconds.

0x2B06 | 0x3306 - VFC: Voltage boost

Voltage boost

Setting range (min. value | unit | max. value)			Lenze setting	
0.0	V	100.0	0.0 V	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	UNSIGNED_16

5.15.4 Setting the load adjustment

Stop!

If the load adjustment is too high, the motor current may increase in idle state and the motor may overheat!

0x2B07 | 0x3307 - VFC: Load adjustment parameter

Sub.	Name	Lenze setting	Data type
1	VFC: Load adjustment - direction of rotation	0: Passive load (e.g. friction)	UNSIGNED_8
2	VFC: Load adjustment - value	20.00 %	UNSIGNED_32

Subindex 1: **VFC: Load adjustment - direction of rotation**			
Adaptation of the characteristic depending on the load in the case of CW and CCW rotation			
Selection list(Lenze setting printed in bold)		Info	
0	**Passive load (e.g. friction)**	The motor operates in motor mode in both directions.	
1	Active load, CCW (e.g. hoist)	Application example: Hoist without counterweight	
2	Active load, CW (e.g. hoist)	Application example: Dancer-controlled unwinder	
☑ Write access ☑ CINH ☐ OSC ☑ P ☐ RX ☐ TX			UNSIGNED_8

Subindex 2: **VFC: Load adjustment - value**				
Load adjustment in [%] proportionally to the rated torque, in order to obtain a correspondingly "rigid" drive behaviour even after the starting action. • When starting torque = rated torque, a load adjustment of 50 % is suitable for most applications.				
Setting range (min. value | unit | max. value)			Lenze setting	
0.00	%	200.00	20.00 %	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/100	UNSIGNED_32

5.15.5 Defining the behaviour at the current limit (Imax controller)

The maximum output current or the current limit is determined by the object "Max. current" (0x6073 or 0x6873 for axis B). In case of the V/f characteristic control, an Imax controller is implemented for complying with this limit. If the motor current exceeds the set maximum value, the Imax controller is activated.

- The Imax controller changes the field frequency so that the motor current does not exceed the current limit. In motor mode, the frequency is reduced and in generator mode it is increased.
- The gain and reset time of the Imax controller can be parameterised.
  - For the automatic calculation of these two parameters, the "VFC: Calculate Imax controller parameters" function is provided via object 0x2822 (or 0x3022 for axis B).

Optimising the Imax controller

The automatic calculation serves to determine starting parameters of the Imax controller which are sufficient for many applications. Thus, an optimisation is not required for most of the applications.

The parameters of the Imax controller have to be adapted if

- a power control is implemented with great moments of inertia.
  Recommendation:
  Step 1: Increase reset time in 0x2B08:2 (or 0x3308:2 for axis B)
  Step 2: Reduce gain in 0x2B08:1 (or 0x3308:1 for axis B)
- vibrations occur with V/f characteristic control during the operation of the Imax controller.
  Recommendation:
  Step 1: Increase reset time in 0x2B08:2 (or 0x3308:2 for axis B)
  Step 2: Reduce gain in 0x2B08:1 (or 0x3308:1 for axis B)
- overcurrent errors occur due to load impulses or too high acceleration/deceleration ramps.

Recommendation:
Step 1: Reduce reset time in 0x2B08:2 (or 0x3308:2 for axis B)
Step 2: Increase gain in 0x2B08:1 (or 0x3308:1 for axis B)

If the connected mechanics and the conditions in the machine allow it, it may be helpful to determine the optimal parameters of the current controller by a practical acceleration test with a reduced maximum current (0x6073 or 0x6873 for axis B). Mass inertia and acceleration/deceleration considerably determine the requirements of the Imax control loop.

Recommendation: Execute the adjustment with the real mass inertia and optimise the parameters step by step with an increasing acceleration/deceleration. The oscilloscope function of the »PLC Designers«/»EASY motor starter« serves to record the following objects:

- VFC: Setpoint frequency (0x2B0B or 0x330B for axis B)
- Motor: Actual voltage (0x2D82 or 0x3582 for axis B)
- Motor: Actual current (0x2DD1:5 or 0x35D1:5 for axis B)
- Device: Max. current (0x6073 or 0x6873 for axis B)
- Device: Actual output frequency (0x2DDD or 0x35DD for axis B)

0x2B08 | 0x3308 - VFC: Imax controller - Parameter

Sub.	Name	Lenze setting	Data type
1	VFC: Imax controller - gain	0.001 Hz/A	UNSIGNED_32
2	VFC: Imax controller - reset time	100.0 ms	UNSIGNED_32

Subindex 1: **VFC: Imax controller - gain**				
Gain for Imax controller				
Setting range (min. value | unit | max. value)			Lenze setting	
0.000	Hz/A	1000.000	0.001 Hz/A	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/1000	UNSIGNED_32

Subindex 2: **VFC: Imax controller - reset time**				
Reset time for Imax controller				
Setting range (min. value | unit | max. value)			Lenze setting	
1.0	ms	2000.0	100.0 ms	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	UNSIGNED_32

5.15.6 Deactivating or parameterising feedbacks

The encoder feedback system and the temperature feedback are always evaluated regardless of the control mode selected.

In order to avoid error messages when using a usually encoderless V/f characteristic control, the following monitoring functions must be deactivated:

- 0x2C45 = 0, no response in case of wire breakage
- 0x2C41:004 = 0, no response in case of communication monitoring with Hiperface encoder
- 0x2D49:004, set maximum value of the temperature threshold.

Special feature: Monitoring of available encoders

Since encoder monitoring is independent of the control mode used, encoders available in the machine can be evaluated and monitored.

Note!

If no encoder is connected, a sin/cos encoder must be parameterised as follows:

- 0x2C40 / 0x3440 = 1

If this parameterisation is disregarded, it is prevented that the inverter - by its continuous attempts to read out a hiperface encoder - can be enabled.

Despite a sensorless control, the following objects can be used to evaluate encoders. They are connected to X8 (Sub-D socket) of the device:

- 0x6063 / 0x6863 and
- 0x6064 / 0x6864.

5.15.7 Setting the slip compensation

The slip compensation serves to automatically compensate for a load-dependent speed deviation. The rated slip of the motor is required to ensure that the slip compensation can work correctly.

Note!

Observe correct parameterisation of the rated motor frequency and the rated motor speed. Both parameters serve to calculate the rated motor slip.

0x2B09 | 0x3309 - VFC: Slip compensation - Parameter

Sub.	Name	Lenze setting	Data type
1	VFC: Slip compensation - influence	0.00 %	INTEGER_16
2	VFC: Slip compensation - filter time	2000 ms	UNSIGNED_16

Subindex 1: **VFC: Slip compensation - influence**				
Adjustment of the slip calculated • Required due to deviations (tolerances) between the real motor data and the nameplate data. • For instance, with a setting of 100 %: fully effective slip compensation (assumption: at rated load, the values for the real slip and calculated slip are equal.)				
Setting range (min. value | unit | max. value)			Lenze setting	
-200.00	%	200.00	0.00 %	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/100	INTEGER_16

Subindex 2: **VFC: Slip compensation - filter time**				
Adjustment of the time-dependent behaviour of the slip compensation				
Setting range (min. value | unit | max. value)			Lenze setting	
1	ms	6000	2000 ms	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

5.15.8 Setting the oscillation damping

The oscillation damping serves to reduce the oscillations during no-load operation which are caused by energy oscillating between the mechanical system (mass inertia) and the electrical system (DC bus). Furthermore, the oscillation damping can also be used to compensate resonances.

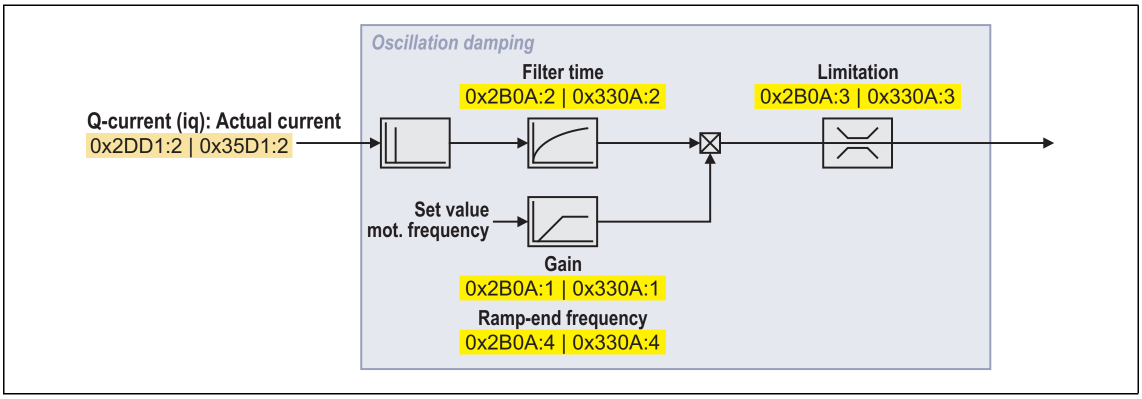
Note!

Observe the following restrictions:

- Damping is possible only for constant oscillations at a steady-state operating point.
- Oscillations occurring sporadically cannot be damped.
- Oscillation damping is not suitable for oscillations occurring during dynamic processes (e.g. accelerations or load changes).
- Oscillation damping is only active if the setpoint speed is greater than 10 rpm and the DC-bus voltage exceeds a value of 100 V.

Function

The determination of the oscillation is based on the active current. In order to obtain the alternating component of the active current, this current is differentiated. This signal is then passed through a PT1 filter.



Identification of the oscillation

Before the oscillation damping can be parameterised, the oscillation has to be identified. One option is to examine the motor current at switched-off oscillation damping (gain = 0 %). The oscilloscope function of the »PLC Designer« serves to record the following currents:

- Q current 0x2DD1:2 / 0x35D1:2
- Total current 0x2DD1:5 or 0x35D1:5

Passive loads cause a constant current in continuous operation with constant speed (steady-state operation). If the drive oscillates, this oscillation also takes place in the motor current. Thus, it is possible to determine the frequency and amplitude of the oscillation by means of the AC component in the motor current. In the following, this AC component is called "current oscillation".

Parameter setting

The gain of the oscillation damping has to be set according to the following equation:

$$\text{Gain of the oscillation damping} = \frac{\text{Current amplitude}}{\sqrt{2} \cdot \text{Maximum device current}} \cdot 100\ \%$$

The time constant of the PT1 filter has to be set in such a way that the oscillation can be damped and higher-frequency components are filtered out of the signal. The time constant is determined from the reciprocal value of the double frequency of the current oscillation.

$$\text{Time constant} = \frac{1}{2 \cdot \text{Oscillation frequency}}$$

The oscillation frequency calculated can be limited before it is added to the field frequency. The maximum frequency can be derived from the amplitude of the current oscillation, the rated motor current, and the slip frequency of the connected motor:

$$\text{Max. frequency} = \frac{2 \cdot \text{Amplitude of the current oscillation}}{\text{Rated motor current}} \cdot \text{Rated slip frequency}$$

0x2B0A | 0x330A - VFC: Oscillation damping - Parameter

Sub.	Name	Lenze setting	Data type
▸ 1	VFC: Oscillation damping - gain	20 %	INTEGER_16
▸ 2	VFC: Oscillation damping - filter time	5 ms	UNSIGNED_16
▸ 3	VFC: Oscillation damping - limitation	0.2 Hz	UNSIGNED_16
▸ 4	VFC: Oscillation damping - ramp-end frequency	0 %	UNSIGNED_8

Subindex 1: **VFC: Oscillation damping - gain**				
Gain of the oscillation signal • If the setting is "0 %", the oscillation damping is switched off.				
Setting range (min. value | unit | max. value)			Lenze setting	
-100	%	100	20 %	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				INTEGER_16

Subindex 2: **VFC: Oscillation damping - filter time**				
Time constant of the PT1 filter				
Setting range (min. value | unit | max. value)			Lenze setting	
1	ms	600	5 ms	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 3: **VFC: Oscillation damping - limitation**				
Limitation of the calculated oscillation frequency before it is added to the field frequency.				
Setting range (min. value | unit | max. value)			Lenze setting	
0.1	Hz	20.0	0.2 Hz	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	UNSIGNED_16

Subindex 4: **VFC: Oscillation damping - final ramp frequency**				
Ramp end frequency from which the gain factor is to have reached its rated value. • By setting a ramp end frequency, a possible negative impact of the oscillation damping on the concentricity factor in the lower speed range can be reduced. • The ramp end frequency refers to the rated motor frequency in percent.				
Setting range (min. value | unit | max. value)			Lenze setting	
0	%	100	0 %	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_8

5.15.9 Optimising pull-out slip limitation

The following object serves to adapt the pull-out slip function or the maximally permissible motor current in the field weakening range.

- If the motor is to be stalled in the field weakening range, the override point can be offset by reducing the value so that a motor stalling can be prevented.
- If the motor cannot provide enough torque in the field weakening range, the value has to be increased.

0x2B0C | 0x330C - VFC: Override point of field weakening

From software version V01.03.xx onwards
Offset of the override point for field weakening

Setting range (min. value | unit | max. value)			Lenze setting	
-500.0	Hz	500.0	0.0 Hz	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	INTEGER_16

5.15.10 Display parameter

0x2B0B | 0x330B - VFC: Setpoint frequency

Display range (min. value | unit | max. value)			Initialisation	
-600.0	Hz	600.0	0.0 Hz	
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: 1/10	INTEGER_16

5.15.11 "Flying restart" function

As a protective function against high compensation currents, the i700 servo inverters provides the flying restart function. High compensation currents may occur in the case of the V/f characteristic control if the drive is not at standstill at the time of controller enable. The flying restart function determines the motor speed by means of a test current and, by this information, presets the frequency setpoint. The speed determined is also provided to the axis control.

STOP

Stop!

If the flying restart function is deactivated and the controller is not enabled at standstill, the output voltage and output frequency do not match the current motor speed. High compensation currents may flow!

- The drive is first braked towards 0 Hz and is then accelerated again!

0x2BA0 | 0x33A0 - Flying restart: Activate

Activation of the additional "flying restart process" function

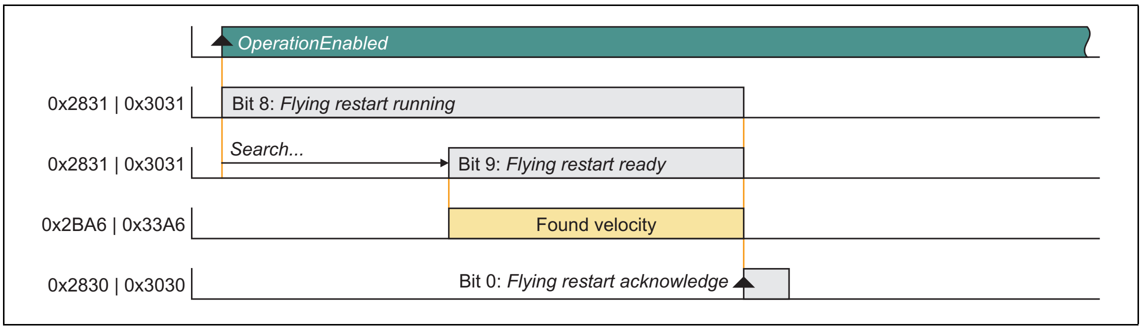
- If the flying restart function is activated ("1: On"), a flying restart process for the determination of the current motor speed is started automatically after the controller inhibit has been deactivated if the following conditions are met:
  - The V/f characteristic control is set as motor control.
  - The CiA402 mode is selected as drive mode.
  - The flying restart function is not blocked via bit 2 in the Lenze control word (0x2830 or 0x3030 for axis B).
  - No DC-injection braking is active.
  - No motor phase failure has been detected.

Selection list(Lenze setting printed in bold)			
0	**Off**		
1	On		
2	Without confirmation or disable		
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			UNSIGNED_8

Flying restart process

If the "flying restart process" function is activated, the flying restart process will start after controller enable:

1. The i700 servo inverters reports the started flying restart process to the controller via bit 8 in the Lenze status word (0x2831 or 0x3031 for axis B).
2. If a speed has been found, it is reported to the controller via bit 9 in the Lenze-status word.
3. The controller gives the i700 servo inverters a feedback on the fact that the found speed has been accepted via bit 0 in the Lenze control word (0x2830 or 0x3030 for axis B). As long as the feedback is not provided, further flying restart processes are not possible.



[5-15] Signals during a flying restart process

Parameter setting

Note!

- The flying restart algorithm requires the motor voltage as exact as possible. Therefore it is absolutely necessary to predetermine the inverter error characteristic.
  ▸ Compensating for inverter influence on output voltage
- In addition to the exact motor voltage, the exact stator resistance must also be known. If the flying restart process does not work as desired, the setting of the stator resistance is to be slightly adapted in object 0x2C01:2 (or 0x3401:2 for axis B).
- A flying restart process can be blocked via bit 1 in the Lenze control word (0x2830 or 0x3030 for axis B).

A control loop is involved in the flying restart process the controller parameter integration time of which has to be adapted to the motor in the object 0x2BA3 (or 0x33A3 for axis B). For the automatic calculation of this parameter, the "VFC: Flying restart process controller - calculate controller parameters" function is available via the object 0x2822 (or 0x3022 for axis B).

The actual flying restart process can be adjusted via the following parameters:

0x2BA1 | 0x33A1 - Flying restart: Current

Current injected in the motor by the flying restart algorithm to identify the current speed.
- Selection in [%] based on the rated motor current (0x6075 or 0x6875 for axis B).
- The higher the current the higher the torque imposed upon the motor.
- If the current is too low, a wrong speed can be detected.

Setting range (min. value | unit | max. value)			Lenze setting	
0	%	100	15 %	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

0x2BA2 | 0x33A2 - Flying restart: Start frequency

Start frequency of the flying restart algorithm
- If it can be anticipated at which frequency the motor "caught" most frequently in a flying restart process, this frequency is to be set here.

Setting range (min. value | unit | max. value)			Lenze setting	
-600.0	Hz	600.0	20.0 Hz	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	INTEGER_16

0x2BA3 | 0x33A3 - Flying restart: Integration time

Integration time of the angle controller
- The Lenze setting is adapted to machines with an average power.
- A guide value for the integration time can be calculated as a function of the motor power with the following equation: Ti = 1.1 µ/W * rated motor power + 9.4 ms
- For accelerating the search process, this guide value can be reduced.
- If the flying restart frequency oscillates too much, the integration time has to be increased again.
- A longer integration time increases the time for "catching" the drive.

Setting range (min. value | unit | max. value)			Lenze setting	
1	ms	60000	600 ms	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

0x2BA4 | 0x33A4 - Flying restart: Min. deviation

Setting range (min. value | unit | max. value)			Lenze setting	
0.00	°	90.00	5.00 °	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/100	UNSIGNED_16

0x2BA5 | 0x33A5 - Flying restart: Delay time

To avoid starting a flying restart process at short-time controller inhibit, a time can be set here for the minimum active controller inhibit time before a flying restart process is activated.
- Since a pulse inhibit > 500 ms causes a controller inhibit, this also applies to the pulse inhibit.

Setting range (min. value | unit | max. value)			Lenze setting	
0	ms	10000	0 ms	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

0x2BA6 | 0x33A6 - Flying restart: Result

Output of the speed found for acceptance in the control.
- This information is pending until the control reports acceptance of the found speed to the i700 servo inverter via bit 0 in the Lenze control word (0x2830 or 0x3030 for axis B).

Sub.	Name	Lenze setting	Data type
▸ 1	Flying restart: Determined speed [rpm]		INTEGER_16
▸ 2	Flying restart: Determined speed [n unit]		INTEGER_32

Subindex 1: **Flying restart: Determined speed [rpm]**				
Display range (min. value | unit | max. value)			Initialisation	
-32768	r/min	32767		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX				INTEGER_16

Subindex 2: **Flying restart: Determined speed [n unit]**				
Display range (min. value | unit | max. value)			Initialisation	
-480000	[n unit]	480000		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: $480000/2^{31}$	INTEGER_32

5.16 "DC-injection braking" function

The control modes for asynchronous motors provide the opportunity to use "DC-injection braking" for the braking process.

DC-injection braking can be

- parameterised via bit 6 in the Lenze control word (0x2830 or 0x3030 for axis B) or
- as response to light errors. ▶ Response of the device in the event of an error (📖 320)

Application cases for DC-injection braking

- DC-injection braking via bit 6 of the Lenze control word
  The motor system itself can be used as an energy converter. This option is advantageous if
  - a brake resistor required for absorbing the braking energy is not provided in the system. A prerequisite for this process is that a sufficient braking torque can be reached using DC-injection braking.
  - the power of the brake chopper to be converted is limited and thus has to be used exclusively for the main drives of the DC network. The quality of the deceleration ramp via DC-injection braking is sufficient for auxiliary drives and unloads the brake chopper.
  - a fan drive has to be braked in V/f characteristic mode.
- DC-injection braking as response to light errors
  A light error can be an error of an encoder of an asynchronous machine as servo axis.
  Due to the error, the quick stop function cannot be executed anymore. Optionally, the controlled shutdown with low deceleration via DC-injection braking can be used.

Functional description

In this case, the motor control injects a DC current the amplitude of which can be set in the object 0x2B80 (or 0x3380 for axis B). For this purpose, it is required that the current control is adapted to the corresponding motor. ▶ Setting and optimising the current controller (📖 146)

0x2B80 | 0x3380 - DC-injection braking: Current

Braking current for DC-injection braking

Setting range (min. value | unit | max. value)			Lenze setting	
0.00	A	500.00	0.00 A	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/100	UNSIGNED_16

> **Note!**
>
> The r.m.s. value of the current is calculated from the time values by means of the use of the factor "√2". This also happens in case of the field frequency 0 Hz (DC current). In case of the DC injection braking, the r.m.s. value is thus given by the factor "√ too low in 0x2DD1:5 (or 0x35D1:5 for axis B).

5.17 "Short-circuit braking" function

The control modes for synchronous motors provide the opportunity to use "short-circuit braking" for the braking process.

Short-circuit braking can be

- parameterised via bit 6 in the Lenze control word (0x2830 or 0x3030 for axis B) or
- as response to light errors. ▸ Response of the device in the event of an error (📖 320)

Stop!

In some constellations, it is not possible to decelerate the motor speed of a synchronous motor to zero by means of "short-circuit braking". The braking effect cannot be compared to the "quick stop" function where the motor is energised under optimal conditions!

Cases of application

- for short-circuit braking via bit 6:
  - The braking energy cannot be converted into heat in a brake resistor.
  - An external control wants to request short-circuit braking as e.g. an error has been detected in the encoder system which does not permit braking via quick stop.
- for short-circuit braking as response to light errors:
  - Due to an encoder error, no quick stop is possible anymore.

Functional description

Note!

The short-circuit current is set freely according to the motor voltage (k_E * speed) and the internal resistance of the system. Thus, it is mandatory that the ampacity of the i700 servo inverters is oriented towards the short-circuit current to be maximally expected.

Guide value: I_{max_device} (3 s) ≥ 1.5 * I_{max_motor} (according to the data sheet/catalogue)

In case of a deviating assignment, a rating based on the currently possible parameters (max. speed, max. motor current, field weakening, etc.) is required!

The effect of the short-circuit braking on the deceleration behaviour depends on the motor features, the effective cable length, the mass inertia and the initial value of the speed (application point).

Primarily, short-circuit braking can be used to convert a part of the kinetic energy into heat energy which serves to unload external brake assemblies and limit position dampers.

If short-circuit braking is to be used as the only holistically effective deceleration unit, it is recommended to verify the feasibility by tests. For this purpose, bit 6 in the Lenze control word (0x2830 or 0x3030 for axis B) serves to trigger the short-circuit braking. The oscilloscope function of the »PLC Designer«/»EASY Starter« serves to record the following important objects:

- Speed: Setpoint speed (0x2DD3:1 or 0x35D3:1 for axis B)
- Speed: Actual speed (0x606C or 0x686C for axis B)
- Phase currents U, V, W (0x2D83:2...4 or 0x3583:2...4 for axis B)

5.18 Setting the switching frequency

The i700 servo inverters can be operated with the switching frequencies listed below.

0x2939 | 0x3139 - Switching frequency

Note: In case of "8 kHz var." and "16 kHz var.", variable switching frequencies are involved. If the permissible continuous current for the respective switching frequency is exceeded, it is automatically switched back to the next-lower switching frequency.

Selection list(Lenze setting printed in bold)		Info	
0	16 kHz variable		
1	8 kHz variable		
2	**4 kHz**		
10	16 kHz, permanent	From software version V01.06.xx onwards	
11	8 kHz fix	From software version V01.06.xx onwards	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			UNSIGNED_8

5.19 Frequency and speed limitations

Output frequency

Note!

By limiting the maximum output frequency to ± 599 Hz, the devices are not subject to the export restrictions of the "EC-Dual-Use Regulation" (EG 428/2009). This applies to devices supplied from the middle of the year 2015.

For certain applications, the supply of devices with the current maximum output frequency of ± 1999 Hz. If required, get in touch with your Lenze contact person.

The output frequency of the i700 servo inverters is limited to a maximum value whose amount corresponds to the lower of the two following values:

$$f_{lim} = \frac{f_{chop}}{8} \quad \text{or} \quad f_{lim} = f_{max\_device}$$

f_{lim}	Maximum output frequency
f_{chop}	Switching frequency (0x2939 or 0x3139 for axis B)
f_{max_device}	Maximum device output frequency (0x2835:2 or 0x3035:2 for axis B, depending on the device type ± 599 Hz or ± 1999 Hz)

By restricting the "Dual-Use Regulation" (EC 428/2009), values of + 599 Hz to + 1000 Hz or - 599 Hz to - 1000 Hz do not cause an increase of the output frequency. Please observe the dead band that occurs in this case.

Speed setpoint

If servo control is used, the speed setpoint is limited depending on the number of motor pole pairs:

$$n_{lim} = \frac{f_{lim}}{zp}$$

n_{lim}	Speed limit value
f_{lim}	Maximum output frequency
zp	Number of motor pole pairs

- If the speed setpoint is limited, bit 1 ("Speed: Setpoint 1 limited") or bit 5 ("Speed: Setpoint 2 limited") is set in the Lenze status word (0x2831 or 0x3031 for axis B).
- The behaviour corresponds to the behaviour which is shown when the set maximum speed (0x6080 or 0x6880 for axis B) is reached.
- The sequence is: Limit the speed to 0x6080 (or 0x6880 for axis B) first, then limit it to the speed limit value n_{lim}.

Frequency setpoint

If V/f characteristic control is used, the frequency setpoint is limited in addition to the speed setpoint.

- If the frequency setpoint is limited, bit 10 ("Output frequency limited") is set in the Lenze status word (0x2831 or 0x3031 for axis B).

6 Holding brake control

This device function is used to control a motor holding brake connected to the i700 servo inverters.

Note!

- When the brake is open, a slight knocking sound can be noticed in the motor. This stems from test pulses for monitoring the motor brake control.
- In the "Fault" device status, the holding brake is applied.

Objects described in this chapter

Object		Name	Data type
Axis A	Axis B		
0x2820	0x3020	Brake control: Settings	RECORD

6.1 Operating modes

For holding brake control, the following three operating modes available for the i700 servo inverters in the object 0x2820:1 (or 0x3020:1 for axis B):

Operating mode	Application
Triggering via control word from external controller (Lenze setting)	Operation in the Lenze system, i.e. • Lenze Controller as external control • FAST Motion application Advantage: All settings required are carried out easily and in a concise fashion via the corresponding dialogs in the Engineering tools
Triggering via state machine of device	Operation with an external control by an original equipment manufacturer. Here, a handshake via the device status machine serves to open the brake and carry out the operation. • Including automatic torque feedforward control
No brake connected	-

Tip!

We recommend the use of the preset "Triggering via control word from external control" mode. In this operating mode, the triggering is carried out via the application program in the Controller.

You can find detailed information on the respective operating mode in the following subchapters.

6.1.1 Triggering via control word via external control (Lenze setting)

Object		Name	Required setting
Axis A	Axis B		
0x2820:1	0x3020:1	Brake: Operating mode	1: Activation via control word by ext. control system

In this preset operating mode, the triggering takes place via the application program in the Controller via bit 14 in the CiA402 control word (0x6040 or 0x6840 for axis B).

- Bit 14 in the CiA402 control word = "0" ➞ close holding brake
- Bit 14 in the CiA402 control word = "1" ➞ release holding brake

L_MC1P_BrakeInterface function block

For Lenze Controllers, the **L_MC1P_BrakeInterface** FAST Motion FB is provided for the purpose of controlling, containing the following functionality:

- Torque feedforward control in case of release in order to prevent a sagging for active loads (hoists).
- Considering the delay times of the holding brake during the opening and closing process.
  - The delay times which have been transferred as parameters to the function block from the application are considered.
  - The brake parameters "closing time" and "opening time" of the i700 servo inverters are not effective in this operating mode.
- Option to manually release the holding brake for service purposes via a control input.

The basic conditions and timing are displayed in the illustration "Principal signal flow of the holding brake control".

6.1.2 Triggering via state machine of device

Object		Name	Required setting
Axis A	Axis B		
0x2820:1	0x3020:1	Brake: Operating mode	0: Triggering via device state machine

In this operating mode, the holding brake is triggered as a function of the device status optionally with automatic torque feedforward control. ▶ Device states (📖 230)

Note!

In the event of an error or when STO ("SafeTorqueOff") is activated

- closing of the brake and inhibit of the operation take place immediately without considering the set brake closing time.
- the drive changes to the Switch on disabled (📖 233) state.

The basic conditions and timing are displayed in the illustration "Principal signal flow of the holding brake control".

6.1.3 No brake connected

Object		Name	Required setting
Axis A	Axis B		
0x2820:1	0x3020:1	Brake: Operating mode	2: No brake connected

In this operating mode, no control, detection and monitoring of the brake takes place during normal operation.

6.2 Opening the brake with a starting torque

If the control is performed via the device state machine (selection 0x2820:1 = 0 or 0x3020:1 = 0), the required torque of the drive is precontrolled while releasing the brake.

Before the actual torque is fully built up, the starting torque (0x2820:9) is specified as setpoint and the brake is only controlled when the actual torque has at least reached 90 % of this setpoint.

Note!

The torque is precontrolled for one second. During this time, the actual torque must have reached 90 % of the setpoint torque, otherwise an error is triggered!

0x2820:9 or 0x3020:9 for axis B is used to first make the basic selection whether a parameterised starting torque or the torque memorised at the last closing process should be used for the feedforward control.

The torque feedforward control can be activated by setting a starting torque for a jerk-free load transfer:

- 0x2820:9 = 0
- 0x2820:9 = 1 and 0x2820:10 ≠ 0

6.3 Display of the holding brake status

The CiA402 status word (0x6041 or 0x6841 for axis B) displays the holding brake status via bit 14:

- Bit 14 in the CiA402 status word = "0" means "Holding brake applied"
- Bit 14 in the CiA402 status word = "1" means "Holding brake released"

6.4 Overriding the selected brake mode

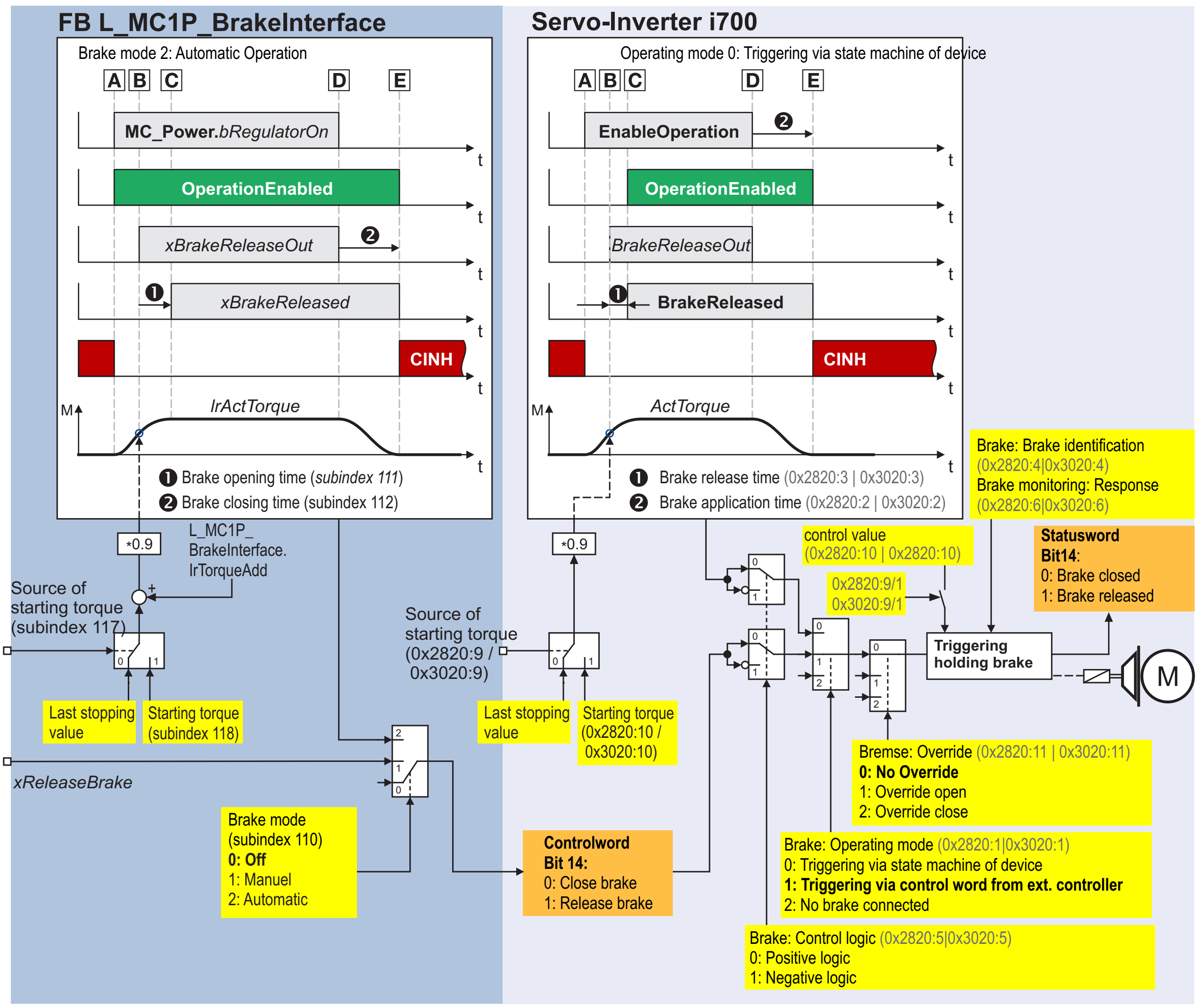
The 0x2820:11 or 0x3020:11 object is used for a forced release or forced application of the brake independent of its operating mode and the operating status of the i700 servo inverters.

This function, for instance, is advantageous in the event of an error because the forced release of the brake allows the axis to move manually.

6.5 Basic signal flow

The following signal flow clarifies the interaction between the **L_SMC_BrakeControl** function block (in the application program of the Lenze Controller) and the device function integrated in the i700 servo inverters for holding brake control.

Principal signal flow of the holding brake control (with the respective operating mode compared to the automatic control)



	L_MC1P_BrakeInterface - process of the automatic control:		**i700 servo inverters** - Triggering process via device state machine:
A	When **MC_Power**.*bRegulatorOn* = TRUE, the i700 is enabled and the starting torque for the torque feedforward control is specified simultaneously.	A	The "Enable operation" command serves to enable the i700 and simultaneously specify the starting torque for the torque feedforward control.
B	When the actual torque is = 0.9 x starting torque, the holding brake is triggered to be released. The triggering is signalled at the *xBrakeReleaseOut* output.	B	When the actual torque is = 0.9 x starting torque, the starting torque is triggered to be released.
C	After the brake opening time has elapsed, the "holding brake released" status is reported via the *xBrakeReleased* output. ➞ From now on, the i700 servo inverters is able to move.	C	After the brake opening time has elapsed, the "holding brake released" status is reported via the CiA402 status word (bit 14). ➞ From now on, the i700 servo inverters is able to move.
D	When **MC_Power**.*bRegulatorOn* = FALSE, the holding brake is triggered to be closed.	D	The Disable operation" command serves to trigger the holding brake to be closed.
E	Only after the brake closing time has elapsed, the i700 is inhibited. The *xBrakeReleased* output reports the "Holding brake closed" status.	E	Only after the brake closing time has elapsed, the i700 is inhibited. The CiA402 status word (bit 14) serves to report the "Holding brake closed" status.

6.6 Settings

0x2820 | 0x3020 - brake control: settings

Sub.	Name	Lenze setting	Data type
1	Brake: Operating mode	1: Activation via control word by ext. control system	UNSIGNED_8
2	Brake: Application time	100 ms	UNSIGNED_16
3	Brake: Release time	100 ms	UNSIGNED_16
4	Brake: Brake identification	0: Identification not active	UNSIGNED_16
5	Brake: Control logic	0: Positive logic	UNSIGNED_8
6	Brake monitoring: Response	1: Trouble	UNSIGNED_8
9	Holding brake: Starting torque source	0: Stopping value	UNSIGNED_16
10	Holding brake: Starting torque		INTEGER_16
11	Holding brake: Override of the holding brake control		UNSIGNED_8
21	Holding brake: Actual torque		INTEGER_16

Subindex 1: **Brake: operating mode**		
Note: In the case of the manual control (0x2825 = "4" or 0x3025 = "4" for axis B), the holding brake is controlled automatically. The setting of the operating mode is irrelevant in this case.		
Selection list(Lenze setting printed in bold)		Info
0	Triggering via state machine of device	This operating mode serves to release the brake automatically if controller enable is set. Device states
1	**Triggering via control word from external controller**	In this operating mode the brake can be controlled manually via bit 14 in the Controlword (0x6040 or 0x6840 for axis B).
2	No brake connected	During normal operation, no control, detection, and monitoring of the brakes take place.
☑ Write access ☐ CINH ☐ OSC ☑ P ☑ RX ☐ TX		UNSIGNED_8

Subindex 2: **Brake: application time**			
Setting range (min. value | unit | max. value)			Lenze setting
0	ms	10000	100 ms
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			UNSIGNED_16

Subindex 3: **Brake: release time**			
Setting range (min. value | unit | max. value)			Lenze setting
0	ms	10000	100 ms
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			UNSIGNED_16

Subindex 4: **Brake: brake identification**
In the case of the manual control (0x2825 = "4" or 0x3025 = "4" for axis B), during the changeover from the "Switched on" to "Operation enabled" device status, it is automatically determined whether a holding brake is connected. The brake identification is executed again after every controller enable.

Subindex 4: **Brake: brake identification**			
Selection list (read only)			
0	**Detection not active**		
1	Brake detection is running...		
2	No brake detected		
3	Brake detected		
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			UNSIGNED_16

Subindex 5: **Brake: Control logic**			
The control logic of the holding brake can be inverted.			
Selection list(Lenze setting printed in bold)			
0	**Positive logic**		
1	Negative logic		
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			UNSIGNED_8

Subindex 6: **Brake monitoring: Response**			
In the controlled state, the holding brake is monitored cyclically for the presence of the brake current. Since, after connecting the brake, the brake current builds up in a time-delayed manner as a function of inductance, an open circuit, a terminal short circuit, or a missing brake supply is detected with a delay. When monitoring trips, the response set here will be activated. **Note:** The brake is not monitored in the non-triggered state!			
Selection list(Lenze setting printed in bold)			
0	No response		
1	**Interference**		
2	Warning		
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			UNSIGNED_8

Subindex 9: **Holding brake: Starting torque source**			
Source for the holding brake torque			
Selection list(Lenze setting printed in bold)		Info	
0	**Stopping value**	In case of feedforward control with the last memorised stopping value, the setpoint that has been automatically memorised during the last closing process is used as starting torque.	
1	Starting torque of 0x2820/10	In case of feedforward control with parameterised starting torque, the parameterised starting torque from subindex 10 is used. The starting torque can be mapped.	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			UNSIGNED_16

Subindex 10: **Holding brake: Starting torque**				
Feedforward control value for the automatic operation (subindex 9 = 1)				
Setting range (min. value | unit | max. value)			Lenze setting	
-3276.7	%	3276.7		
☑ Write access ☐ CINH ☐ OSC ☑ P ☑ RX ☐ TX			Scaling: 1/10	INTEGER_16

Subindex 11: **Holding brake: Override of the holding brake control**			
Mode for override or forced release/application of the holding brake independent of the operating mode. • In the event of an error and activated function for forced release, the brake is not applied.			
Selection list(Lenze setting printed in bold)		Info	
0	No override active	No override active	
1	Release holding brake	Open brake	
2	Apply holding brake	Apply brake	
☑ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			UNSIGNED_8

Subindex 21: **Holding brake: Actual torque**				
Display range (min. value | unit | max. value)			Initialisation	
-3276.8	%	3276.7		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☐ TX			Scaling: 1/10	INTEGER_16

7 CiA402 device profile

The CiA402 device profile describes predefined functions or modes of a variable-speed drive. The individual process data for the modes are clearly defined. The axis parameterisation and the bus configuration (PDO mapping) are carried out via standardised indexes.

Objects described in this chapter

Object Axis A	 Axis B	Name	Data type
0x2500		Touch probe (TP): Filter time	UNSIGNED_16
0x2946	0x3146	Cyclic sync torque mode: Speed limitation	RECORD
0x2D00	0x3500	Touch probe (TP): Delay time	RECORD
0x2D01	0x3501	Touch probe (TP): Time stamp	RECORD
0x6040	0x6840	Controlword	UNSIGNED_16
0x6041	0x6841	Statusword	UNSIGNED_16
0x6042	0x6842	vl target velocity	INTEGER_16
0x6043	0x6843	vl velocity demand	INTEGER_16
0x6044	0x6844	vl velocity actual value	INTEGER_16
0x6046	0x6846	vl velocity min max amount	RECORD
0x6048	0x6848	vl velocity acceleration	RECORD
0x6049	0x6849	vl velocity deceleration	RECORD
0x605A	0x685A	Quick stop option code	INTEGER_16
0x6060	0x6860	Operating mode: Selection	INTEGER_8
0x6061	0x6861	Modes of operation display	INTEGER_8
0x6062	0x6862	Position demand value	INTEGER_32
0x6063	0x6863	Position actual internal value	INTEGER_32
0x6064	0x6864	Position actual value	INTEGER_32
0x6065	0x6865	Following error window	UNSIGNED_32
0x6066	0x6866	Following error time out	UNSIGNED_16
0x6067	0x6867	Position window	UNSIGNED_32
0x6068	0x6868	Position window time	UNSIGNED_16
0x606C	0x686C	Velocity actual value	INTEGER_32
0x6071	0x6871	Target torque	INTEGER_16
0x6072	0x6872	Max torque	UNSIGNED_16
0x6073	0x6873	Max current	UNSIGNED_16
0x6074	0x6874	Target torque	INTEGER_16
0x6077	0x6877	Torque actual value	INTEGER_16
0x6078	0x6878	Current actual value	INTEGER_16
0x6079	0x6879	DC bus: Actual voltage	UNSIGNED_32
0x607A	0x687A	Position demand value	INTEGER_32
0x607E	0x687E	Polarity	UNSIGNED_8
0x6080	0x6880	Max motor speed	UNSIGNED_32
0x6085	0x6885	Quick stop deceleration	UNSIGNED_32
0x608F	0x688F	Position encoder resolution	RECORD
0x6090	0x6890	Velocity encoder resolution	RECORD

Object		Name	Data type
Axis A	Axis B		
0x60B1	0x68B1	Velocity offset	INTEGER_32
0x60B2	0x68B2	Torque offset	INTEGER_16
0x60B8	0x68B8	Touch probe function	UNSIGNED_16
0x60B9	0x68B9	Touch probe status	UNSIGNED_16
0x60BA	0x68BA	Touch probe pos1 pos value	INTEGER_32
0x60BB	0x68BB	Touch probe pos1 neg value	INTEGER_32
0x60BC	0x68BC	Touch probe pos2 pos value	INTEGER_32
0x60BD	0x68BD	Touch probe pos2 neg value	INTEGER_32
0x60C0	0x68C0	Interpolation sub mode select	INTEGER_16
0x60C2	0x68C2	Interpolation time period	RECORD
0x60E0	0x68E0	Positive torque limit value	UNSIGNED_16
0x60E1	0x68E1	Negative torque limit value	UNSIGNED_16
0x60F4	0x68F4	Following error actual value	INTEGER_32
0x60FA	0x68FA	Control effort	INTEGER_32
0x60FC	0x68FC	Position demand internal value	INTEGER_32
0x60FD	0x68FD	Digital inputs	UNSIGNED_32
0x60FF	0x68FF	Target velocity	INTEGER_32
0x6404	0x6C04	Motor manufacturer	STRING(50)
0x6502	0x6D02	Supported drive modes	UNSIGNED_32
0x67FF	0x6FFF	ECAT: Device Profile Number	UNSIGNED_32

CiA402 objects described in other chapters:

Object		Name	Data type
Axis A	Axis B		
▸ Indication of fault and warning (error code)			
0x603F	0x683F	Error code	UNSIGNED_16
0x605E	0x685E	Fault reaction option code	INTEGER_16
▸ Set motor parameters manually			
0x6075	0x6875	Motor rated current	UNSIGNED_32
0x6076	0x6876	Motor rated torque	UNSIGNED_32

7.1 Supported drive modes

CiA402 - drive modes	Can be used with	
	Servo control	V/f characteristic control
Velocity mode (vl) • closed-loop speed control	●	●
Cyclic sync velocity mode (csv) • Speed control with interpolation of the speed setpoint.	●	●
Cyclic sync torque mode (cst) • Torque control with interpolation of the torque setpoint.	●	
Cyclic sync position mode (csp) • Position control with interpolation of the position setpoint.	●	

Setpoint interpolation in the operating modes with a cyclic setpoint selection

When an operating mode with cyclic setpoint selection is selected, first all setpoints are controlled via interpolators, which divide the setpoint step-changes down from the bus cycle to the cycle time of the control loops. All interpolators are parameterised commonly via the "Interpolation time period" object (0x60C2 or 0x68C2 for axis B).

7.2 Applied units and scaling for position and velocity

Name	Unit	Standardisation
CiA402 position	Pos unit	2^{16} ... 2^{30} [Pos unit] ≡ 360° Depending on the position encoder resolution set (0x608F or 0x688F for axis B).
Internal position	Incr.	2^{32} [Incr.] ≡ 360°
CiA402 velocity	r/min	
Internal velocity	n-unit	$\pm 2^{31}$ [n-unit] ≡ ±480000 rpm

7.3 General CiA402 parameters

This chapter describes the general CiA402 parameters.

All objects correspond to the CiA402 specification, but some of them have only a restricted value range.

Objects described in this chapter:

Object Axis A	Object Axis B	Name	Data type
0x60FD	0x68FD	Digital inputs	UNSIGNED_32
0x6404	0x6C04	Motor manufacturer	STRING(50)
0x6502	0x6D02	Supported drive modes	UNSIGNED_32
0x67FF	0x6FFF	ECAT: Device Profile Number	UNSIGNED_32
Greyed out = read access only			

Objects described in other chapters:

Object Axis A	Object Axis B	Name	Data type
0x603F	0x683F	Error code	UNSIGNED_16
Greyed out = read access only			

0x60FD | 0x68FD - Digital inputs

Display of the current status of the digital inputs

Display range (min. value | unit | max. value)			Initialisation
0		4294967295	0
Value is bit-coded:			Info
Bit 0-3	Not specified		
Bits 4-15	Reserved		
Bit 16	Level at digital input 1		0: LOW level 1: HIGH level
Bit 17	Level at digital input 2		0: LOW level 1: HIGH level
Bit 18-31	Reserved		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			UNSIGNED_32

Related topics:
- Touch probe (TP)
- Delay times of the digital inputs and required minimum signal duration

0x6404 | 0x6C04 - Motor manufacturer

☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX		STRING(50)

0x6502 | 0x6D02 - Supported drive modes

Bit coded display of the operating modes in which the drive can be actuated.
0: Operating mode is not supported.
1: Operating mode is supported.

Display range (min. value | unit | max. value)			Initialisation
0x00000000		0xFFFFFFFF	0x00000382
Value is bit-coded:			Info
Bit 0	Not specified		Not supported
Bit 1	Velocity mode		▸ Velocity mode
Bit 2	Not specified		Not supported
Bit 3	Not specified		Not supported
Bit 4	Reserved		
Bit 5	Not specified		Not supported
Bit 6	Not specified		Not supported
Bit 7	Cyclic sync position mode		▸ Cyclic sync position mode
Bit 8	Cyclic sync velocity mode		▸ Cyclic sync velocity mode
Bit 9	Cyclic sync torque mode		▸ Cyclic sync torque mode
Bit 10-15	Reserved		
Bit 16-31	Reserved		
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			UNSIGNED_32

0x67FF | 0x6FFF - Device profile number

Display range (min. value | unit | max. value)			Initialisation
0x00000000		0xFFFFFFFF	0x00020192
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			UNSIGNED_32

7.4 Drive control

The objects described in this chapter serve to control the states of the i700 servo inverters and select the operating mode.

All objects correspond to the CiA402 specification, but some of them have only a restricted value range.

Objects described in this chapter:

Object		Name	Data type
Axis A	Axis B		
0x6040	0x6840	Controlword	UNSIGNED_16
0x6041	0x6841	Statusword	UNSIGNED_16
0x605A	0x685A	Quick stop option code	INTEGER_16
0x6060	0x6860	Operating mode: Selection	INTEGER_8
0x6061	0x6861	Modes of operation display	INTEGER_8
0x6085	0x6885	Quick stop deceleration	UNSIGNED_32
Greyed out = read access only			

0x6040 | 0x6840 - CiA402 controlword

Control word for the drive

Setting range (min. value | unit | max. value)			Lenze setting
0		65535	0
Value is bit-coded: (☑ = bit set)			Info
Bit 0 ☐	Switching on		
Bit 1 ☐	DC bus: Establish readiness for operation		
Bit 2 ☐	Activate quick stop		
Bit 3 ☐	Enable operation		
Bit 4-6 ☐	Depending on the operating mode		
Bit 7 ☐	Fault reset		▸ Indication of fault and warning (error code)
Bit 8 ☐	Not specified		
Bit 9 ☐	Depending on the operating mode		
Bit 10 ☐	Reserved		
Bit 11 ☐	Reserved		
Bit 12 ☐	Reserved		
Bit 13 ☐	Reserved		
Bit 14 ☐	Release brake		▸ Holding brake control
Bit 15 ☐	Reserved		
☑ Write access ☐ CINH ☑ OSC ☐ P ☑ RX ☐ TX			UNSIGNED_16

0x6041 | 0x6841 - Statusword

Status word from the drive

Display range (min. value | unit | max. value)			Initialisation
0		65535	
Value is bit-coded:			Info
Bit 0	Ready to switch on		
Bit 1	Switched on		
Bit 2	Operation enabled		
Bit 3	Trouble active		
Bit 4	Voltage enabled		
Bit 5	Quick stop		If bit 0 ("ready to start") = 1: • "Quick stop active", bit 4 = 0 • "Quick stop not active", bit 4 = 1 In all other cases, the value of bit 4 is optional.
Bit 6	Switch on disabled		
Bit 7	Warning is active		
Bit 8	Deactivate RPDOs		Cyclic PDOs have been deactivated in 0x2824 (or 0x3024 for axis B).
Bit 9	Remote		Drive can receive commands via EtherCAT.
Bit 10	Target reached		The actual position is located in the window (0x6067 or 0x6867 for axis B).
Bit 11	Internal limitation is active		For details see Lenze status word (0x2831 or 0x3031 for axis B).
Bit 12	Drive Follows Command		Operation is enabled and no test mode is activated (no internal setpoint generation active).
Bit 13	Following error		The position could not be approached.
Bit 14	Brake released		
Bit 15	STO is not active		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			UNSIGNED_16

0x605A | 0x685A - Quick stop option code

Device status after exiting the quick stop ramp
2 = automatic changeover to the "Switch-on disabled" device status
6 = the axis remains in the "Quick stop active" device status.

Setting range (min. value | unit | max. value)			Lenze setting
2		6	2
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			INTEGER_16

0x6060 | 0x6860 - Modes of operation

Selection of the operating mode:
0 = no operating mode (standstill)
2 = Velocity mode
8 = Cyclic sync position mode
9 = Cyclic sync velocity mode
10 = Cyclic sync torque mode

Setting range (min. value | unit | max. value)			Lenze setting	
0		10	0	
☑ Write access ☐ CINH ☑ OSC ☑ P ☑ RX ☐ TX				INTEGER_8

0x6061 | 0x6861 - Modes of operation display

Display of the current operating mode:
0 = no operating mode (standstill)
2 = Velocity mode
8 = Cyclic sync position mode
9 = Cyclic sync velocity mode
10 = Cyclic sync torque mode

Display range (min. value | unit | max. value)			Initialisation	
0		10		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX				INTEGER_8

0x6085 | 0x6885 - Quick stop deceleration

Change in velocity used for deceleration to a standstill if quick stop is activated.
- In consideration of the position encoder resolution set in 0x608F (or 0x688F for axis B) (see sample calculation below).

Setting range (min. value | unit | max. value)			Lenze setting	
0	[Pos unit/s²]	2147483647	2147483647 [Pos unit/s²]	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_32

Example: Calculation of the deceleration to be set at a given ramp time

Specifications:
- Position encoder resolution (0x608F) = "16 bits/revolution".
  ➞ A mechanical motor revolution is indicated with 65536 increments.
- Initial speed of the motor = 1000 rpm
- Duration of the ramp until standstill = 2.5 s

Deceleration to be set:

$$\text{Deceleration} = \frac{\text{Initial speed}}{\text{continuous}} \cdot \frac{1}{60} \cdot \frac{\text{0x608F:1}}{\text{0x608F:2}}$$

$$\text{0x6085} = \frac{1000\ [\text{r/min}]}{2.5\ [\text{s}]} \cdot \frac{1}{60\ [\text{s/min}]} \cdot \frac{65536\ [\text{increments}]}{1\ [\text{r}]}$$

$$\text{0x6085} = 436907 \left[\frac{\text{Increments}}{s^2}\right]$$

7.4.1 Commands for the device status control

Via the CiA402 control word (0x6040 or 0x6840 for axis B) commands can be executed to make the i700 servo inverters change to a specific device status:

Command	Bit pattern in the Controlword							
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	Fault reset	Control bits depending on the operating mode			Enable operation	Activate quick stop	Enable voltage	Switching on
Shutdown	0	X	X	X	X	1	1	0
Switching on	0	X	X	X	0	1	1	1
Enable operation	0	X	X	X	1	1	1	1
Activate quick stop	0	X	X	X	X	0	1	X
Disable operation	0	X	X	X	0	1	1	1
Pulse inhibit	0	X	X	X	X	X	0	X
Fault reset	0 ↗ 1	X	X	X	X	X	X	X
X = Status not significant								

Tip!

The greyed out control bits listed in the table are not important for the activation of commands but only serve to improve readability of the bit patterns.

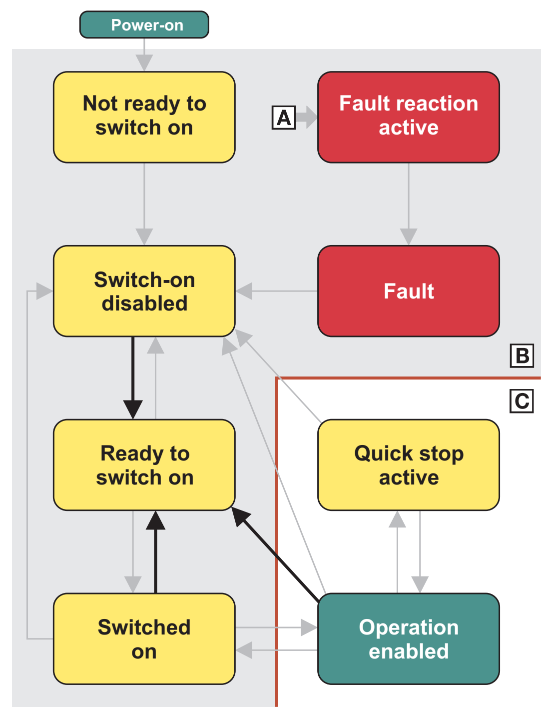
A PLC program of a PLCopen control can, for instance, trigger several commands for state changes in a row by the level change at the *bRegulatorOn* input of the "MC_Power" block. In the mentioned example, these device commands are "switch-off" and "switch-on" in this order.

Detailed information on the different commands can be obtained from the following subchapters.

Further Lenze-specific control bits (bits 8 ... 15)

Device control command	Bit pattern in the Controlword							
	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
	Reserved	Release brake	Reserved					
Apply brake	X	0	X	X	X	X	X	X
Release brake	X	1	X	X	X	X	X	X
X = Status not significant								

7.4.1.1 Shutdown



A From all states
B Power section inhibited (pulse inhibit)
C Power section enabled

This command changes the device status from "Switch on disabled" to "Ready to switch on".

If the pulse inhibit has already been deactivated und the device status of the i700 servo inverters is "Operation enabled", this command can be used to set the pulse inhibit again.

- If automatic brake operation is activated, the parameterised brake closing time is observed: The system waits until the brake is applied before the pulse inhibit is set.
- The motor becomes torqueless.
- The device status "Switched on" or "Operation enabled" changes back to "Ready to switch on".

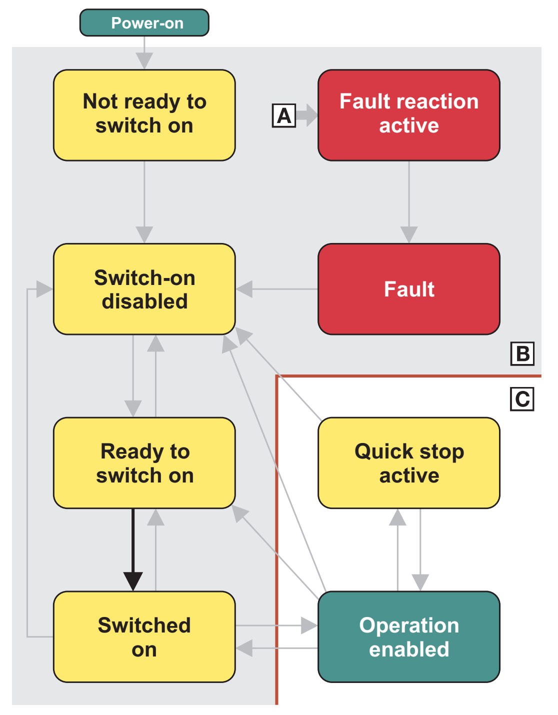
Danger!

If the motor gets torqueless and a load is connected, motors without a holding brake may move in an uncontrolled way!
In case no load is connected, the motor is coasting.

Bit pattern for the "Shutdown" command in the Controlword (0x6040 or 0x6840 for axis B):

Bits 15 - 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	Fault reset	Control bits depending on the operating mode			Enable operation	Activate quick stop	Enable voltage	Switching on
X	0	X	X	X	X	1	1	0

7.4.1.2 Switching on



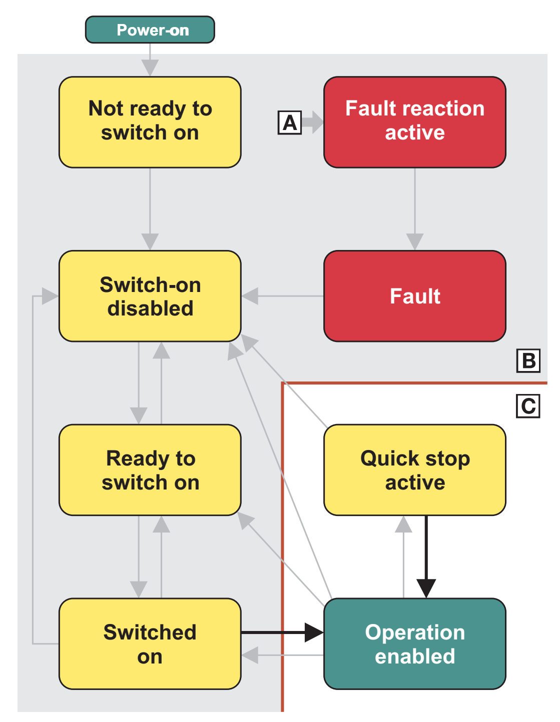
A From all states
B Power section inhibited (pulse inhibit)
C Power section enabled

This command serves to deactivate the switch on inhibit which is active after switch on or after the reset (acknowledgement) of an error.

- A changeover to the "Switched on" device status takes place.

Bit pattern for the "Switch on" command in the Controlword (0x6040 or 0x6840 for axis B):								
Bits 15 - 8	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
	Fault reset	Control bits depending on the operating mode			Enable operation	Activate quick stop	Enable voltage	Switching on
X	0	X	X	X	0	1	1	1

7.4.1.3 Enable operation



A From all states
B Power section inhibited (pulse inhibit)
C Power section enabled

This command serves to enable the operation and stop an active quick stop again.

- A changeover to the "Operation enabled" device status takes place.
- The output stages of the i700 servo inverters become active.

Bit pattern for the "Enable operation" command in the Controlword (0x6040 or 0x6840 for axis B):								
Bits 15 - 8	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
	Fault reset	Control bits depending on the operating mode			Enable operation	Activate quick stop	Enable voltage	Switching on
X	0	X	X	X	1	1	1	1

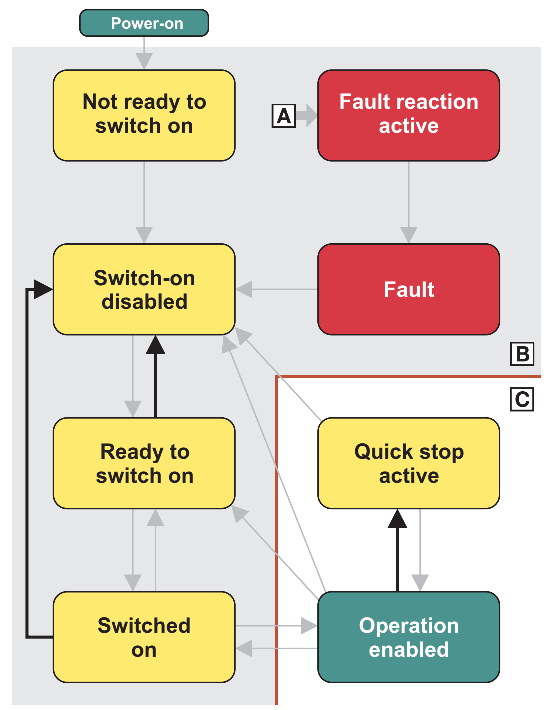
Note!

The signalling of the "Operation enabled" device status in the CiA402 status word can be delayed in the following cases:

- If in case of the synchronous motor servo control the "pole position identification option has been activated before the start in 0x2C63 (or 0x3463 for axis B) and is just running (few milliseconds).
- If the brake is in the "control via device state machine" mode and the brake opening time (0x2820 or 0x3020 for axis B) has not elapsed yet.
- If an asynchronous motor is used which has not been magnetised yet.
  Check the→ setting of the rated motor current (0x6075 or 0x6875 for axis B) and the maximum device current (0x6073 or 0x6873 for axis B).

Only when the "Operation enabled" device status is signalled in the CiA402 status word, the points mentioned before are concluded and the i700 servo inverters is ready for the acceptance of setpoints of the Controller.

7.4.1.4 Activate quick stop



A From all states
B Power section inhibited (pulse inhibit)
C Power section enabled

This command serves to activate quick stop when operation is enabled.

- Irrespective of the setpoint specified, the drive is brought to a standstill with the deceleration set for quick stop (0x6085 or 0x6885 for axis B).
- A changeover to the "Quick stop active" device status takes place.

If the operation has not been enabled yet (status "Ready to switch on" or "Switched on"), this command triggers a change to the "Switch on disabled" device status.

Bit pattern for the "Quick stop" in the Controlword (0x6040 or 0x6840 for axis B):								
Bits 15 - 8	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
	Fault reset	Control bits depending on the operating mode			Enable operation	Activate quick stop	Enable voltage	Switching on
X	0	X	X	X	X	0	1	X

Note!

During quick stop, the drive executes the setpoint generation and no longer follows the setpoint defined by the controller!

If several drives execute a chained synchronous motion, the quick stop function has to be coordinated by the Controller by means of a quick stop profile (master function). In this case, quick stop cannot be activated via the control bit 6!

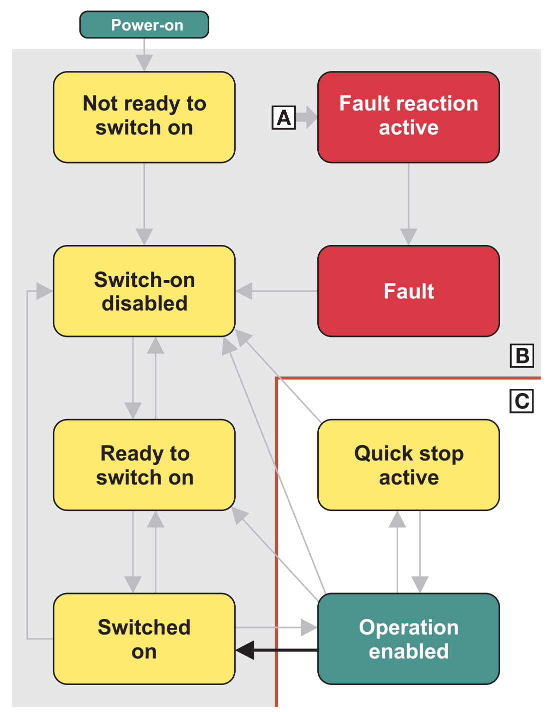
Up to and including software version V01.06.3x

Triggering the quick stop causes the drive to be stopped along the current limit of 0x6073 / 0x6873. The torque limits from 0x60E0 / 0x68E0, 0x60E1 / 0x68E1 and 0x6072 / 0x6872 are not effective during the quick stop!

From software version V01.07.00 onwards

During the quick stop, both the current limit 0x6073 and the torque limit 0x6072 are active. The smaller of the two limits determine the output motor torque. The torque limits from 0x60E0 and 0x60E1 have no effect during the quick stop.

7.4.1.5 Disable operation



A From all states
B Power section inhibited (pulse inhibit)
C Power section enabled

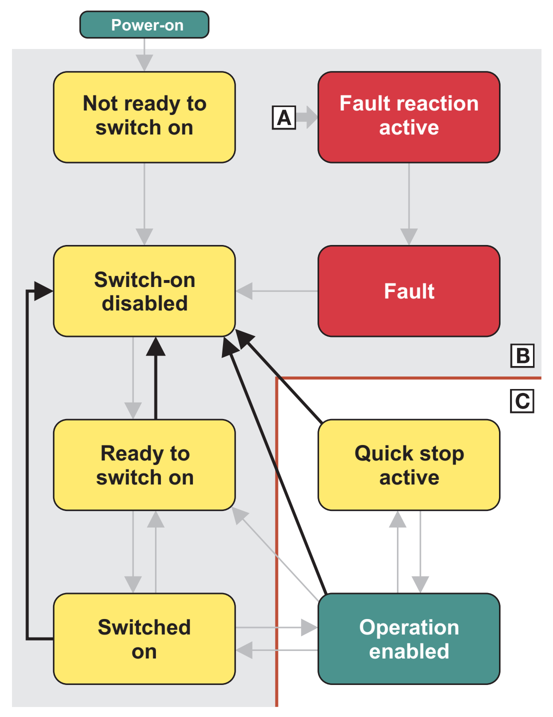
This command serves to inhibit the enabled operation again.

- The pulse inhibit is set.
- If automatic brake operation is activated, the parameterised brake closing time is observed: The system waits until the brake is applied before the pulse inhibit is set.
- A changeover back to the "Switched on" device status takes place.

Bit pattern for the "Disable operation" command in the Controlword (0x6040 or 0x6840 for axis B):

Bits 15 - 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	Fault reset	Control bits depending on the operating mode			Enable operation	Activate quick stop	Enable voltage	Switching on
X	0	X	X	X	0	1	1	1

7.4.1.6 Pulse inhibit



A From all states
B Power section inhibited (pulse inhibit)
C Power section enabled

This command serves to inhibit the output stages of the i700 servo inverters again.

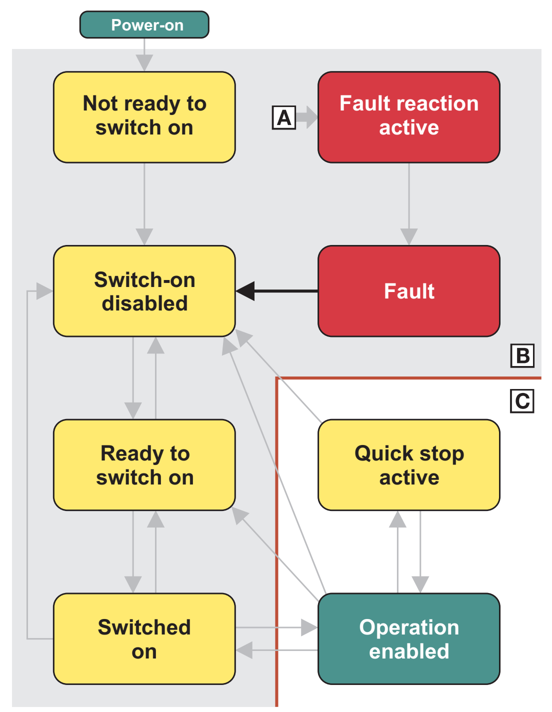
- The pulse inhibit is set (unless they are set already),i.e. the pulses of the i700 servo inverters are inhibited.
- The motor becomes torqueless.
- A changeover back to the "Switch on disabled" device status takes place.

Danger!

If the motor gets torqueless and a load is connected, motors without a holding brake may move in an uncontrolled way!
In case no load is connected, the motor is coasting.

Bit pattern for the "Disable voltage" in the Controlword (0x6040 or 0x6840 for axis B):								
Bits 15 - 8	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
	Fault reset	Control bits depending on the operating mode			Enable operation	Activate quick stop	Enable voltage	Switching on
X	0	X	X	X	X	X	0	X

7.4.1.7 Fault reset



[A] From all states
[B] Power section inhibited (pulse inhibit)
[C] Power section enabled

This command serves to reset an existing fault if the cause of the fault has been eliminated.

- The pulse inhibit remains set.
- A changeover to the "Switch on disabled" device status takes place, i. e. the switch on inhibit is active.

Bit pattern for the "Reset fault" in the Controlword (0x6040 or 0x6840 for axis B):

Bits 15 - 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	Fault reset	Control bits depending on the operating mode			Enable operation	Activate quick stop	Enable voltage	Switching on
X	0 ↗ 1	X	X	X	X	X	X	X

7.4.2 Device states

The current device status of the drive can be read via the Statusword (0x6041 or 0x6841 for axis B):

Device status	Bit pattern in the Statusword							
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	Warning is active	Switch on disabled	Quick stop	Voltage enabled	Fault active	Operation enabled	Switched on	Ready to switch on
Not ready to switch on	X	0	X	X	0	0	0	0
Switch on disabled	X	1	X	X	0	0	0	0
Ready to switch on	X	0	1	X	0	0	0	1
Switched on	X	0	1	X	0	0	1	1
Operation enabled	X	0	1	X	0	1	1	1
Quick stop active	X	0	0	X	0	1	1	1
Fault reaction active	X	0	X	X	1	1	1	1
Fault	X	0	X	X	1	0	0	0
X = Status not significant								

Tip!

Bits 4 ("Voltage enabled") and 7 ("Warning active") are not relevant as regards the device status and merely serve the purpose of a better readability of the bit patterns here.

Detailed information on the different device states can be obtained from the following subchapters.

"Warning active" status bit

Via bit 7 in the Statusword, a warning is indicated.

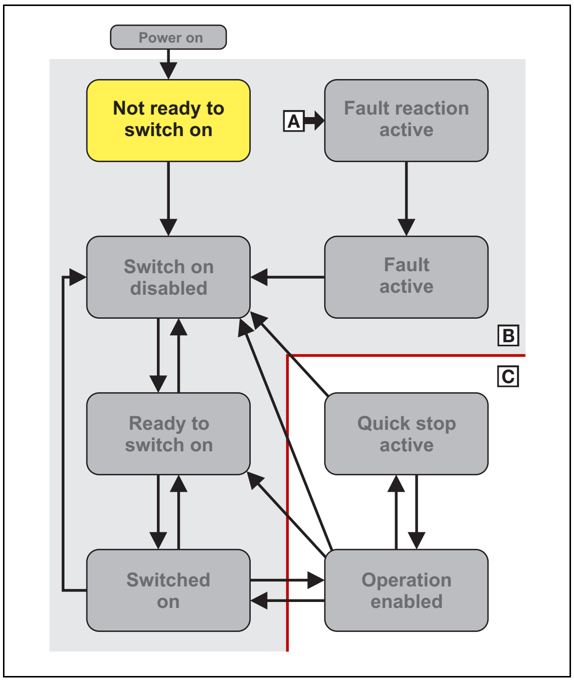
- The occurrence of a warning does not cause a state change.
- Warnings do not need to be reset.

Further Lenze-specific status bits (bits 8 ... 15)

Device status	Bit pattern in the status word							
	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
	STO is not active	Brake released	Following error	Drive follows setpoint selection	Internal limitation is active	Target reached	Remote	RPDOs deactivated
Brake applied	X	0	X	X	X	X	X	X
Brake released	X	1	X	X	X	X	X	X
STO is active	0	X	X	X	X	X	X	X
STO is not active	1	X	X	X	X	X	X	X

X = Status not significant

7.4.2.1 Not ready to switch on



This is the device state of the i700 servo inverters controller directly after switching on the supply voltage.

- In this device status, the device is initialised.
- Communication is not possible yet.
- The i700 servo inverters cannot be parameterised yet and no drive commands can be carried out yet.
- The motor brake, if available, is closed.
- The inverter is inhibited.

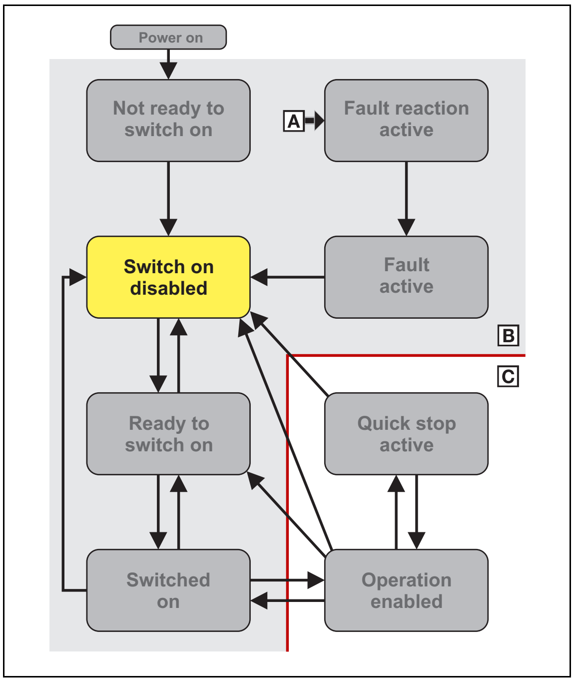
Bit pattern for the "Not ready to switch on" device status in the Statusword (0x6041 or 0x6841 for axis B):

Bits 15 - 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	Warning is active	Switch on disabled	Quick stop	Voltage enabled	Fault active	Operation enabled	Switched on	Ready to switch on
X	X	0	X	X	0	0	0	0

Note!

The i700 servo inverters changes to "Switch on disabled" if the EtherCAT bus is in the "Operational" status or PDO communication is deactivated via object 0x2824 (or 0x3024 for axis B).

7.4.2.2 Switch on disabled

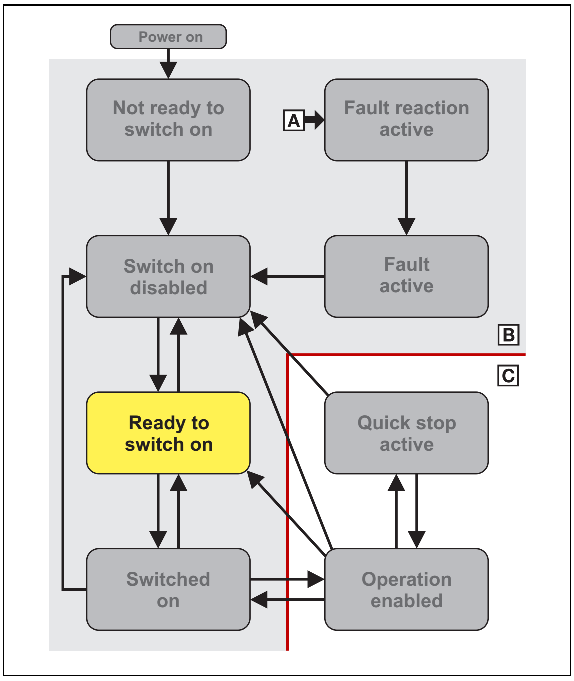


This is the status of the i700 servo inverters after the device has been initialised successfully.

- The process data monitoring is active.
- Communication is possible.
- DC-bus voltage is available.
- The i700 servo inverters can be parameterised.
- If the internal holding brake control is active in the i700 servo inverter (0x2820:1 = 0), the motor brake is closed!
- The inverter is inhibited.

Bit pattern for the "Switch on disabled" device status in the Statusword (0x6041 or 0x6841 for axis B):								
Bits 15 - 8	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
	Warning is active	Switch on disabled	Quick stop	Voltage enabled	Fault active	Operation enabled	Switched on	Ready to switch on
X	X	1	X	X	0	0	0	0

7.4.2.3 Ready to switch on



This is the device status of the i700 servo inverters after the device has been initialised successfully and after the controller has received the "Shutdown" command.

- The process data monitoring is active.
- Communication is possible.
- DC-bus voltage is available.
- The i700 servo inverters can be parameterised.
- If the internal holding brake control is active in the i700 servo inverter (0x2820:1 = 0), the motor brake is closed!
- The inverter is inhibited.

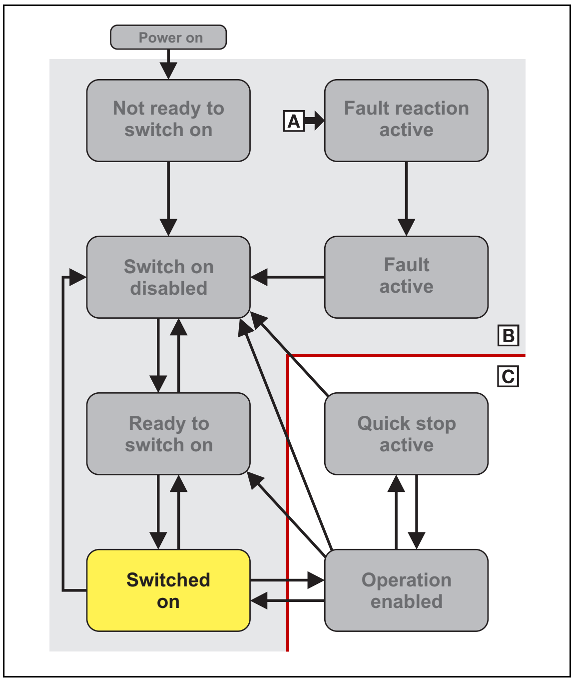
Bit pattern for the "Ready to switch on" device status in the Statusword (0x6041 or 0x6841 for axis B):								
Bits 15 - 8	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
	Warning is active	Switch on disabled	Quick stop	Voltage enabled	Fault active	Operation enabled	Switched on	Ready to switch on
X	X	0	1	X	0	0	0	1

Note!

A changeover to this device status is also effected if, in the "Switched on" or "Operation enabled" device status, the "Shutdown" command is activated.

The change to the sequential "Switched on" state is effected by activation of the "Switching on" command.

7.4.2.4 Switched on



This is the status of the i700 servo inverters after, in the "Ready to switch on" device status, it has received the "Switching on" command.

- The process data monitoring is active.
- Communication is possible.
- DC-bus voltage is available.
- The i700 servo inverters can be parameterised.
- If the internal holding brake control is active in the i700 servo inverter (0x2820:1 = 0), the motor brake is closed!
- The inverter is inhibited.

Bit pattern for the "Switched on" device status in the Statusword (0x6041 or 0x6841 for axis B):								
Bits 15 - 8	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
	Warning is active	Switch on disabled	Quick stop	Voltage enabled	Fault active	Operation enabled	Switched on	Ready to switch on
X	X	0	1	X	0	0	1	1

Note!

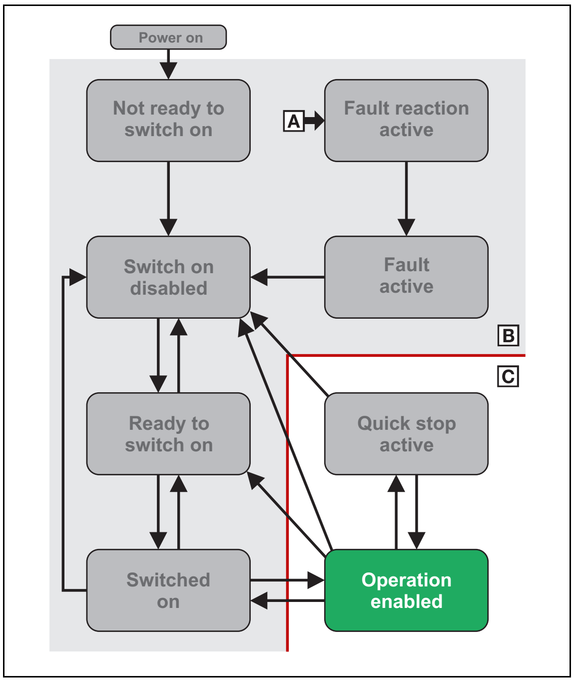
The changeover to the "Operation enabled" device status, and thus to operation enable, is effected by triggering the "Enable operation" command.

The signalling of the "Operation enabled" device status in the CiA402 status word can be delayed in the following cases:

- If in case of the synchronous motor servo control the "pole position identification option has been activated before the start in 0x2C63 (or 0x3463 for axis B) and is just running (few milliseconds).
- If the brake is in the "control via device state machine" mode and the brake opening time (0x2820 or 0x3020 for axis B) has not elapsed yet.
- If an asynchronous motor is used which has not been magnetised yet. Check the→ setting of the rated motor current (0x6075 or 0x6875 for axis B) and the maximum device current (0x6073 or 0x6873 for axis B).

Only when the "Operation enabled" device status is signalled in the CiA402 status word, the points mentioned before are concluded and the i700 servo inverters is ready for the acceptance of setpoints from the control.

7.4.2.5 Operation enabled

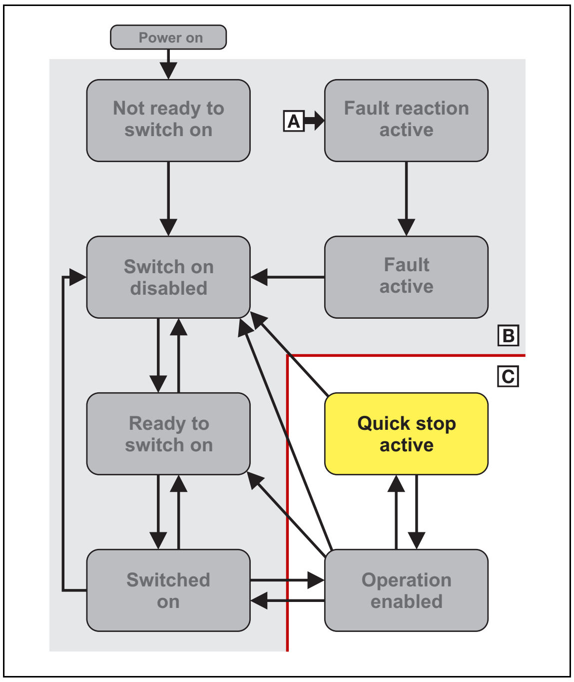


This device status represents normal operation. The operation in the operating mode selected is enabled and no errors are pending.

- Only the parameters of the i700 servo inverters can be changed that do not require controller inhibit.
- A possibly available motor brake is released if the automatic operating mode of the Holding brake control is activated.
- The drive control is working.

Bit pattern for the "Operation enabled" device status in the Statusword (0x6041 or 0x6841 for axis B):								
Bits 15 - 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	Warning is active	Switch on disabled	Quick stop	Voltage enabled	Fault active	Operation enabled	Switched on	Ready to switch on
X	X	0	1	X	0	1	1	1

7.4.2.6 Quick stop active



This device status is active if quick stop is executed or is active.

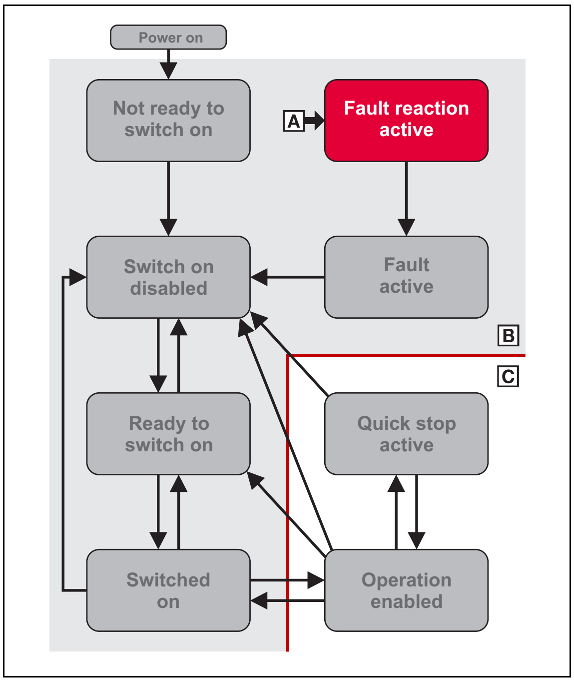
- Only the parameters of the i700 servo inverters can be changed that do not require controller inhibit.
- If the internal holding brake control is active in the i700 servo inverter (0x2820:1 = 0), the motor brake is closed!
- The drive control is working.

Bit pattern for the "Quick stop active" device status in the Statusword (0x6041 or 0x6841 for axis B):								
Bits 15 - 8	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
	Warning is active	Switch on disabled	Quick stop	Voltage enabled	Fault active	Operation enabled	Switched on	Ready to switch on
X	X	0	0	X	0	1	1	1

Note!

The "Enable operation" command serves to deactivate an active quick stop again.

7.4.2.7 Fault reaction active



If a minor fault occurs, i. e. the drive is still able to actuate the motor in a controlled manner, this device status becomes active and the drive is brought to a standstill irrespective of the setpoint specified with the deceleration set for quick stop (0x6085 or 0x6885 for axis B).

- Only the parameters of the i700 servo inverters can be changed that do not require controller inhibit.
- If the internal holding brake control is active in the i700 servo inverter (0x2820:1 = 0), the motor brake is closed!
- The drive control is working.

Bit pattern for the "Fault reaction active" device status in the Statusword (0x6041 or 0x6841 for axis B):

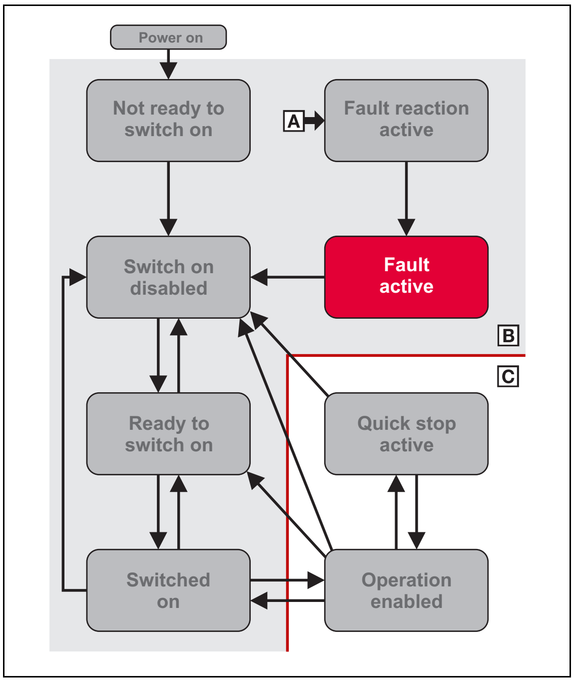
Bits 15 - 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	Warning is active	Switch on disabled	Quick stop	Voltage enabled	Fault active	Operation enabled	Switched on	Ready to switch on
X	X	0	X	X	1	1	1	1

Note!

After quick stop has been executed, i. e. when the drive is at standstill, an automatic changeover to the "Fault" device status is effected.

The "Fault" device status can only be exited using the "Fault reset" command if the cause of the fault is eliminated.

7.4.2.8 Fault



If a grave error occurs, i.e. the drive is not able to operate the motor in a controlled manner anymore, the drive is switched off immediately and this device state becomes active.

- Pulse inhibit is set, i.e. the pulses of the i700 servo inverters are inhibited.
- The motor is torqueless.
- The motor brake, if available, is closed.
- The inverter is inhibited.
- The inverter can be parameterised.

Bit pattern for the "Fault" device status in the Statusword (0x6041 or 0x6841 for axis B):

Bits 15 - 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	Warning is active	Switch on disabled	Quick stop	Voltage enabled	Fault active	Operation enabled	Switched on	Ready to switch on
X	X	0	X	X	1	0	0	0

Note!

This device status can only be left using the "Fault reset" command if the cause of the fault is eliminated.

_ _

7.4.2.9 STO ("Safe Torque Off")

The status of "Safe Torque Off" is included in bit 15 as part of the CiA402 status word. Bit 15 contains information whether STO is active or not.

This status information is required since the activation of STO causes all integral control parts to be deleted. In case of hoists, for instance, the drive would be sagging without any corrective measures after completing STO.

In order to prevent this unwanted state, the control has to be preloaded with a starting value after completing STO:

- In case of a control type with encoder, the integral-action component of the speed controller has to be preloaded.
- In case of the "VFC" control type, the slip frequency has to be preloaded.

Note!

If STO is inhibited, a changeover to the "Switch on disabled" device status takes place and a warning is transmitted.

7.4.3 Selection of the operating mode

The drive behaviour depends on the selected operating mode. The operating mode is selected by the Controller via the 0x6060 object (or 0x6860 for axis B) in the communication statuses "Pre-Operational" or "Operational". Access can be effected via SDO or PDO.

- Only one operating mode at a time can be active.
- The operating modes differ by the type of setpoint selection.
- The following table shows the permissible operating modes with the corresponding object value:

Operating mode	Object value	Setpoint selection
Velocity mode (vl)	2	Velocity
Cyclic sync position mode (csp)	8	Position
Cyclic sync velocity mode (csv)	9	Velocity
Cyclic sync torque mode (cst)	10	Torque
-	All other values	- (Standstill)

Tip!

The operating mode currently set can be read via the 0x6061 object (or 0x6861 for axis B). For this object, too, access is possible via SDO or PDO.

7.5 Parameters for the scaling of physical values

Objects described in this chapter:

Object Axis A	Axis B	Name	Data type
0x607E	0x687E	Polarity	UNSIGNED_8
0x6080	0x6880	Max motor speed	UNSIGNED_32
0x608F	0x688F	Position encoder resolution	RECORD
0x6090	0x6890	Velocity encoder resolution	RECORD

0x607E | 0x687E - Polarity

Mounting position
- Only the setting "0" is accepted.

Setting range (min. value | unit | max. value)			Lenze setting	
0		0	0	
☑ Write access ☑ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_8

0x6080 | 0x6880 - Max. motor speed

Speed limitation

Setting range (min. value | unit | max. value)			Lenze setting	
0	r/min	480000	6075 rpm	
☑ Write access ☐ CINH ☑ OSC ☑ P ☑ RX ☐ TX				UNSIGNED_32

0x608F | 0x688F - Position encoder resolution

Resolution of the position detection by the motor encoder

Sub.	Name	Lenze setting	Data type
1	Encoder increments	0x00010000: 16 Bit	UNSIGNED_32
2	Motor revolutions	1	UNSIGNED_32

Subindex 1: **Encoder increments**		
Setting of the number of bits with which a mechanical motor revolution is to be resolved.		
Selection list (Lenze setting printed in bold)		
0x00010000	**16 bits**	
0x00040000	18 bits	
0x00100000	20 bits	
0x00400000	22 bits	
0x01000000	24 bits	
0x04000000	26 bits	
0x10000000	28 bits	
0x40000000	30 bits	
☑ Write access ☑ CINH ☐ OSC ☑ P ☐ RX ☐ TX		UNSIGNED_32

Subindex 2: **Motor revolutions**				
Number of motor revolutions • Only the setting "1" is accepted.				
Setting range (min. value | unit | max. value)			Lenze setting	
1		1	1	
☑ Write access ☑ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_32

Explanation

The 0x608F index can be used to select a specific part of a motor position for position detection. The selection also determines the resolution of the position setpoints and the quality or rigidity of a position control.

The selection to be made in the 0x608F:1 index directly determines the resolution of the motor revolutions and indirectly determines the resolution of the motor revolutions to be counted.

An example illustrates this circumstance: Index 0x608F:1 = "18 Bit" means that a motor revolution is displayed in a resolution of 18 bits. The i700 servo inverter is internally dimensioned to variables with 32 bits. Thus, the remaining 14 bits are used for counting the motor revolutions.

The Lenze setting considers the use of a Lenze motor with resolver feedback. The resolver provides a value for a motor revolution in a resolution of approximately 16 bits.

Selection criteria

If the number of motor revolutions to be distinguished on the part of the process is not important, the following recommendation can be given regarding the setting of 0x608F:

g (encoder increment) / Log(2) + 10 bi

0x608F is selected in such a way that the values to be parameterised only slightly exceed the display option of the feedback system.

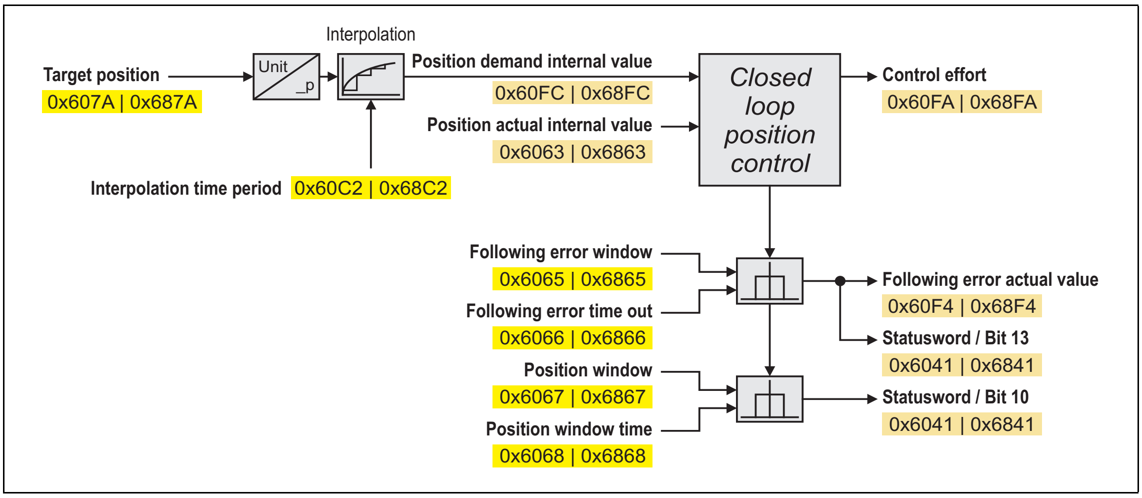
0x6090 | 0x6890 - Velocity encoder resolution

Note: This object is not used in the firmware and only available for compatibility reasons. Do not use the contained values for a calculation on the control level.

Sub.	Name	Lenze setting	Data type
1	Incr./s	33554432	UNSIGNED_32
2	Motor revolutions/s	125	UNSIGNED_32
☑ Write access ☑ CINH ☐ OSC ☑ P ☐ RX ☐ TX			

7.6 Parameters for actuation of the position control

The objects described in this chapter actuate the position control (including the following error and in-position recognition). All objects comply with the CiA402 specification.



[7-1] Access to the position control via the CiA402 device profile

Input data

Object		Name	Data type
Axis A	Axis B		
0x60FC	0x68FC	Position demand internal value	INTEGER_32
0x6062	0x6862	Position demand value	INTEGER_32
0x6065	0x6865	Following error window	UNSIGNED_32
0x6066	0x6866	Following error time out	UNSIGNED_16
0x6067	0x6867	Position window	UNSIGNED_32
0x6068	0x6868	Position window time	UNSIGNED_16
Greyed out = read access only			

Output data

Object		Name	Data type
Axis A	Axis B		
0x6063	0x6863	Position actual internal value	INTEGER_32
0x6064	0x6864	Position actual value	INTEGER_32
0x60F4	0x68F4	Following error actual value	INTEGER_32
0x60FA	0x68FA	Control effort	INTEGER_32
▶ Drive control			
0x6041	0x6841	Statusword	UNSIGNED_16
Greyed out = read access only			

0x6062 | 0x6862 - Position demand value

Interpolated set position for the position control

Display range (min. value | unit | max. value)			Initialisation	
-2147483648	[Pos unit]	2147483647	0 [Pos unit]	
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☐ TX				INTEGER_32

0x6063 | 0x6863 - Position actual internal value

Actual position in internal unit

Display range (min. value | unit | max. value)			Initialisation	
-2147483648	Incr.	2147483647		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX				INTEGER_32

0x6064 | 0x6864 - Position actual value

Actual position

Display range (min. value | unit | max. value)			Initialisation	
-2147483648	[Pos unit]	2147483647		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX				INTEGER_32

siehe Erläuterungen zum Index 0x608F

0x6065 | 0x6865 - Following error window

Symmetrical tolerance window around the set position for the purpose of following error detection
- The following error detection is active as soon as the **cyclic sync position mode** (0x6060 = "8") is set. It cannot be deactivated directly but only by entering very high values.
- If the tolerance window is set > "0", the following error detection is activated, i.e. a following error will be detected if the actual position is outside this tolerance window.
- If the following error detection is active: If the following error is detected for a period longer than that defined in [ms] in object 0x6066 (or 0x6866 for axis B), bit 13 ("Following error") is set in the Statusword (0x6041 or 0x6841 for axis B).
- Object 0x60F4 (or 0x68F4 for axis B) shows the current deviation of the actual position to the set position.

Setting range (min. value | unit | max. value)			Lenze setting	
0	[Pos unit]	4294967295	1000 [Pos unit]	
☑ Write access ☐ CINH ☑ OSC ☑ P ☑ RX ☐ TX				UNSIGNED_32

0x6066 | 0x6866 - Following error time out

Time monitoring for following error detection
0 = following error is evaluated without time delay.

Setting range (min. value | unit | max. value)			Lenze setting	
0	ms	0	0 ms	
☑ Write access ☐ CINH ☑ OSC ☑ P ☑ RX ☐ TX				UNSIGNED_16

0x6067 | 0x6867 - Position window

Symmetrical tolerance window around the target position for in-position recognition
- If the actual position is within this tolerance window for a longer period than for the time defined in [ms] in object 0x6068 (or 0x6868 for axis B) , the target position is considered as reached, and bit 10 ("Target position reached") is set in the Statusword (0x6041 or 0x6841 for axis B).

Setting range (min. value | unit | max. value)			Lenze setting	
0	[Pos unit]	4294967295	1000 [Pos unit]	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_32

0x6068 | 0x6868 - Position window time

Time monitoring for the status message "target position reached"
0 = position in the target window is evaluated without time delay.

Setting range (min. value | unit | max. value)			Lenze setting	
0	ms	0	0 ms	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

0x60F4 | 0x68F4 - Following error actual value

Current following error

Display range (min. value | unit | max. value)			Initialisation	
-2147483648	[Pos unit]	2147483647		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX				INTEGER_32

0x60FA | 0x68FA - Control effort

Target position for the position control

Display range (min. value | unit | max. value)			Initialisation	
-480000	[n unit]	480000		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: $480000/2^{31}$	INTEGER_32

0x60FC | 0x68FC - Position demand internal value

Interpolated set position for the position control in the internal unit

Display range (min. value | unit | max. value)			Initialisation	
-2147483648	Incr.	2147483647	0 incr.	
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☐ TX				INTEGER_32

7.7 Velocity mode (vl)

Selection of the operating mode

"Velocity mode" is selected with the setting "2" in 0x6060 (or 0x6860 for axis B).

7.7.1 Default mapping

The default mapping for the "Velocity mode" is defined in the following objects:

Object Axis A	 Axis B	Name	Data type
0x1603	0x1613	RPDO-->Axis A/B: Velocity mode (vl)	RECORD
0x1A03	0x1A13	Axis A/B-->TPDO: Velocity mode (vl)	RECORD

Date received from the controller (RPDO)

Object Axis A	 Axis B	Name	Data type
0x6040	0x6840	Controlword	UNSIGNED_16
0x2830	0x3030	Lenze control word	UNSIGNED_16
0x6060	0x6860	Operating mode: Selection	INTEGER_8
0x6042	0x6842	vl target velocity	INTEGER_16

Data transmitted to the controller (TPDO)

Object Axis A	 Axis B	Name	Data type
0x6041	0x6841	Statusword	UNSIGNED_16
0x2831	0x3031	Lenze status word	UNSIGNED_16
0x6061	0x6861	Modes of operation display	INTEGER_8
0x603F	0x683F	Error code	UNSIGNED_16
0x6044	0x6844	vl velocity actual value	INTEGER_16

7.7.2 Object description

The following two tables provide an overview of the most important objects for this operating mode (without motor parameters, motor control parameters, and feedback parameters).

All objects correspond to the CiA402 specification, but some of them have only a restricted value range.

Objects described in other chapters:

Object Axis A	 Axis B	Name	Data type
▸ Lenze control and status word			
0x2830	0x3030	Lenze control word	UNSIGNED_16
0x2831	0x3031	Lenze status word	UNSIGNED_16
▸ Drive control			
0x6040	0x6840	Controlword	UNSIGNED_16
0x6060	0x6860	Operating mode: Selection	INTEGER_8
0x6041	0x6841	Statusword	UNSIGNED_16
0x6061	0x6861	Modes of operation display	INTEGER_8
Greyed out = read access only			

Objects described in this chapter:

Object Axis A	 Axis B	Name	Data type
0x6042	0x6842	vl target velocity	INTEGER_16
0x6043	0x6843	vl velocity demand	INTEGER_16
0x6044	0x6844	vl velocity actual value	INTEGER_16
0x6046	0x6846	vl velocity min max amount	RECORD
0x6048	0x6848	vl velocity acceleration	RECORD
0x6049	0x6849	vl velocity deceleration	RECORD

0x6042 | 0x6842 - vl target velocity

Setting range (min. value | unit | max. value)			Lenze setting	
-32768	r/min	32767	0 rpm	
☑ Write access ☐ CINH ☑ OSC ☐ P ☑ RX ☐ TX				INTEGER_16

0x6043 | 0x6843 - vl velocity demand

Display range (min. value | unit | max. value)			Initialisation	
-32768	r/min	32767		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX				INTEGER_16

0x6044 | 0x6844 - vl velocity actual value

Display range (min. value | unit | max. value)			Initialisation	
-32768	r/min	32767		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX				INTEGER_16

0x6046 | 0x6846 - vl velocity min max amount

Sub.	Name	Lenze setting	Data type
1	vl velocity min amount	0 rpm	UNSIGNED_32
2	vl velocity max amount	2147483647 rpm	UNSIGNED_32

Subindex 1: **vl velocity min amount**				
Setting range (min. value | unit | max. value)			Lenze setting	
0	r/min	0	0 rpm	
☑ Write access ☐ CINH ☑ OSC ☐ P ☑ RX ☐ TX				UNSIGNED_32

Subindex 2: **vl velocity max amount**				
Setting range (min. value | unit | max. value)			Lenze setting	
2147483647	r/min	2147483647	2147483647 rpm	
☑ Write access ☐ CINH ☑ OSC ☐ P ☑ RX ☐ TX				UNSIGNED_32

0x6048 | 0x6848 - vl velocity acceleration

Sub.	Name	Lenze setting	Data type
1	Delta speed	0 rpm	UNSIGNED_32
2	Delta time	10 s	UNSIGNED_16

Subindex 1: **Delta speed**				
Setting range (min. value | unit | max. value)			Lenze setting	
0	r/min	2147483647	0 rpm	
☑ Write access ☐ CINH ☑ OSC ☑ P ☑ RX ☐ TX				UNSIGNED_32

Subindex 2: **Delta time**				
Setting range (min. value | unit | max. value)			Lenze setting	
0	s	65535	10 s	
☑ Write access ☐ CINH ☑ OSC ☑ P ☑ RX ☐ TX				UNSIGNED_16

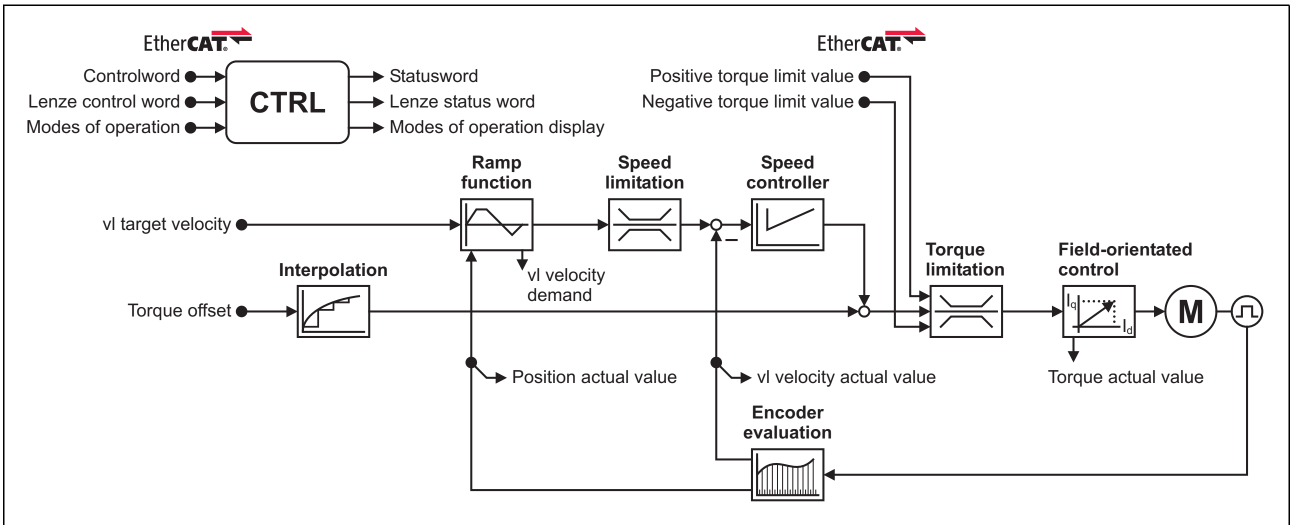
0x6049 | 0x6849 - vl velocity deceleration

Sub.	Name	Lenze setting	Data type
1	Delta speed	0 rpm	UNSIGNED_32
2	Delta time	10 s	UNSIGNED_16

Subindex 1: **Delta speed**				
Setting range (min. value | unit | max. value)			Lenze setting	
0	r/min	2147483647	0 rpm	
☑ Write access ☐ CINH ☑ OSC ☑ P ☑ RX ☐ TX				UNSIGNED_32

Subindex 2: **Delta time**				
Setting range (min. value | unit | max. value)			Lenze setting	
0	s	65535	10 s	
☑ Write access ☐ CINH ☑ OSC ☑ P ☑ RX ☐ TX				UNSIGNED_16

7.7.3 Signal flow (servo control)



[7-2] Signal flow of the servo control in "Velocity mode" (simplified representation)

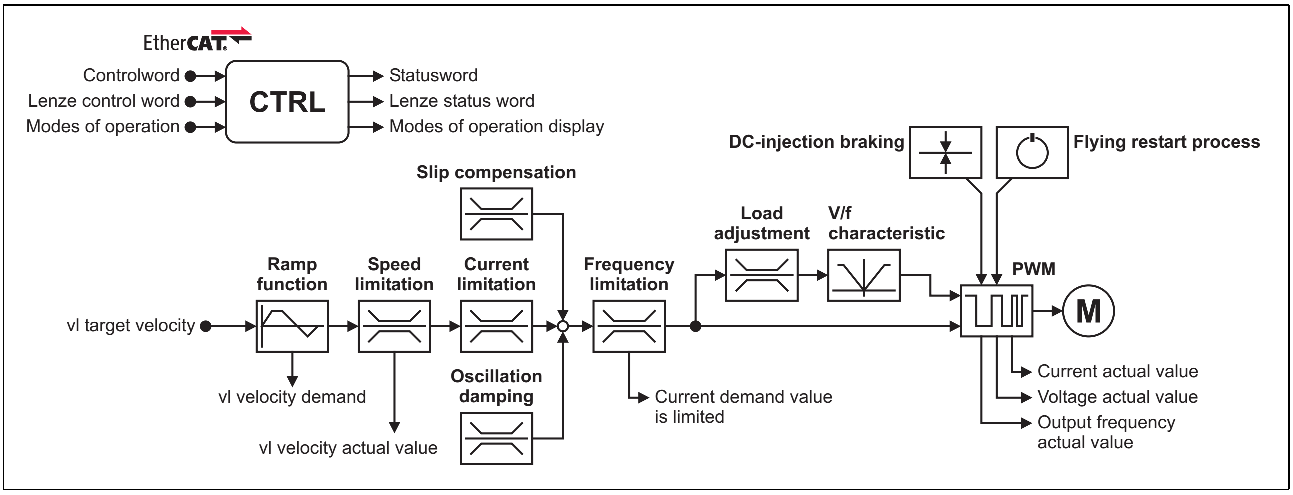
Short overview of the most important parameters:

Function	Object		Name
	Axis A	Axis B	
Input data	0x6040	0x6840	Controlword
	0x2830	0x3030	Lenze control word
	0x6060	0x6860	Operating mode: Selection
	0x6042	0x6842	vl target velocity
	0x60B2	0x68B2	Torque offset
	0x60E0	0x68E0	Positive torque limit value
	0x60E1	0x68E1	Negative torque limit value

Function	Object		Name
	Axis A	Axis B	
Output data	0x6041	0x6841	Statusword
	0x2831	0x3031	Lenze status word
	0x6061	0x6861	Modes of operation display
	0x6043	0x6843	vl velocity demand
	0x6064	0x6864	Position actual value
	0x606C	0x686C	Velocity actual value
	0x6077	0x6877	Torque actual value
Interpolation	0x60C2	0x68C2	Interpolation time period
Ramp function	0x6048:1	0x6848:1	Delta speed
	0x6048:2	0x6848:2	Delta time
	0x6049:1	0x6849:1	Delta speed
	0x6049:2	0x6849:2	Delta time
Speed limitation	0x6080	0x6880	Max motor speed
	0x2903	0x3103	Speed: Speed setpoint - filter time
Speed controller	0x2900:1	0x3100:1	Speed controller: Gain
	0x2900:2	0x3100:2	Speed controller: Reset time
	0x2900:3	0x3100:3	Speed controller: Rate time
	0x2901	0x3101	Speed controller: Gain - adaptation
	0x2902:0	0x3102:0	Speed controller: Load I component
Torque limitation	0x60E0	0x68E0	Positive torque limit value
	0x60E1	0x68E1	Negative torque limit value
	0x6076	0x6876	Motor rated torque
	0x6072	0x6872	Max torque
	0x2944:1	0x3144:1	Notch filter 1: Frequency
	0x2944:2	0x3144:2	Notch filter 1: Bandwidth
	0x2944:3	0x3144:3	Notch filter 1: Damping
	0x2944:4	0x3144:4	Notch filter 2: Frequency
	0x2944:5	0x3144:5	Notch filter 2: Bandwidth
	0x2944:6	0x3144:6	Notch filter 2: Damping

Function	Object		Name
	Axis A	Axis B	
Field-oriented control	0x6073	0x6873	Max current
	0x6075	0x6875	Motor rated current
	0x2941	0x3141	Current controller: Feedforward control
	0x2942:1	0x3142:1	Current controller: Gain
	0x2942:2	0x3142:2	Current controller: Reset time
	0x29E2	0x31E2	DC link circuit voltage: Filter time
	0x29E3	0x31E3	Motor voltage act. value: Filter time
	0x29E0:1	0x31E0:1	Field weakening controller: Gain
	0x29E0:2	0x31E0:2	Field weakening controller: Reset time
	0x29E1	0x31E1	Field set value limitation
	0x29C0:1	0x31C0:1	Field controller: Gain
	0x29C0:2	0x31C0:2	Field controller: Reset time
	0x2939	0x3139	Switching frequency

7.7.4 Signal flow (V/f characteristic control)



[7-3] Signal flow of the V/f characteristic control in "Velocity mode" (simplified representation)

Short overview of the most important parameters:

Function	Object		Name
	Axis A	Axis B	
Input data	0x6040	0x6840	Controlword
	0x2830	0x3030	Lenze control word
	0x6060	0x6860	Operating mode: Selection
	0x6042	0x6842	vl target velocity

Function	Object		Name
	Axis A	Axis B	
Output data	0x6041	0x6841	Statusword
	0x2831	0x3031	Lenze status word
	0x6061	0x6861	Modes of operation display
	0x6043	0x6843	vl velocity demand
	0x606C	0x686C	Velocity actual value
	0x6078	0x6878	Current actual value
	0x2D82	0x3582	Motor: Actual voltage - Veff, phase-phase
	0x2DDD	0x35DD	Device: Actual output frequency
Ramp function	0x6048:1	0x6848:1	Delta speed
	0x6048:2	0x6848:2	Delta time
	0x6049:1	0x6849:1	Delta speed
	0x6049:2	0x6849:2	Delta time
Speed limitation	0x6080	0x6880	Max motor speed
	0x2903	0x3103	Speed: Speed setpoint - filter time
Slip compensation	0x2B09:1	0x3309:1	VFC: Slip compensation - influence
	0x2B09:2	0x3309:2	VFC: Slip compensation - filter time
Oscillation damping	0x2B0A:1	0x330A:1	VFC: Oscillation damping - gain
	0x2B0A:2	0x330A:2	VFC: Oscillation damping - filter time
	0x2B0A:3	0x330A:3	VFC: Oscillation damping - limitation
	0x2B0A:4	0x330A:4	VFC: Oscillation damping - ramp-end frequency
Load adjustment	0x2B07:1	0x2B07:1	VFC: Load adjustment - direction of rotation
	0x2B07:2	0x2B07:2	VFC: Load adjustment - value
V/f characteristic	0x2B01:1	0x3301:1	VFC: V/f characteristic - voltage at reference point
	0x2B01:2	0x3301:2	VFC: V/f characteristic - frequency at reference point
	0x2B06	0x3306	VFC: Voltage boost
	0x2B04	0x3304	VFC: Voltage vector control - current setpoint
	0x2B00	0x3300	VFC: V/f characteristic - shape
	0x2B02:x	0x3302:x	VFC: User defined V/f characteristic • Frequency grid points (x1 ... x11)
	0x2B03:x	0x3303:x	VFC: User defined V/f characteristic • Voltage grid points (y1 ... y11)
DC-injection braking	0x2B80	0x3380	DC-injection braking: Current
Flying restart function	0x2BA0	0x33A0	Flying restart: Activate
	0x2BA1	0x33A1	Flying restart: Current
	0x2BA2	0x33A2	Flying restart: Start frequency
	0x2BA3	0x33A3	Flying restart: Integration time
	0x2BA4	0x33A4	Flying restart: Min. deviation
	0x2BA5	0x33A5	Flying restart: Delay time
	0x2BA6	0x33A6	Flying restart: Result

Tip!

A detailed representation of the signal flow with all relevant parameters can be found in the »PLC Designer« on the **Signal flow** tab for the i700 servo inverters.

7.8 Cyclic sync position mode (csp)

This operating mode provides a quick position follower with speed/torque/feed force feedforward control. The motion profile to be processed is defined by the controller.

Note!

This operating mode can only be actuated reasonably by the use of a servo control.

▶ Motor control & motor settings

Subfunctions of the operating mode

- Calculation of the position setpoint
- Calculation of the speed feedforward control value
- Calculation of the speed feedforward control value
- Interpolation between communication cycle and control cycle
- Position control
- Velocity control
- Closed-loop torque control
- Update of the actual values for position, speed, and torque

Selection of the operating mode

"Cyclic sync position mode" is selected with the setting "8" in 0x6060 (or 0x6860 for axis B).

7.8.1 Default mapping

The default mapping for the cyclic sync position mode is defined in the following objects:

Object Axis A	Axis B	Name	Data type
0x1600	0x1610	RPDO-->Axis A/B: Cyclic syncs position (csp)	RECORD
0x1606	0x1616	RPDO-->Axis A/B: Torque limits	RECORD
0x1A00	0x1A10	Axis A/B-->TPDO: Cyclic sync position (csp)	RECORD

Date received from the controller (RPDO)

Object Axis A	Axis B	Name	Data type
0x6040	0x6840	Controlword	UNSIGNED_16
0x2830	0x3030	Lenze control word	UNSIGNED_16
0x6060	0x6860	Operating mode: Selection	INTEGER_8
0x60B2	0x68B2	Torque offset	INTEGER_16
0x607A	0x687A	Position demand value	INTEGER_32
0x60B1	0x68B1	Velocity offset	INTEGER_32
0x2902	0x3102	Speed controller: Load I component	INTEGER_16
0x60E0	0x68E0	Positive torque limit value	UNSIGNED_16
0x60E1	0x68E1	Negative torque limit value	UNSIGNED_16

Data transmitted to the controller (TPDO)

Object Axis A	Axis B	Name	Data type
0x6041	0x6841	Statusword	UNSIGNED_16
0x2831	0x3031	Lenze status word	UNSIGNED_16
0x6061	0x6861	Modes of operation display	INTEGER_8
0x603F	0x683F	Error code	UNSIGNED_16
0x606C	0x686C	Velocity actual value	UNSIGNED_16
0x6077	0x6877	Torque actual value	INTEGER_16
0x6064	0x6864	Position actual value	INTEGER_32
0x60F4	0x68F4	Following error actual value	INTEGER_32

7.8.2 Object description

The following two tables provide an overview of the most important objects for this operating mode (without motor parameters, motor control parameters, and feedback parameters).

All objects correspond to the CiA402 specification, but some of them have only a restricted value range.

Objects described in other chapters:

Object Axis A	 Axis B	Name	Data type
Lenze control and status word			
0x2830	0x3030	Lenze control word	UNSIGNED_16
0x2831	0x3031	Lenze status word	UNSIGNED_16
Drive control			
0x6040	0x6840	Controlword	UNSIGNED_16
0x6060	0x6860	Modes of operation	INTEGER_8
0x6041	0x6841	Statusword	UNSIGNED_16
0x6061	0x6861	Modes of operation display	INTEGER_8
Parameters for actuation of the position control			
0x6065	0x6865	Following error window	UNSIGNED_32
0x6066	0x6866	Following error time out	UNSIGNED_16
0x6067	0x6867	Position window	UNSIGNED_32
0x6068	0x6868	Position window time	UNSIGNED_16
0x6062	0x6862	Position demand value	INTEGER_32
0x6063	0x6863	Position actual internal value	INTEGER_32
0x6064	0x6864	Position actual value	INTEGER_32
0x60F4	0x68F4	Following error actual value	INTEGER_32
0x60FC	0x68FC	Position demand internal value	INTEGER_32
Cyclic sync velocity mode (csv)			
0x60B1	0x68B1	Velocity offset	INTEGER_32
0x60FF	0x68FF	Target velocity	INTEGER_32
0x606C	0x686C	Velocity actual value	UNSIGNED_16
Cyclic sync torque mode (cst)			
0x60B2	0x68B2	Torque offset	INTEGER_16
0x6071	0x6871	Target torque	INTEGER_16
0x6077	0x6877	Torque actual value	INTEGER_16
Greyed out = read access only			

Objects described in this chapter:

Object Axis A	 Axis B	Name	Data type
0x607A	0x687A	Position demand value	INTEGER_32
0x60C0	0x68C0	Interpolation sub mode select	INTEGER_16
0x60C2	0x68C2	Interpolation time period	RECORD
0x60E0	0x68E0	Positive torque limit value	UNSIGNED_16
0x60E1	0x68E1	Negative torque limit value	UNSIGNED_16

0x607A | 0x687A - Target position

Setting range (min. value | unit | max. value)			Lenze setting	
-2147483648	[Pos unit]	2147483647	0 [Pos unit]	
☑ Write access ☐ CINH ☑ OSC ☐ P ☑ RX ☐ TX				INTEGER_32

siehe Erläuterungen zum Index 0x608F

0x60C0 | 0x68C0 - Interpolation sub mode select

From software version V01.03.xx onwards
Interpolation sub mode select
0 = linear position interpolation
-1 = square position interpolation
All other values have no function and are rejected.

Setting range (min. value | unit | max. value)			Lenze setting	
-32768		32767	-1	
☑ Write access ☐ CINH ☐ OSC ☑ P ☑ RX ☐ TX				INTEGER_16

0x60C2 | 0x68C2 - Interpolation time period

Here the cycle time for the process data communication used by the L-force Controller is to be set.
- Preset interpolation time interval = 1 * 10^{-3} s = 1 ms

Sub.	Name	Lenze setting	Data type
▸ 1	Interpolation time period	1 10^(Interpolation time index) s	UNSIGNED_8
▸ 2	Interpolation time index	-3	INTEGER_8

Subindex 1: **Interpolation time period**				
Basic multiplier for interpolation time interval				
Setting range (min. value | unit | max. value)			Lenze setting	
0	10^(Interpolation time index) s	255	1 10^(Interpolation time index) s	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_8

Subindex 2: **Interpolation time index**				
Exponent for interpolation time period				
Setting range (min. value | unit | max. value)			Lenze setting	
-6		0	-3	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				INTEGER_8

$$\text{Interpolation time period}[s] = \text{Time period value} \cdot 10^{\text{Index}}[s]$$

[7-4] Definition of the interpolation time period

Tip!

For an interpolation cycle of 2 ms, for instance, the following values are to be set:
- Subindex 1 (Time period value) = "2"
- Subindex 2 (Time index) = "-3"

0x60E0 | 0x68E0 - Positive torque limit value

100 % ≡ rated motor torque (0x6076 or 0x6876 for axis B)

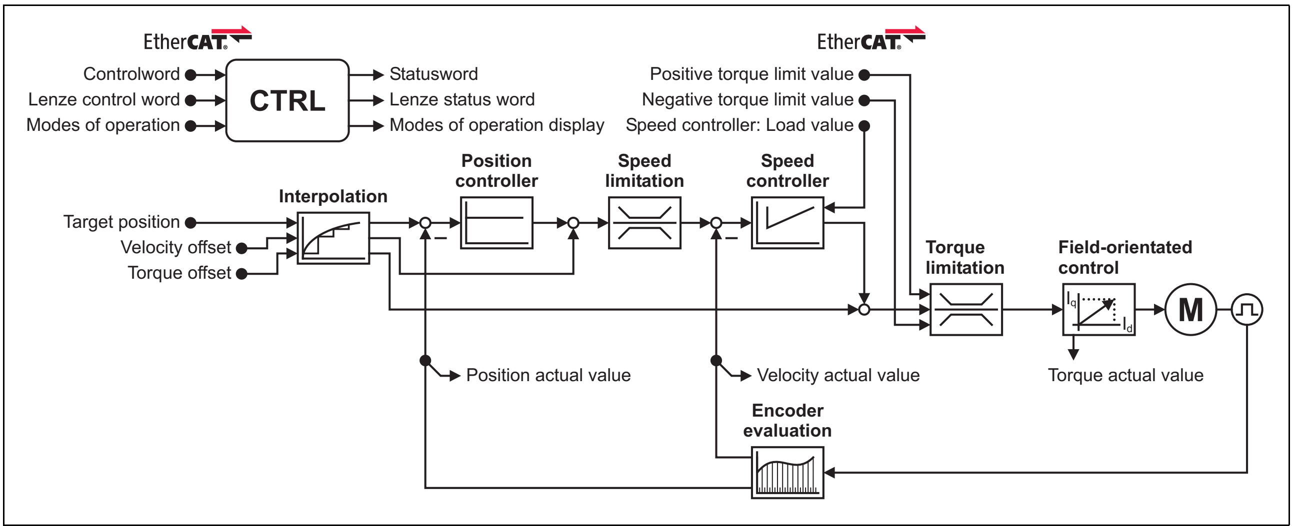
Setting range (min. value | unit | max. value)			Lenze setting	
0.0	%	3276.7	100.0 %	
☑ Write access ☐ CINH ☑ OSC ☑ P ☑ RX ☐ TX			Scaling: 1/10	UNSIGNED_16

0x60E1 | 0x68E1 - Negative torque limit value

100 % ≡ rated motor torque (0x6076 or 0x6876 for axis B)

Setting range (min. value | unit | max. value)			Lenze setting	
0.0	%	3276.7	100.0 %	
☑ Write access ☐ CINH ☑ OSC ☑ P ☑ RX ☐ TX			Scaling: 1/10	UNSIGNED_16

7.8.3 Signal flow



[7-5] Signal flow of the servo control in cyclic sync position mode (simplified representation)

Short overview of the most important parameters:

Function	Object		Name
	Axis A	Axis B	
Input data	0x6040	0x6840	Controlword
	0x2830	0x3030	Lenze control word
	0x6060	0x6860	Operating mode: Selection
	0x607A	0x687A	Position demand value
	0x60B1	0x68B1	Velocity offset
	0x60B2	0x68B2	Torque offset
	0x60E0	0x68E0	Positive torque limit value
	0x60E1	0x68E1	Negative torque limit value
	0x2902	0x3102	Speed controller: Load I component
Output data	0x6041	0x6841	Statusword
	0x2831	0x3031	Lenze status word
	0x6061	0x6861	Modes of operation display
	0x6064	0x6864	Position actual value
	0x606C	0x686C	Velocity actual value
	0x6077	0x6877	Torque actual value
Interpolation	0x60C0	0x68C0	Interpolation sub mode select
	0x60C2	0x68C2	Interpolation time period
Position controller	0x2980	0x3180	Position controller: Gain
	0x2981	0x3181	Position controller: Gain - adaptation
	0x2982	0x3182	Position controller: Output signal limitation
	0x2983	0x3183	Position: Select new actual position
	0x2984	0x3184	Determine target position: Mode
	0x2985	0x3185	Position controller: Internal adaptation - characteristics
	0x2986	0x3186	Position controller: Resulting gain adaptation
Speed limitation	0x6080	0x6880	Max motor speed
	0x2903	0x3103	Speed: Speed setpoint - filter time
Speed controller	0x2900:1	0x3100:1	Speed controller: Gain
	0x2900:2	0x3100:2	Speed controller: Reset time
	0x2900:3	0x3100:3	Speed controller: Rate time
	0x2901	0x3101	Speed controller: Gain - adaptation
	0x2902	0x3102	Speed controller: Load I component
Torque limitation	0x60E0	0x68E0	Positive torque limit value
	0x60E1	0x68E1	Negative torque limit value
	0x6076	0x6876	Motor rated torque
	0x6072	0x6872	Max torque
	0x2944:1	0x3144:1	Notch filter 1: Frequency
	0x2944:2	0x3144:2	Notch filter 1: Bandwidth
	0x2944:3	0x3144:3	Notch filter 1: Damping
	0x2944:4	0x3144:4	Notch filter 2: Frequency
	0x2944:5	0x3144:5	Notch filter 2: Bandwidth
	0x2944:6	0x3144:6	Notch filter 2: Damping

Function	Object		Name
	Axis A	Axis B	
Field-oriented control	0x6073	0x6873	Max current
	0x6075	0x6875	Motor rated current
	0x2941	0x3141	Current controller: Feedforward control
	0x2942:1	0x3142:1	Current controller: Gain
	0x2942:2	0x3142:2	Current controller: Reset time
	0x29E2	0x31E2	DC link circuit voltage: Filter time
	0x29E3	0x31E3	Motor voltage act. value: Filter time
	0x29E0:1	0x31E0:1	Field weakening controller: Gain
	0x29E0:2	0x31E0:2	Field weakening controller: Reset time
	0x29E1	0x31E1	Field set value limitation
	0x29C0:1	0x31C0:1	Field controller: Gain
	0x29C0:2	0x31C0:2	Field controller: Reset time
	0x2939	0x3139	Switching frequency

Tip!

A detailed representation of the signal flow with all relevant parameters can be found in the »PLC Designer« on the **Signal flow** tab for the i700 servo inverters.

7.8.4 Control commands & status information

The following control commands can be executed in cyclic sync position mode via the Controlword (0x6040 or 0x6840 for axis B):

Control word	Status	Function
Bit 4	0	*Reserved* (bit must be set to "0")
Bit 5	0	*Reserved* (bit must be set to "0")
Bit 6	0	*Reserved* (bit must be set to "0")
Bit 8	0 ↗ 1	Halt,

The following status information is output in cyclic sync position mode via the Statusword (0x6041 or 0x6841 for axis B):

Status word	Status	Meaning
Bit 12	0	Cyclic sync position mode inactive
	1	Cyclic sync position mode active

7.9 Cyclic sync velocity mode (csv)

This operating mode provides a quick speed follower with torque/feed force feedforward control. The motion profile to be processed is defined by the controller.

Subfunctions of the operating mode

- Calculation of the speed setpoint
- Calculation of the speed feedforward control value
- Interpolation between communication cycle and control cycle
- Velocity control
- Closed-loop torque control
- Limitation of the motor speed
- Update of the actual values for position, speed, and torque

Selection of the operating mode

Cyclic sync velocity mode is selected with the setting "9" in 0x6060 (or 0x6860 for axis B).

7.9.1 Default mapping

The default mapping for the cyclic sync velocity mode is defined in the following objects:

Object		Name	Data type
Axis A	Axis B		
0x1602	0x1612	RPDO-->Axis A/B: Cyclic sync velocity mode (csv)	RECORD
0x1606	0x1616	RPDO-->Axis A/B: Torque limits	RECORD
0x1A02	0x1A12	Axis A/B-->TPDO: Cyclic sync velocity mode (csv)	RECORD

Date received from the controller (RPDO)

Object		Name	Data type
Axis A	Axis B		
0x6040	0x6840	Controlword	UNSIGNED_16
0x2830	0x3030	Lenze control word	UNSIGNED_16
0x6060	0x6860	Operating mode: Selection	INTEGER_8
0x60B2	0x68B2	Torque offset	INTEGER_16
0x60FF	0x68FF	Target velocity	INTEGER_32
0x60E0	0x68E0	Positive torque limit value	UNSIGNED_16
0x60E1	0x68E1	Negative torque limit value	UNSIGNED_16

Data transmitted to the controller (TPDO)

Object		Name	Data type
Axis A	Axis B		
0x6041	0x6841	Statusword	UNSIGNED_16
0x2831	0x3031	Lenze status word	UNSIGNED_16
0x6061	0x6861	Modes of operation display	INTEGER_8
0x603F	0x683F	Error code	UNSIGNED_16
0x606C	0x686C	Velocity actual value	UNSIGNED_16
0x6077	0x6877	Torque actual value	INTEGER_16
0x6064	0x6864	Position actual value	INTEGER_32

7.9.2 Object description

The following two tables provide an overview of the most important objects for this operating mode (without motor parameters, motor control parameters, and feedback parameters).

All objects correspond to the CiA402 specification, but some of them have only a restricted value range.

Objects described in other chapters:

Object Axis A	Axis B	Name	Data type
▸ Lenze control and status word			
0x2830	0x3030	Lenze control word	UNSIGNED_16
0x2831	0x3031	Lenze status word	UNSIGNED_16
▸ Drive control			
0x6040	0x6840	Controlword	UNSIGNED_16
0x6060	0x6860	Operating mode: Selection	INTEGER_8
0x6041	0x6841	Statusword	UNSIGNED_16
0x6061	0x6861	Modes of operation display	INTEGER_8
▸ Parameters for actuation of the position control			
0x6064	0x6864	Position actual value	INTEGER_32
▸ Cyclic sync position mode (csp)			
0x60C2	0x68C2	Interpolation time period	RECORD
0x60E0	0x68E0	Positive torque limit value	UNSIGNED_16
0x60E1	0x68E1	Negative torque limit value	UNSIGNED_16
▸ Cyclic sync torque mode (cst)			
0x60B2	0x68B2	Torque offset	INTEGER_16
0x6071	0x6871	Target torque	INTEGER_16
0x6077	0x6877	Torque actual value	INTEGER_16
Greyed out = read access only			

Objects described in this chapter:

Object Axis A	Axis B	Name	Data type
0x606C	0x686C	Velocity actual value	INTEGER_32
0x60B1	0x68B1	Velocity offset	INTEGER_32
0x60FF	0x68FF	Target velocity	INTEGER_32
Greyed out = read access only			

0x606C | 0x686C - Velocity actual value

Current velocity

Display range (min. value | unit | max. value)			Initialisation	
-480000	[n unit]	480000		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: $480000/2^{31}$	INTEGER_32

0x60B1 | 0x68B1 - Velocity offset

Additive value for target velocity or velocity feedforward control

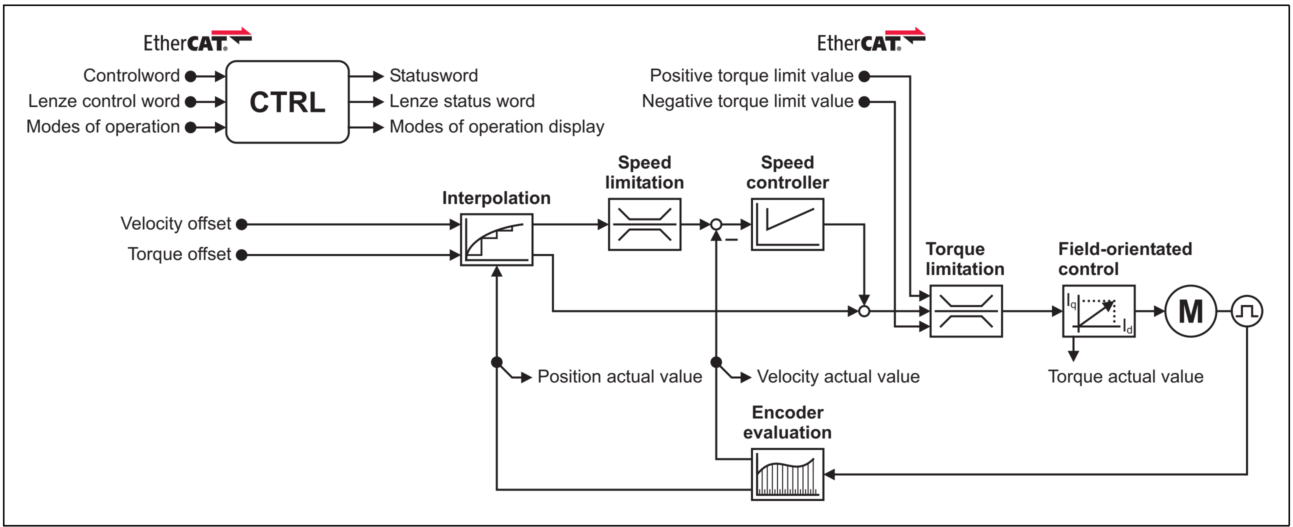
Setting range (min. value | unit | max. value)			Lenze setting	
-480000	[n unit]	480000	0 [n-unit]	
☑ Write access ☐ CINH ☑ OSC ☑ P ☑ RX ☐ TX			Scaling: $480000/2^{31}$	INTEGER_32

0x60FF | 0x68FF - Target velocity

Target velocity or velocity feedforward control

Setting range (min. value | unit | max. value)			Lenze setting	
-480000	[n unit]	480000	0 [n-unit]	
☑ Write access ☐ CINH ☑ OSC ☐ P ☑ RX ☐ TX			Scaling: $480000/2^{31}$	INTEGER_32

7.9.3 Signal flow (servo control)



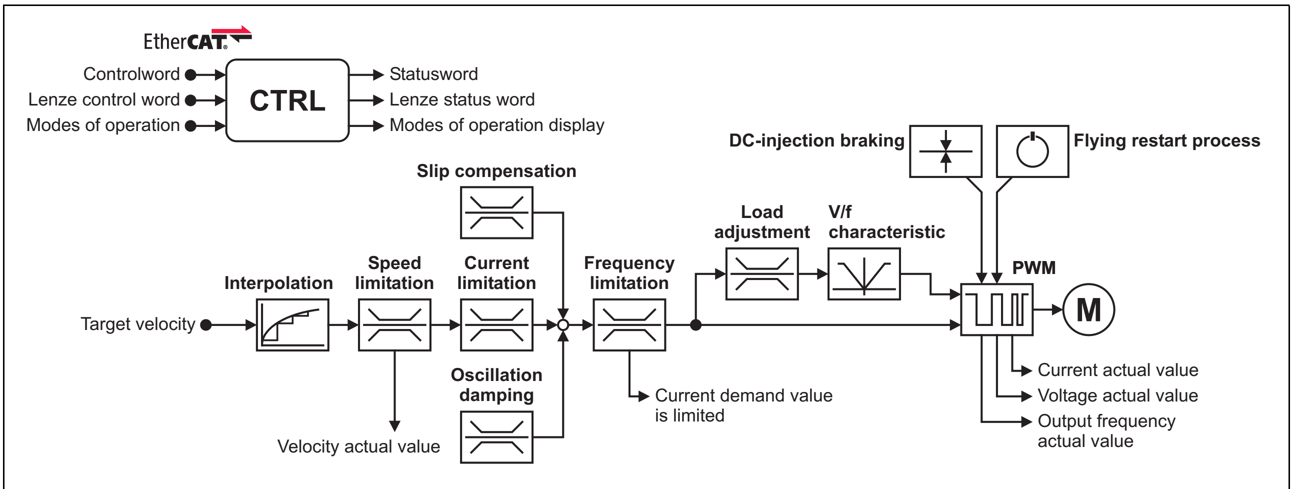
[7-6] Signal flow of the servo control in cyclic sync velocity mode (simplified representation)

Short overview of the most important parameters:

Function	Object		Name
	Axis A	Axis B	
Input data	0x6040	0x6840	Controlword
	0x2830	0x3030	Lenze control word
	0x6060	0x6860	Operating mode: Selection
	0x60B1	0x68B1	Velocity offset
	0x60B2	0x68B2	Torque offset
	0x60E0	0x68E0	Positive torque limit value
	0x60E1	0x68E1	Negative torque limit value
Output data	0x6041	0x6841	Statusword
	0x2831	0x3031	Lenze status word
	0x6061	0x6861	Modes of operation display
	0x6064	0x6864	Position actual value
	0x606C	0x686C	Velocity actual value
	0x6077	0x6877	Torque actual value
Interpolation	0x60C2	0x68C2	Interpolation time period
Speed limitation	0x6080	0x6880	Max motor speed
	0x2903	0x3103	Speed: Speed setpoint - filter time
Speed controller	0x2900:1	0x3100:1	Speed controller: Gain
	0x2900:2	0x3100:2	Speed controller: Reset time
	0x2900:3	0x3100:3	Speed controller: Rate time
	0x2901	0x3101	Speed controller: Gain - adaptation
	0x2902:0	0x3102:0	Speed controller: Load I component
Torque limitation	0x60E0	0x68E0	Positive torque limit value
	0x60E1	0x68E1	Negative torque limit value
	0x6076	0x6876	Motor rated torque
	0x6072	0x6872	Max torque
	0x2944:1	0x3144:1	Notch filter 1: Frequency
	0x2944:2	0x3144:2	Notch filter 1: Bandwidth
	0x2944:3	0x3144:3	Notch filter 1: Damping
	0x2944:4	0x3144:4	Notch filter 2: Frequency
	0x2944:5	0x3144:5	Notch filter 2: Bandwidth
	0x2944:6	0x3144:6	Notch filter 2: Damping

Function	Object		Name
	Axis A	Axis B	
Field-oriented control	0x6073	0x6873	Max current
	0x6075	0x6875	Motor rated current
	0x2941	0x3141	Current controller: Feedforward control
	0x2942:1	0x3142:1	Current controller: Gain
	0x2942:2	0x3142:2	Current controller: Reset time
	0x29E2	0x31E2	DC link circuit voltage: Filter time
	0x29E3	0x31E3	Motor voltage act. value: Filter time
	0x29E0:1	0x31E0:1	Field weakening controller: Gain
	0x29E0:2	0x31E0:2	Field weakening controller: Reset time
	0x29E1	0x31E1	Field set value limitation
	0x29C0:1	0x31C0:1	Field controller: Gain
	0x29C0:2	0x31C0:2	Field controller: Reset time
	0x2939	0x3139	Switching frequency

7.9.4 Signal flow (V/f characteristic control)



[7-7] Signal flow of the V/f characteristic control in cyclic sync velocity mode (simplified representation)

Short overview of the most important parameters:

Function	Object		Name
	Axis A	Axis B	
Input data	0x6040	0x6840	Controlword
	0x2830	0x3030	Lenze control word
	0x6060	0x6860	Operating mode: Selection
	0x60FF	0x68FF	Target velocity

Function	Object		Name
	Axis A	Axis B	
Output data	0x6041	0x6841	Statusword
	0x2831	0x3031	Lenze status word
	0x6061	0x6861	Modes of operation display
	0x606C	0x686C	Velocity actual value
	0x6078	0x6878	Current actual value
	0x2D82	0x3582	Motor: Actual voltage - Veff, phase-phase
	0x2DDD	0x35DD	Device: Actual output frequency
Interpolation	0x60C2	0x68C2	Interpolation time period
Speed limitation	0x6080	0x6880	Max motor speed
	0x2903	0x3103	Speed: Speed setpoint - filter time
Slip compensation	0x2B09:1	0x3309:1	VFC: Slip compensation - influence
	0x2B09:2	0x3309:2	VFC: Slip compensation - filter time
Oscillation damping	0x2B0A:1	0x330A:1	VFC: Oscillation damping - gain
	0x2B0A:2	0x330A:2	VFC: Oscillation damping - filter time
	0x2B0A:3	0x330A:3	VFC: Oscillation damping - limitation
	0x2B0A:4	0x330A:4	VFC: Oscillation damping - ramp-end frequency
Load adjustment	0x2B07:1	0x2B07:1	VFC: Load adjustment - direction of rotation
	0x2B07:2	0x2B07:2	VFC: Load adjustment - value
V/f characteristic	0x2B01:1	0x3301:1	VFC: V/f characteristic - voltage at reference point
	0x2B01:2	0x3301:2	VFC: V/f characteristic - frequency at reference point
	0x2B06	0x3306	VFC: Voltage boost
	0x2B04	0x3304	VFC: Voltage vector control - current setpoint
	0x2B00	0x3300	VFC: V/f characteristic - shape
	0x2B02:x	0x3302:x	VFC: User defined V/f characteristic • Frequency grid points (x1 ... x11)
	0x2B03:x	0x3303:x	VFC: User defined V/f characteristic • Voltage grid points (y1 ... y11)
DC-injection braking	0x2B80	0x3380	DC-injection braking: Current
Flying restart function	0x2BA0	0x33A0	Flying restart: Activate
	0x2BA1	0x33A1	Flying restart: Current
	0x2BA2	0x33A2	Flying restart: Start frequency
	0x2BA3	0x33A3	Flying restart: Integration time
	0x2BA4	0x33A4	Flying restart: Min. deviation
	0x2BA5	0x33A5	Flying restart: Delay time
	0x2BA6	0x33A6	Flying restart: Result

Tip!

A detailed representation of the signal flow with all relevant parameters can be found in the »PLC Designer« on the **Signal flow** tab for the i700 servo inverters.

7.9.5 Control commands & status information

The following control commands can be executed in "Cyclic synchronous velocity mode" via the Controlword (0x6040 or 0x6840 for axis B):

Control word	Status	Function
Bit 4	0	*Reserved* (bit must be set to "0")
Bit 5	0	*Reserved* (bit must be set to "0")
Bit 6	0	*Reserved* (bit must be set to "0")
Bit 8	0 ↗ 1	Halt,

The following status information is output in "Cyclic synchronous velocity mode" via the Statusword (0x6041 or 0x6841 for axis B):

Status word	Status	Meaning
Bit 12	0	Cyclic sync velocity mode inactive
	1	Cyclic sync velocity mode active

7.10 Cyclic sync torque mode (cst)

This operating mode provides a quick torque follower with speed limitation. The torque profile to be processed is defined by the controller.

Note!

- This operating mode can only be actuated reasonably if the servo control is set as motor control. ▶ Motor control & motor settings
- Since this is a cyclic mode, setpoints in the bus cycle are expected.

If communication is interrupted and thus telegrams fail to appear, the internal "quick stop" function takes over the braking of the motor. Afterwards, the controller is inhibited.

Up to and including software version V01.06.3x

Triggering the quick stop causes the drive to be stopped along the current limit of 0x6073 / 0x6873. The torque limits from 0x60E0 / 0x68E0, 0x60E1 / 0x68E1 and 0x6072 / 0x6872 are not effective during the quick stop!

From software version V01.07.00 onwards

During the quick stop, both the current limit 0x6073 and the torque limit 0x6072 are active. The smaller of the two limits determine the output motor torque. The torque limits from 0x60E0 and 0x60E1 have no effect during the quick stop.

Subfunctions of the operating mode

- Torque control with speed limitation
- Calculation of the torque setpoint
- Interpolation of the setpoints from the communication cycle to the control cycle
- Limitation of the motor speed
- Update of the actual values for position, speed, and torque

Selection of the operating mode

"Cyclic synchronous torque mode" is selected with the setting "10" in 0x6060 (or 0x6860 for axis B).

7.10.1 Default mapping

The default mapping for "Cyclic synchronous torque mode" is defined in the following objects:

Object		Name	Data type
Axis A	Axis B		
0x1601	0x1611	RPDO-->Axis A/B: Cyclic sync torque mode (cst)	RECORD
0x1A01	0x1A11	Axis A/B-->TPDO: Cyclic sync torque mode (cst)	RECORD

Date received from the controller (RPDO)

Object		Name	Data type
Axis A	Axis B		
0x6040	0x6840	Controlword	UNSIGNED_16
0x2830	0x3030	Lenze control word	UNSIGNED_16
0x6060	0x6860	Operating mode: Selection	INTEGER_8
0x60B2	0x68B2	Torque offset	INTEGER_16
0x6071	0x6871	Target torque	INTEGER_16
0x2946:1	0x3146:1	Speed limitation: Upper speed limit	INTEGER_32
0x2946:2	0x3146:2	Speed limitation: Lower speed limit	INTEGER_32

Data transmitted to the controller (TPDO)

Object		Name	Data type
Axis A	Axis B		
0x6041	0x6841	Statusword	UNSIGNED_16
0x2831	0x3031	Lenze status word	UNSIGNED_16
0x6061	0x6861	Modes of operation display	INTEGER_8
0x603F	0x683F	Error code	UNSIGNED_16
0x606C	0x686C	Velocity actual value	UNSIGNED_16
0x6077	0x6877	Torque actual value	INTEGER_16

7.10.2 Object description

The following two tables provide an overview of the most important objects for this operating mode (without motor parameters, motor control parameters, and feedback parameters).

All objects correspond to the CiA402 specification, but some of them have only a restricted value range.

Objects described in other chapters:

Object Axis A	 Axis B	Name	Data type
Lenze control and status word			
0x2830	0x3030	Lenze control word	UNSIGNED_16
0x2831	0x3031	Lenze status word	UNSIGNED_16
Drive control			
0x6040	0x6840	Controlword	UNSIGNED_16
0x6060	0x6860	Operating mode: Selection	INTEGER_8
0x6041	0x6841	Statusword	UNSIGNED_16
0x6061	0x6861	Modes of operation display	INTEGER_8
Parameters for actuation of the position control			
0x6064	0x6864	Position actual value	INTEGER_32
Cyclic sync position mode (csp)			
0x60C2	0x68C2	Interpolation time period	RECORD
Cyclic sync velocity mode (csv)			
0x606C	0x686C	Velocity actual value	INTEGER_32
Set motor parameters manually			
0x6075	0x6875	Motor rated current	UNSIGNED_32
0x6076	0x6876	Motor rated torque	UNSIGNED_32
Greyed out = read access only			

Objects described in this chapter:

Object Axis A	 Axis B	Name	Data type
0x2946	0x3146	Cyclic sync torque mode: Speed limitation	RECORD
0x6071	0x6871	Target torque	INTEGER_16
0x6072	0x6872	Max torque	UNSIGNED_16
0x6073	0x6873	Max current	UNSIGNED_16
0x6074	0x6874	Target torque	INTEGER_16
0x6077	0x6877	Torque actual value	INTEGER_16
0x6078	0x6878	Current actual value	INTEGER_16
0x6079	0x6879	DC bus: Actual voltage	UNSIGNED_32
0x60B2	0x68B2	Torque offset	INTEGER_16
Greyed out = read access only			

0x2946 | 0x3146 - Cyclic sync torque mode: Speed limitation

The torque control contains a speed limitation as a protective function against very high speeds. High speeds can occur in the case of a mere torque selection if no counter-torque is available (load-free machine).
The speed limitation function takes effect if the actual motor speed reaches the limit values set here: The motor speed is kept at the limit value in each case. If the machine is decelerated by a counter-torque again, the speed limitation hands control back to the external setpoint via the setpoint torque.
Note!
The upper limit value must be set to a greater value than the lower limit value. There is no plausibility check in the i700 servo inverter.

Sub.	Name	Lenze setting	Data type
▸ 1	Speed limitation: Upper speed limit	0 [n-unit]	INTEGER_32
▸ 2	Speed limitation: Lower speed limit	0 [n-unit]	INTEGER_32

Subindex 1: **Speed limitation: Upper speed limit**				
Upper limit for the speed limitation				
Setting range (min. value | unit | max. value)			Lenze setting	
-480000	[n unit]	480000	0 [n-unit]	
☑ Write access ☐ CINH ☑ OSC ☑ P ☑ RX ☐ TX			Scaling: $480000/2^{31}$	INTEGER_32

Subindex 2: **Speed limitation: Lower speed limit**				
Lower limit for the speed limitation				
Setting range (min. value | unit | max. value)			Lenze setting	
-480000	[n unit]	480000	0 [n-unit]	
☑ Write access ☐ CINH ☑ OSC ☑ P ☑ RX ☐ TX			Scaling: $480000/2^{31}$	INTEGER_32

0x6071 | 0x6871 - Torque demand

Target torque or torque feedforward control
- 100 % ≡ rated motor torque (0x6076 or 0x6876 for axis B)

Setting range (min. value | unit | max. value)			Lenze setting	
-3276.8	%	3276.7	0.0 %	
☑ Write access ☐ CINH ☑ OSC ☐ P ☑ RX ☐ TX			Scaling: 1/10	INTEGER_16

0x6072 | 0x6872 - Max torque

Symmetrical selection of the maximum permissible torque
- 100 % ≡ rated motor torque (0x6076 or 0x6876 for axis B)
- This parameter serves to implement a statically and bipolarly acting torque limitation. This can be used, for instance, as overload protection of the mechanical transmission path/elements starting at the motor shaft.
- This limitation acts independently of the unipolarly acting torque limitations which act as process data on the objects 0x60E0 (or 0x60E0 for axis B) and 0x60E1 (or 0x60E1 for axis B). For details see signal flow.

Setting range (min. value | unit | max. value)			Lenze setting	
0.0	%	3276.7	250.0 %	
☑ Write access ☐ CINH ☑ OSC ☑ P ☑ RX ☐ TX			Scaling: 1/10	UNSIGNED_16

0x6073 | 0x6873 - Max current

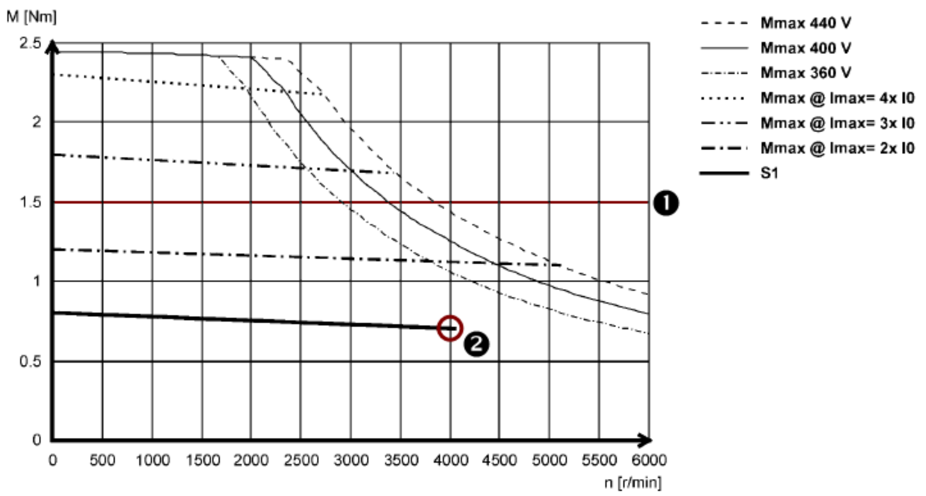
Maximum current (maximum overload current of the device)
- 100 % ≡ rated motor current (0x6075 or 0x6875 for axis B)
- This parameter serves to limit the inverter output current.
- The value to be set here results from the maximum torque required for the process and the connected motor. The value is calculated according to the following rule of thumb:
  - For synchronous motors (SM):
    Axis A: 0x6073 [%] = $M_{max\_process}$ [Nm] / 0x6076 [Nm] * 100
    Axis B: 0x6873 [%] = $M_{max\_process}$ [Nm] / 0x6876 [Nm] * 100
  - For asynchronous motors (ASM):
    Axis A: 0x6073 [%] = $M_{max\_process}$ [Nm] / 0x6076 [Nm] * 115
    Axis B: 0x6873 [%] = $M_{max\_process}$ [Nm] / 0x6876 [Nm] * 115
  - For both motor types:
    Axis A: 0x6072 [%] = $M_{max\_process}$ [Nm] / 0x6076 [Nm] * 100
    Axis B: 0x6872 [%] = $M_{max\_process}$ [Nm] / 0x6876 [Nm] * 100
- Depending on the motor frame size, the Lenze setting of 0x6073 (or 0x6873 for axis B) and 0x6072 (or 0x6872 for axis B) serves to achieve a rated current of 150 % in case of synchronous motors and a rated current of 130 % in case of asynchronous motors. If the process (application) needs torques above these limits, these two parameters have to be set accordingly based on the motor data.
- If a value is set here which, multiplied by the rated motor current (0x6075 or 0x6875 for axis B) leads to a higher current value than the maximum device current, the value is limited device-internally to a maximum device current / rated motor current of 100 % * and output as warning in the logbook. The value set here, however, is not changed.

Note!
This parameter is not identical to the ultimate motor current I_{ULT} (0x2D46:1 or 0x3546:1 for axis B!
- The ultimate motor current is a limit value for synchronous motors in order to protect their magnets.
- The value to be set here should always be considerably below the ultimate motor current!

Setting range (min. value | unit | max. value)			Lenze setting	
0.0	%	3276.7	150.0 %	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	UNSIGNED_16

Example calculation based on the torque characteristics for MCS06C41 Lenze motor

The data apply to a mains voltage 3 x 400 V



❶	$M_{max\_process}$	0x6073 = $M_{max\_process}$ / M_{rated_motor} * 100 % = 1.5 Nm / 0.7 Nm * 100 % = 215 %
❷	M_{rated_motor}	

Tip!

More torque characteristics can be found on the Internet:

http://www.lenze.com/dsc

0x6074 | 0x6874 - torque: Interpol. setpoint torque

Interpolated target torque
- 100 % ≡ rated motor torque (0x6076 or 0x6876 for axis B)

Display range (min. value | unit | max. value)			Initialisation	
-3276.8	%	3276.7	0.0 %	
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☐ TX			Scaling: 1/10	INTEGER_16

0x6077 | 0x6877 - Torque actual value

Current torque
- 100 % ≡ rated motor torque (0x6076 or 0x6876 for axis B)

Display range (min. value | unit | max. value)			Initialisation	
-3276.8	%	3276.7		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: 1/10	INTEGER_16

0x6078 | 0x6878 - Current actual value

Actual motor current
- 100 % ≡ rated motor current (0x6075 or 0x6875 for axis B)

Display range (min. value | unit | max. value)			Initialisation	
-3276.8	%	3276.7		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: 1/10	INTEGER_16

0x6079 | 0x6879 - DC link circuit voltage

Actual DC-bus voltage

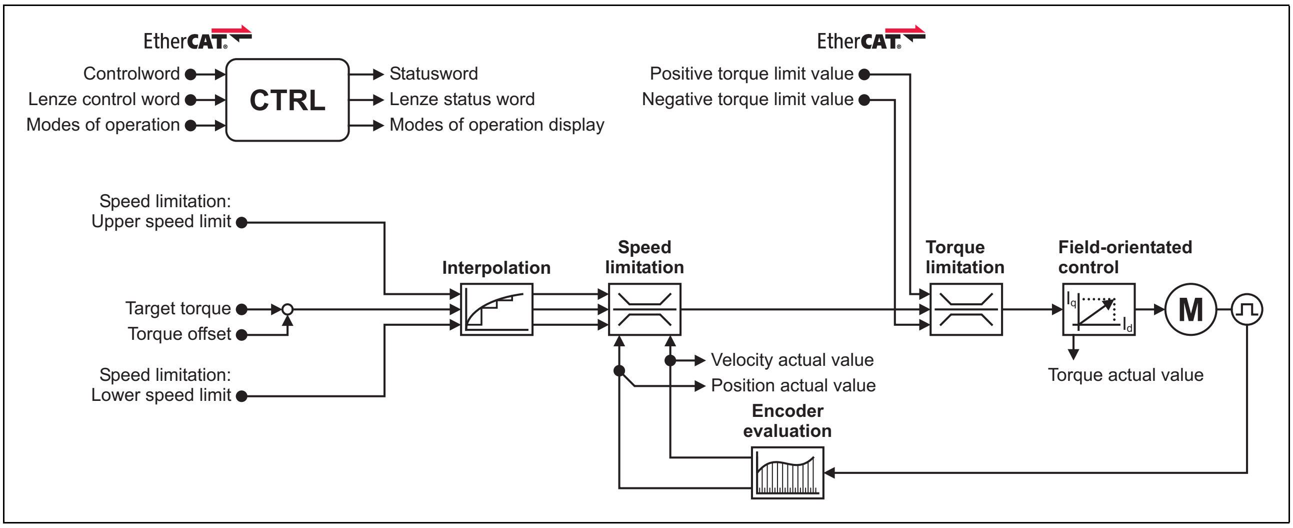
Display range (min. value | unit | max. value)			Initialisation	
0.000	V	4294967.295		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: 1/1000	UNSIGNED_32

0x60B2 | 0x68B2 - Torque offset

Additive value for the target torque or torque feedforward control
- 100 % ≡ rated motor torque (0x6076 or 0x6876 for axis B)

Setting range (min. value | unit | max. value)			Lenze setting	
-3276.8	%	3276.7	0.0 %	
☑ Write access ☐ CINH ☑ OSC ☑ P ☑ RX ☐ TX			Scaling: 1/10	INTEGER_16

7.10.3 Signal flow



[7-8] Signal flow of the servo control in "Cyclic sync torque mode" (simplified representation)

Short overview of the most important parameters:

Function	Object		Name
	Axis A	Axis B	
Input data	0x6040	0x6840	Controlword
	0x2830	0x3030	Lenze control word
	0x6060	0x6860	Operating mode: Selection
	0x2946:1	0x3146:1	Speed limitation: Upper speed limit
	0x60B2	0x68B2	Torque offset
	0x6071	0x6871	Target torque
	0x2946:2	0x3146:2	Speed limitation: Lower speed limit
	0x60E0	0x68E0	Positive torque limit value
	0x60E1	0x68E1	Negative torque limit value
Output data	0x6041	0x6841	Statusword
	0x2831	0x3031	Lenze status word
	0x6061	0x6861	Modes of operation display
	0x6064	0x6864	Position actual value
	0x606C	0x686C	Velocity actual value
	0x6077	0x6877	Torque actual value
Interpolation	0x60C2	0x68C2	Interpolation time period
Speed limitation	0x6080	0x6880	Max motor speed
	0x2903	0x3103	Speed: Speed setpoint - filter time

Function	Object		Name
	Axis A	Axis B	
Torque limitation	0x60E0	0x68E0	Positive torque limit value
	0x60E1	0x68E1	Negative torque limit value
	0x6076	0x6876	Motor rated torque
	0x6072	0x6872	Max torque
	0x2944:1	0x3144:1	Notch filter 1: Frequency
	0x2944:2	0x3144:2	Notch filter 1: Bandwidth
	0x2944:3	0x3144:3	Notch filter 1: Damping
	0x2944:4	0x3144:4	Notch filter 2: Frequency
	0x2944:5	0x3144:5	Notch filter 2: Bandwidth
	0x2944:6	0x3144:6	Notch filter 2: Damping
Field-oriented control	0x6073	0x6873	Max current
	0x6075	0x6875	Motor rated current
	0x2941	0x3141	Current controller: Feedforward control
	0x2942:1	0x3142:1	Current controller: Gain
	0x2942:2	0x3142:2	Current controller: Reset time
	0x29E2	0x31E2	DC link circuit voltage: Filter time
	0x29E3	0x31E3	Motor voltage act. value: Filter time
	0x29E0:1	0x31E0:1	Field weakening controller: Gain
	0x29E0:2	0x31E0:2	Field weakening controller: Reset time
	0x29E1	0x31E1	Field set value limitation
	0x29C0:1	0x31C0:1	Field controller: Gain
	0x29C0:2	0x31C0:2	Field controller: Reset time
	0x2939	0x3139	Switching frequency

Tip!

A detailed representation of the signal flow with all relevant parameters can be found in the »PLC Designer« on the **Signal flow** tab for the i700 servo inverters.

7.10.4 Control commands & status information

The following control commands can be executed in "Cyclic synchronous torque mode" via the Controlword (0x6040 or 0x6840 for axis B):

Control word	Status	Function
Bit 4	0	*Reserved* (bit must be set to "0")
Bit 5	0	*Reserved* (bit must be set to "0")
Bit 6	0	*Reserved* (bit must be set to "0")
Bit 8	0 ↗ 1	Halt,

The following status information is output in "Cyclic synchronous torque mode" via the Statusword (0x6041 or 0x6841 for axis B):

Status word	Status	Meaning
Bit 12	0	Cyclic sync torque mode inactive
	1	Cyclic sync torque mode active

7.11 Touch probe (TP)

A "Touch probe" (short: "TP") is an event which can for instance be actuated in an edge-controlled manner via a digital input to detect an actual value (that changes quickly) at the time of activation and to process it further within the program afterwards.

- Typical touch probe applications:
  - Homing
  - Mark synchronisation
  - Measurements of lengths
- Up to 2 touch probe channels can be used in parallel for each axis.
- Possible touch probe sources:
  - TP1 : Zero pulse position encoder or digital input DI1
  - TP2 : Zero pulse position encoder or digital input DI2

Note!

The digital inputs DI1 and DI2 can be evaluated additionally as "normal" digital inputs via the 0x60FD object any time (or 0x68FD for axis B).

Objects described in other chapters:

Object Axis A	 Axis B	Name	Data type
▸ General CiA402 parameters			
0x60FD	0x68FD	Digital inputs	UNSIGNED_32
Greyed out = read access only			

Objects described in this chapter:

Object Axis A	 Axis B	Name	Data type
Lenze-specific objects			
0x2500		Touch probe (TP): Filter time	UNSIGNED_16
0x2D00	0x3500	Touch probe (TP): Delay time	RECORD
0x2D01	0x3501	Touch probe (TP): Time stamp	RECORD
Objects according to CiA402 specification (version 3)			
0x60B8	0x68B8	Touch probe function	UNSIGNED_16
0x60B9	0x68B9	Touch probe status	UNSIGNED_16
0x60BA	0x68BA	Touch probe pos1 pos value	INTEGER_32
0x60BB	0x68BB	Touch probe pos1 neg value	INTEGER_32
0x60BC	0x68BC	Touch probe pos2 pos value	INTEGER_32
0x60BD	0x68BD	Touch probe pos2 neg value	INTEGER_32
Greyed out = read access only			

7.11.1 Default mapping

The default mapping for a touch probe detection is specified in the following objects:

Object Axis A	 Axis B	Name	Data type
0x1604	0x1614	RPDO-->Axis A/B: Touch probe (TP)	RECORD
0x1A04	0x1A14	Axis A/B-->TPDO: Touch probe (TP)	RECORD

Date received from the controller (RPDO)

Object Axis A	 Axis B	Name	Data type
0x60B8	0x68B8	Touch probe function	UNSIGNED_16

Data transmitted to the controller (TPDO)

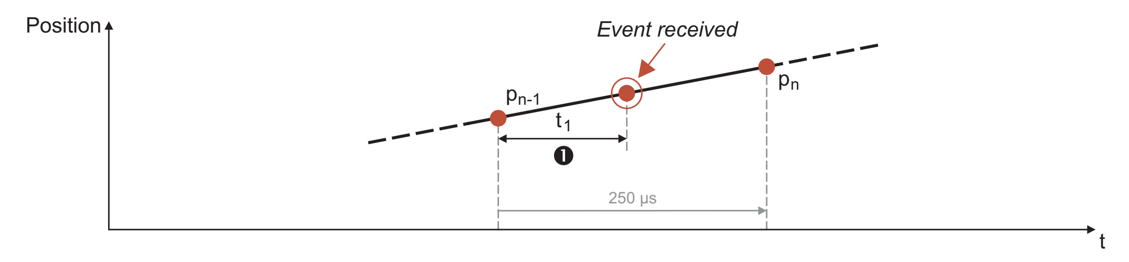
Object Axis A	 Axis B	Name	Data type
0x60B9	0x68B9	Touch probe status	UNSIGNED_16
0x60BA	0x68BA	Touch probe pos1 pos value	INTEGER_32
0x60BB	0x68BB	Touch probe pos1 neg value	INTEGER_32
0x60BC	0x68BC	Touch probe pos2 pos value	INTEGER_32
0x60BD	0x68BD	Touch probe pos2 neg value	INTEGER_32

7.11.2 General functional principle

If an event occurs at the touch probe source configured, a time stamp is recorded in the i700 servo inverters.

The time stamp recorded is related to the system time and can thus be separated into two parts: One part is the control cycle within which the event has occurred. The other part is the time difference between the start of the control cycle determined and the actual detection of the event.

A history buffer enables the i700 servo inverters to know the last n position values. Thus, the actual position at the start and end of the control cycle within which the event has occurred is known. Linear interpolation is executed between these two position grid points. The result is the exact position at the motor shaft at the time the event is triggered:



P_{n-1} Actual position grid point 1

P_n Actual position grid point 2

❶ Time difference between the start of the control cycle determined and the actual detection of the event

[7-9] Determination of the exact position by linear interpolation (principle)

The position grid points are detected in the i700 servo inverters in a grid of 250 µs. After a touch probe is tripped, the input is deactivated for up to 250 µs in order to avoid bouncing. Thus the maximum frequency for the tripping of touch probes is 4 kHz.

If, in contrast to the steady motion outlined in figure [7-9], an accelerated motion is taken as a basis, the 250 µs grid also allows for a very good linear position reconstruction, since the velocity change at the motor shaft can only have a minor effect during 250 µs.

7.11.3 Filtering the touch probe signal

A common filter time (debounce time) can be parameterised for the touch probe inputs in order to debounce the TP signals so that there is no response to external interference signals.

- Every 31 µs, the signal status at the TP input is detected for the debounce filter and a new value is assigned to the filter.
- Via the following object, the filter time for all touch probe inputs of both axes of the device is set. A separate setting for a touch probe or an axis is not possible.

0x2500 - Touch probe (TP): Debounce time

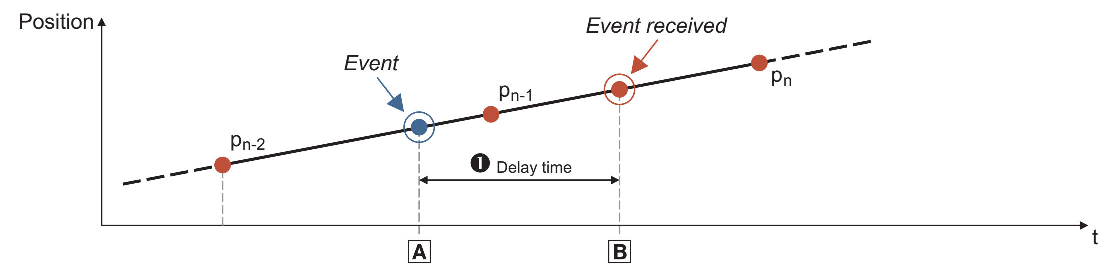
Note: Since the filter is scanned with 32 kHz, there are discrete, adjustable values. After entering an optional filter time between 0 and 1984 µs, the value is automatically internally rounded down to the next adjustable value and is also displayed on read request.
- The filter time is automatically taken into consideration in the TP calculation.
- If the setting is "0", the filter is deactivated.

Setting range (min. value | unit | max. value)			Lenze setting	
0	us	1984	0 us	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

7.11.4 Compensation of runtime delays

In reality, both the input connection in the i700 servo inverters and the touch probe sensor have runtime delays (latencies) themselves. They can be taken into consideration for calculating the actual tripping time and thus the actual position at the time of tripping.

In the following illustration, the event is detected at the time B in the i700 servo inverters. Due to the input connection and the sensor used, however, the signal has undergone a runtime delay; the actual physical event has already occurred at the time A. In order to compensate this runtime delay, a corresponding delay time which is taken into consideration for the determination of the control cycle and interpolation of the position (see figure [7-9]) can be set for each touch probe channel.



A Actual physical event

B Electrical recognition of the event in the i700 servo inverters

❶ Delay time between the actual physical event and the electrical recognition

[7-10] Compensation of runtime delays (principle)

Delay times of the digital inputs and required minimum signal duration

The table below lists the delay times typical for the digital inputs of the i700 servo inverters and the required minimum signal duration:

Digital signal	Typical delay time	Minimum signal duration
Rising edge (HIGH pulse)	5 - 7 µs	7 µs
Falling edge (LOW pulse)	approx. 40 µs	50 µs

0x2D00 | 0x3500 - Touch probe (TP): Dead time compensation

Delay time for the compensation of runtime delays (latencies)
- Set the sum of individual delays for each touch probe channel here (sensor, cable, and input connection).

Sub.	Name	Lenze setting	Data type
1	Dead time compensation: TP1 dead time	0.000 ms	UNSIGNED_16
2	Dead time compensation: TP2 dead time	0.000 ms	UNSIGNED_16

Subindex 1: **Dead time compensation: TP1 delay time**				
Setting range (min. value | unit | max. value)			Lenze setting	
0.000	ms	7.000	0.000 ms	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/1000	UNSIGNED_16

Subindex 2: **Dead time compensation: TP2 delay time**				
Setting range (min. value | unit | max. value)			Lenze setting	
0.000	ms	7.000	0.000 ms	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/1000	UNSIGNED_16

7.11.5 Touch probe function

0x60B8 | 0x68B8 - Touch probe function

Control word for the configuration of the touch probe functionality

Setting range (min. value | unit | max. value)			Lenze setting
0		65535	0
Value is bit-coded: (☑ = bit set)			Info
Bit 0 ☐	Enable touch probe 1		0: Switch off touch probe 1 1: Enable touch probe 1
Bit 1 ☐	Trigger on first event (0) or continous (1)		Event for touch probe channel 1: 0: Record first event only 1: Record all events
Bit 2 ☐	TP: Source - TP input (0) or position encoder zero pulse (1)		Source for touch probe channel 1: 0: Digital input 1 1: Zero pulse position encoder
Bit 3 ☐	Reserved		
Bit 4 ☐	TP1: Activate sampling - rising edge		0: Deactivate sampling 1: Activate sampling
Bit 5 ☐	TP1: Activate sampling - falling edge		0: Deactivate sampling 1: Activate sampling
Bit 6 ☐	Reserved		
Bit 7 ☐	Reserved		
Bit 8 ☐	Enable touch probe 2		0: Switch off touch probe 2 1: Enable touch probe 2
Bit 9 ☐	Trigger on first event (0) or continous (1)		Event for touch probe channel 2: 0: Record first event only 1: Record all events
Bit 10 ☐	TP: Source - TP input (0) or position encoder zero pulse (1)		Source for touch probe channel 2: 0: Digital input 2 1: Zero pulse position encoder
Bit 11 ☐	Reserved		
Bit 12 ☐	TP2: Activate sampling - rising edge		0: Deactivate sampling 1: Activate sampling
Bit 13 ☐	TP2: Activate sampling - falling edge		0: Deactivate sampling 1: Activate sampling
Bit 14 ☐	Reserved		
Bit 15 ☐	Reserved		
☑ Write access ☐ CINH ☑ OSC ☑ P ☑ RX ☐ TX			UNSIGNED_16

7.11.6 Touch probe status

0x60B9 | 0x68B9 - Touch probe status

Status of the touch probe functionality

Display range (min. value | unit | max. value)			Initialisation
0		65535	0
Value is bit-coded:			Info
Bit 0	TP1: Active		0: Touch probe channel 1 deactivated 1: Touch probe channel 1 activated
Bit 1	TP1: Position detected - rising edge		0: Position not detected 1: Position detected
Bit 2	TP1: Position detected - falling edge		0: Position not detected 1: Position detected
Bit 3	Reserved		
Bit 4	Reserved		
Bit 5	Reserved		
Bit 6	Touch probe (TP): Level at release time		Level at detection via touch probe channel 1: 0: LOW level 1: HIGH level
Bit 7	Reserved		
Bit 8	TP2: Active		0: Touch probe channel 2 deactivated 1: Touch probe channel 2 activated
Bit 9	TP2: Position detected - rising edge		0: Position not detected 1: Position detected
Bit 10	TP2: Position detected - falling edge		0: Position not detected 1: Position detected
Bit 11	Reserved		
Bit 12	Reserved		
Bit 13	Reserved		
Bit 14	Touch probe (TP): Level at release time		Level at detection via touch probe channel 2: 0: LOW level 1: HIGH level
Bit 15	Reserved		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			UNSIGNED_16

7.11.7 Time stamps and positions detected

Note!

In the event of a "Continuous trigger" touch probe configuration, a newly detected value overwrites the value detected before, even if it has not been retrieved by the controller.

0x2D01 | 0x3501 - Touch probe (TP): Time stamp

Time stamp for the positions detected via touch probe
- From software version V01.05.xx onwards, the time stamps of the TPs are coupled to the Ethercat DC time and directly map the lower 32 bits of the DC time. Thus, an absolute time of the TP event is available for use in the system.

Sub.	Name	Lenze setting	Data type
1	TP1: Time stamp - rising edge	0 ns	UNSIGNED_32
2	TP1: Time stamp - falling edge	0 ns	UNSIGNED_32
3	TP2: Time stamp - rising edge	0 ns	UNSIGNED_32
4	TP2: Time stamp - falling edge	0 ns	UNSIGNED_32
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			

0x60BA | 0x68BA - Touch probe pos1 pos value

Touch probe position 1 detected with a rising edge

Display range (min. value | unit | max. value)			Initialisation	
-2147483648	[Pos unit]	2147483647	0 [Pos unit]	
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX				INTEGER_32

0x60BB | 0x68BB - Touch probe pos1 neg value

Touch probe position 1 detected with a falling edge

Display range (min. value | unit | max. value)			Initialisation	
-2147483648	[Pos unit]	2147483647	0 [Pos unit]	
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX				INTEGER_32

0x60BC | 0x68BC - Touch probe pos2 pos value

Touch probe position 2 detected with a rising edge

Display range (min. value | unit | max. value)			Initialisation	
-2147483648	[Pos unit]	2147483647	0 [Pos unit]	
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX				INTEGER_32

0x60BD | 0x68BD - Touch probe pos2 neg value

Touch probe position 2 detected with a falling edge

Display range (min. value | unit | max. value)			Initialisation	
-2147483648	[Pos unit]	2147483647	0 [Pos unit]	
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX				INTEGER_32

8 Monitoring functions

The i700 servo inverters is provided with various monitoring functions, which are described in the following subchapters.

Objects described in this chapter

Object Axis A	 Axis B	Name	Data type
0x2D40	0x3540	Ixt utilisation	RECORD
0x2D44	0x3544	Motor speed monitoring	RECORD
0x2D45	0x3545	Motor phase failure detection	RECORD
0x2D46	0x3546	Monitoring: Ultimate motor current	RECORD
0x2D49	0x3549	Motor temperature monitoring: Parameter	RECORD
0x2D4C	0x354C	Motor utilisation (I^2xt): Parameter for the thermal model	RECORD
0x2D4D	0x354D	Motor utilisation (I^2xt): User-definable characteristic	RECORD
0x2D4E	0x354E	Motor utilisation (I^2xt): Motor overload warning threshold	UNSIGNED_16
0x2D4F	0x354F	Motor utilisation (I^2xt): Actual utilisation	UNSIGNED_16
0x2D50	0x3550	Motor utilisation (I^2xt): Motor overload error	UNSIGNED_32
0x2D84	0x3584	Heatsink temperature	RECORD

8.1 24-V supply voltage monitoring

The 24 V supply voltage is monitored by means of two specified voltage thresholds.

- The following table describes these thresholds and shows the device response resulting from the level of the supply voltage:

Threshold	Voltage	Device response	
U24 warn	21.45 V ±5 %	Above threshold:	Normal operation
		Below threshold:	Normal operation with warning In this case, all inverters that are operated in parallel in the same DC bus connection must be stopped in order to be protected from internal destruction.
U24 down	18.15 V ±5 %	Above threshold:	Normal operation with warning
		Below threshold:	No operation • Response with regard to the EtherCAT master and adjacent EtherCAT slaves is undefined.

Related topics:

▶ Monitoring of the DC-bus voltage (📖 55)

8.2 Monitoring of the power section and device utilisation (Ixt)

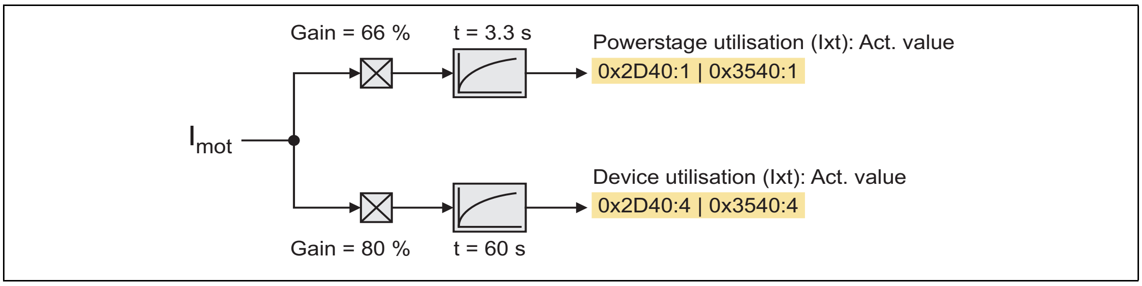
The monitoring of the device utilisation primarily protects the power section. Indirectly, also other components such as conductors and terminals are protected. The monitoring system is designed as simply as possible, in order to facilitate the determination of the reserves until disconnection, and to carry out the drive dimensioning accordingly. Only cycles the arithmetic mean (AV) of which does not exceed the continuous device current are permissible.

"i700 servo inverters" hardware manual

In order to determine the reserves until Ixt disconnection, the rated data depending on the switching frequency and mains operation such as the rated device current and maximum current for the respective i700 design can be gathered from the hardware manual. (See chapter 4.2.4 "Axis modules" and chapter 4.3 "Overcurrent operation".)

The motor current is evaluated depending on the continuously permissible rated current at the effective switching frequency and mains voltage and is provided to two PT1 timing elements:

- The PT1 timing element with a time constant of t = 3.3 s simulates the thermal response of the power section (power semiconductor).
- The PT1 timing element with a time constant of t = 60 s serves to monitor the cycle with maximum current in order to protect the device from thermal overload (heatsink temperature).



[8-1] Ixt monitoring signal flow

The outputs of the two PT1 timing elements show the current utilization of power section and the entire device.

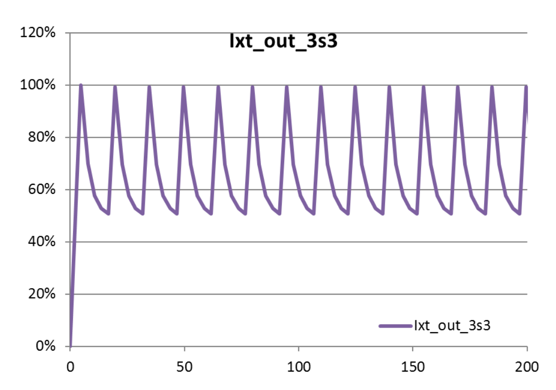
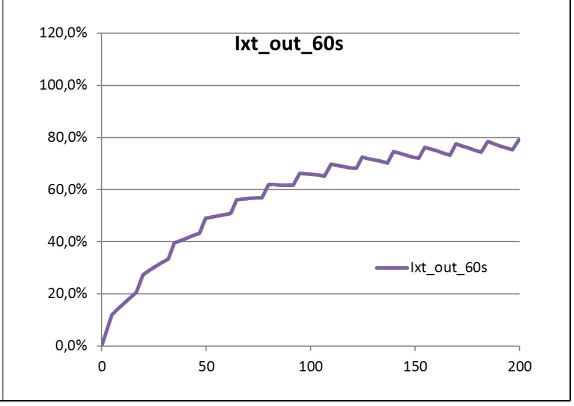
Gain factors and filter time constants are defined such that within a certain cycle time a specific overload current can be active for a certain period with subsequent recovery phase (75 % rated current).

The following load change cycles are defined for the i700 servo inverters:

- 15-second cycle:
  200 % rated current for 3 s, recovery phase 12 s with 75 % rated current
- 24-second cycle:
  175 % rated current for 6 s, recovery phase 18 s with 75 % rated current
- 3-minute cycle:
  150 % rated current for 60 s, recovery phase 120 s with 75 % rated current

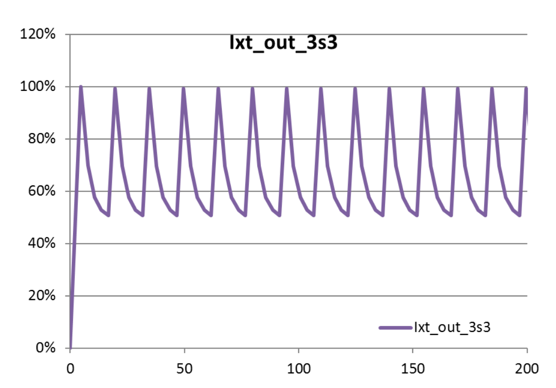
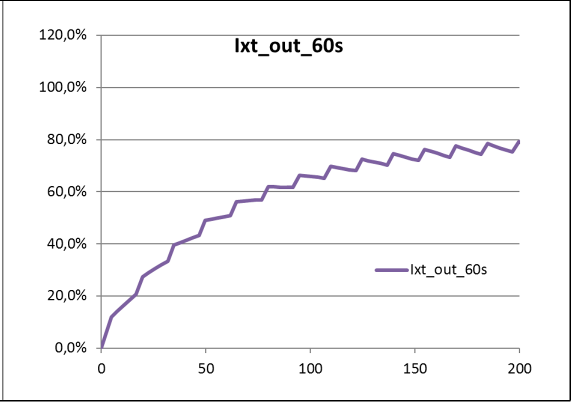
1. Load cycle

15-second cycle with a 200 %/3 s overload phase and a 75 %/12 s recovery phase.

PT1 with 3.3 s filter time constant for power section protection	PT1 with 60 s filter time constant for device protection
	

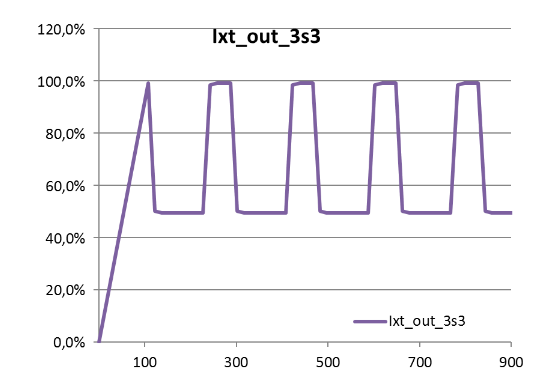
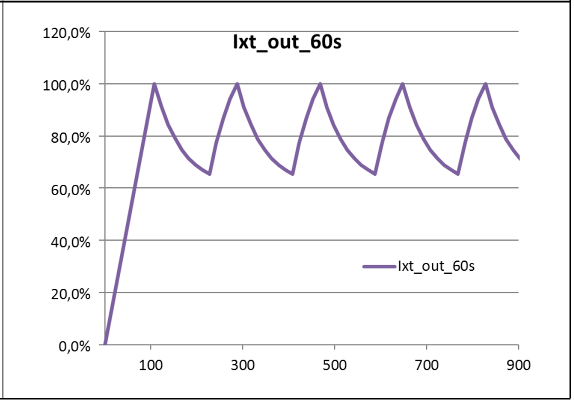
2. Load cycle

24-second cycle with a 175 %/6 s overload phase and a 75 %/18 s recovery phase.

PT1 with 3.3 s filter time constant for power section protection	PT1 with 60 s filter time constant for device protection
	

3. Load cycle

3 minute cycle with a 150 %/60 s overload phase and a 75 %/120 s recovery phase.

PT1 with 3.3 s filter time constant for power section protection	PT1 with 60 s filter time constant for device protection
	

Note!

Though continuous operation over 100 % is allowed for by the Ixt monitoring function, this is nevertheless an impermissible working point!

Parameterisation and evaluation of the monitoring function

The Ixt monitoring function can be parameterised and evaluated via the following indices. Here, a warning limit and an error limit can be set for each PT1 timing element.

- If the warning threshold (Lenze setting: 95 %) is exceeded, a warning is output in order that the master is still able to respond if required.
- If the error threshold is exceeded (Lenze setting: 101 %), the i700 servo inverters is switched off for protecting the device and generates an error message.

0x2D40 | 0x3540 - Ixt utilisation

Sub.	Name	Lenze setting	Data type
1	Power section utilisation (Ixt): Actual utilisation	0 %	UNSIGNED_16
2	Power stage utilisation (Ixt): Warning threshold	95 %	UNSIGNED_16
3	Power stage utilisation (Ixt): Error threshold	101 %	UNSIGNED_16
4	Device utilisation (Ixt): Actual utilisation	0 %	UNSIGNED_16
5	Device utilisation (Ixt): Warning threshold	95 %	UNSIGNED_16
6	Device utilisation (Ixt): Error threshold	101 %	UNSIGNED_16

Subindex 1: **Power stage utilisation (Ixt): Actual utilisation**				
Display range (min. value | unit | max. value)			Initialisation	
0	%	101	0 %	
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX				UNSIGNED_16

Subindex 2: **Power stage utilisation (Ixt): Warning threshold**				
Setting range (min. value | unit | max. value)			Lenze setting	
0	%	101	95 %	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 3: **Power stage utilisation (Ixt): Error threshold**				
Display range (min. value | unit | max. value)			Initialisation	
0	%	101	101 %	
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 4: **Device utilisation (Ixt): Actual utilisation**				
Display range (min. value | unit | max. value)			Initialisation	
0	%	101	0 %	
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX				UNSIGNED_16

Subindex 5: **Device utilisation (Ixt): Warning threshold**				
Setting range (min. value | unit | max. value)			Lenze setting	
0	%	101	95 %	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 6: **Device utilisation (Ixt): Error threshold**				
Display range (min. value | unit | max. value)			Initialisation	
0	%	101	101 %	
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX				UNSIGNED_16

8.3 Monitoring of the heatsink temperature

In order to avoid impermissible heating of the device, the heatsink temperature is detected and monitored.

Note!

In the temperature range 0 ... 80 °C, the heatsink temperature is measured with a tolerance of -2 ... +4 °C. Outside this temperature range, the measuring accuracy strongly decreases.

0x2D84 | 0x3584 - Heatsink temperature

Sub.	Name	Lenze setting	Data type
1	Heatsink temperature: Actual temperature		INTEGER_16
2	Heatsink temperature: Warning threshold	80.0 °C	INTEGER_16
3	Heatsink temperature: Threshold - switch-on fan	55.0 °C	INTEGER_16
4	Heatsink temperature: Threshold - switch-off fan	45.0 °C	INTEGER_16

Subindex 1: **Heatsink temperature: Actual temperature**				
Display of the current heatsink temperature				
Display range (min. value | unit | max. value)			Initialisation	
-3276.8	°C	3276.7		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: 1/10	INTEGER_16

Subindex 2: **Heatsink temperature: Warning threshold**				
Warning threshold for temperature monitoring • If the heatsink temperature exceeds the threshold value set here, the device reports a warning. The warning message is reset with a hysteresis of approx. 5 °C. • If the heatsink temperature increases further and exceeds the non-adjustable error threshold (100 °C), the device changes to the "Fault" device status. Further operation of the i700 servo inverter is disabled by controller inhibit.				
Setting range (min. value | unit | max. value)			Lenze setting	
50.0	°C	100.0	80.0 °C	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	INTEGER_16

Subindex 3: **Heatsink temperature: Threshold for fan switch-on**				
Switch-on threshold for device fans • If the heatsink temperature exceeds the threshold value set here, the device fan is switched on. This also happens if the switch-off threshold is parameterised higher than the switch-on threshold by mistake. • In order to activate the fan, the switch-on and switch-off threshold can be set to the maximum value as then the device simultaneously switches to the "Fault" status. • Bigger designs of the Servo-Inverter i700 are provided with an internal fan. This is triggered at the same time as the heatsink fan and does not have internal switch-on or switch-off thresholds. **Note!** Increasing the switch-on threshold can cause higher device temperatures. This may reduce the service life of the device!				
Setting range (min. value | unit | max. value)			Lenze setting	
0.0	°C	100.0	55.0 °C	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	INTEGER_16

Subindex 4: **Heatsink temperature: Threshold for fan switch-off**				
Switch-off threshold for device fans • If the heatsink temperature falls below the threshold value set here, the device fan is switched off. This only happens if the switch-off threshold is parameterised lower than the switch-on threshold. • Bigger designs of the Servo-Inverter i700 are provided with an internal fan. This is triggered at the same time as the heatsink fan and does not have internal switch-on or switch-off thresholds.				
Setting range (min. value | unit | max. value)			Lenze setting	
0.0	°C	100.0	45.0 °C	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	INTEGER_16

8.4 Monitoring of the motor utilisation (I²xt)

This monitoring detects the thermal utilisation of the motor by calculating the thermal motor utilisation from the detected motor currents based on a mathematical model and displaying it in the object 0x2D4F (or 0x354F for axis B).

In case of permanent overload and excess of the warning threshold set in the object 0x2D4E (or 0x354E for axis B), a warning is output in order that the higher-level Controller is still able to respond and reduce the motor load or interrupt the operation.

From software version V01.04.xx onwards, an error response in the object 0x2D50:1 (or 0x3550:1 for axis B) can also be parameterised if the disconnection is not to be or cannot be executed by a higher-level controller.

STOP

Stop!

Monitoring the motor utilisation (I²xt) is not a means for full motor protection!

Since the motor utilisation calculated in the thermal model gets lost after mains switching, the following operating states cannot be determined correctly:

- Restarting (after mains switching) of a motor that is already very hot.
- Change of the cooling conditions (e.g. cooling air flow interrupted or too warm).

Full motor protection requires additional measures such as the evaluation of temperature sensors that are located directly in the winding or the use of thermal contacts.

During the calculation, the speed dependence of the permissible motor load and thus of the permissible current (difference between the standstill current and rated current is taken into consideration. This is done via object 0x2D4D (or 0x354D for axis B).

The threshold value for the warning output can be adapted via the following object:

0x2D4E | 0x354E - Motor utilisation (I²xt): Motor overload warning threshold

Setting range (min. value | unit | max. value)			Lenze setting	
0	%	250	100 %	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

The current thermal motor utilisation is shown in the following object:

0x2D4F | 0x354F - Motor utilisation (I²xt): Actual utilisation

Display range (min. value | unit | max. value)			Initialisation	
0	%	250		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX				UNSIGNED_16

0x2D50 | 0x3550 - motor utilisation (I²xt): Motor overload error

From software version V01.04.xx onwards
If the disconnection must not or cannot be executed by a higher-level Controller, the operation can be interrupted by the following parameter setting:
- As a response to the excess of the adjustable error threshold, set "1: Fault" in subindex 1.
- In subindex 2, set the threshold value as error threshold which in case of being exceeded triggers an interruption of the operation.

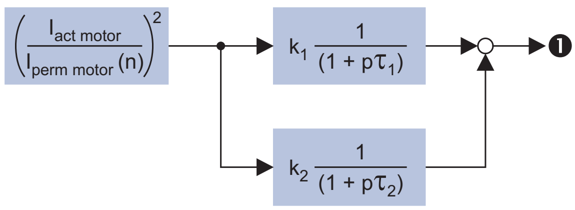
Sub.	Name	Lenze setting	Data type
▸ 1	Motor utilisation (I²xt): Response	1: Trouble	UNSIGNED_8
▸ 2	Motor utilisation (I²xt): Error threshold	105 %	UNSIGNED_16

Subindex 1: **Motor utilisation (I²xt): Response**			
Selection list(Lenze setting printed in bold)			
0	No response		
1	**Interference**		
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			UNSIGNED_8

Subindex 2: **Motor utilisation (I²xt): Error threshold**				
Setting range (min. value | unit | max. value)			Lenze setting	
0	%	250	105 %	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

The introduction of a two-component model with two time constants (one for the winding and the housing/laminated core, respectively) serves to display the thermal behaviour of the motors up to 500% of the rated current.

Structure of the monitoring



❶ Thermal utilisation of the motor in [%]

Axis A	Axis B	Symbol	Description	Dimension unit
-	-	$I_{act\ motor}$	Actual motor current	A
-	-	$I_{perm\ motor}$	Permissible motor current (speed-dependent)	A
0x2D4C:1	0x354C:1	τ_1	Therm. time constant coil	s
0x2D4C:2	0x354C:2	τ_2	Therm. time constant of the laminated core	s
0x2D4C:3	0x354C:3	k_1	Percentage of the winding in the final temperature	%
-	-	k_2	Percentage of the laminated core in the final temperature: $k_2 = 100\ \% - k_1$	%

0x2D4C | 0x354C - Motor utilisation (I²xt): Parameter for the thermal model

Sub.	Name	Lenze setting	Data type
▸ 1	Motor utilisation (I²xt): Thermal time constant - winding	60 s	UNSIGNED_16
▸ 2	Motor utilisation (I²xt): Thermal time constant - laminations	852 s	UNSIGNED_16
▸ 3	Motor utilisation (I²xt): Influence of the winding	27 %	UNSIGNED_8
▸ 4	Motor utilisation (I²xt): Starting value	0 %	UNSIGNED_16

Subindex 1: **Motor utilisation (I²xt): Thermal time constant of the winding**				
Setting range (min. value | unit | max. value)			Lenze setting	
1	s	36000	60 s	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 2: **Motor utilisation (I²xt): Thermal time constant - laminated core**				
Setting range (min. value | unit | max. value)			Lenze setting	
1	s	36000	852 s	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 3: **Motor utilisation (I²xt): Influence of the winding**				
Setting range (min. value | unit | max. value)			Lenze setting	
0	%	100	27 %	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_8

Subindex 4: **Motor utilisation (I²xt): Starting value**				
Setting range (min. value | unit | max. value)			Lenze setting	
0	%	250	0 %	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Calculation with only one time constant

If k1 = "0 %" is set, the part of the winding is not taken into consideration and the thermal model is only calculated using the time constant set for the housing/laminated core. This setting is e.g. required if only the time constant of the laminated core (T2) is known.

Parameter setting of the time constant and the influence of the winding on motors of other manufacturers

When the influence of the winding is activated, the i²xt monitoring becomes more sensible as if only the influence of the laminated core would be used for monitoring purposes.

The necessity to activate the influence of the winding rises with the increasing utilisation of the motor overload capacity. It also rises with applications where the motor is at standstill for longer periods or cyclically and a load ≥ permanent standstill current is applied.

For determining the values for the thermal time constant, try to get the data from the motor manufacturer. If this is not possible, you can use the data of a comparable Lenze motor.

Conditions for comparability are similar values in case of the following motor features:

- Square dimensions of the motor (active part)
- Length of the active part (if available)
- Permanent standstill current Io [A_RMS]
- Peak current/overload capacity [A_RMS]
- Copper resistance of the winding at 20 °C [Rphase]

Example:

Motor features	Data of the third-party motor	Data of a comparable Lenze motor (from motor catalogue)
Square dimension	95 mm	MCS09xxx = 89 mm
Standstill current	2.2 A	MCS09F38 = 3.0 A
Peak current	7.3 A	MCS09F38 = 15 A
Phase resistance	5.1 Ohm	MCS09F38 = 5.2 Ohm

When the "MCS09F38" Lenze motor is selected from the motor catalogue of the »PLC Designer«m the following values are displayed:

- Thermal time constant of the winding = 126 s ➞ 0x2D4C:1 (or 0x354C:1 for axis B)
- Influence of the winding = 27 % ➞ 0x2D4C:3 (or 0x354C:3 for axis B)

Speed-dependent evaluation of the motor current

By selecting a characteristic, the permissible motor current is evaluated depending on speed for calculating the thermal motor utilisation. For this purpose, up to four operating points on the S1 characteristic of a motor can be used.

- The S1 characteristic can be found in the technical data sheet/catalogue of the respective motor.
- The representation in the objects /characteristic is carried out as relative values with reference to rated values.

Note!

When you select a Lenze motor from the catalogue and transfer its parameters into the i700 servo inverters a typical characteristic is automatically set for the selected motor. A deviating parameterisation is only required if the motor is operated in ambient conditions which demand a general derating. Example: Use in site altitudes > 1000 m.

In case of motors of other manufacturers, the operating points have to be parameterised based on the data sheet information.

Operating points		
❶	Standstill n01-I01	For motors, this operating point is often described with the no-Io values.
❷	Reference point n02-I02	If the value falls below the speed n02, a derating in the current is required because: • the motor cooling of self-ventilated motors deteriorates considerably. • a DC current load causes an increased power loss in a winding. For motors, this operating point is also described with the no*-Io* values.
❸	Rated point (n03=nrated)-(I03=Irated)	Rated values of the motor are the reference for all operating points of the i²xt monitoring.
❹	Field weakening n04-I04	This operating point should be parameterised independently of the use in the current application.

0x2D4D | 0x354D - Motor utilisation (I²xt): User-definable characteristic

Sub.	Name	Lenze setting	Data type
1	I²xt: x1 = n01/nN (n01 ~ 0)	0 %	UNSIGNED_16
2	I²xt: y1 = i01/iN (x = n01 ~ 0)	100 %	UNSIGNED_16
3	I²xt: x2 = n02/nN (n02 = limit - reduced cooling)	0 %	UNSIGNED_16
4	I²xt: y2 = i02/iN (x = n02 = limit - reduced cooling)	100 %	UNSIGNED_16
5	I²xt: x3 = n03/nN (n03 = rated speed)	100 %	UNSIGNED_16
6	I²xt: y3 = i03/iN (x = n03 = rated speed)	100 %	UNSIGNED_16
7	I²xt: x4 = n04/nN (n04 = limit - field weakening)	100 %	UNSIGNED_16
8	I²xt: y4 = i04/iN (x = n04 = limit - field weakening)	100 %	UNSIGNED_16
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			

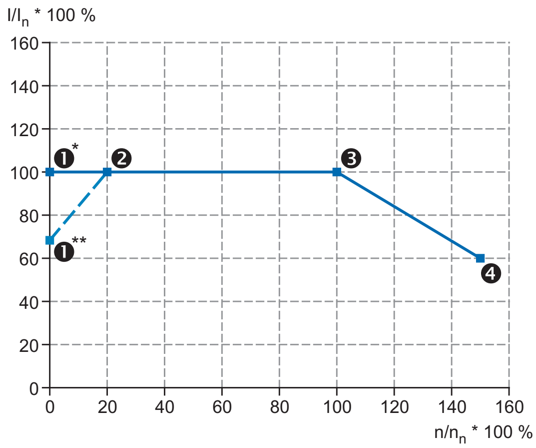
An example of how to enter the characteristic can be found in the following subchapter.

8.4.1 Example of how to enter the characteristic for standard and servo motors

The required data of the operating points result from the S1 characteristic of the prevailing motor:

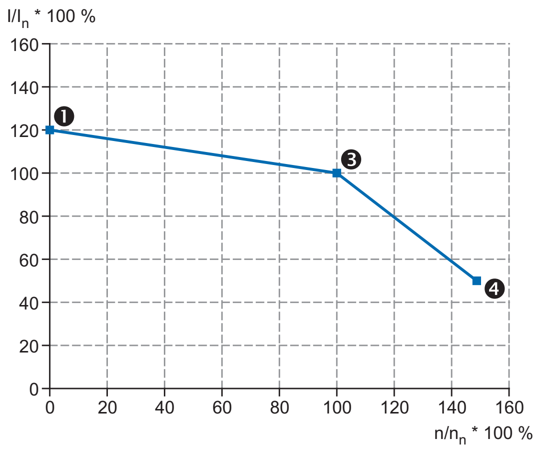
Examples for S1 characteristics

Standard motor

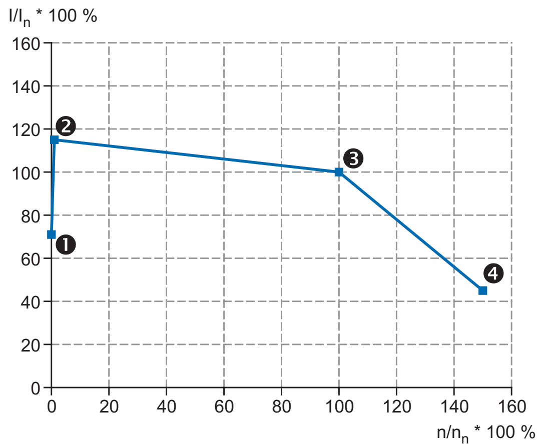


* Standard motor forced ventilated
** Standard motor self-ventilated

Servo motor



Servo motor with derating at standstill



Object		Characteristic point		Info
Axis A	Axis B			
0x2D4D:1	0x354D:1	❶	n_1/n_n * 100 %	Speed = "0" (standstill)
0x2D4D:2	0x354D:2		I_1/I_n * 100 %	Permissible motor current at standstill
0x2D4D:3	0x354D:3	❷	n_2/n_n * 100 %	Speed from which the current must be reduced for self-ventilated motors. • Below this speed the cooling air flow of the integral fan is not sufficient anymore.
0x2D4D:4	0x354D:4		I_2/I_n * 100 %	Permissible motor current at speed n_2 (torque reduction)
0x2D4D:5	0x354D:5	❸	n_3/n_n * 100 %	Rated speed
0x2D4D:6	0x354D:6		I_3/I_n * 100 %	Permissible motor current at rated speed
0x2D4D:7	0x354D:7	❹	n_4/n_n * 100 %	Speed above the rated speed (in the field weakening range for asynchronous motors)
0x2D4D:8	0x354D:8		I_4/I_n * 100 %	Permissible motor current at speed n_4 (field weakening)

The speed-dependent evaluation of the permissible motor current can actually be switched off by parameterising all 8 characteristic points to "100 %".

Stop!

Applications with standard motors

Self-ventilated standard motors are insufficiently protected at low speeds if the speed-dependent evaluation of the permissible motor current is not effective (all characteristic points = "100 %").

Applications with PM synchronous motors

Please check for every individual case which r.m.s. value can be used to permanently operate the motor at standstill.

In case of some motors, a derating I1/In < 100 % is required when n1/nn = 0 %. This serves to prevent an overload of individual motor phases as their power loss doubles with continuous DC current load. (It is called DC current load as the field frequency amounts to 0 Hz at standstill.)

8.4.2 UL 508-compliant I²xt motor overload monitoring

If the compliance with the UL 508 standard is required for the operation of the motor and the UL 508 compliant motor load monitoring is implemented by the mathematical model of the I²xt monitoring, the following conditions have to be met.

Note!

The settings described have been taken from the current UL 508 standard.

The i700 servo inverters only outputs one warning if the warning limit has been exceeded (0x2D4E or 0x354E for axis B). The i700 servo inverters cannot interrupt the operation of the motor. The motor load has to be interrupted by a higher-level Controller.

Stop!

From software version V01.04.xx onwards:

If the disconnection must not or cannot be executed by a higher-level Controller, the operation has to be interrupted by the following parameter setting:

1. Set the response "1: Fault" to the excess of the adjustable error threshold in the object 0x2D50:1 (or 0x3550:1 for axis B).
2. Set the threshold value as error threshold which in case of being exceeded triggers an interruption of the operation in the object 0x2D50:2 (or 0x3550:2 for axis B).

UL 508 condition 1:

With a motor load of 600 %, the I²xt warning has to be output within 20 seconds.

- A motor load of 600 % occurs if the r.m.s. value of the total motor current displayed in the object 0x2DD1:5 (or 0x35D1:5 for axis B) corresponds to 6 times the rated motor current (0x6075 or 0x6875 for axis B).

This condition can be met by the following parameter setting:

Object		Name	setting
Axis A	Axis B		
0x2D4C:2	0x354C:2	Motor utilisation (I²xt): Thermal time constant - laminations	≤ 709 s
0x2D4C:3	0x354C:3	Motor utilisation (I²xt): Influence of the winding	0 %
0x2D4E	0x354E	Motor utilisation (I²xt): Motor overload warning threshold	100 %
From software version V01.04.xx onwards:			
0x2D50:1	0x3550:1	Motor utilisation (I²xt): Response	1: Fault
0x2D50:2	0x3550:2	Motor utilisation (I²xt): Error threshold	100 %

UL 508 condition 2:

With a motor load of 110 % and a motor rotating field frequency of 10 Hz, the I²xt warning has to be output faster than with a motor rotating field frequency of 20 Hz.

- The current motor rotating field frequency is displayed in the object 0x2DDD (or 0x35DD for axis B).
- A motor load of 110 % occurs if the r.m.s. value of the total motor current displayed in the object 0x2DD1:5 (or 0x35D1:5 for axis B) corresponds to 1.1 times the rated motor current (0x6075 or 0x6875 for axis B).

This condition can be met by the following parameter setting:

Object		Name	setting
Axis A	Axis B		
0x2D4D:1	0x354D:1	I²xt: x1 = n01/nN (n01 ~ 0)	0 %
0x2D4D:2	0x354D:2	I²xt: y1 = i01/iN (x = n01 ~ 0)	< 100 %
0x2D4D:3	0x354D:3	I²xt: x2 = n02/nN (n02 = limit - reduced cooling) Setting: 100 % * 20 Hz / rated motor frequency 0x2C01:5 or 0x3401:5)	
0x2D4D:4	0x354D:4	I²xt: y2 = i02/iN (x = n02 = limit - reduced cooling)	≥ 100 %

UL 508 condition 3:

After mains switching and a motor load > 100 %, the I²xt warning has to be output faster than in the same load case before mains switching.

- A motor load of > 100 % occurs if the r.m.s. value of the total motor current displayed in the object 0x2DD1:5 (or 0x35D1:5 for axis B) is higher than the rated motor current (0x6075 or 0x6875 for axis B).

This condition can be met by the following parameter setting:

Object		Name	setting
Axis A	Axis B		
0x2D4C:4	0x354C:4	Motor utilisation (I²xt): Starting value	> 0 %

8.5 Motor temperature monitoring

0x2D49 | 0x3549 - Motor temperature monitoring: Parameters

Sub.	Name	Lenze setting	Data type
1	Motor temperature monitoring: Sensor type	0: KTY83-110	UNSIGNED_8
2	Motor temperature monitoring: Response	1: Trouble	UNSIGNED_8
3	Motor temperature monitoring: Warning threshold	145.0 °C	INTEGER_16
4	Motor temperature monitoring: Error threshold	155.0 °C	INTEGER_16
5	Motor temperature monitoring: Actual motor temperature		INTEGER_16
6	Thermal sensor characteristic: Grid point 1 - temperature	25.0 °C	INTEGER_16
7	Thermal sensor characteristic: Grid point 2 - temperature	150.0 °C	INTEGER_16
8	Spec. charact.: resistance sampling point 1	1000 Ohm	INTEGER_16
9	Spec. charact.: resistance sampling point 2	2225 ohms	INTEGER_16

Subindex 1: **Motor temperature monitoring: Sensor type**		
Selection of the motor temperature sensor used		
Selection list(Lenze setting printed in bold)		Info
0	**KTY83-110**	
1	KTY83-110 + 2 x PTC 150 °C (series connection)	
2	KTY84-130	
3	Spec. charact. curve	From software version V01.03.xx onwards Spec. charact. curve for motor temperature sensor
4	Reserved	
5	PT1000	From software version V01.10.xx or V02.10.xx onwards
6	PT1000 + 2 x PTC 150 °C (series connection)	From software version V01.10.xx or V02.10.xx onwards
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX		UNSIGNED_8

Subindex 2: **Motor temperature monitoring: Response**		
Response when tripping the motor temperature monitoring • In order to exclude an unwanted tripping of the fault during initial commissioning without connected motor, the error response is a warning as long as the device is in the "switch-on inhibited" status. If this status is left, the error response parameterised here is triggered.		
Selection list(Lenze setting printed in bold)		
0	No response	
1	**Interference**	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX		UNSIGNED_8

Subindex 3: **Motor temperature monitoring: Warning threshold**				
Warning threshold for motor temperature monitoring • Reset is effected with a hysteresis of 5 °C.				
Setting range (min. value | unit | max. value)			Lenze setting	
-3276.8	°C	3276.7	145.0 °C	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	INTEGER_16

Subindex 4: **Motor temperature monitoring: Error threshold**				
Error threshold for motor temperature monitoring • Reset is effected with a hysteresis of 5 °C.				
Setting range (min. value | unit | max. value)			Lenze setting	
-3276.8	°C	3276.7	155.0 °C	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	INTEGER_16

Subindex 5: **Motor temperature monitoring: Actual motor temperature**				
Display of the current motor temperature				
Display range (min. value | unit | max. value)			Initialisation	
-666.0	°C	666.0		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: 1/10	INTEGER_16

Subindex 6: **Spec. charact.: temperature sampling point 1**				
From software version V01.03.xx onwards The special thermal sensor characteristic is selected by setting "3" in 0x2D49:1 (or 0x3549:1 for axis B)				
Setting range (min. value | unit | max. value)			Lenze setting	
0.0	°C	255.0	25.0 °C	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	INTEGER_16

Subindex 7: **Spec. charact.: temperature sampling point 2**				
From software version V01.03.xx onwards The special thermal sensor characteristic is selected by setting "3" in 0x2D49:1 (or 0x3549:1 for axis B)				
Setting range (min. value | unit | max. value)			Lenze setting	
0.0	°C	255.0	150.0 °C	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	INTEGER_16

Subindex 8: **Spec. charact.: resistance sampling point 1**				
From software version V01.03.xx onwards The special thermal sensor characteristic is selected by setting "3" in 0x2D49:1 (or 0x3549:1 for axis B)				
Setting range (min. value | unit | max. value)			Lenze setting	
0	Ohm	30000	1000 Ohm	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				INTEGER_16

Subindex 9: **Spec. charact.: resistance sampling point 2**				
From software version V01.03.xx onwards The special thermal sensor characteristic is selected by setting "3" in 0x2D49:1 (or 0x3549:1 for axis B)				
Setting range (min. value | unit | max. value)			Lenze setting	
0	Ohm	30000	2225 ohms	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				INTEGER_16

8.5.1 Spec. charact. curve for motor temperature sensor

This function extension is available from software version V01.03.xx onwards!

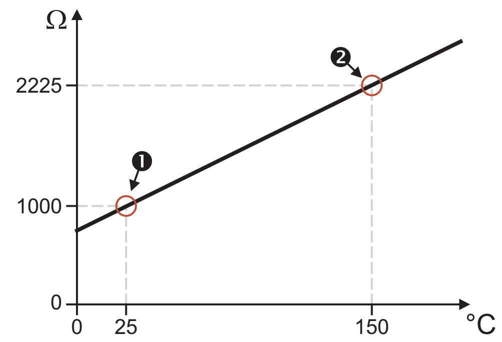
If required, you can define and activate a special characteristic for the motor temperature sensor.

Stop!

The implemented solution is not suitable as a replacement for a tripping unit for the thermal protection of rotating electrical machines (EN 60947-8:2003)!

- The special characteristic is activated by setting 0x2D49:1 = "3" (or 0x3549:1 = "3" for axis B).
- The special characteristic is defined based on two parameterisable sampling points. The two sampling points define a line which is extrapolated to the right and to the left. In the Lenze setting, the special characteristic is defined as follows:

Lenze setting of the special characteristic



Axis A	Axis B	Description	Lenze setting
0x2D49:6	0x3549:6	Sampling point ❶ - Temperature	25.0 °C
0x2D49:8	0x3549:8	Sampling point ❶ - resistance	1000 Ohm
0x2D49:7	0x3549:7	Sampling point ❷ - Temperature	150.0 °C
0x2D49:9	0x3549:9	Sampling point ❷ - resistance	2225 Ohms

Note!

- Selecting a motor from the motor catalogue causes the parameters of the special characteristic to be overwritten!
- If the i700 servo inverters measures a resistance of below 122 Ω, this is assessed as a sensor error and a temperature of -666 °C is output.
- In individual cases, a short circuit is a wanted status (e.g. temperature contact, closed below 140 °C). For this purpose, the sampling point 1 (0x2D49:8 or 0x3549:8 for axis B) must be below 122 Ω to not trigger a sensor error anymore. The temperature is continued to be calculated.

8.6 Motor speed monitoring

0x2D44 | 0x3544 - Motor speed monitoring

Sub.	Name	Lenze setting	Data type
▸ 1	Motor speed monitoring: Threshold	8000 r/min	UNSIGNED_16
▸ 2	Motor speed monitoring: Response	1: Trouble	UNSIGNED_8

Subindex 1: **Motor speed monitoring: Threshold**				
Warning/error threshold for motor speed monitoring				
Setting range (min. value | unit | max. value)			Lenze setting	
50	r/min	50000	8000 r/min	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 2: **Motor speed monitoring: Response**			
Response when tripping the motor speed monitoring			
Selection list(Lenze setting printed in bold)			
0	No response		
1	**Interference**		
2	Warning		
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			UNSIGNED_8

8.7 Motor phase failure monitoring

Note!

In the Lenze setting, the monitoring function is not activated!

0x2D45 | 0x3545 - Motor phase failure detection

Sub.	Name	Lenze setting	Data type
1	Motor phase failure 1: Response	0: No Response	UNSIGNED_8
2	Motor phase failure 1: Current threshold	5.0 %	UNSIGNED_8
3	Motor phase failure 1: Voltage threshold	10.0 V	UNSIGNED_16
4	Motor phase failure 2: Response	0: No Response	UNSIGNED_8

Subindex 1: **Motor phase failure 1: Response**		
Monitoring 1 (during operation): Response during tripping		
Selection list(Lenze setting printed in bold)		
0	**No response**	
1	Interference	
2	Warning	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX		UNSIGNED_8

Subindex 2: **Motor phase failure 1: Current threshold**				
Monitoring 1 (during operation): Selection of the current threshold for activation • 100 % ≡ maximum axis current (0x2DDF:2 or 0x35DF:2 for axis B) • Background: In order to be able to reliably detect the failure of a motor phase, a certain motor current for the current sensors must flow first. The monitoring function therefore is only activated if, in the case of a servo control, the setpoint of the motor current, and in the case of a V/f characteristic control, the actual value of the motor current (display in 0x6078 or 0x6878 for axis B) has exceeded the current threshold set here.				
Setting range (min. value | unit | max. value)			Lenze setting	
1.0	%	10.0	5.0 %	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	UNSIGNED_8

Subindex 3: **Motor phase failure 1: Voltage threshold**				

Monitoring 1 (during operation): Selection of the voltage threshold for monitoring several motor phases
• The V/f characteristic control enables the detection of several failed motor phases during operation.
• Monitoring with regard to a failure of several motor phases is active if
  • a response other than "0: No response" is set in subindex 1, and
  • the motor voltage exceeds the voltage threshold set here.
• The "Motor disconnected" error message is output if the motor current is lower than the device-dependent threshold value for more than 20 ms.
• The monitoring function for the failure of several motor phases can be deactivated if the value "1000.0 V" is set here.				

Setting range (min. value | unit | max. value)			Lenze setting	
0.0	V	100.0	10.0 V	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	UNSIGNED_16

Subindex 4: **Motor phase failure 2: Response**			
Monitoring 2 (prior to operation): Response during activation			
Selection list(Lenze setting printed in bold)			
0	**No response**		
1	Interference		
2	Warning		
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			UNSIGNED_8

8.7.1 Limits of the motor phase failure monitoring

The motor phase failure monitoring can be activated for both synchronous and asynchronous motors. However, it is possible that a current flow cannot be detected for sure in the case of certain operating states of synchronous motors that are connected correctly. Hence, a fault is triggered. The following table provides an overview:

Operating status		Synchronous motor	Async. motor
Check of the motor phases prior to operation		●	●
Check of the motor phases during operation			
I_q < current threshold value	at standstill	○	●
	when motor is rotating	○	●
$I_q \geq$ current threshold value	at standstill	○	●
	when motor is rotating	●	●
Field weakening	when motor is rotating	●	○

- ● Phase failure is detected for sure
- ○ The monitoring function may be activated without a fault pending.
- I_q Torque-forming current component

Note!

Monitoring during operation serves especially for applications which are operated with constant load and speed. In all other cases, transient processes or unfavourable operating points can cause maloperation.

Special case: Operation in generator mode (e.g. hoist)

The motor phase failure monitoring may trigger a fault message in an operation in generator mode if the asynchronous motor applied for this purpose reaches the following working point:

- The hoist moves downwards, i.e. the motor is in generator mode.
- The slip frequency equals the field frequency in terms of amount. Both frequencies mutually neutralise themselves due to their opposite effective directions.

8.7.2 Monitoring 2: In the "enable operation" state transition

On the basis of test signals, this extended monitoring function for motor phase failure can detect a phase failure and check the presence of the motor. Only after a successful check, the actual operation is continued.

- Monitoring is only active for a short time after controller enable (when the device status changes from "Switched on" to "Operation enabled") if
  - a response other than "0: No response" has been set for this monitoring function in object 0x2D45:4 (or 0x3545:4 for axis B), and
  - no test mode and no identification mode (0x2825 or 0x3025 for axis B) are active.
- Before the actual operation, the motor is supplied with a maximum DC current, the max. level of which corresponds to the lower of the following two values:

$$50\ \% \cdot \sqrt{2} \cdot \text{Rated device current}$$

or

$$50\ \% \cdot \sqrt{2} \cdot \text{Rated motor current}$$

- The response set is triggered if one or several motor phase currents have not reached a specific threshold value within 5 ms after controller enable. The threshold value depends on the maximum device current and cannot be parameterised.
- The check is completed successfully if all three motor phase currents have exceeded the threshold value. Then the actual operation is continued immediately.

Note!

- As the check is cancelled immediately if all three motor phase currents have exceeded the threshold value, the setpoint current usually is not achieved.
- In order to be able to achieve the threshold value used for the check, the rated motor current must at least be 10 % of the maximum device current.

8.7.3 Monitoring 1: In "operation enabled" status

If a current-carrying motor phase (U, V, W) fails during operation, the response set for this monitoring is tripped if two conditions are met:

- Condition 1: Monitoring is activated
  - In order to be able to reliably detect the failure of a motor phase, a certain motor current for the current sensors must flow. The monitoring function therefore is only activated if, in the case of servo control, the setpoint of the motor current, and in the case of a V/f characteristic control, the actual value of the motor current (display in 0x6078 or 0x6878 for axis B) has exceeded a parameterisable current threshold (0x2D45:2 or 0x3545:2 for axis B).
- Condition 2: A specific commutation angle has been covered without the detection of a current flow.
  - In this case monitoring works according to the principle of checking for each motor phase that a current flows depending on the commutation angle.
  - Monitoring responds if a rotating field is output and hence a specific commutation angle (approx. 150°, electric) is covered without the current having exceeded a (non-parameterisable) threshold that depends on the device power.

Note!

The dependence on the commutation angle also causes a dependence on the motor type used:

- The commutation angle and the angle at the shaft (number of pole pairs) of a synchronous motor are proportional. This makes it possible to predict which shaft angle is maximally covered in the event of an error.
- There is still a slip between the commutation angle and the angle at the shaft of an asynchronous motor. This results in a load dependency which makes it impossible to predict a maximally covered shaft angle in the event of an error.

For some applications (e.g. when a hoist is lowered at non-zero speed) it may happen that there is no rotating field anymore, but a DC current is flowing. In this case, condition 2 is no longer met.

8.7.4 Monitoring with regard to short circuit and earth fault

The motor phases are monitored with regard to short circuit and earth fault by means of a hardware circuit. If this monitoring function is activated, the i700 servo inverters reports an error and changes to the "Pulse inhibit" error status. This error can only be reset after an inhibit time of 5 s has elapsed.

The following table lists possible causes of a short-circuit message:

Cause	Remedies
The is a physical connection between two motor phases.	Correct the wiring and remove the short circuit.
A transient current control process causes the phase currents to increase above the signalling threshold. This can happen if it is operated at the current limit of the device and a) the current controller is set incorrectly or b) a synchronous motor is operated in the field weakening range and the current controller feedforward control in 0x2941 (or 0x3141 for axis B) has been deactivated. A transient current control process can be triggered, for instance, if an EtherCAT fault causes a quick stop. In such a situation, normally some bus cycles pass by until the error is triggered. During this period, the setpoints remain frozen which causes an unsteady setpoint profile and thus a dominant current control process.	Remedy for a): Set current controller according to alignment instructions. ▶ Setting and optimising the current controller Remedy for b): If a synchronous motor is operated at the limits or within the field weakening range, activate the current controller feedforward control in 0x2941 (or 0x3141 for axis B).

8.8 Monitoring of the ultimate motor current

Stop!

The parameterisable ultimate motor current I_{ULT} is a limit value the exceedance of which causes non-reversible damages of the motor!

Examples:

- Demagnetisation of single rotor magnets when the PM servo motor is operated.
- Destruction of the stator winding.

This limit value must not be travelled cyclically in the drive process!

The maximum current (0x6073 or 0x6873 for axis B) parameterised should provide a sufficient distance to this ultimate motor current.

Note!

When you select a Lenze motor from the catalogue and transfer the parameters of the motor to the i700 servo inverters, the setting of the limit value I_{ULT} is automatically adapted to the selected motor.

In case of motors of other manufacturers, we recommend the parameterisation of the limit value I_{ULT} in 0x2D46:1 (or 0x3546:1 for axis B) to be executed as one of the first commissioning steps.

If the instantaneous value of the motor current exceeds the limit value set, the response set for this monitoring function is triggered in order to protect the motor.

0x2D46 | 0x3546 - Monitoring: Ultimate motor current

Sub.	Name	Lenze setting	Data type
1	Ultimate motor current: Threshold	5.4 A	UNSIGNED_16
2	Ultimate motor current: Response	1: Trouble	UNSIGNED_8

Subindex 1: **Ultimate motor current: Threshold**				
Warning/error threshold for motor current monitoring				
Setting range (min. value | unit | max. value)			Lenze setting	
0.0	A	500.0	5.4 A	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			Scaling: 1/10	UNSIGNED_16

Subindex 2: **Ultimate motor current: Response**			
Response when tripping the motor current monitoring			
Selection list(Lenze setting printed in bold)			
0	No response		
1	**Interference**		
2	Warning		
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			UNSIGNED_8

8.9 Monitoring of the position and velocity error

0x2D51 - monitoring: Position and velocity error

Sub.	Name	Lenze setting	Data type
▸ 1	Monitoring: Velocity error - error threshold	50 rpm	UNSIGNED_32
▸ 2	Monitoring: Velocity error - minimum error time	0 ms	UNSIGNED_16
▸ 3	Monitoring: Velocity error - response	0: No Response	UNSIGNED_8
▸ 4	Monitoring: Position error - error threshold	360 °	UNSIGNED_32
▸ 5	Monitoring: Position error - minimum error time	0 ms	UNSIGNED_16
▸ 6	Monitoring: Position error - response	0: No Response	UNSIGNED_8

Subindex 1: **Monitoring: Velocity error - error threshold**				
Setting range (min. value | unit | max. value)			Lenze setting	
1	r/min	2147483647	50 rpm	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_32

Subindex 2: **Monitoring: Velocity error - minimum error time**				
Setting range (min. value | unit | max. value)			Lenze setting	
0	ms	50	0 ms	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 3: **Monitoring: Velocity error - response**			
Selection list(Lenze setting printed in bold)			
0	**No response**		
1	Interference		
2	Warning		
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			UNSIGNED_8

Subindex 4: **Monitoring: Position error - error threshold**				
Setting range (min. value | unit | max. value)			Lenze setting	
1	°	2147483647	360 °	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_32

Subindex 5: **Monitoring: Position error - minimum error time**				
Setting range (min. value | unit | max. value)			Lenze setting	
0	ms	50	0 ms	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_16

Subindex 6: **Monitoring: Position error - response**			
Selection list(Lenze setting printed in bold)			
0	**No response**		
1	Interference		
2	Warning		
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX			UNSIGNED_8

Both monitoring modes can be switched off via the PDO if, for instance, in case of homing, positioning is to be carried out against a mechanical limit stop.

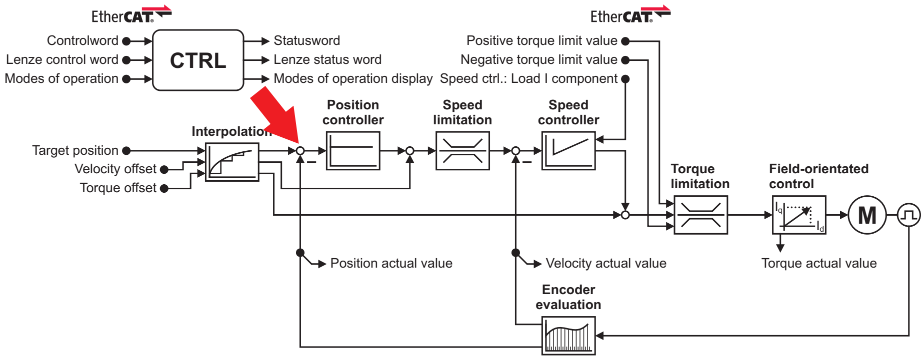
8.9.1 Monitoring of the position error

This function extension is available from software version V01.09.xx onwards!

For the following control modes, the monitoring function of the position error can be used in combination with the CiA402 operating mode "csp":

- Servo control for synchronous motor (SM)
- Servo control for asynchronous motor (ASM)
- V/f characteristic control for asynchronous motor (ASM)

The following error monitoring is effective for an operating mode with position controller. At the input of the position controller (see red arrow in the figure below), the system deviation (i.e. the following error) is compared to the set following error tolerance.



The error response set in the 0x2D51:6 object (or 0x3551:6 for axis B) is executed if

1. the set following error tolerance (0x2D51:4 / 0x3551:4) has been exceeded and
2. the duration of the exceedance is pending uninterruptedly as long as set in the 0x2D51:5 / 0x3551:5 object.

Additional following error monitoring mode in the Controller

According to the manufacturer spanning device profile CiA 402, the following error in the higher-level Controller is monitored in addition to the following error monitoring in the i700 servo inverters. The following error monitoring in the Controller uses its own objects.

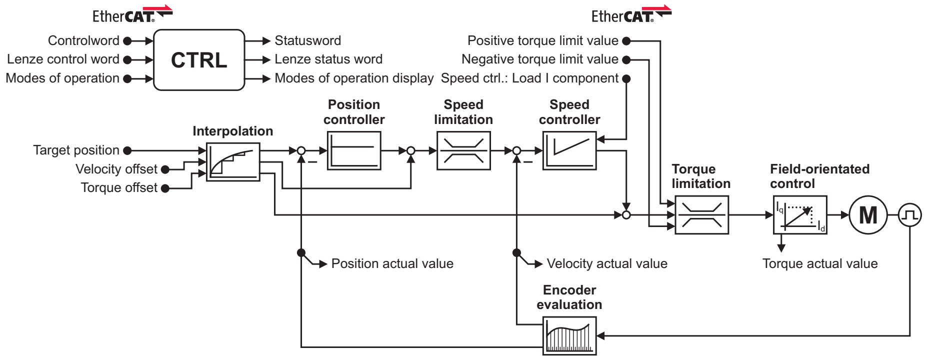
8.9.2 Monitoring of the speed deviation

This function extension is available from software version V01.09.xx onwards!

Monitoring of the speed deviation can only be used in the following control modes:

- Servo control for synchronous motor (SM)
- Servo control for asynchronous motor (ASM)

Monitoring of the speed deviation is effective in the operating modes with contained speed controller. It monitors the system deviation at the input of the speed controller (see blue arrow):



The error response set in the object 0x2D51:3 (or 0x3551:3 for axis B) is executed if

1. the set tolerance of the speed deviation (0x2D51:1 / 0x3551:1) has been exceeded and
2. the duration of the exceedance is pending uninterruptedly as long as set in the 0x2D51:2 / 0x3551:2 object.

9 Diagnostics & error management

This chapter contains information on the drive diagnostics, error handling, and fault analysis.

Objects described in this chapter

Object Axis A	 Axis B	Name	Data type
Identification data			
0x1000		Device: Type	UNSIGNED_32
0x1001		Error memory	UNSIGNED_8
0x1008		ECAT: Manufacturer device name	STRING(50)
0x1009		Device: Hardware version	STRING(50)
0x100A		Device: Software version	STRING(50)
0x1018		ECAT: Identity object	RECORD
Error management			
0x10F3		Diagnostics: History buffer	RECORD
0x2826	0x3026	Quick stop: Duration in case of trouble	UNSIGNED_32
0x2840	0x3040	Delay time: Reset error	INTEGER_32
0x2841	0x3041	Error reset	UNSIGNED_8
0x284F	0x304F	Current fault	ARRAY [0..63] OF BYTE
0x2860	0x3060	PRBS excitation	UNSIGNED_16
0x603F	0x683F	Error code	UNSIGNED_16
0x605E	0x685E	Setting/response in the event of an error	INTEGER_16
Diagnostics parameters			
0x10F8		ECAT DC: Current time	UNSIGNED_64
0x2D81	0x3581	Counter: Operating time	RECORD
0x2D82	0x3582	Motor: Actual voltage - Veff, phase-phase	UNSIGNED_32
0x2D83	0x3583	Motor: Phase currents	RECORD
0x2D8A	0x358A	Speed monitoring: Actual speed error	INTEGER_32

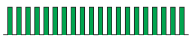
9.1 LED status display

The LED status displays on the front of the i700 servo inverters provide quick information on some operating states.

- The two LEDs "RDY" and "ERR" serve to indicate the device status.

LED	Status	Meaning
RDY	Off	24 V supply voltage missing
		Single axis: Axis is inhibited Double axis: Both axes are inhibited
		Single axis: Axis is enabled Double axis: One or both axes are enabled
ERR	Off	No error.
		Single axis: Device error or axis error Double axis: Device error or axis error in one axis or both axes

- Three green LEDs at the EtherCAT interfaces (RJ45 sockets X4 and X5) serve to indicate the EtherCAT bus status and the connection status of the input and output socket. The arrangement of the LEDs can be obtained from the "i700 servo inverters" hardware manual.

LED	Status	Meaning
RUN	Off	EtherCAT status "Init"
		EtherCAT state "Pre-Operational"
		EtherCAT state "Safe-Operational"
		EtherCAT state "Operational"
		EtherCAT state "Bootstrap"
L/A	Off	No EtherCAT connection
		EtherCAT communication active
		EtherCAT connection available

9.2 Indication of fault and warning (error code)

The indication of fault and warning is implemented according to CiA 301/402:

- In the error-free state, error code "0" is displayed.
- The fault/warning that has occurred first is always displayed.
- Faults overwrite warnings.
- In general, warnings do not lock up the system. If the cause of the warning disappears, the i700 servo inverters resets the warning display and bit 7 in the Statusword (0x6041 or 0x6841 for axis B) after the PDO update has been carried out in the direction of control flow.

0x603F | 0x683F - Error code

Display range (min. value | unit | max. value)			Initialisation	
0x0000		0xFFFF		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX				UNSIGNED_16

Tip!

Chapter "CiA402 error codes / error messages" lists all possible error codes of the i700 servo inverters including causes and possible remedies.

0x284F | 0x304F - Current fault

From software version V01.06.xx onwards
Text display of the current error
- This object contains a reference to the text to be displayed from the ESI file as well as the replacement values for all wildcards in this text.
- The text has the same structure as the diagnostics messages in the history buffer.

Display range (min. value | unit | max. value)			Initialisation	
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX				BYTE

Document "ETG.1020 Protocol Enhancements"

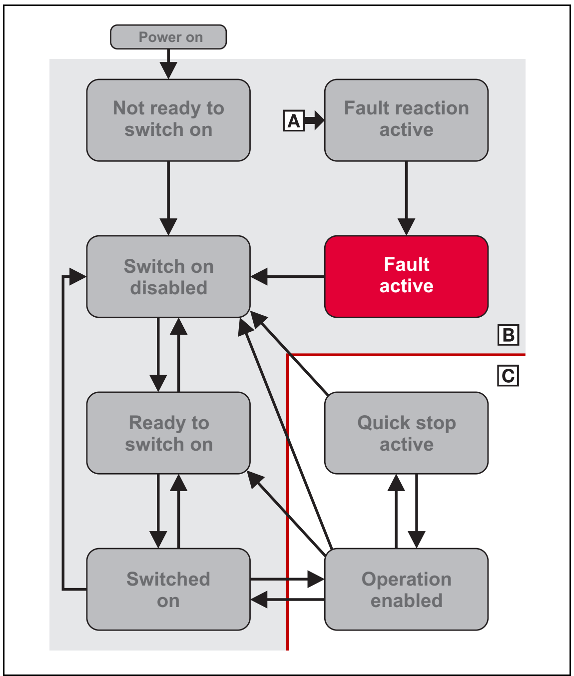
See chapter 13.3 of document "ETG.1020 Protocol Enhancements" provided by the EtherCAT Technology Group (ETG) for detailed information on the structure of the diagnostic messages.

Further indications of faults and warnings

- In the Statusword (0x6041 or 0x6841 for axis B)
  - a fault is indicated via bit 3,
  - a warning is indicated via bit 7.
- Further diagnostics can be carried out with the history buffer (0x10F3). It contains the last 32 messages of the i700 servo inverters.

9.2.1 Response of the device in the event of an error

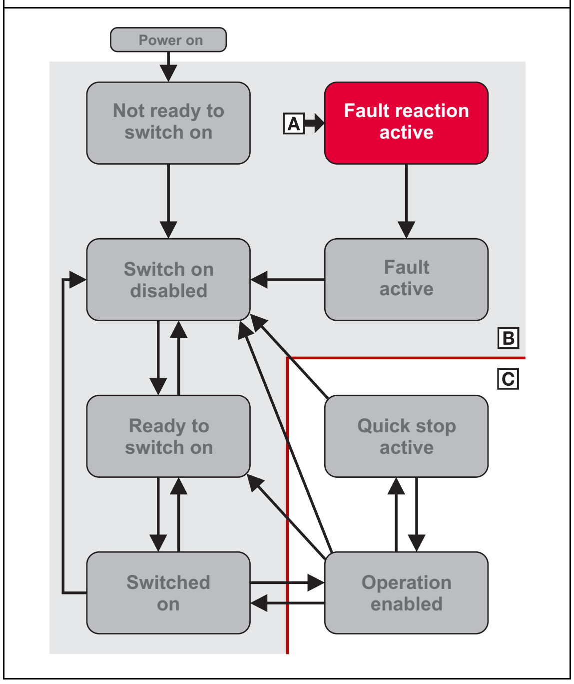
There are two different interference classes in the i700 servo inverters:



Class I: Fatal errors

Evading the "Fault reaction active" device status, errors of class I immediately cause the "Fault" device status.

The i700 servo inverters is switched off immediately (pulse inhibit) so that a guided standstill can only be realised by using a mechanical brake.



Class II: Light errors

In case of errors of class II, the error response selected in the object 0x605E (or 0x685E for axis B) takes place in the Fault reaction active.

This classification enables the i700 servo inverters to actively support the machines to reach standstill when the following light errors occur.

Light errors of class II are:

- Error code 0x4380: Motor temperature sensor
- Error code 0x7303: Error in feedback system (from software version V01.06.xx onwards)
- Error code 0x7380: Hiperface communication error
- Error code 0x8181: EtherCAT communication
- Error code 0x8700: Sync controller

0x605E | 0x685E - Response to error

From software version V01.05.xx onwards
Selection of the response to errors of class II:
-2 = quick stop if possible, otherwise DC current or short-circuit braking.
0 = pulse inhibit
2 = quick stop
Further information on the respective response can be found in the "Info" column further down.

Setting range (min. value | unit | max. value)			Lenze setting	
-32768		32767	-2	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				INTEGER_16

Setting/response in the event of an error		Info
-2	**Quick stop if possible, otherwise DC current or short-circuit braking**	In case of servo control, first quick stop is activated as error response, see selection 2 ▶ Quick stop . If an encoder is detected, ASM servo control immediately switches to the "DC-injection braking" function and SM servo control immediately switches to the "short-circuit braking" function. • This setting is reasonable for drives without service brake and with limited travel way. • The braking is maintained for the entire monitoring time (0x2826 or 0x3026 for axis B) since a standstill of the motor cannot be recognised due to the missing feedback. • Afterwards, the drive changes to the "Fault" device status and inhibits the inverter. If a holding brake is parameterised and connected, it will be applied. • The braking current for DC-injection braking (ASM) has to be set in 0x2B80 (or 0x3380 for axis B).
		Background: The servo control requires a feedback of position/speed signals of the motor encoder. This basic condition also applies when the "quick stop" function is activated. If the i700 servo inverters has no access to this information anymore, it cannot realise a guided operation of the motor into the drift-free standstill. When synchronous motors are operated, the i700 servo inverters loses the current pole position in addition, which can lead to a non-guided behaviour.
2	Quick stop	Quick stop is activated as error response. • The deceleration for quick stop can be set in 0x6085 (or 0x6885 for axis B). • After the deceleration has stopped or the monitoring time for quick stop has elapsed (0x2826 or 0x3026 for axis B), the drive changes to the "Fault" device status and inhibits the inverter. If a holding brake is parameterised and connected, it will be applied.
0	Pulse inhibit	Evading the "Fault reaction active" device status, it is immediately changed to the "Fault" device status. • The drive is immediately switched off (pulse inhibit). • Thus the behaviour is identical to the behaviour in case of fatal errors (class I). Up to and including software version V01.06.3x Triggering the quick stop causes the drive to be stopped along the current limit of 0x6073 / 0x6873. The torque limits from 0x60E0 / 0x68E0, 0x60E1 / 0x68E1 and 0x6072 / 0x6872 are not effective during the quick stop! From software version V01.07.00 onwards During the quick stop, both the current limit 0x6073 and the torque limit 0x6072 are active. The smaller of the two limits determine the output motor torque. The torque limits from 0x60E0 and 0x60E1 have no effect during the quick stop.

0x2826 | 0x3026 - Quick stop: Duration in the event of a fault

If, in the "Fault reaction active" device status, quick stop is not reached within the period set here, the drive changes to the "Fault" device status.

Setting range (min. value | unit | max. value)			Lenze setting	
0	s	100	4 s	
☑ Write access ☐ CINH ☐ OSC ☑ P ☐ RX ☐ TX				UNSIGNED_32

9.2.2 Resetting the error/fault

- A fault can only be reset via bit 7 in the Controlword (0x6040 or 0x6840 for axis B).
  - If there is another fault pending after resetting the fault, bit 3 in the status word remains set and object 0x603F (or 0x683F for axis B) now shows the error code of this fault.
- Certain errors can only be reset by 24 V mains switching. These are, for instance, internal errors due to internal communication faults, initialisation errors, and checksum errors of the firmware or the persistent data (elapsed-hour meters and power-on hour meters, calibration values, etc.).
- Certain errors (e.g. earth fault or short circuit of the motor phases) can cause a delay time. In this case, the fault can only be reset via bit 7 in the Controlword after the delay time has elapsed.
  - Active delay times are displayed via bit 14 in the Lenze status word (0x2831 or 0x3031 for axis B).
  - The remaining delay time can be read out via the following object:

0x2840 | 0x3040 - Delay time: Reset error

Display range (min. value | unit | max. value)			Initialisation	
-2147483648	ms	2147483647	0 ms	
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX				INTEGER_32

0x2841 | 0x3041 - Reset error

Setting range (min. value | unit | max. value)			Lenze setting	
0		1	0	
☑ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX				UNSIGNED_8

9.3 History buffer

Via the history buffer (0x10F3), the controller can access the last 32 messages of the i700 servo inverters.

- The history buffer is saved persistently to the i700 servo inverters.
- The structure of the history buffer corresponds to a ring buffer:
  - As long as there is free history buffer space available, a message is placed in the next free slot in the buffer.
  - If all buffer slots are full, the oldest message is deleted for a new one.
  - The latest messages will always remain available.
- See the "Diag code" (32-bit word) of every single message to find out which axis the message refers to:

Diag code bits 0 ... 15	Diag code bits 16 ... 31
0xE000 ➞ device error/warning	CiA402 error code (device-specific)
0xE001 ➞ fault/warning axis A	CiA402 error code (axis-specific)
0xE002 ➞ fault/warning axis B	CiA402 error code (axis-specific)

▸ CiA402 error codes / error messages (📖 325)

Document "ETG.1020 Protocol Enhancements"

See chapter 13.3 of document "ETG.1020 Protocol Enhancements" provided by the EtherCAT Technology Group (ETG) for detailed information on the structure of the diagnostic messages.

0x10F3 - Diagnostics: History buffer

Sub.	Name	Lenze setting	Data type
▸ 1	Max. number of messages	32	UNSIGNED_8
▸ 2	Latest message	0	UNSIGNED_8
▸ 3	Latest acknowledged message	0	UNSIGNED_8
▸ 4	New active message	0	UNSIGNED_8
▸ 5	Control bits	0	UNSIGNED_16

Subindex 1: **Max. number of messages**				
Maximum number of messages which can be stored in the history buffer (from subindex 6).				
Display range (min. value | unit | max. value)			Initialisation	
0		255	32	
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX				UNSIGNED_8

Subindex 2: **Latest message**				
Subindex of the most recent message				
Display range (min. value | unit | max. value)			Initialisation	
0		255	0	
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX				UNSIGNED_8

Subindex 3: **Latest acknowledged message**				
Subindex of the most recent message acknowledged by the EtherCAT master.				
Setting range (min. value | unit | max. value)			Lenze setting	
0		255	0	
☑ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX				UNSIGNED_8

Subindex 4: **New active message**				
TRUE if messages are pending that have not been acknowledged by the EtherCAT master yet.				
Display range (min. value | unit | max. value)			Initialisation	
0		255	0	
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☑ TX				UNSIGNED_8

Subindex 5: **Control bits**				
Settings for the transmission and storage of the messages				
Setting range (min. value | unit | max. value)			Lenze setting	
0		65535	0	
☑ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX				UNSIGNED_16

9.4 CiA402 error codes / error messages

Short overview

The table below contains all error messages of the inverter operating system in numerical order.

Error number dec representation	Error number hex representation	Error message
8992	0x2320	Short circuit or earth leakage at the motor end
9024	0x2340	Short circuit at the motor end
9041	0x2351	Motor utilisation (I^2xt) > [value] %
9088	0x2380	Fault - utilisation of the power section (Ixt) too high
9089	0x2381	Warning - utilisation of the power section (Ixt) too high
9090	0x2382	Fault - device utilisation (Ixt) too high
9091	0x2383	Warning - device utilisation (Ixt) too high
9092	0x2384	Ultimate motor current reached
9093	0x2385	Output current > [value] A (maximum device current)
9094	0x2386	Clamp is active
9095	0x2387	Clamp responded too often
9104	0x2390	Fault - utilisation of the power section (Ixt) too high
9105	0x2391	Warning - utilisation of the power section (Ixt) too high
9106	0x2392	Fault - device utilisation (Ixt) too high
9107	0x2393	Warning - device utilisation (Ixt) too high
12816	0x3210	DC bus - overvoltage [value] V
12832	0x3220	DC bus - undervoltage [value] V
16912	0x4210	Module temperature too high, [value]
17024	0x4280	Internal fault - module temperature monitoring
17168	0x4310	Motor temperature too high, [value]
17280	0x4380	Motor temperature sensor
20754	0x5112	24 V supply
24592	0x6010	Watchdog reset
24704	0x6080	The new firmware is incompatible
25360	0x6310	Incorrect parameter set download
25376	0x6320	Parameter error in object [value]
28961	0x7121	PLI - motor blocked
29443	0x7303	Error in feedback system
29568	0x7380	Hiperface communication error
29569	0x7381	Hiperface: Wrong absolute value - motor speed too high
33152	0x8180	ECAT DC - synchronisation required
33153	0x8181	EtherCAT communication
33408	0x8280	Sync Manager - address [value]
33409	0x8281	Sync Manager - current size [value]
33410	0x8282	Sync Manager - settings [value]
33411	0x8283	PDO mapping: Object unknown (index [value])
33412	0x8284	No PDO mapping object (index [value])
33413	0x8285	Too many mapped objects (max. number = [value])
33414	0x8286	PDO mapping - error
34560	0x8700	Sync controller
34561	0x8701	Time-out during synchronisation with EtherCAT sync signal
65280	0xff00	Fatal internal error
65281	0xff01	Fatal internal communication error, cycle [value]
65282	0xff02	Brake
65283	0xff03	Fatal internal error overflow task
65284	0xff04	PLI - motor movement too large

Error number dec representation	Error number hex representation	Error message
65285	0xff05	STO inhibited
65286	0xff06	Max. speed reached
65287	0xff07	Impermissible during identification or in test mode
65288	0xff08	Impossible during identification
65289	0xff09	Motor phase failure
65290	0xff0a	Motor phase failure - phase U
65291	0xff0b	Motor phase failure - phase V
65292	0xff0c	Motor phase failure - phase W
65293	0xff0d	No resolver connected --> command cannot be executed.
65294	0xff0e	Speed too slow --> command cannot be executed
65295	0xff0f	No resolver connected --> command cannot be executed
65296	0xff10	Time-out
65297	0xff11	Numerical problem
65298	0xff12	Inverter error too big
65299	0xff13	Identification aborted
65300	0xff14	Not possible in the 'operation enabled' or 'quick stop active' state
65301	0xff15	No Hiperface absolute value encoder connected --> command cannot be executed
65302	0xff16	Connected Hiperface absolute value encoder is not supported
65303	0xff17	Connected Hiperface absolute value encoder is not supported --> Command cannot be executed
65304	0xff18	Communication timeout in manual mode control
65305	0xff19	Internal error during identification
65312	0xff20	The check of the Replacement protection has failed
65313	0xff21	Timeout - holding brake feedforward control
65314	0xff22	Speed error occurred
65315	0xff23	Position error occurred
65316	0xff24	Sync0 cycle time is invalid
65317	0xff25	The hardware is not supported
65318	0xff26	Cable Check - motor movement too great
65319	0xff27	Cable Check - incorrect direction of rotation
65320	0xFF28	Master password entered incorrectly 4 times

Note!

- When an error message is output, the source (device, axis A or axis B) is preceded.
- This data [value] of an error message text serves as wildcard for the value detected during the runtime of the i700 servo inverters.
- The indices for axis A and axis B are given directly in succession in the following descriptions for a better readability (index of axis A | index of axis B)

Error code 0x0000: No error

Text ID: 0x00

Cause	Remedies	Response
-	-	-

Error code 0x2320: Short circuit or earth leakage at the motor end

Text ID: 0x04

Cause	Remedies	Response
• Earth fault in motor cable. • Excessive capacitive charging current in the motor cable.	• Check motor cable. • Use motor cable that is shorter or has a lower capacitance.	Interference **Note:** Error reset only possible after 5 s.

Error code 0x2340: Short circuit at the motor end

Text ID: 0x05

Cause	Remedies	Response
• Short circuit/earth fault in motor cable. • Excessive capacitive charging current in the motor cable.	• Check motor cable. • Use motor cable that is shorter or has a lower capacitance.	Interference **Note:** Error reset only possible after 5 s.

Error code 0x2351: Motor utilisation (I^2xt) > [value] %

Text ID: 0x38

Cause	Remedies	Response
Motor is thermally overloaded, e.g. due to: • impermissible continuous current. • Frequent or too long acceleration processes.	• Check drive dimensioning. • Check setting or the warning or error threshold.	Warning • Warning threshold can be adapted in 0x2D4E | 0x354E. From software version V01.05.xx onwards, an error response can be parameterised in 0x2D50:1 | 0x3550:1. • Error threshold can be adapted in 0x2D50:2 | 0x3550:2.

▸ Monitoring of the motor utilisation (I^2xt)

Error code 0x2380: Fault - utilisation of the power section (Ixt) too high

Text ID: 0x06

Cause	Remedies	Response
Frequent and too long acceleration processes with overcurrent.	Check drive dimensioning.	Interference

Error code 0x2381: Warning - utilisation of the power section (Ixt) too high

Text ID: 0x07

Cause	Remedies	Response
Frequent and too long acceleration processes with overcurrent.	Check drive dimensioning.	Warning

Error code 0x2382: Fault - device utilisation (Ixt) too high

Text ID: 0x08

Cause	Remedies	Response
Frequent and too long acceleration processes with overcurrent.	Check drive dimensioning.	Interference

Error code 0x2383: Warning - device utilisation (Ixt) too high

Text ID: 0x09

Cause	Remedies	Response
Frequent and too long acceleration processes with overcurrent.	Check drive dimensioning.	Warning

Error code 0x2384: Ultimate motor current reached

Text ID: 0x22

Cause	Remedies	Response
The motor current has exceeded the "ultimate" motor current. The "ultimate" motor current corresponds to the current level at which the permanent magnets of a synchronous motor suffer permanent damage.	Reduce maximum current (0x6073 | 0x6873) and/or adjust the current controller (0x2942 | 0x3142) in order to reduce the current overshoot.	Fault (adjustable in 0x2D46:2 | 0x3546:2) • Threshold can be adapted in 0x2D46:1 | 0x3546:1.

Error code 0x2385: Output current > [value] A (maximum device current)

Text ID: 0x24

Cause	Remedies	Response
The desired maximum current (0x6073 | 0x6873) is higher than the maximum current of the axis (display in 0x2DDF:2 | 0x35DF:2).	The permissible maximum current (0x6073 | 0x6873) is: $\frac{\text{Maximum axis current}}{\text{Rated motor current}}$ • Setting of rated motor current in 0x6075 | 0x6875.	Warning

Error code 0x2386: Clamp is active

Text ID: 0x2F

Cause	Remedies	Response
Maximum current of the axis (display in 0x2DDF:2 | 0x35DF:2) has been reached.	• Select a flatter speed ramp. • Reduce the load. • Set the Imax controller (0x2B08 | 0x3308) more dynamically.	Warning

Error code 0x2387: Clamp responded too often

Text ID: 0x30

Cause	Remedies	Response
Maximum current of the axis (display in 0x2DDF:2 | 0x35DF:2) was reached too often in a row.	• Select a flatter speed ramp. • Reduce the load. • Set the Imax controller (0x2B08 | 0x3308) more dynamically.	Interference

Error code 0x2390: Fault - utilisation of the power section (Ixt) too high

Text ID: 0x4B

Cause	Remedies	Response
Frequent and too long acceleration processes with overcurrent in combination of both axes.	Check drive sizing of both axes.	Error

Error code 0x2391: Warning - utilisation of the power section (Ixt) too high

Text ID: 0x4C

Cause	Remedies	Response
Frequent and too long acceleration processes with overcurrent in combination of both axes.	Check drive sizing of both axes.	Warning

Error code 0x2392: Fault - device utilisation (Ixt) too high

Text ID: 0x4D

Cause	Remedies	Response
Frequent and too long acceleration processes with overcurrent in combination of both axes.	Check drive sizing of both axes.	Error

Error code 0x2393: Warning - device utilisation (Ixt) too high

Text ID: 0x4E

Cause	Remedies	Response
Frequent and too long acceleration processes with overcurrent in combination of both axes.	Check drive sizing of both axes.	Warning

Error code 0x3210: DC bus - overvoltage [value] V

Text ID: 0x0A
Delay time: 5 ms

Cause	Remedies	Response
Due to a too high braking energy, the DC-bus voltage exceeds the overvoltage threshold which results from the mains voltage setting.	• Connect brake resistor to power supply module or use regenerative module. • Check mains setting. • Check drive profile.	Warning or fault (depending on the height of the voltage) • Warning threshold adaptable in 0x2540:5. • Fixed error threshold.

Error code 0x3220: DC bus - undervoltage [value] V

Text ID: 0x0B

Cause	Remedies	Response
DC bus voltage is lower than the undervoltage threshold resulting from the mains setting.	• Check mains voltage. • Check DC-bus voltage. • Check mains settings.	Warning or fault (depending on the height of the voltage) • Warning threshold adaptable in 0x2540:2. • Fixed error threshold.

Error code 0x4210: Module temperature too high, [value]

Text ID: 0x0C

Cause	Remedies	Response
Heatsink temperature higher than fixed temperature limit (100 °C): • i700 servo inverters ambient temperature too high. • Dirty fan or ventilation slots. • Fan defective. • Warning threshold set too low.	• Check control cabinet temperature. • Clean filter. • Clean i700 servo inverters. • Replace fan.	Warning or fault (depending on the height of the temperature) • Warning threshold can be adapted in 0x2D84:2 | 0x3584:2. • Fixed error threshold.

▸ Monitoring of the heatsink temperature

Error code 0x4280: Internal fault - module temperature monitoring

Text ID: 0x26

Cause	Remedies	Response
Thermal sensor error	• 24 V mains switching required. • If the error occurs frequently, please contact Lenze.	Interference

Error code 0x4310: Motor temperature too high, [value]

Text ID: 0x0D

Cause	Remedies	Response
Motor temperature higher than variable temperature limit: • Motor too hot due to impermissibly high currents or frequent and too long acceleration processes. • No KTY connected. • Error or warning threshold set too low.	• Check drive dimensioning. • Connect KTY or switch off monitoring. • Set a higher warning or error threshold.	Fault (adjustable in 0x2D49:2 | 0x3549:2) • Warning threshold can be adapted in 0x2D49:3 | 0x3549:3. • Error threshold adaptable in 0x2D49:4 | 0x3549:4.

▸ Motor temperature monitoring

Error code 0x4380: Motor temperature sensor

Text ID: 0x25

Cause	Remedies	Response
Motor temperature sensor	• Connect KTY or switch off monitoring. • Set a higher warning or error threshold.	Class II fault (adjustable in 0x2D49:2 | 0x3549:2) • If "fault" is set, the response set in 0x605E | 0x685E takes place. • Warning threshold can be adapted in 0x2D49:3 | 0x3549:3. • Error threshold adaptable in 0x2D49:4 | 0x3549:4.

▸ Motor temperature monitoring

Error code 0x5112: 24 V supply

Text ID: 0x0E

Cause	Remedies	Response
24 V supply has failed or has fallen below the warning threshold (21.45 V) or error threshold (18.15 V).	Check 24 V voltage supply.	Warning or fault (depending on the height of the voltage) • Thresholds are fixed.

Error code 0x6010: Watchdog reset

Text ID: 0x0F

Cause	Remedies	Response
• Reset was triggered via device command 1100. • Internal error	If the reset was not triggered via the device command, consult Lenze if the error occurs frequently.	Warning if restart of the drive is requested via command. Otherwise fault.

Error code 0x6080: The new firmware is incompatible

Text ID: 0x48

Cause	Remedies	Response
• The redownloaded firmware is incompatible to the one on the device. • The new firmware has not been accepted.	• Download a compatible firmware version (same major version). • If the current firmware shall be maintained, the error can simply be acknowledged.	Error

Error code 0x6310: Incorrect parameter set download

Text ID: 0x10

Cause	Remedies	Response
Error when downloading a parameter set: • CRC error in the parameter set detected. • Vendor, product code or revision of the object directory are different. • Unknown parameter set.	• Repeat parameter set download. • Recreate parameter set.	Warning if single parameters cannot be imported. Fault if all parameters cannot be imported.

Error code 0x6320: Parameter error in object [value]

Text ID: 0x11

Cause	Remedies	Response
Unknown index in the parameter set.	Recreate parameter set.	Warning

Error code 0x7121: PLI - motor blocked

Text ID: 0x12

Cause	Remedies	Response
An error occurred during the pole position identification. The pole position identification could not be completed successfully: • Too many deviations during the identification. • Motor is blocked.	• Check whether all requirements for an identification of the pole position are fulfilled. • Ensure that the machine is not braked or blocked during the pole position identification. • Repeat the pole position identification.	Fault (adjustable in 0x2C60 | 0x3460)

▶ Synchronous motor (SM): Pole position identification

Error code 0x7303: Error in feedback system

Text ID: 0x13

Cause	Remedies	Response
• Encoder cable interrupted. • Encoder defective.	• Check encoder cable. • Check encoder. • Switch off monitoring if feedback is not used.	Warning in the states "Not ready to switch on" and "Switch on disabled". Otherwise class II fault (adjustable in 0x2C45 | 0x3445) • If "fault" is set, the response set in 0x605E | 0x685E takes place.

▸ Encoder open-circuit monitoring

Error code 0x7380: Hiperface communication error

Text ID: 0x3A

Cause	Remedies	Response
Communication with HIPERFACE® absolute value encoder is disturbed.	• Check the supply voltage of the encoder (0x2C42:2 | 0x3442:2). • Check wiring to the encoder.	Up to and including software version V01.05.xx: Fault (adjustable in 0x2C41:4 | 0x3441:4)
		From software version V01.06.xx onwards: Class II fault (adjustable in 0x2C41:4 | 0x3441:4) • If "fault" is set, the response set in 0x605E | 0x685E takes place.

Error code 0x7381: Hiperface: Wrong absolute value - motor speed too high

Text ID: 0x41

Cause	Remedies	Response
The absolute position of the HIPERFACE® absolute value encoder cannot be accepted as the motor rotates too fast.	Let the motor coast.	Fault (adjustable in 0x2C41:4 | 0x3441:4)

Error code 0x8180: ECAT DC - synchronisation required

Text ID: 0x15

Cause	Remedies	Response
EtherCAT DC mode not activated. (DC = Distributed Clock)	Activate DC mode.	Warning

Error code 0x8181: EtherCAT communication

Text ID: 0x19

Cause	Remedies	Response
EtherCAT communication error	• Check EtherCAT bus cabling. • Select a longer PDI watchdog time.	Class II fault • The response set in 0x605E | 0x685E takes place.

Error code 0x8280: Sync Manager - address [value]

Text ID: 0x16

Cause	Remedies	Response
Incorrect Sync Manager setting of the EtherCAT master.	• Check the devices on the bus. • Check cabling or sequence of the devices on the bus. • Remove the i700 servo inverter from the project tree and rescan the bus. • Check the device description of the i700 servo inverter.	Interference

Error code 0x8281: Sync Manager - current size [value]

Text ID: 0x17

Cause	Remedies	Response
Incorrect Sync Manager setting of the EtherCAT master.	• Check the devices on the bus. • Check cabling or sequence of the devices on the bus. • Remove the i700 servo inverter from the project tree and rescan the bus. • Check the device description of the i700 servo inverter.	Interference

Error code 0x8282: Sync Manager - settings [value]

Text ID: 0x18

Cause	Remedies	Response
Incorrect Sync Manager setting of the EtherCAT master.	• Check the devices on the bus. • Check cabling or sequence of the devices on the bus. • Remove the i700 servo inverter from the project tree and rescan the bus. • Check the device description of the i700 servo inverter.	Interference

Error code 0x8283: PDO mapping: Object unknown (index [value])

Text ID: 0x2B

Cause	Remedies	Response
Unknown index in the PDO mapping.	Check PDO mapping.	Interference

Error code 0x8284: No PDO mapping object (index [value])

Text ID: 0x2C

Cause	Remedies	Response
Index cannot be mapped as PDO.	Check PDO mapping.	Interference

Error code 0x8285: Too many objects mapped (max. number = [value])

Text ID: 0x2D

Cause	Remedies	Response
Max. number of PDO mapping objects has been exceeded (> 64).	Reduce the number of mapping objects in the "free mapping".	Interference

Error code 0x8286: PDO mapping - error

Text ID: 0x2E

Cause	Remedies	Response
Incorrect PDO mapping.	Check PDO mapping.	Interference

Error code 0x8700: Sync controller

Text ID: 0x14

Cause	Remedies	Response
• EtherCAT Sync0 signal has failed. • PDO transmitted or received too late.	• Check in EtherCAT master configuration whether DC mode is set. • Restart EtherCAT configuration. • Select a later instant of reception, outside the user shift time. • Sync Error Counter - increase monitoring limit (0x10F1:2).	Class II fault • The response set in 0x605E | 0x685E takes place.

Error code 0x8701: Time-out during synchronisation with EtherCAT sync signal

Text ID: 0x39

Cause	Remedies	Response
During the EtherCAT state change from "Pre-Operational" to "Safe-Operational": • EtherCAT Sync0 signal has failed. • Jitter of the EtherCAT Sync0 signal is too high.	• Check EtherCAT master configuration for DC deviation. • Restart EtherCAT configuration.	Interference

Error code 0xFF00: Fatal internal error

Text ID: 0x1A

Cause	Remedies	Response
Internal error	• 24 V mains switching required. • If the error occurs frequently, please contact Lenze.	Interference

Error code 0xFF01: Fatal internal communication error, cycle [value]

Text ID: 0x1B

Cause	Remedies	Response
Internal error	• 24 V mains switching required. • If the error occurs frequently, please contact Lenze.	Interference

Error code 0xFF02: Brake

Text ID: 0x1C

Cause	Remedies	Response
Brake error due to short circuit or cable break.	Check brake cable or brake.	Warning in "manual control" mode. Otherwise class II fault (adjustable in 0x2820:6 | 0x3020:6) • If "fault" is set, the response set in 0x605E | 0x685E takes place.

Error code 0xFF03: Fatal internal error overflow task

Text ID: 0x1D

Cause	Remedies	Response
Internal error	• 24 V mains switching required. • If the error occurs frequently, please contact Lenze.	Interference

Error code 0xFF04: PLI - motor movement too large

Text ID: 0x1E

Cause	Remedies	Response
Motor moved too much during pole position identification.	• Check settings of the pole position identification in 0x2C62 | 0x3462. • If this error takes place in pole position identification without any motion, the motor has to be stalled.	Fault (adjustable in 0x2C60 | 0x3460)

▶ Synchronous motor (SM): Pole position identification

Error code 0xFF05: STO inhibited

Text ID: 0x1F

Cause	Remedies	Response
The servo inverter is inhibited via STO terminals, despite controller enable via the control word.	• Deactivate STO or inhibit the servo inverter. • Check the cabling of the STO terminals.	Warning with change to the "Switch on disabled" device status.

Error code 0xFF06: Max. speed reached

Text ID: 0x20

Cause	Remedies	Response
Motor speed too high.	• Reduce motor speed. • Check resolver or cabling.	Fault (adjustable in 0x2D44:2 | 0x3544:2) • Threshold can be adapted in 0x2D44:1 | 0x3544:1.

Error code 0xFF07: Impermissible during identification or in test mode

Text ID: 0x21

Cause	Remedies	Response
A parameter was attempted to change which influences the identification in progress or the currently activated test mode.	Postpone changing the parameter until the identification is completed or the test mode is deactivated again.	Warning

Error code 0xFF08: Impossible during identification

Text ID: 0x23

Cause	Remedies	Response
A parameter was attempted to change which influences the identification in progress.	Postpone changing the parameter until the identification is completed.	Warning

Error code 0xFF09: Motor phase failure

Text ID: 0x27

Cause	Remedies	Response
Multiple motor phases are not connected.	• Correct the wiring between the servo inverter and the motor. • In the case of a false alarm, increase the threshold values for monitoring in 0x2D45 | 0x3545.	Default setting: No response Monitoring takes place before and during operation. Both monitoring modes can trigger an error. For parameter setting of the two monitoring modes, see 0x2D45 | 0x3545.

Motor phase failure monitoring

Error code 0xFF0A: Motor phase failure phase U

Text ID: 0x28

Cause	Remedies	Response
Motor phase U is not connected.	• Correct the wiring between the servo inverter and the motor. • In the case of a false alarm, increase the threshold values for monitoring in 0x2D45 | 0x3545.	Default setting: No response Monitoring takes place before and during operation. Both monitoring modes can trigger an error. For parameter setting of the two monitoring modes, see 0x2D45 | 0x3545.

Motor phase failure monitoring

Error code 0xFF0B: Motor phase failure phase V

Text ID: 0x29

Cause	Remedies	Response
Motor phase V is not connected.	• Correct the wiring between the servo inverter and the motor. • In the case of a false alarm, increase the threshold values for monitoring in 0x2D45 | 0x3545.	Default setting: No response Monitoring takes place before and during operation. Both monitoring modes can trigger an error. For parameter setting of the two monitoring modes, see 0x2D45 | 0x3545.

Motor phase failure monitoring

Error code 0xFF0C: Motor phase failure phase W

Text ID: 0x2A

Cause	Remedies	Response
Motor phase W is not connected.	• Correct the wiring between i700 servo inverters and motor. • In the case of a false alarm, increase the threshold values for monitoring in 0x2D45 | 0x3545.	Default setting: No response Monitoring takes place before and during operation. Both monitoring modes can trigger an error. For parameter setting of the two monitoring modes, see 0x2D45 | 0x3545.

Motor phase failure monitoring

Error code 0xFF0D: No resolver connected --> command cannot be executed.

Text ID: 0x31

Cause	Remedies	Response
The identification of a resolver error was attempted during V/f characteristic control without feedback system.	The resolver error can only be identified if the resolver is used as feedback system during servo control.	Warning

Error code 0xFF0E: Speed too low --> command cannot be executed.

Text ID: 0x32

Cause	Remedies	Response
The actual speed is too low for the resolver error to be identified.	Repeat identification at higher speed.	Warning

Error code 0xFF0F: No resolver connected --> command cannot be executed.

Text ID: 0x33

Cause	Remedies	Response
The identification of a resolver error was attempted but an encoder is used as feedback system.	Identification is only possible with devices with resolver feedback.	Warning

Error code 0xFF10: Time-out

Text ID: 0x34

Cause	Remedies	Response
The actual speed is too low for the resolver error to be identified.	Repeat identification at higher speed.	Warning

Error code 0xFF11: Numerical problem

Text ID: 0x35

Cause	Remedies	Response
The actual speed is too low for the resolver error to be identified.	Repeat identification at higher speed.	Warning

Error code 0xFF12: Inverter error too big

Text ID: 0x36

Cause	Remedies	Response
The results of the inverter error identification are not plausible.	Check shielding of the motor cable. Connect the shield with a surface as large as possible.	Warning

Error code 0xFF13: Identification aborted

Text ID: 0x37

Cause	Remedies	Response
A started identification was aborted due to an error or a controller inhibit.	Repeat the identification and avoid the cause of the abort.	Faulty pole position identification (adjustable in 0x2C60 | 0x3460) Otherwise warning

Error code 0xFF14: Impossible in the 'operation enabled' or 'quick stop active' state

Text ID: 0x3B

Cause	Remedies	Response
Reading out and accepting the hiperface data is only permitted when the controller is inhibited.	Inhibit controller.	Warning

Error code 0xFF15: No Hiperface absolute value encoder connected --> command cannot be executed

Text ID: 0x3C

Cause	Remedies	Response
When the "resolver" device version or the sin/cos encoder has been selected, an attempt is being made to read the data from a HIPERFACE® absolute value encoder.	The command is only possible with the "encoder" device version. Moreover, the "2: Hiperface absolute value encoder" selection has to be set in 0x2C40 | 0x3440.	Warning

Error code 0xFF16: Connected Hiperface absolute value encoder is not supported

Text ID: 0x3D

Cause	Remedies	Response
The HIPERFACE® absolute value encoder has a type code which is neither stored in the firmware nor corresponds to the type code set in 0x2C41:2 | 0x3441:2.	Set type code in 0x2C41:2 | 0x3441:2.	Fault (adjustable in 0x2C41:4 | 0x3441:4)

▸ Additional settings for SinCos absolute value encoders with HIPERFACE® protocol

Error code 0xFF17: Connected Hiperface absolute value encoder is not supported --> Command cannot be executed

Text ID: 0x3E

Cause	Remedies	Response
The HIPERFACE® absolute value encoder is not supported by the device. Hence, no data can be detected.	Connect a HIPERFACE® absolute value encoder which is supported by the device.	Warning

▸ Supported encoder types with HIPERFACE® protocol

Error code 0xFF18: Communication time-out in manual control mode

Text ID: 0x3F

Cause	Remedies	Response
In the "manual control" mode, writing the setpoint data failed to appear for a longer period of time than set in 0x2836:5 | 0x3036:5.	• Check connection between device, Controller and »EASY Starter« or »PLC Designer«. • Increase the time period in 0x2836:5 | 0x3036:5. • Restart manual jog dialog.	Interference

▸ Manual control

Error code 0xFF19: Internal error during identification

Text ID: 0x40

Cause	Remedies	Response
Internal error during identification.	Please contact Lenze.	Warning

Error code 0xFF20: The check of the Replacement Protection has failed

Text ID: 0x42

Cause	Remedies	Response
The check of the Replacement Protection has failed.	• Enter correct Replacement Protection ID. • Replace device by an OEM device. • Reprogram Replacement Protection ID.	Warning

Error code 0xFF21: Time-out - holding brake feedforward control

Text ID: 0x43

Cause	Remedies	Response
The given feedforward control torque could not be built up within one second.	-	Error

Error code 0xFF22: Speed error occurred

Text ID: 0x44

Cause	Remedies	Response
The speed error exceeds the limit from 0x2D51:1 / 0x3551:1 for a time period > 0x2D51:2 / 0x3551:2.	• Increase time-out in 0x2D51:2 or 0x3551:2. • Increase limit in 0x2D51:1 or 0x3551:1. • Increase speed controller gain. • Increase torque and/or current limit.	Error

Error code 0xFF23: Position error occurred

Text ID: 0x45

Cause	Remedies	Response
The check of the Replacement Protection has failed.	• Increase time-out in 0x2D51:4 or 0x3551:4. • Increase limit in 0x2D51:2 or 0x3551:2. • Increase position controller gain.	Error

Error code 0xFF24: Sync0 cycle time is invalid

Text ID: 0x46

Cause	Remedies	Response
The requested Sync0 cycle on the Ethercat is not within the limits 125 µs ≤ x ≤ 10 ms anymore supported by this device.	Select another Sync0 cycle in the control for this bus interconnection that is within the limits 125 µs ≤ x ≤ 10 ms.	Error

Error code 0xFF25: The hardware is not supported

Text ID: 0x47

Cause	Remedies	Response
Internal error	Exchange software with older version (< 1.10.xx).	Error

Error code 0xFF26: Cable Check - motor movement too great

Text ID: 0x49

Cause	Remedies	Response
During the Cable Check, the motor "jumps" to a new position.		Error
• Phase shifter in the motor cable.	• Swap two phases of the motor cable.	
• Axis was moved manually.	• Stop axis.	
• Motor parameters (number of pole pairs, motor type) are not set correctly.	• Set motor parameters (number of pole pairs, motor type) correctly.	
• Synchronous machine: the pole position is set incorrectly	• Synchronous machine: set pole position correctly.	
• Motor with brake: the brake was released early by an external source	• Motor with brake: Prevent the brake to be released by an external source.	

Error code 0xFF27: Cable Check - incorrect direction of rotation

Text ID: 0x4A

Cause	Remedies	Response
During the wiring test, the motor cannot follow the specified angle.		Error
• Motor and encoder cable do not lead to the same motor.	Up to and including Softwarestand V01.10.xx / V02.10.xx: • Identify a common motor and encoder cable. From software version V02.11.xx onwards: • Switch off the current supply and correct the wiring. • Then switch on the voltage supply and execute the **Cable Check** function again.	
• Phase shifter in the motor cable.	Up to and including Softwarestand V01.10.xx / V02.10.xx: • Switch off the current supply. • Swap two phases of the motor cable. From software version V02.11.xx onwards: • Switch off the current supply. • Swap two phases of the motor cable. • Then switch on the voltage supply and execute the **Cable Check** function again.	
• Missing motor phase(s)	• Check motor phase(s)	
• Motor without brake: • Test current is not high enough. • Machine is blocked.	• Check mechanics.	
• Motor with brake: • The brake is so strong that no movement is detected despite a correct wiring.	• Check mechanics. • Ensure that the motor can rotate freely.	
• Incorrect pole position	• Switch off the current supply and correct the wiring. • Then switch on the current supply and carry out the pole position identification. • Then execute the **Cable Check** function again.	
• Motor and encoder data are possibly incorrect	• Compare the data of the parameter with the data of the motor nameplate and correct the data in the parameters, if necessary.	

Error code 0xFF28: Master password entered incorrectly 4 times

Text ID: 0x4F

Cause	Remedies	Response
An incorrect master password was used for the log-in too frequently.	• Use the correct master password for the log-in. • Switch mains to delete the warning.	Warning

9.5 Identification data

0x1000 - Device: Type

Display range (min. value | unit | max. value)			Initialisation	
0x00000000		0xFFFFFFFF	0x02020192	
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX				UNSIGNED_32

0x1001 - Error memory

The error memory displays the error cause in a bit coded manner. It is also possible that several errors are pending at the same time.

Display range (min. value | unit | max. value)			Initialisation	
0		255	0	
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☑ TX				UNSIGNED_8

0x1008 - ECAT: Manufacturer's device name

☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX		STRING(50)

0x1009 - Device: Hardware version

☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX		STRING(50)

0x100A - Device: Software version

☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX		STRING(50)

0x1018 - ECAT: Identification data

Note:
The fixed identification data depend on the device version of the i700 servo inverter.
Examples of identification data for a double axis are provided below.

Sub.	Name	Lenze setting	Data type
1	ECAT: Vendor ID	59	UNSIGNED_32
2	ECAT: Product code	0x69070002: i700 (Double Inverter, Safety STO)	UNSIGNED_32
3	ECAT: ESI revision	131085	UNSIGNED_32
4	ECAT: Serial number		UNSIGNED_32
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

9.6 Diagnostics parameters

0x10F8 - ECAT DC: Current time

Display of the time information the i700 servo inverter is currently using (time of the device if you will).
▸ Real-time information (Distributed Clock)

Display range (min. value | unit | max. value)			Initialisation	
0	ns	2^{64}-1		
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☑ TX				UNSIGNED_64

0x2D81 | 0x3581 - Counter: Operating time

Sub.	Name	Lenze setting	Data type
▸ 1	Device: Operating time		UNSIGNED_32
▸ 2	Device: Power-on time		UNSIGNED_32

Subindex 1: **Device: Operating time**				
Display of the seconds the i700 servo inverter has been operated (device status "Operation enabled") so far.				
Display range (min. value | unit | max. value)			Initialisation	
0	s	4294967295		
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX				UNSIGNED_32

Subindex 2: **Device: Power-on time**				
Display of the seconds the i700 servo inverter has been switched on so far.				
Display range (min. value | unit | max. value)			Initialisation	
0	s	4294967295		
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX				UNSIGNED_32

0x2D82 | 0x3582 - motor: Actual voltage - Veff, phase-phase

Display of the current motor voltage

Display range (min. value | unit | max. value)			Initialisation	
0.0	V	429496729.5	0.0 V	
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: 1/10	UNSIGNED_32

0x2D83 | 0x3583 - Motor: Phase currents

Display of the current motor current of each individual motor phase

Sub.	Name	Lenze setting	Data type
▸ 1	Zero system current		INTEGER_32
▸ 2	Current - phase U		INTEGER_32
▸ 3	Current - phase V		INTEGER_32
▸ 4	Current - phase W		INTEGER_32

Subindex 1: **Zero system current**				
Display range (min. value | unit | max. value)			Initialisation	
-21474836.48	A	21474836.47		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: 1/100	INTEGER_32

Subindex 2: **Current - phase U**				
Display range (min. value | unit | max. value)			Initialisation	
-21474836.48	A	21474836.47		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: 1/100	INTEGER_32

Subindex 3: **Current - phase V**				
Display range (min. value | unit | max. value)			Initialisation	
-21474836.48	A	21474836.47		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: 1/100	INTEGER_32

Subindex 4: **Current - phase W**				
Display range (min. value | unit | max. value)			Initialisation	
-21474836.48	A	21474836.47		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: 1/100	INTEGER_32

0x2D8A | 0x358A - Speed monitoring: Current speed error

Display range (min. value | unit | max. value)			Initialisation	
-480000	r/min	480000		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX				INTEGER_32

10 Service/internal

Objects described in this chapter

Object		Name	Data type
Axis A	Axis B		
0x2DD0	0x35D0	Field: Values	RECORD
0x2DD1	0x35D1	Motor: Currents	RECORD
0x2DD2	0x35D2	Position: Target position interpolated	INTEGER_32
0x2DD3	0x35D3	Target speeds	RECORD
0x2DD4	0x35D4	Speed controller: Output signal	RECORD
0x2DD5	0x35D5	Target torque	INTEGER_32
0x2DD6	0x35D6	Torque: Filter cascade	RECORD
0x2DD7	0x35D7	Voltage values	RECORD
0x2DDC	0x35DC	Slip: Actual slip	INTEGER_16
0x2DDD	0x35DD	Device: Actual output frequency	INTEGER_16
0x2DDE	0x35DE	Motor: Actual position of rotor angle	INTEGER_16
0x2DDF	0x35DF	Axis: Device data	RECORD

0x2DD0 | 0x35D0 - field: Values

Sub.	Name	Lenze setting	Data type
▸ 1	Field: Actual field		UNSIGNED_16
▸ 2	Field: Field setpoint	0 %	UNSIGNED_16

Subindex 1: **Field: actual field**				
Display range (min. value | unit | max. value)			Initialisation	
0	%	65535		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX				UNSIGNED_16

Subindex 2: **Field: target field**				
Display range (min. value | unit | max. value)			Initialisation	
0	%	65535	0 %	
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX				UNSIGNED_16

0x2DD1 | 0x35D1 - motor: Currents

Sub.	Name	Lenze setting	Data type
▸ 1	D-current (id): Actual D-current		INTEGER_16
▸ 2	Q-current (iq): Actual Q-current		INTEGER_16
▸ 3	D-current (id): D-current setpoint		INTEGER_16
▸ 4	Q-current (iq): Q-current setpoint		INTEGER_16
▸ 5	Motor current Ieff		INTEGER_16

Subindex 1: **D current (id): Actual D-current**				
Display range (min. value | unit | max. value)			Initialisation	
-327.67	A	327.67		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: 1/100	INTEGER_16

Subindex 2: **Q current (iq): Actual Q-current**				
Display range (min. value | unit | max. value)			Initialisation	
-327.67	A	327.67		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: 1/100	INTEGER_16

Subindex 3: **D current (id): Target D current**				
Display range (min. value | unit | max. value)			Initialisation	
-327.67	A	327.67		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: 1/100	INTEGER_16

Subindex 4: **Q current (iq): Target Q current**				
Display range (min. value | unit | max. value)			Initialisation	
-327.67	A	327.67		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: 1/100	INTEGER_16

Subindex 5: **Motor current Ieff**				
Display range (min. value | unit | max. value)			Initialisation	
-327.67	A	327.67		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: 1/100	INTEGER_16

0x2DD2 | 0x35D2 - Position: Target position interpolated

Display range (min. value | unit | max. value)			Initialisation	
-2147483647	[Pos unit]	2147483647	0 [Pos unit]	
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX				INTEGER_32

0x2DD3 | 0x35D3 - Target speeds

Sub.	Name	Lenze setting	Data type
▸ 1	Speed: Target speed	0 rpm	INTEGER_32
▸ 2	Speed: Target speed 2	0 rpm	INTEGER_32
▸ 3	Speed: Target speed limited	0 rpm	INTEGER_32

Subindex 1: **Speed: target speed**				
Display range (min. value | unit | max. value)			Initialisation	
-480000	r/min	480000	0 rpm	
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX				INTEGER_32

Subindex 2: **Speed: target speed 2**	
Display range (min. value | unit | max. value)	Initialisation

Subindex 2: **Speed: target speed 2**				
-480000	r/min	480000	0 rpm	
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX				INTEGER_32

Subindex 3: **Speed: target speed limited**				
Display range (min. value | unit | max. value)			Initialisation	
-480000	r/min	480000	0 rpm	
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX				INTEGER_32

0x2DD4 | 0x35D4 - Speed controller: Output signal

- 100 % ≡ rated motor torque (0x6076 or 0x6876 for axis B)

Sub.	Name	Lenze setting	Data type
1	Speed controller: Output signal 1	0.0 %	INTEGER_16
2	Speed controller: Output signal 2	0.0 %	INTEGER_16

Subindex 1: **speed controller: output signal 1**				
Display range (min. value | unit | max. value)			Initialisation	
-3276.7	%	3276.7	0.0 %	
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: 1/10	INTEGER_16

Subindex 2: **speed controller: output signal 2**				
Display range (min. value | unit | max. value)			Initialisation	
-3276.7	%	3276.7	0.0 %	
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: 1/10	INTEGER_16

0x2DD5 | 0x35D5 - torque: Target torque

- 100 % ≡ rated motor torque (0x6076 or 0x6876 for axis B)

Display range (min. value | unit | max. value)			Initialisation	
-21474836.47	Nm	21474836.47	0.00 Nm	
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: 1/100	INTEGER_32

0x2DD6 | 0x35D5 - Torque: Filter cascade

- 100 % ≡ rated motor torque (0x6076 or 0x6876 for axis B)

Sub.	Name	Lenze setting	Data type
1	Torque: Filter cascade - starting value	0.0 %	INTEGER_16
2	Torque: Notch filter 1 - input value	0.0 %	INTEGER_16
3	Torque: Notch filter 2 - input value	0.0 %	INTEGER_16
4	Torque: Target torque filtered	0.0 %	INTEGER_16

Subindex 1: **Torque: filter cascade - starting value**				
Display range (min. value | unit | max. value)			Initialisation	
-3276.7	%	3276.7	0.0 %	
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: 1/10	INTEGER_16

Subindex 2: **Torque: notch filter 1 - input value**				
Display range (min. value | unit | max. value)			Initialisation	
-3276.7	%	3276.7	0.0 %	
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: 1/10	INTEGER_16

Subindex 3: **Torque: notch filter 2 - input value**				
Display range (min. value | unit | max. value)			Initialisation	
-3276.7	%	3276.7	0.0 %	
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: 1/10	INTEGER_16

Subindex 4: **Torque: target torque filtered**				
Display range (min. value | unit | max. value)			Initialisation	
-3276.7	%	3276.7	0.0 %	
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: 1/10	INTEGER_16

0x2DD7 | 0x35D7 - Voltage values

Sub.	Name	Lenze setting	Data type
▸ 1	Current motor voltage limit: Actual voltage		INTEGER_16
▸ 2	D-current controller: Output signal		INTEGER_16
▸ 3	Q-current controller: Output signal		INTEGER_16
▸ 4	D-voltage (magnetisation)		INTEGER_16
▸ 5	Q-voltage (torque)		INTEGER_16

Subindex 1: **Current motor voltage limit: actual voltage**				
Display range (min. value | unit | max. value)			Initialisation	
-3276.7	V	3276.7		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: 1/10	INTEGER_16

Subindex 2: **D-current controller: output signal**				
Display range (min. value | unit | max. value)			Initialisation	
-3276.7	V	3276.7		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: 1/10	INTEGER_16

Subindex 3: **Q-current controller: output signal**				
Display range (min. value | unit | max. value)			Initialisation	
-3276.7	V	3276.7		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: 1/10	INTEGER_16

Subindex 4: **D-voltage (magnetisation)**	
Display range (min. value | unit | max. value)	Initialisation

Subindex 4: **D-voltage (magnetisation)**				
-3276.7	V	3276.7		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: 1/10	INTEGER_16

Subindex 5: **Q-voltage (torque)**				
Display range (min. value | unit | max. value)			Initialisation	
-3276.7	V	3276.7		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: 1/10	INTEGER_16

0x2DDC | 0x35DC - Slip: Actual slip

Display range (min. value | unit | max. value)			Initialisation	
-3276.7	Hz	3276.7		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: 1/10	INTEGER_16

0x2DDD | 0x35DD - Device: Actual output frequency

Display range (min. value | unit | max. value)			Initialisation	
-3276.7	Hz	3276.7	0.0 Hz	
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: 1/10	INTEGER_16

0x2DDE | 0x35DE - motor: Actual position of rotor angle

Display range (min. value | unit | max. value)			Initialisation	
0		2047		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX				INTEGER_16

0x2DDF | 0x35DF - Axis: Device data

Sub.	Name	Lenze setting	Data type
▸ 1	Axis: Rated current		UNSIGNED_16
▸ 2	Axis: Maximum current		UNSIGNED_16
▸ 5	Axis: Supported feedback type	0: Product-defined	UNSIGNED_8

Subindex 1: **axis: rated current**				
Display range (min. value | unit | max. value)			Initialisation	
0.00	A	655.35		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: 1/100	UNSIGNED_16

Subindex 2: **axis: maximum current**				
Display range (min. value | unit | max. value)			Initialisation	
0.00	A	655.35		
☐ Write access ☐ CINH ☑ OSC ☐ P ☐ RX ☑ TX			Scaling: 1/100	UNSIGNED_16

Subindex 5: **Axis: Supported feedback type**			
From software version V01.03.xx onwards			
Selection list (read only)			
0	**Product-defined**		
1	No feedback system		
2	Resolver		
3	SinCos + Hiperface Encoder feedback		
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☑ TX			UNSIGNED_8

10.1 Firmware update

The drive component firmware can be replaced by means of the available EtherCAT interface (RJ45 socket) via FoE. In order to carry out a firmware download, the drive component must be in the "Bootstrap" EtherCAT status. Only one firmware with the name "i700.bin" is accepted.

10.2 Reserved objects

Note!

For the i700 servo inverter, various "reserved objects" are provided, which must be used solely device-internally and must not be written to by the user!

The following reserved objects are provided:

- 0x2C10 ▸ Sensorless synchronous control: Low speed range, open loop
- 0x2C11 ▸ Sensorless synchronous control: High speed range
- 0x2C12 ▸ Sensorless synchronous control: Low speed range, controlled
- 0x2C13 ▸ Sensorless synchronous control: Low speed range, operating mode

11 Appendix

11.1 Table of attributes

The table of attributes contains information required for a communication with the i700 servo inverters via objects.

How to read the table of attributes:

Column	Entry	Meaning
Index	*0xXXXX*	Object index
Name	*Text*	Object or parameter name
	Subindex: Text	Number and parameter name of the subindex
Data type	Data type of the parameter:	
	INTEGER_8	1 byte, with sign
	INTEGER_16	2 bytes with sign
	INTEGER_32	4 bytes with sign
	UNSIGNED_8	1 byte without sign
	UNSIGNED_16	2 bytes without sign
	UNSIGNED_32	4 bytes without sign
	UNSIGNED_64	8 bytes without sign
	STRING(xx)	ASCII string (with character length xx)
	ARRAY [] OF...	ARRAY
Scaling	Scaling of the parameter	
Attributes	P	Parameter can be persisted.
	OSC	Parameter can be recorded with the oscilloscope function.
	Tx	Parameter can be mapped into the TPDO.
	Rx	Parameter can be mapped into the RPDO.
	CINH	Parameter can only be written in case of controller inhibit.

Index	Name	Data type	Factor	Attributes				
Communication Area								
0x1000	Device: Type	UNSIGNED_32	-					
0x1001	Error memory	UNSIGNED_8	-				Tx	
0x1008	ECAT: Manufacturer device name	STRING(50)	-					
0x1009	Device: Hardware version	STRING(50)	-					
0x100A	Device: Software version	STRING(50)	-					
0x1018	ECAT: Identity object							
	1: ECAT: Manufacturer ID	UNSIGNED_32	-					
	2: ECAT: Product code	UNSIGNED_32	-					
	3: ECAT: ESI revision	UNSIGNED_32	-					
	4: ECAT: Serial number	UNSIGNED_32	-					
0x10F1	ECAT: Behaviour in case of error							
	1: Internal device response	UNSIGNED_32	-					
	2: Synchronisation: Error threshold	UNSIGNED_32	-	P				
greyed out = read-only access								

Index	Name	Data type	Factor	Attributes				
0x10F3	Diagnostics: History buffer							
	1: Max. number of messages	UNSIGNED_8	-					
	2: Latest message	UNSIGNED_8	-					
	3: Latest acknowledged message	UNSIGNED_8	-					
	4: Latest active message	UNSIGNED_8	-				Tx	
	5: Control bits	UNSIGNED_16	-					
0x10F8	ECAT DC: Current time	UNSIGNED_64	-				Tx	
Manufacturer specific area								
0x2000	Device: Data							
	1: Device: Product designation	STRING(50)	-					
	2: Device: Serial number	STRING(50)	-					
	3: Device: Manufacturing date	STRING(50)	-					
0x2001	Device: Name	STRING(128)	-	P				
0x2020	EoE information							
	1: Virtual MAC address	STRING(32)	-					
	2: IP address	STRING(32)	-					
	3: Subnet mask	STRING(32)	-					
	4: Standard gateway	STRING(32)	-					
	5: DNS server	STRING(32)	-					
	6: DNS name	STRING(50)	-					
	7: Received packages	UNSIGNED_32	-					
	8: Sent packages	UNSIGNED_32	-					
0x2021	Device: Optical recognition							
	1: Start optical recognition	UNSIGNED_8	-					
	2: Optical recognition: Blinking time	UNSIGNED_16	-					
0x2022	Device command	UNSIGNED_32	-					
0x2030	Parameter set: Validity check (CRC)	UNSIGNED_32	-					
0x2100	Brand protection							
	1: Brand protection: Set pin	INTEGER_32	-					
	2: Brand protection: Enter pin	INTEGER_32	-					
	3: Brand protection: Encryption	UNSIGNED_8	-					
0x2105	Set master password							
	1: Set master password	STRING(16)	-					
	2: Log-in with master password	STRING(16)	-					
0x2500	Touch probe (TP): Debounce time	UNSIGNED_16	-	P				
0x2540	Device: Voltage values							
	1: Mains: Rated voltage	UNSIGNED_8	-	P				
	2: Undervoltage (LU): Warning threshold	UNSIGNED_16	-	P				
	3: Undervoltage (LU): Error threshold	UNSIGNED_16	-	P				
	4: Undervoltage (LU): Threshold 'Reset error'	UNSIGNED_16	-	P				
	5: Overvoltage (OU): Warning threshold	UNSIGNED_16	-	P				
	6: Overvoltage (OU): Error threshold	UNSIGNED_16	-	P				
	7: Overvoltage (OU): Threshold 'Reset error'	UNSIGNED_16	-	P				
	8: Critical DC-bus voltage	UNSIGNED_8	-				Tx	
0x2580	ECAT DC: Real-time information							
	1: ECAT DC: Status real-time information	UNSIGNED_8	-					
	2: ECAT DC: Time stamp first real-time information	UNSIGNED_64	-					
	3: ECAT DC: Time stamp last real-time information	UNSIGNED_64	-					
	4: ECAT DC: Current time	UNSIGNED_64	-					
greyed out = read-only access								

Index	Name	Data type	Factor	Attributes				
0x25D0	Status of device utilisation Ixt							
	1: Power stage utilisation (Ixt): Actual utilisation	UNSIGNED_16	-		OSC		Tx	
	2: Power stage utilisation (Ixt): Warning threshold	UNSIGNED_16	-	P				
	3: Power stage utilisation (Ixt): Error threshold	UNSIGNED_16	-					
	4: Device utilisation (Ixt): Actual utilisation	UNSIGNED_16	-		OSC		Tx	
	5: Device utilisation (Ixt): Warning threshold	UNSIGNED_16	-	P				
	6: Device utilisation (Ixt): Error limit	UNSIGNED_16	-					
Axis A - holding brake control								
0x2820	Brake control: Settings							
	1: Brake: Mode	UNSIGNED_8	-	P		Rx		
	2: Brake: Application time	UNSIGNED_16	-	P				
	3: Brake: Release time	UNSIGNED_16	-	P				
	4: Brake: Brake recognition	UNSIGNED_16	-					
	5: Brake: Control logic	UNSIGNED_8	-	P				
	6: Brake monitoring: Response	UNSIGNED_8	-	P				
	9: Holding brake: Starting torque source	UNSIGNED_16	-	P				
	10: Holding brake: Starting torque	INTEGER_16	1/10	P		Rx		
	11: Holding brake: Override of the holding brake control	UNSIGNED_8	-					
	21: Holding brake: Actual torque	INTEGER_16	1/10		OSC			
Axis A - device control								
0x2822	Axis command	UNSIGNED_32	-					
0x2823	Axis command: Progress	UNSIGNED_8	-					
0x2824	Device control via PDO: Activation	UNSIGNED_8	-	P				CINH
0x2825	Mode: Selection	UNSIGNED_8	-					CINH
0x2826	Quick stop: Duration in case of trouble	UNSIGNED_32	-	P				
0x2830	Lenze control word	UNSIGNED_16	-		OSC	Rx		
0x2831	Lenze status word	UNSIGNED_16	-		OSC		Tx	
0x2832	Identification: Status word	UNSIGNED_16	-		OSC			
0x2833	Lenze statusword 2	UNSIGNED_16	-		OSC		Tx	
0x2835	Manual test mode: Settings							
	1: Manual test mode: Setpoint current	INTEGER_16	-		OSC	Rx		
	2: Manual test mode: Frequency	INTEGER_16	1/10		OSC	Rx		
	3: Manual test mode: Starting angle	INTEGER_16	1/10					
	4: Manual test mode: Frequency 2	INTEGER_16	1/10					
0x2836	Manual jog: Settings							
	1: Manual jog: Setpoint current	UNSIGNED_16	-			Rx		
	2: Manual jog: Frequency	INTEGER_16	1/10		OSC	Rx		
	3: Manual jog: Ramp time - current	UNSIGNED_16	-	P				
	4: Manual jog: Ramp time - frequency	UNSIGNED_16	-	P				
	5: Manual jog: Time monitoring	UNSIGNED_32	-	P				
	6: Manual jog: Current controller gain	UNSIGNED_32	1/100	P				
	7: Manual jog: Current controller reset time	UNSIGNED_32	1/100	P				
0x2837	PRBS excitation							
	1: PRBS stimulation: Cycle time	UNSIGNED_16	-	P				
		UNSIGNED_16	1/10	P				
		UNSIGNED_16	-	P				
		UNSIGNED_16	1/10	P				
		UNSIGNED_8	-	P				CINH
		INTEGER_16	-	P				CINH
Axis A - error management								
greyed out = read-only access								

Index	Name	Data type	Factor	Attributes				
0x2840	Delay time: Reset error	INTEGER_32	-					
0x2841	Error reset	UNSIGNED_8	-					
0x284F	Current fault	BYTE	-					
0x2860	Reserved							
	1: Reserved 1	UNSIGNED_16	-	P				
		UNSIGNED_16	1/10	P				
Axis A - controller settings								
0x2900	Speed controller: Parameter							
	1: Speed controller: Gain	UNSIGNED_32	1/100000	P				
	2: Speed controller: Reset time	UNSIGNED_16	1/10	P				
	3: Speed controller: Rate time	UNSIGNED_16	1/100	P				
0x2901	Speed controller: Gain - adaptation	UNSIGNED_16	1/100	P	OSC	Rx		
0x2902	Speed controller: I component load value	INTEGER_16	1/10	P	OSC	Rx		
0x2903	Speed: Speed setpoint - filter time	UNSIGNED_16	1/10	P				
0x2904	Speed: Actual speed - filter time	UNSIGNED_16	1/10	P				
0x2907	Limitation for SLS							
	1: Add. speed limitation for SLS	UNSIGNED_32	-		OSC	Rx		
0x2910	Moments of inertia							
	1: Moment of inertia: Motor	UNSIGNED_32	1/100	P				
	2: Moment of inertia: Load	UNSIGNED_32	1/100	P				
	3: Moment of inertia: Motor-load coupling	UNSIGNED_8	-	P				
0x2939	Switching frequency	UNSIGNED_8	-	P				
0x2941	Current controller: Feedforward control	UNSIGNED_8	-	P				
0x2942	Current controller: Parameter							
	1: Current controller: Gain	UNSIGNED_32	1/100	P				
	2: Current controller: Reset time	UNSIGNED_32	1/100	P				
0x2943	Motor: Current setpoint - filter time	UNSIGNED_16	1/100	P				
0x2944	Torque: Notch filter torque setpoint							
	1: Notch filter 1: Frequency	UNSIGNED_16	1/10	P				
	2: Notch filter 1: Bandwidth	UNSIGNED_16	1/10	P				
	3: Notch filter 1: Damping	UNSIGNED_8	-	P				
	4: Notch filter 2: Frequency	UNSIGNED_16	1/10	P				
	5: Notch filter 2: Bandwidth	UNSIGNED_16	1/10	P				
	6: Notch filter 2: Damping	UNSIGNED_8	-	P				
0x2945	Torque: Jerk limitation setpoint	UNSIGNED_16	1/10	P				
0x2946	Cyclic sync torque mode: Speed limitation							
	1: Speed limitation: Upper speed limit	INTEGER_32	480000/ 2^{31}	P	OSC	Rx		
	2: Speed limitation: Lower speed limit	INTEGER_32	480000/ 2^{31}	P	OSC	Rx		
greyed out = read-only access								

Index	Name	Data type	Factor	Attributes				
0x2947	Inverter characteristic: Voltage grid points (y)							
	1: IC: y1 = U01 (x = 0.00 %)	UNSIGNED_16	1/100	P				
	2: IC: y2 = U02 (x = 6.25 %)	UNSIGNED_16	1/100	P				
	3: IC: y3 = U03 (x = 12.50 %)	UNSIGNED_16	1/100	P				
	4: IC: y4 = U04 (x = 18.75 %)	UNSIGNED_16	1/100	P				
	5: IC: y5 = U05 (x = 25.00 %)	UNSIGNED_16	1/100	P				
	6: IC: y6 = U06 (x = 31.25 %)	UNSIGNED_16	1/100	P				
	7: IC: y7 = U07 (x = 37.50 %)	UNSIGNED_16	1/100	P				
	8: IC: y8 = U08 (x = 42.75 %)	UNSIGNED_16	1/100	P				
	9: IC: y9 = U09 (x = 50.00 %)	UNSIGNED_16	1/100	P				
	10: IC: y10 = U10 (x = 56.25 %)	UNSIGNED_16	1/100	P				
	11: IC: y11 = U11 (x = 62.50 %)	UNSIGNED_16	1/100	P				
	12: IC: y12 = U12 (x = 68.75 %)	UNSIGNED_16	1/100	P				
	13: IC: y13 = U13 (x = 75.00 %)	UNSIGNED_16	1/100	P				
	14: IC: y14 = U14 (x = 81.25 %)	UNSIGNED_16	1/100	P				
	15: IC: y15 = U15 (x = 87.50 %)	UNSIGNED_16	1/100	P				
	16: IC: y16 = U16 (x = 93.25 %)	UNSIGNED_16	1/100	P				
	17: IC: y17 = U17 (x = 100.00 %)	UNSIGNED_16	1/100	P				
0x2980	Position controller: Gain	UNSIGNED_32	1/100	P				
0x2981	Position controller: Gain - adaptation	UNSIGNED_16	1/100	P	OSC	Rx		
0x2982	Position controller: Output signal limitation	UNSIGNED_32	480000/ 2^{31}	P	OSC	Rx		
0x2983	Set actual position: Starting value	INTEGER_32	-	P	OSC	Rx		
0x2984	Set actual position: Mode	UNSIGNED_8	-	P				
0x2985	Position controller: Internal adaptation - characteristics							
	1: Speed 1	UNSIGNED_16	-	P				
	2: Vp 1	UNSIGNED_16	-	P				
	3: Speed 2	UNSIGNED_16	-	P				
	4: Vp 2	UNSIGNED_16	-	P				
	5: Speed 3	UNSIGNED_16	-	P				
	6: Vp 3	UNSIGNED_16	-	P				
	7: Speed 4	UNSIGNED_16	-	P				
	8: Vp 4	UNSIGNED_16	-	P				
	9: Speed 5	UNSIGNED_16	-	P				
	10: Vp 5	UNSIGNED_16	-	P				
	11: Activation of adapted curve	UNSIGNED_8	-	P				
0x2986	Position controller:Resulting gain adaptation	UNSIGNED_16	1/100		OSC			
0x29C0	Field controller: Parameter							
	1: Field controller: Gain	UNSIGNED_32	1/100	P				
	2: Field controller: Reset time	UNSIGNED_16	1/10	P				
0x29E0	Field weakening controller: Parameter							
	1: Field weakening controller: Gain	UNSIGNED_32	1/1000	P				
	2: Field weakening controller: Reset time	UNSIGNED_32	1/10	P				
0x29E1	Field set value limitation	UNSIGNED_16	1/100	P	OSC	Rx		
0x29E2	DC link circuit voltage: Filter time	UNSIGNED_16	1/10	P				
0x29E3	Motor voltage act. value: Filter time	UNSIGNED_16	1/10	P				
0x29E4	Voltage reserve	UNSIGNED_8	-	P				
Axis A - V/f operation								
0x2B00	VFC: V/f characteristic - shape	UNSIGNED_8	-	P				CINH
greyed out = read-only access								

Index	Name	Data type	Factor	Attributes				
0x2B01	VFC: V/f characteristic - define reference point							
	1: VFC: V/f characteristic - voltage at reference point	UNSIGNED_16	-	P				
	2: VFC: V/f characteristic - frequency at reference point	UNSIGNED_16	-	P				
0x2B02	VFC: User-definable V/f characteristic - frequency grid points (x)							
	1: V/f: x1 = f01	INTEGER_16	-	P				
	2: V/f: x2 = f02	INTEGER_16	-	P				
	3: V/f: x3 = f03	INTEGER_16	-	P				
	4: V/f: x4 = f04	INTEGER_16	-	P				
	5: V/f: x5 = f05	INTEGER_16	-	P				
	6: V/f: x6 = f06	INTEGER_16	-	P				
	7: V/f: x7 = f07	INTEGER_16	-	P				
	8: V/f: x8 = f08	INTEGER_16	-	P				
	9: V/f: x9 = f09	INTEGER_16	-	P				
	10: V/f: x10 = f10	INTEGER_16	-	P				
	11: V/f: x11 = f11	INTEGER_16	-	P				
0x2B03	VFC: User-definable V/f characteristic - voltage grid points (y)							
	1: V/f: y1 = U01 (x = f01)	UNSIGNED_32	1/100	P				
	2: V/f: y2 = U02 (x = f02)	UNSIGNED_32	1/100	P				
	3: V/f: y3 = U03 (x = f03)	UNSIGNED_32	1/100	P				
	4: V/f: y4 = U04 (x = f04)	UNSIGNED_32	1/100	P				
	5: V/f: y5 = U05 (x = f05)	UNSIGNED_32	1/100	P				
	6: V/f: y6 = U06 (x = f06)	UNSIGNED_32	1/100	P				
	7: V/f: y7 = U07 (x = f07)	UNSIGNED_32	1/100	P				
	8: V/f: y8 = U08 (x = f08)	UNSIGNED_32	1/100	P				
	9: V/f: y9 = U09 (x = f09)	UNSIGNED_32	1/100	P				
	10: V/f: y10 = U10 (x = f10)	UNSIGNED_32	1/100	P				
	11: V/f: y11 = U11 (x = f11)	UNSIGNED_32	1/100	P				
0x2B04	VFC: Voltage vector control - current setpoint	UNSIGNED_32	1/100	P				
0x2B05	VFC: Voltage vector control parameter							
	1: VFC: Voltage vector controller - gain	UNSIGNED_32	1/100	P				
	2: VFC: Voltage vector controller - reset time	UNSIGNED_32	1/100	P				
0x2B06	VFC: Voltage boost	UNSIGNED_16	1/10	P				
0x2B07	VFC: Load adjustment - parameter							
	1: VFC: Load adjustment - direction of rotation	UNSIGNED_8	-	P				CINH
	2: VFC: Load adjustment - value	UNSIGNED_32	1/100	P				
0x2B08	VFC: Imax controller - parameter							
	1: VFC: Imax controller - gain	UNSIGNED_32	1/1000	P				
	2: VFC: Imax controller - reset time	UNSIGNED_32	1/10	P				
0x2B09	VFC: Slip compensation - parameter							
	1: VFC: Slip compensation - influence	INTEGER_16	1/100	P				
	2: VFC: Slip compensation - filter time	UNSIGNED_16	-	P				
0x2B0A	VFC: Oscillation damping - parameter							
	1: VFC: Oscillation damping - gain	INTEGER_16	-	P				
	2: VFC: Oscillation damping - filter time	UNSIGNED_16	-	P				
	3: VFC: Oscillation damping - limitation	UNSIGNED_16	1/10	P				
	4: VFC: Oscillation damping - final ramp frequency	UNSIGNED_8	-	P				
0x2B0B	VFC: Setpoint frequency	INTEGER_16	1/10		OSC		Tx	
0x2B0C	VFC: Override point of field weakening	INTEGER_16	1/10	P				
0x2B80	DC-injection braking: Current	UNSIGNED_16	1/100	P				
greyed out = read-only access								

Index	Name	Data type	Factor	Attributes				
0x2BA0	Flying restart: Activate	UNSIGNED_8	-	P				
0x2BA1	Flying restart: Current	UNSIGNED_16	-	P				
0x2BA2	Flying restart: Start frequency	INTEGER_16	1/10	P				
0x2BA3	Flying restart: Integration time	UNSIGNED_16	-	P				
0x2BA4	Flying restart: Min. deviation	UNSIGNED_16	1/100	P				
0x2BA5	Flying restart: Delay time	UNSIGNED_16	-	P				
0x2BA6	Flying restart: Result							
	1: Flying restart: Determined speed [rpm]	INTEGER_16	-		OSC		Tx	
	2: Flying restart: Determined speed [n unit]	INTEGER_32	480000/ 2^{31}		OSC		Tx	
Axis A - motor settings								
0x2C00	Motor control	UNSIGNED_8	-	P				CINH
0x2C01	Motor: Common parameters							
	1: Motor: Number of pole pairs	UNSIGNED_8	-					
	2: Motor: Stator resistance (20°C)	UNSIGNED_32	1/10000	P				
	3: Motor: Stator leakage inductance	UNSIGNED_32	1/1000	P				
	4: Motor: Rated speed	UNSIGNED_16	-	P				
	5: Motor: Rated frequency	UNSIGNED_16	1/10	P				
	6: Motor: Rated power	UNSIGNED_16	1/100	P				
	7: Motor: Rated voltage	UNSIGNED_16	-	P				
	8: Motor: Rated cosine phi	UNSIGNED_16	1/100	P				
	9: Motor: Insulation class	UNSIGNED_8	-	P				
	10: Motor: Designation	STRING(50)	-	P				
0x2C02	Motor (ASM): Parameter							
	1: Motor (ASM): Rotor resistance (20°C)	UNSIGNED_32	1/10000	P				
	2: Motor (ASM): Mutual inductance	UNSIGNED_32	1/10	P				
	3: Motor (ASM): Magnetising current	UNSIGNED_16	1/100	P				
0x2C03	Motor (SM): Parameter							
	1: Motor (SM): e.m.f. constant (KE conductor-conductor, 20°C)	UNSIGNED_32	1/10	P				
	2: Motor (SM): Pole position	INTEGER_16	1/10	P				
	3: Motor (SM): Temperature coefficient - magnets (kTN)	INTEGER_16	1/1000	P				
	4: Motor (SM): Pole position	INTEGER_16	1/10	P				
greyed out = read-only access								

Index	Name	Data type	Factor	Attributes				
0x2C04	Motor: Lss saturation characteristic - inductance grid points (y)							
	1: Lss: y1 = L01 (x = 0.00 %)	UNSIGNED_16	-	P				
	2: Lss: y2 = L02 (x = 6.25 %)	UNSIGNED_16	-	P				
	3: Lss: y3 = L03 (x = 12.50 %)	UNSIGNED_16	-	P				
	4: Lss: y4 = L04 (x = 18.75 %)	UNSIGNED_16	-	P				
	5: Lss: y5 = L05 (x = 25.00 %)	UNSIGNED_16	-	P				
	6: Lss: y6 = L06 (x = 31.25 %)	UNSIGNED_16	-	P				
	7: Lss: y7 = L07 (x = 37.50 %)	UNSIGNED_16	-	P				
	8: Lss: y8 = L08 (x = 42.75 %)	UNSIGNED_16	-	P				
	9: Lss: y9 = L09 (x = 50.00 %)	UNSIGNED_16	-	P				
	10: Lss: y10 = L10 (x = 56.25 %)	UNSIGNED_16	-	P				
	11: Lss: y11 = L11 (x = 62.50 %)	UNSIGNED_16	-	P				
	12: Lss: y12 = L12 (x = 68.75 %)	UNSIGNED_16	-	P				
	13: Lss: y13 = L13 (x = 75.00 %)	UNSIGNED_16	-	P				
	14: Lss: y14 = L14 (x = 81.25 %)	UNSIGNED_16	-	P				
	15: Lss: y15 = L15 (x = 87.50 %)	UNSIGNED_16	-	P				
	16: Lss: y16 = L16 (x = 93.25 %)	UNSIGNED_16	-	P				
	17: Lss: y17 = L17 (x = 100.00 %)	UNSIGNED_16	-	P				
	18: Motor: Lss saturation characteristic - activation	UNSIGNED_16	-	P				
0x2C05	Motor: Lss saturation characteristic - reference for current grid points (x)	UNSIGNED_16	1/10	P				
0x2C06	Motor (SM): Magnet characteristic (current) - grid points							
	1: Magnet characteristic: x1 = i01/iN	UNSIGNED_16	-	P				
	2: Magnet characteristic: y1 = kT01/kTN	UNSIGNED_16	-	P				
	3: Magnet characteristic: x2 = i02/iN	UNSIGNED_16	-	P				
	4: Magnet characteristic: y2 = kT02/kTN	UNSIGNED_16	-	P				
	5: Magnet characteristic: x3 = i03/iN	UNSIGNED_16	-	P				
	6: Magnet characteristic: y3 = kT03/kTN	UNSIGNED_16	-	P				
	7: Magnet characteristic: x4 = i04/iN	UNSIGNED_16	-	P				
	8: Magnet characteristic: y4 = kT04/kTN	UNSIGNED_16	-	P				
0x2C07	Motor (ASM): Lh saturation characteristic - inductance grid points (y)							
	1: Lh: y1 = L01 (x = 0.00 %)	UNSIGNED_16	-	P				
	2: Lh: y2 = L02 (x = 6.25 %)	UNSIGNED_16	-	P				
	3: Lh: y3 = L03 (x = 12.50 %)	UNSIGNED_16	-	P				
	4: Lh: y4 = L04 (x = 18.75 %)	UNSIGNED_16	-	P				
	5: Lh: y5 = L05 (x = 25.00 %)	UNSIGNED_16	-	P				
	6: Lh: y6 = L06 (x = 31.25 %)	UNSIGNED_16	-	P				
	7: Lh: y7 = L07 (x = 37.50 %)	UNSIGNED_16	-	P				
	8: Lh: y8 = L08 (x = 43.75 %)	UNSIGNED_16	-	P				
	9: Lh: y9 = L09 (x = 50.00 %)	UNSIGNED_16	-	P				
	10: Lh: y10 = L10 (x = 56.25 %)	UNSIGNED_16	-	P				
	11: Lh: y11 = L11 (x = 62.50 %)	UNSIGNED_16	-	P				
	12: Lh: y12 = L12 (x = 68.75 %)	UNSIGNED_16	-	P				
	13: Lh: y13 = L13 (x = 75.00 %)	UNSIGNED_16	-	P				
	14: Lh: y14 = L14 (x = 81.25 %)	UNSIGNED_16	-	P				
	15: Lh: y15 = L15 (x = 87.50 %)	UNSIGNED_16	-	P				
	16: Lh: y16 = L16 (x = 93.75 %)	UNSIGNED_16	-	P				
	17: Lh: y17 = L17 (x = 100.00 %)	UNSIGNED_16	-	P				
0x2C08	Motor: Motor parameter setting method	UNSIGNED_8	-	P				
greyed out = read-only access								

Index	Name	Data type	Factor	Attributes				
0x2C10	Sensorless synchronous control: Low speed range, open loop							
	1: HF amplitude	UNSIGNED_16	1/10	P				
	2: angle tracking controller: gain	UNSIGNED_16	1/100	P				
	3: angle tracking controller: reset time	UNSIGNED_16	1/100	P				
	4: static compensation: angle	INTEGER_16	1/10	P				
	5: dynamic compensation: angle 1	INTEGER_16	1/10	P				
	6: dynamic compensation: angle 2	INTEGER_16	1/10	P				
	7: filter time constant	UNSIGNED_16	1/10	P				
0x2C11	Sensorless synchronous control: High speed range							
	1: lower limit for "high speed range"	UNSIGNED_16	-	P				
	2: angle tracking controller: gain	UNSIGNED_16	-	P				
	3: angle tracking controller: reset time	UNSIGNED_16	1/100	P				
	4: sensorless synchronous control: decoupling time - speed range	UNSIGNED_16	1/100	P				
0x2C12	Sensorless synchronous control: Low speed range, controlled							
	1: Additive accelerating current	UNSIGNED_16	-	P				
	2: Standstill holding current	UNSIGNED_16	-	P				
0x2C13	Sensorless synchronous control: Low speed range, operating mode	UNSIGNED_8	-	P				
Axis A - feedback system								
0x2C40	Encoder: Type	UNSIGNED_8	-	P				CINH
0x2C41	Hiperface: Parameter							
	1: Hiperface: Determined type code	UNSIGNED_8	-					
	2: Hiperface: User def. encoder - type code	UNSIGNED_8	-	P				CINH
	3: Hiperface: User def. encoder - specifiable revolutions	UNSIGNED_16	-	P				CINH
	4: Hiperface absolute value fault: Response	UNSIGNED_8	-	P				
	5: Hiperface: Serial number	STRING(50)	-					
	6: Hiperface: Raw data - Actual position	UNSIGNED_32	-				Tx	
	7: Hiperface: Detected Increments / revolution	UNSIGNED_16	-				Tx	
	8: Hiperface: Type code supported by firmware	UNSIGNED_8	-				Tx	
	9: Hiperface: Encoder type	UNSIGNED_8	-				Tx	
	10: Hiperface: Period length of linear encoders	UNSIGNED_32	-				Tx	
	11: Hiperface: Data block address	UNSIGNED_16	-					
	12: Hiperface: Data block length	UNSIGNED_8	-					
	13: Hiperface: Data block command	UNSIGNED_8	-					
	14: Hiperface: Data block status	UNSIGNED_16	-					
	15: Hiperface: Data block	BYTE	-					
0x2C42	Encoder: Parameter							
	1: Encoder: Increments / revolution	UNSIGNED_32	-	P				CINH
	2: Encoder: Supply voltage	UNSIGNED_8	1/10	P				CINH
	3: Encoder: Angle drift - Actual angle error	INTEGER_16	1/10		OSC		Tx	
	4: Encoder: Signal quality - Actual amplitude	UNSIGNED_8	-		OSC			
0x2C43	Resolver: Number of pole pairs	UNSIGNED_8	-	P				CINH
0x2C44	Resolver error compensation: Parameter							
	1: Resolver error compensation: Angle	INTEGER_16	-	P				
	2: Resolver error compensation: Cosine track gain	UNSIGNED_16	-	P				
	3: Resolver error compensation: Sine track gain	UNSIGNED_16	-	P				
	4: Identification: search time for v = const.	UNSIGNED_16	-	P				
	5: Identification: Time-out	UNSIGNED_16	-	P				
	6: Identification: Actual status	UNSIGNED_16	-		OSC			
greyed out = read-only access								

Index	Name	Data type	Factor	Attributes				
0x2C45	Feedback system: Response to open circuit	UNSIGNED_8	-	P				
0x2C46	Feedback system: Specifiable number of revolutions	UNSIGNED_16	-					
0x2C47	Feedback system: Open circuit detection sensitivity	UNSIGNED_8	-	P				
0x2C48	Internal							
	1: Internal: Response	UNSIGNED_8	-	P				
	2: Internal: Actual position (raw data)	UNSIGNED_32	-				Tx	
0x2C5F	Feedback system: Parameter CRC	UNSIGNED_32	-					
Axis A - pole position identification								
0x2C60	Monitoring pole position identification: Response	UNSIGNED_8	-	P				
0x2C61	Pole position identification PPI (360°)							
	1: PPI (360°): Current amplitude	UNSIGNED_16	-	P				CINH
	2: PPI (360°): Ramp time	UNSIGNED_16	-	P				CINH
	3: PPI (360°): Direction of rotation	UNSIGNED_8	-	P				CINH
	4: PPI (360°): Fault tolerance	UNSIGNED_8	-	P				
	5: PPI (360°): Absolute current amplitude	UNSIGNED_32	1/100					
0x2C62	Pole position identification PPI (min. movement)							
	1: PPI (min. movement): Current amplitude	UNSIGNED_16	-	P				CINH
	2: PPI (min. movement): Ramp time - current	UNSIGNED_16	-	P				CINH
	3: PPI (min. movement): Gain	UNSIGNED_16	-	P				
	4: PPI (min. movement): Reset time	UNSIGNED_16	1/10	P				
	5: PPI (min. movement): Max. move permitted	UNSIGNED_8	-	P				
	6: PPI (min. movement): Absolute current amplitude	UNSIGNED_32	1/100					
0x2C63	Pole position identification PPI (without movement)							
	1: Behaviour after switch-on	UNSIGNED_8	-	P				CINH
0x2C64	Cable check							
	1: Cable check: Behaviour after switch-on	UNSIGNED_8	-	P				CINH
	2: Cable check: Status word	UNSIGNED_16	-					
Axis A - touch probe								
0x2D00	Touch probe (TP): Dead time compensation							
	1: Dead time compensation: TP1 delay time	UNSIGNED_16	1/1000	P				
	2: Dead time compensation: TP2 delay time	UNSIGNED_16	1/1000	P				
0x2D01	Touch probe (TP): Time stamp							
	1: TP1: Time stamp - rising edge	UNSIGNED_32	-		OSC		Tx	
	2: TP1: Time stamp - falling edge	UNSIGNED_32	-		OSC		Tx	
	3: TP2: Time stamp - rising edge	UNSIGNED_32	-		OSC		Tx	
	4: TP2: Time stamp - falling edge	UNSIGNED_32	-		OSC		Tx	
Axis A - monitoring functions								
0x2D40	Ixt utilisation							
	1: Power stage utilisation (Ixt): Actual utilisation	UNSIGNED_16	-		OSC		Tx	
	2: Power stage utilisation (Ixt): Warning threshold	UNSIGNED_16	-	P				
	3: Power stage utilisation (Ixt): Error threshold	UNSIGNED_16	-					
	4: Device utilisation (Ixt): Actual utilisation	UNSIGNED_16	-		OSC		Tx	
	5: Device utilisation (Ixt): Warning threshold	UNSIGNED_16	-	P				
	6: Device utilisation (Ixt): Error limit	UNSIGNED_16	-					
0x2D44	Motor speed monitoring							
	1: Motor speed monitoring: Threshold	UNSIGNED_16	-	P				
	2: Motor speed monitoring: Response	UNSIGNED_8	-	P				
greyed out = read-only access								

Index	Name	Data type	Factor	Attributes				
0x2D45	Motor phase failure detection							
	1: Motor phase failure 1: Response	UNSIGNED_8	-	P				
	2: Motor phase failure 1: Current threshold	UNSIGNED_8	1/10	P				
	3: Motor phase failure 1: Voltage threshold	UNSIGNED_16	1/10	P				
	4: Motor phase failure 2: Response	UNSIGNED_8	-	P				
0x2D46	Monitoring: Ultimate motor current							
	1: Ultimate motor current: Threshold	UNSIGNED_16	1/10	P				
	2: Ultimate motor current: Response	UNSIGNED_8	-	P				
0x2D49	Motor temperature monitoring: Parameter							
	1: Motor temperature monitoring: Sensor type	UNSIGNED_8	-	P				
	2: Motor temperature monitoring: Response	UNSIGNED_8	-	P				
	3: Motor temperature monitoring: Warning threshold	INTEGER_16	1/10	P				
	4: Motor temperature monitoring: Error threshold	INTEGER_16	1/10	P				
	5: Motor temperature monitoring: Actual motor temperature	INTEGER_16	1/10		OSC		Tx	
	6: Thermal sensor characteristic: Grid point 1 - temperature	INTEGER_16	1/10	P				
	7: Thermal sensor characteristic: Grid point 2 - temperature	INTEGER_16	1/10	P				
	8: Thermal sensor characteristic: grid point 1 - resistance	INTEGER_16	-	P				
	9: Thermal sensor characteristic: Grid point 2 - resistance	INTEGER_16	-	P				
0x2D4C	Motor utilisation (I²xt): Parameter for the thermal model							
	1: Motor utilisation (I²xt): Thermal time constant - winding	UNSIGNED_16	-	P				
	2: Motor utilisation (I²xt): Thermal time constant - laminations	UNSIGNED_16	-	P				
	3: Motor utilisation (I²xt): Influence of the winding	UNSIGNED_8	-	P				
	4: Motor utilisation (I²xt): Starting value	UNSIGNED_16	-	P				
0x2D4D	Motor utilisation (I²xt): User-definable characteristic							
	1: I²xt: x1 = n01/nN (n01 ~ 0)	UNSIGNED_16	-	P				
	2: I²xt: y1 = i01/iN (x = n01 ~ 0)	UNSIGNED_16	-	P				
	3: I²xt: x2 = n02/nN (n02 = limit - reduced cooling)	UNSIGNED_16	-	P				
	4: I²xt: y2 = i02/iN (x = n02 = limit - reduced cooling)	UNSIGNED_16	-	P				
	5: I²xt: x3 = n03/nN (n03 = rated speed)	UNSIGNED_16	-	P				
	6: I²xt: y3 = i03/iN (x = n03 = rated speed)	UNSIGNED_16	-	P				
	7: I²xt: x4 = n04/nN (n04 = limit - field weakening)	UNSIGNED_16	-	P				
	8: I²xt: y4 = i04/iN (x = n04 = limit - field weakening)	UNSIGNED_16	-	P				
0x2D4E	Motor utilisation (I²xt): Motor overload warning threshold	UNSIGNED_16	-	P				
0x2D4F	Motor utilisation (I²xt): Actual utilisation	UNSIGNED_16	-		OSC		Tx	
0x2D50	Motor utilisation (I²xt): Motor overload error							
	1: Motor utilisation (I²xt): Response	UNSIGNED_8	-	P				
	2: Motor utilisation (I²xt): Error threshold	UNSIGNED_16	-	P				

greyed out = read-only access

Index	Name	Data type	Factor	Attributes				
0x2D51	Monitoring: Position and velocity error							
	1: Monitoring: Velocity error - error threshold	UNSIGNED_32	-	P				
	2: Monitoring: Velocity error - minimum error time	UNSIGNED_16	-	P				
	3: Monitoring: Velocity error - response	UNSIGNED_8	-	P				
	4: Monitoring: Position error - error threshold	UNSIGNED_32	-	P				
	5: Monitoring: Position error - minimum error time	UNSIGNED_16	-	P				
	6: Monitoring: Position error - response	UNSIGNED_8	-	P				
0x2D81	Counter: Operating time							
	1: Device: Operating time	UNSIGNED_32	-					
	2: Device: Power-on time	UNSIGNED_32	-					
0x2D82	Motor: Actual voltage - Veff, phase-phase	UNSIGNED_32	1/10		OSC		Tx	
0x2D83	Motor: Phase currents							
	1: Zero system current	INTEGER_32	1/100		OSC		Tx	
	2: Current - phase U	INTEGER_32	1/100		OSC		Tx	
	3: Current - phase V	INTEGER_32	1/100		OSC		Tx	
	4: Current - phase W	INTEGER_32	1/100		OSC		Tx	
0x2D84	Heatsink temperature							
	1: Heatsink temperature: Actual temperature	INTEGER_16	1/10		OSC		Tx	
	2: Heatsink temperature: Warning threshold	INTEGER_16	1/10	P				
	3: Heatsink temperature: Threshold - switch-on fan	INTEGER_16	1/10	P				
	4: Heatsink temperature: Threshold - switch-off fan	INTEGER_16	1/10	P				
0x2D8A	Speed monitoring: Actual speed error	INTEGER_32	-		OSC		Tx	
0x2DD0	Field: Values							
	1: Field: Actual field	UNSIGNED_16	-		OSC		Tx	
	2: Field: Setpoint field	UNSIGNED_16	-		OSC		Tx	
0x2DD1	Motor: Currents							
	1: D current (id): Actual D current	INTEGER_16	1/100		OSC		Tx	
	2: Q current (iq): Actual Q current	INTEGER_16	1/100		OSC		Tx	
	3: D current (id): Setpoint D current	INTEGER_16	1/100		OSC		Tx	
	4: Q current (iq): Setpoint Q current	INTEGER_16	1/100		OSC		Tx	
	5: Motor current Ieff	INTEGER_16	1/100		OSC		Tx	
0x2DD2	Position: Target position interpolated	INTEGER_32	-		OSC		Tx	
0x2DD3	Target speeds							
	1: Speed: Setpoint speed	INTEGER_32	-		OSC		Tx	
	2: Speed: Setpoint speed 2	INTEGER_32	-		OSC		Tx	
	3: Speed: Limited setpoint speed	INTEGER_32	-		OSC		Tx	
0x2DD4	Speed controller: Output signal							
	1: Speed controller: Output signal 1	INTEGER_16	1/10		OSC		Tx	
	2: Speed controller: Output signal 2	INTEGER_16	1/10		OSC		Tx	
0x2DD5	Target torque	INTEGER_32	1/100		OSC		Tx	
0x2DD6	Torque: Filter cascade							
	1: Torque: Filter cascade - starting value	INTEGER_16	1/10		OSC		Tx	
	2: Torque: Notch filter 1 - input value	INTEGER_16	1/10		OSC		Tx	
	3: Torque: Notch filter 2 - input value	INTEGER_16	1/10		OSC		Tx	
	4: Torque: Filtered setpoint torque	INTEGER_16	1/10		OSC		Tx	

greyed out = read-only access

Index	Name	Data type	Factor	Attributes				
0x2DD7	Voltage values							
	1: Actual motor voltage limit: Actual voltage	INTEGER_16	1/10		OSC		Tx	
	2: D-current controller: Output signal	INTEGER_16	1/10		OSC		Tx	
	3: Q-current controller: Output signal	INTEGER_16	1/10		OSC		Tx	
	4: D-voltage (magnetisation)	INTEGER_16	1/10		OSC		Tx	
	5: Q-voltage (torque)	INTEGER_16	1/10		OSC		Tx	
0x2DDC	Slip: Actual slip	INTEGER_16	1/10		OSC		Tx	
0x2DDD	Device: Actual output frequency	INTEGER_16	1/10		OSC		Tx	
0x2DDE	Motor: Actual position of rotor angle	INTEGER_16	-		OSC		Tx	
0x2DDF	Axis: Device data							
	1: Axis: Rated current	UNSIGNED_16	1/100		OSC		Tx	
	2: Axis: Maximum current	UNSIGNED_16	1/100		OSC		Tx	
		UNSIGNED_16	$1/10^9$				Tx	
		UNSIGNED_16	$1/10^8$				Tx	
	5: Axis: Supported feedback type	UNSIGNED_8	-				Tx	
0x2DE0	Advanced settings							
	1: Current controller: Setting for identification	UNSIGNED_8	-					
	2: Sensorless synchronous control: Signal for test mode	UNSIGNED_8	-					
	3: Resolver: Position detection - dynamics	UNSIGNED_16	-					
	4: Resolver: 8 kHz safety signal	UNSIGNED_8	-					
	5: Torque: Internal interpolation	UNSIGNED_8	-	P				
	6: OEM service data: Serial number	UNSIGNED_32	-					
	14: Overwrite CiA control word bit 4	UNSIGNED_8	-	P				
Axis B - holding brake control								
0x3020	Brake control: Settings							
	1: Brake: Mode	UNSIGNED_8	-	P		Rx		
	2: Brake: Application time	UNSIGNED_16	-	P				
	3: Brake: Release time	UNSIGNED_16	-	P				
	4: Brake: Brake recognition	UNSIGNED_16	-					
	5: Brake: Control logic	UNSIGNED_8	-	P				
	6: Brake monitoring: Response	UNSIGNED_8	-	P				
	9: Holding brake: Starting torque source	UNSIGNED_16	-	P				
	10: Holding brake: Starting torque	INTEGER_16	1/10	P		Rx		
	11: Holding brake: Override of the holding brake control	UNSIGNED_8	-					
	21: Holding brake: Actual torque	INTEGER_16	1/10		OSC			
Axis B - device control								
0x3022	Axis command	UNSIGNED_32	-					
0x3023	Axis command: Progress	UNSIGNED_8	-					
0x3024	Device control via PDO: Activation	UNSIGNED_8	-	P				CINH
0x3025	Mode: Selection	UNSIGNED_8	-					CINH
0x3026	Quick stop: Duration in case of trouble	UNSIGNED_32	-	P				
0x3030	Lenze control word	UNSIGNED_16	-		OSC	Rx		
0x3031	Lenze status word	UNSIGNED_16	-		OSC		Tx	
0x3032	Identification: Status word	UNSIGNED_16	-		OSC			
0x3033	Lenze statusword 2	UNSIGNED_16	-		OSC		Tx	
greyed out = read-only access								

Index	Name	Data type	Factor	Attributes				
0x3035	Manual test mode: Settings							
	1: Manual test mode: Setpoint current	INTEGER_16	-		OSC	Rx		
	2: Manual test mode: Frequency	INTEGER_16	1/10		OSC	Rx		
	3: Manual test mode: Starting angle	INTEGER_16	1/10					
	4: Manual test mode: Frequency 2	INTEGER_16	1/10					
0x3036	Manual jog: Settings							
	1: Manual jog: Setpoint current	UNSIGNED_16	-			Rx		
	2: Manual jog: Frequency	INTEGER_16	1/10		OSC	Rx		
	3: Manual jog: Ramp time - current	UNSIGNED_16	-	P				
	4: Manual jog: Ramp time - frequency	UNSIGNED_16	-	P				
	5: Manual jog: Time monitoring	UNSIGNED_32	-	P				
	6: Manual jog: Current controller gain	UNSIGNED_32	1/100	P				
	7: Manual jog: Current controller reset time	UNSIGNED_32	1/100	P				
0x3037	PRBS excitation							
	1: PRBS stimulation: Cycle time	UNSIGNED_16	-	P				
		UNSIGNED_16	1/10	P				
		UNSIGNED_16	-	P				
		UNSIGNED_16	1/10	P				
		UNSIGNED_8	-	P				CINH
		INTEGER_16	-	P				CINH
Axis B - error management								
0x3040	Delay time: Reset error	INTEGER_32	-					
0x3041	Error reset	UNSIGNED_8	-					
0x304F	Current fault	BYTE	-					
0x3060	Reserved							
	1: Reserved 1	UNSIGNED_16	-	P				
		UNSIGNED_16	1/10	P				
Axis B - controller settings								
0x3100	Speed controller: Parameter							
	1: Speed controller: Gain	UNSIGNED_32	1/100000	P				
	2: Speed controller: Reset time	UNSIGNED_16	1/10	P				
	3: Speed controller: Rate time	UNSIGNED_16	1/100	P				
0x3101	Speed controller: Gain - adaptation	UNSIGNED_16	1/100	P	OSC	Rx		
0x3102	Speed controller: I component load value	INTEGER_16	1/10	P	OSC	Rx		
0x3103	Speed: Speed setpoint - filter time	UNSIGNED_16	1/10	P				
0x3104	Speed: Actual speed - filter time	UNSIGNED_16	1/10	P				
0x3107	Limitation for SLS							
	1: Add. speed limitation for SLS	UNSIGNED_32	-		OSC	Rx		
0x3110	Moments of inertia							
	1: Moment of inertia: Motor	UNSIGNED_32	1/100	P				
	2: Moment of inertia: Load	UNSIGNED_32	1/100	P				
	3: Moment of inertia: Motor-load coupling	UNSIGNED_8	-	P				
0x3139	Switching frequency	UNSIGNED_8	-	P				
0x3141	Current controller: Feedforward control	UNSIGNED_8	-	P				
0x3142	Current controller: Parameter							
	1: Current controller: Gain	UNSIGNED_32	1/100	P				
	2: Current controller: Reset time	UNSIGNED_32	1/100	P				
0x3143	Motor: Current setpoint - filter time	UNSIGNED_16	1/100	P				
greyed out = read-only access								

Index	Name	Data type	Factor	Attributes				
0x3144	Torque: Notch filter torque setpoint							
	1: Notch filter 1: Frequency	UNSIGNED_16	1/10	P				
	2: Notch filter 1: Bandwidth	UNSIGNED_16	1/10	P				
	3: Notch filter 1: Damping	UNSIGNED_8	-	P				
	4: Notch filter 2: Frequency	UNSIGNED_16	1/10	P				
	5: Notch filter 2: Bandwidth	UNSIGNED_16	1/10	P				
	6: Notch filter 2: Damping	UNSIGNED_8	-	P				
0x3145	Torque: Jerk limitation setpoint	UNSIGNED_16	1/10	P				
0x3146	Cyclic sync torque mode: Speed limitation							
	1: Speed limitation: Upper speed limit	INTEGER_32	480000/ 2^{31}	P	OSC	Rx		
	2: Speed limitation: Lower speed limit	INTEGER_32	480000/ 2^{31}	P	OSC	Rx		
0x3147	Inverter characteristic: Voltage grid points (y)							
	1: IC: y1 = U01 (x = 0.00 %)	UNSIGNED_16	1/100	P				
	2: IC: y2 = U02 (x = 6.25 %)	UNSIGNED_16	1/100	P				
	3: IC: y3 = U03 (x = 12.50 %)	UNSIGNED_16	1/100	P				
	4: IC: y4 = U04 (x = 18.75 %)	UNSIGNED_16	1/100	P				
	5: IC: y5 = U05 (x = 25.00 %)	UNSIGNED_16	1/100	P				
	6: IC: y6 = U06 (x = 31.25 %)	UNSIGNED_16	1/100	P				
	7: IC: y7 = U07 (x = 37.50 %)	UNSIGNED_16	1/100	P				
	8: IC: y8 = U08 (x = 42.75 %)	UNSIGNED_16	1/100	P				
	9: IC: y9 = U09 (x = 50.00 %)	UNSIGNED_16	1/100	P				
	10: IC: y10 = U10 (x = 56.25 %)	UNSIGNED_16	1/100	P				
	11: IC: y11 = U11 (x = 62.50 %)	UNSIGNED_16	1/100	P				
	12: IC: y12 = U12 (x = 68.75 %)	UNSIGNED_16	1/100	P				
	13: IC: y13 = U13 (x = 75.00 %)	UNSIGNED_16	1/100	P				
	14: IC: y14 = U14 (x = 81.25 %)	UNSIGNED_16	1/100	P				
	15: IC: y15 = U15 (x = 87.50 %)	UNSIGNED_16	1/100	P				
	16: IC: y16 = U16 (x = 93.25 %)	UNSIGNED_16	1/100	P				
	17: IC: y17 = U17 (x = 100.00 %)	UNSIGNED_16	1/100	P				
0x3180	Position controller: Gain	UNSIGNED_32	1/100	P				
0x3181	Position controller: Gain - adaptation	UNSIGNED_16	1/100	P	OSC	Rx		
0x3182	Position controller: Output signal limitation	UNSIGNED_32	480000/ 2^{31}	P	OSC	Rx		
0x3183	Set actual position: Starting value	INTEGER_32	-	P	OSC	Rx		
0x3184	Set actual position: Mode	UNSIGNED_8	-	P				
0x3185	Position controller: Internal adaptation - characteristics							
	1: Speed 1	UNSIGNED_16	-	P				
	2: Vp 1	UNSIGNED_16	-	P				
	3: Speed 2	UNSIGNED_16	-	P				
	4: Vp 2	UNSIGNED_16	-	P				
	5: Speed 3	UNSIGNED_16	-	P				
	6: Vp 3	UNSIGNED_16	-	P				
	7: Speed 4	UNSIGNED_16	-	P				
	8: Vp 4	UNSIGNED_16	-	P				
	9: Speed 5	UNSIGNED_16	-	P				
	10: Vp 5	UNSIGNED_16	-	P				
	11: Activation of adapted curve	UNSIGNED_8	-	P				
0x3186	Position controller:Resulting gain adaptation	UNSIGNED_16	1/100		OSC			
greyed out = read-only access								

Index	Name	Data type	Factor	Attributes				
0x31C0	Field controller: Parameter							
	1: Field controller: Gain	UNSIGNED_32	1/100	P				
	2: Field controller: Reset time	UNSIGNED_16	1/10	P				
0x31E0	Field weakening controller: Parameter							
	1: Field weakening controller: Gain	UNSIGNED_32	1/1000	P				
	2: Field weakening controller: Reset time	UNSIGNED_32	1/10	P				
0x31E1	Field set value limitation	UNSIGNED_16	1/100	P	OSC	Rx		
0x31E2	DC link circuit voltage: Filter time	UNSIGNED_16	1/10	P				
0x31E3	Motor voltage act. value: Filter time	UNSIGNED_16	1/10	P				
0x31E4	Voltage reserve	UNSIGNED_8	-	P				
Axis B - V/f operation								
0x3300	VFC: V/f characteristic - shape	UNSIGNED_8	-	P				CINH
0x3301	VFC: V/f characteristic - define reference point							
	1: VFC: V/f characteristic - voltage at reference point	UNSIGNED_16	-	P				
	2: VFC: V/f characteristic - frequency at reference point	UNSIGNED_16	-	P				
0x3302	VFC: User-definable V/f characteristic - frequency grid points (x)							
	1: V/f: x1 = f01	INTEGER_16	-	P				
	2: V/f: x2 = f02	INTEGER_16	-	P				
	3: V/f: x3 = f03	INTEGER_16	-	P				
	4: V/f: x4 = f04	INTEGER_16	-	P				
	5: V/f: x5 = f05	INTEGER_16	-	P				
	6: V/f: x6 = f06	INTEGER_16	-	P				
	7: V/f: x7 = f07	INTEGER_16	-	P				
	8: V/f: x8 = f08	INTEGER_16	-	P				
	9: V/f: x9 = f09	INTEGER_16	-	P				
	10: V/f: x10 = f10	INTEGER_16	-	P				
	11: V/f: x11 = f11	INTEGER_16	-	P				
0x3303	VFC: User-definable V/f characteristic - voltage grid points (y)							
	1: V/f: y1 = U01 (x = f01)	UNSIGNED_32	1/100	P				
	2: V/f: y2 = U02 (x = f02)	UNSIGNED_32	1/100	P				
	3: V/f: y3 = U03 (x = f03)	UNSIGNED_32	1/100	P				
	4: V/f: y4 = U04 (x = f04)	UNSIGNED_32	1/100	P				
	5: V/f: y5 = U05 (x = f05)	UNSIGNED_32	1/100	P				
	6: V/f: y6 = U06 (x = f06)	UNSIGNED_32	1/100	P				
	7: V/f: y7 = U07 (x = f07)	UNSIGNED_32	1/100	P				
	8: V/f: y8 = U08 (x = f08)	UNSIGNED_32	1/100	P				
	9: V/f: y9 = U09 (x = f09)	UNSIGNED_32	1/100	P				
	10: V/f: y10 = U10 (x = f10)	UNSIGNED_32	1/100	P				
	11: V/f: y11 = U11 (x = f11)	UNSIGNED_32	1/100	P				
0x3304	VFC: Voltage vector control - current setpoint	UNSIGNED_32	1/100	P				
0x3305	VFC: Voltage vector control parameter							
	1: VFC: Voltage vector controller - gain	UNSIGNED_32	1/100	P				
	2: VFC: Voltage vector controller - reset time	UNSIGNED_32	1/100	P				
0x3306	VFC: Voltage boost	UNSIGNED_16	1/10	P				
0x3307	VFC: Load adjustment - parameter							
	1: VFC: Load adjustment - direction of rotation	UNSIGNED_8	-	P				CINH
	2: VFC: Load adjustment - value	UNSIGNED_32	1/100	P				
0x3308	VFC: Imax controller - parameter							
	1: VFC: Imax controller - gain	UNSIGNED_32	1/1000	P				
	2: VFC: Imax controller - reset time	UNSIGNED_32	1/10	P				
greyed out = read-only access								

Index	Name	Data type	Factor	Attributes				
0x3309	VFC: Slip compensation - parameter							
	1: VFC: Slip compensation - influence	INTEGER_16	1/100	P				
	2: VFC: Slip compensation - filter time	UNSIGNED_16	-	P				
0x330A	VFC: Oscillation damping - parameter							
	1: VFC: Oscillation damping - gain	INTEGER_16	-	P				
	2: VFC: Oscillation damping - filter time	UNSIGNED_16	-	P				
	3: VFC: Oscillation damping - limitation	UNSIGNED_16	1/10	P				
	4: VFC: Oscillation damping - final ramp frequency	UNSIGNED_8	-	P				
0x330B	VFC: Setpoint frequency	INTEGER_16	1/10		OSC		Tx	
0x330C	VFC: Override point of field weakening	INTEGER_16	1/10	P				
0x3380	DC-injection braking: Current	UNSIGNED_16	1/100	P				
0x33A0	Flying restart: Activate	UNSIGNED_8	-	P				
0x33A1	Flying restart: Current	UNSIGNED_16	-	P				
0x33A2	Flying restart: Start frequency	INTEGER_16	1/10	P				
0x33A3	Flying restart: Integration time	UNSIGNED_16	-	P				
0x33A4	Flying restart: Min. deviation	UNSIGNED_16	1/100	P				
0x33A5	Flying restart: Delay time	UNSIGNED_16	-	P				
0x33A6	Flying restart: Result							
	1: Flying restart: Determined speed [rpm]	INTEGER_16	-		OSC		Tx	
	2: Flying restart: Determined speed [n unit]	INTEGER_32	480000/ 2^{31}		OSC		Tx	
Axis B - motor settings								
0x3400	Motor control	UNSIGNED_8	-	P				CINH
0x3401	Motor: Common parameters							
	1: Motor: Number of pole pairs	UNSIGNED_8	-					
	2: Motor: Stator resistance (20°C)	UNSIGNED_32	1/10000	P				
	3: Motor: Stator leakage inductance	UNSIGNED_32	1/1000	P				
	4: Motor: Rated speed	UNSIGNED_16	-	P				
	5: Motor: Rated frequency	UNSIGNED_16	1/10	P				
	6: Motor: Rated power	UNSIGNED_16	1/100	P				
	7: Motor: Rated voltage	UNSIGNED_16	-	P				
	8: Motor: Rated cosine phi	UNSIGNED_16	1/100	P				
	9: Motor: Insulation class	UNSIGNED_8	-	P				
	10: Motor: Designation	STRING(50)	-	P				
0x3402	Motor (ASM): Parameter							
	1: Motor (ASM): Rotor resistance (20°C)	UNSIGNED_32	1/10000	P				
	2: Motor (ASM): Mutual inductance	UNSIGNED_32	1/10	P				
	3: Motor (ASM): Magnetising current	UNSIGNED_16	1/100	P				
0x3403	Motor (SM): Parameter							
	1: Motor (SM): e.m.f. constant (KE conductor-conductor, 20°C)	UNSIGNED_32	1/10	P				
	2: Motor (SM): Pole position	INTEGER_16	1/10	P				
	3: Motor (SM): Temperature coefficient - magnets (kTN)	INTEGER_16	1/1000	P				
	4: Motor (SM): Pole position	INTEGER_16	1/10	P				
greyed out = read-only access								

Index	Name	Data type	Factor	Attributes				
0x3404	Motor: Lss saturation characteristic - inductance grid points (y)							
	1: Lss: y1 = L01 (x = 0.00 %)	UNSIGNED_16	-	P				
	2: Lss: y2 = L02 (x = 6.25 %)	UNSIGNED_16	-	P				
	3: Lss: y3 = L03 (x = 12.50 %)	UNSIGNED_16	-	P				
	4: Lss: y4 = L04 (x = 18.75 %)	UNSIGNED_16	-	P				
	5: Lss: y5 = L05 (x = 25.00 %)	UNSIGNED_16	-	P				
	6: Lss: y6 = L06 (x = 31.25 %)	UNSIGNED_16	-	P				
	7: Lss: y7 = L07 (x = 37.50 %)	UNSIGNED_16	-	P				
	8: Lss: y8 = L08 (x = 42.75 %)	UNSIGNED_16	-	P				
	9: Lss: y9 = L09 (x = 50.00 %)	UNSIGNED_16	-	P				
	10: Lss: y10 = L10 (x = 56.25 %)	UNSIGNED_16	-	P				
	11: Lss: y11 = L11 (x = 62.50 %)	UNSIGNED_16	-	P				
	12: Lss: y12 = L12 (x = 68.75 %)	UNSIGNED_16	-	P				
	13: Lss: y13 = L13 (x = 75.00 %)	UNSIGNED_16	-	P				
	14: Lss: y14 = L14 (x = 81.25 %)	UNSIGNED_16	-	P				
	15: Lss: y15 = L15 (x = 87.50 %)	UNSIGNED_16	-	P				
	16: Lss: y16 = L16 (x = 93.25 %)	UNSIGNED_16	-	P				
	17: Lss: y17 = L17 (x = 100.00 %)	UNSIGNED_16	-	P				
	18: Motor: Lss saturation characteristic - activation	UNSIGNED_16	-	P				
0x3405	Motor: Lss saturation characteristic - reference for current grid points (x)	UNSIGNED_16	1/10	P				
0x3406	Motor (SM): Magnet characteristic (current) - grid points							
	1: Magnet characteristic: x1 = i01/iN	UNSIGNED_16	-	P				
	2: Magnet characteristic: y1 = kT01/kTN	UNSIGNED_16	-	P				
	3: Magnet characteristic: x2 = i02/iN	UNSIGNED_16	-	P				
	4: Magnet characteristic: y2 = kT02/kTN	UNSIGNED_16	-	P				
	5: Magnet characteristic: x3 = i03/iN	UNSIGNED_16	-	P				
	6: Magnet characteristic: y3 = kT03/kTN	UNSIGNED_16	-	P				
	7: Magnet characteristic: x4 = i04/iN	UNSIGNED_16	-	P				
	8: Magnet characteristic: y4 = kT04/kTN	UNSIGNED_16	-	P				
0x3407	Motor (ASM): Lh saturation characteristic - inductance grid points (y)							
	1: Lh: y1 = L01 (x = 0.00 %)	UNSIGNED_16	-	P				
	2: Lh: y2 = L02 (x = 6.25 %)	UNSIGNED_16	-	P				
	3: Lh: y3 = L03 (x = 12.50 %)	UNSIGNED_16	-	P				
	4: Lh: y4 = L04 (x = 18.75 %)	UNSIGNED_16	-	P				
	5: Lh: y5 = L05 (x = 25.00 %)	UNSIGNED_16	-	P				
	6: Lh: y6 = L06 (x = 31.25 %)	UNSIGNED_16	-	P				
	7: Lh: y7 = L07 (x = 37.50 %)	UNSIGNED_16	-	P				
	8: Lh: y8 = L08 (x = 43.75 %)	UNSIGNED_16	-	P				
	9: Lh: y9 = L09 (x = 50.00 %)	UNSIGNED_16	-	P				
	10: Lh: y10 = L10 (x = 56.25 %)	UNSIGNED_16	-	P				
	11: Lh: y11 = L11 (x = 62.50 %)	UNSIGNED_16	-	P				
	12: Lh: y12 = L12 (x = 68.75 %)	UNSIGNED_16	-	P				
	13: Lh: y13 = L13 (x = 75.00 %)	UNSIGNED_16	-	P				
	14: Lh: y14 = L14 (x = 81.25 %)	UNSIGNED_16	-	P				
	15: Lh: y15 = L15 (x = 87.50 %)	UNSIGNED_16	-	P				
	16: Lh: y16 = L16 (x = 93.75 %)	UNSIGNED_16	-	P				
	17: Lh: y17 = L17 (x = 100.00 %)	UNSIGNED_16	-	P				
0x3408	Motor: Motor parameter setting method	UNSIGNED_8	-	P				

greyed out = read-only access

Index	Name	Data type	Factor	Attributes				
0x3412	Sensorless synchronous control: Low speed range, controlled							
	1: Additive accelerating current	UNSIGNED_16	-	P				
	2: Standstill holding current	UNSIGNED_16	-	P				
0x3413	Sensorless synchronous control: Low speed range, operating mode	UNSIGNED_8	-	P				
Axis B - feedback system								
0x3440	Encoder: Type	UNSIGNED_8	-	P				CINH
0x3441	Hiperface: Parameter							
	1: Hiperface: Determined type code	UNSIGNED_8	-					
	2: Hiperface: User def. encoder - type code	UNSIGNED_8	-	P				CINH
	3: Hiperface: User def. encoder - specifiable revolutions	UNSIGNED_16	-	P				CINH
	4: Hiperface absolute value fault: Response	UNSIGNED_8	-	P				
	5: Hiperface: Serial number	STRING(50)	-					
	6: Hiperface: Raw data - Actual position	UNSIGNED_32	-				Tx	
	7: Hiperface: Detected Increments / revolution	UNSIGNED_16	-				Tx	
	8: Hiperface: Type code supported by firmware	UNSIGNED_8	-				Tx	
	9: Hiperface: Encoder type	UNSIGNED_8	-				Tx	
	10: Hiperface: Period length of linear encoders	UNSIGNED_32	-				Tx	
	11: Hiperface: Data block address	UNSIGNED_16	-					
	12: Hiperface: Data block length	UNSIGNED_8	-					
	13: Hiperface: Data block command	UNSIGNED_8	-					
	14: Hiperface: Data block status	UNSIGNED_16	-					
	15: Hiperface: Data block	BYTE	-					
0x3442	Encoder: Parameter							
	1: Encoder: Increments / revolution	UNSIGNED_32	-	P				CINH
	2: Encoder: Supply voltage	UNSIGNED_8	1/10	P				CINH
	3: Encoder: Angle drift - Actual angle error	INTEGER_16	1/10		OSC		Tx	
	4: Encoder: Signal quality - Actual amplitude	UNSIGNED_8	-		OSC			
0x3443	Resolver: Number of pole pairs	UNSIGNED_8	-	P				CINH
0x3444	Resolver error compensation: Parameter							
	1: Resolver error compensation: Angle	INTEGER_16	-	P				
	2: Resolver error compensation: Cosine track gain	UNSIGNED_16	-	P				
	3: Resolver error compensation: Sine track gain	UNSIGNED_16	-	P				
	4: Identification: search time for v = const.	UNSIGNED_16	-	P				
	5: Identification: Time-out	UNSIGNED_16	-	P				
	6: Identification: Actual status	UNSIGNED_16	-		OSC			
0x3445	Feedback system: Response to open circuit	UNSIGNED_8	-	P				
0x3446	Feedback system: Specifiable number of revolutions	UNSIGNED_16	-					
0x3447	Feedback system: Open circuit detection sensitivity	UNSIGNED_8	-	P				
0x3448	Internal							
	1: Internal: Response	UNSIGNED_8	-	P				
	2: Internal: Actual position (raw data)	UNSIGNED_32	-				Tx	
0x345F	Feedback system: Parameter CRC	UNSIGNED_32	-					
Axis B - pole position identification								
0x3460	Monitoring pole position identification: Response	UNSIGNED_8	-	P				
greyed out = read-only access								

Index	Name	Data type	Factor	Attributes				
0x3461	Pole position identification PPI (360°)							
	1: PPI (360°): Current amplitude	UNSIGNED_16	-	P				CINH
	2: PPI (360°): Ramp time	UNSIGNED_16	-	P				CINH
	3: PPI (360°): Direction of rotation	UNSIGNED_8	-	P				CINH
	4: PPI (360°): Fault tolerance	UNSIGNED_8	-	P				
	5: PPI (360°): Absolute current amplitude	UNSIGNED_32	1/100					
0x3462	Pole position identification PPI (min. movement)							
	1: PPI (min. movement): Current amplitude	UNSIGNED_16	-	P				CINH
	2: PPI (min. movement): Ramp time - current	UNSIGNED_16	-	P				CINH
	3: PPI (min. movement): Gain	UNSIGNED_16	-	P				
	4: PPI (min. movement): Reset time	UNSIGNED_16	1/10	P				
	5: PPI (min. movement): Max. move permitted	UNSIGNED_8	-	P				
	6: PPI (min. movement): Absolute current amplitude	UNSIGNED_32	1/100					
0x3463	Pole position identification PPI (without movement)							
	1: Behaviour after switch-on	UNSIGNED_8	-	P				CINH
0x3464	Cable check							
	1: Cable check: Behaviour after switch-on	UNSIGNED_8	-	P				CINH
	2: Cable check: Status word	UNSIGNED_16	-					
Axis B - touch probe								
0x3500	Touch probe (TP): Dead time compensation							
	1: Dead time compensation: TP1 delay time	UNSIGNED_16	1/1000	P				
	2: Dead time compensation: TP2 delay time	UNSIGNED_16	1/1000	P				
0x3501	Touch probe (TP): Time stamp							
	1: TP1: Time stamp - rising edge	UNSIGNED_32	-		OSC		Tx	
	2: TP1: Time stamp - falling edge	UNSIGNED_32	-		OSC		Tx	
	3: TP2: Time stamp - rising edge	UNSIGNED_32	-		OSC		Tx	
	4: TP2: Time stamp - falling edge	UNSIGNED_32	-		OSC		Tx	
Axis B - monitoring functions								
0x3540	Ixt utilisation							
	1: Power stage utilisation (Ixt): Actual utilisation	UNSIGNED_16	-		OSC		Tx	
	2: Power stage utilisation (Ixt): Warning threshold	UNSIGNED_16	-	P				
	3: Power stage utilisation (Ixt): Error threshold	UNSIGNED_16	-					
	4: Device utilisation (Ixt): Actual utilisation	UNSIGNED_16	-		OSC		Tx	
	5: Device utilisation (Ixt): Warning threshold	UNSIGNED_16	-	P				
	6: Device utilisation (Ixt): Error limit	UNSIGNED_16	-					
0x3544	Motor speed monitoring							
	1: Motor speed monitoring: Threshold	UNSIGNED_16	-	P				
	2: Motor speed monitoring: Response	UNSIGNED_8	-	P				
0x3545	Motor phase failure detection							
	1: Motor phase failure 1: Response	UNSIGNED_8	-	P				
	2: Motor phase failure 1: Current threshold	UNSIGNED_8	1/10	P				
	3: Motor phase failure 1: Voltage threshold	UNSIGNED_16	1/10	P				
	4: Motor phase failure 2: Response	UNSIGNED_8	-	P				
0x3546	Monitoring: Ultimate motor current							
	1: Ultimate motor current: Threshold	UNSIGNED_16	1/10	P				
	2: Ultimate motor current: Response	UNSIGNED_8	-	P				
greyed out = read-only access								

Index	Name	Data type	Factor	Attributes				
0x3549	Motor temperature monitoring: Parameter							
	1: Motor temperature monitoring: Sensor type	UNSIGNED_8	-	P				
	2: Motor temperature monitoring: Response	UNSIGNED_8	-	P				
	3: Motor temperature monitoring: Warning threshold	INTEGER_16	1/10	P				
	4: Motor temperature monitoring: Error threshold	INTEGER_16	1/10	P				
	5: Motor temperature monitoring: Actual motor temperature	INTEGER_16	1/10		OSC		Tx	
	6: Thermal sensor characteristic: Grid point 1 - temperature	INTEGER_16	1/10	P				
	7: Thermal sensor characteristic: Grid point 2 - temperature	INTEGER_16	1/10	P				
	8: Thermal sensor characteristic: grid point 1 - resistance	INTEGER_16	-	P				
	9: Thermal sensor characteristic: Grid point 2 - resistance	INTEGER_16	-	P				
0x354C	Motor utilisation (I^2xt): Parameter for the thermal model							
	1: Motor utilisation (I^2xt): Thermal time constant - winding	UNSIGNED_16	-	P				
	2: Motor utilisation (I^2xt): Thermal time constant - laminations	UNSIGNED_16	-	P				
	3: Motor utilisation (I^2xt): Influence of the winding	UNSIGNED_8	-	P				
	4: Motor utilisation (I^2xt): Starting value	UNSIGNED_16	-	P				
0x354D	Motor utilisation (I^2xt): User-definable characteristic							
	1: I^2xt: x1 = n01/nN (n01 ~ 0)	UNSIGNED_16	-	P				
	2: I^2xt: y1 = i01/iN (x = n01 ~ 0)	UNSIGNED_16	-	P				
	3: I^2xt: x2 = n02/nN (n02 = limit - reduced cooling)	UNSIGNED_16	-	P				
	4: I^2xt: y2 = i02/iN (x = n02 = limit - reduced cooling)	UNSIGNED_16	-	P				
	5: I^2xt: x3 = n03/nN (n03 = rated speed)	UNSIGNED_16	-	P				
	6: I^2xt: y3 = i03/iN (x = n03 = rated speed)	UNSIGNED_16	-	P				
	7: I^2xt: x4 = n04/nN (n04 = limit - field weakening)	UNSIGNED_16	-	P				
	8: I^2xt: y4 = i04/iN (x = n04 = limit - field weakening)	UNSIGNED_16	-	P				
0x354E	Motor utilisation (I^2xt): Motor overload warning threshold	UNSIGNED_16	-	P				
0x354F	Motor utilisation (I^2xt): Actual utilisation	UNSIGNED_16	-		OSC		Tx	
0x3550	Motor utilisation (I^2xt): Motor overload error							
	1: Motor utilisation (I^2xt): Response	UNSIGNED_8	-	P				
	2: Motor utilisation (I^2xt): Error threshold	UNSIGNED_16	-	P				
0x3551	Monitoring: Position and velocity error							
	1: Monitoring: Velocity error - error threshold	UNSIGNED_32	-	P				
	2: Monitoring: Velocity error - minimum error time	UNSIGNED_16	-	P				
	3: Monitoring: Velocity error - response	UNSIGNED_8	-	P				
	4: Monitoring: Position error - error threshold	UNSIGNED_32	-	P				
	5: Monitoring: Position error - minimum error time	UNSIGNED_16	-	P				
	6: Monitoring: Position error - response	UNSIGNED_8	-	P				
0x3581	Counter: Operating time							
	1: Device: Operating time	UNSIGNED_32	-					
	2: Device: Power-on time	UNSIGNED_32	-					
0x3582	Motor: Actual voltage - Veff, phase-phase	UNSIGNED_32	1/10		OSC		Tx	
greyed out = read-only access								

Index	Name	Data type	Factor	Attributes				
0x3583	Motor: Phase currents							
	1: Zero system current	INTEGER_32	1/100		OSC		Tx	
	2: Current - phase U	INTEGER_32	1/100		OSC		Tx	
	3: Current - phase V	INTEGER_32	1/100		OSC		Tx	
	4: Current - phase W	INTEGER_32	1/100		OSC		Tx	
0x3584	Heatsink temperature							
	1: Heatsink temperature: Actual temperature	INTEGER_16	1/10		OSC		Tx	
	2: Heatsink temperature: Warning threshold	INTEGER_16	1/10	P				
	3: Heatsink temperature: Threshold - switch-on fan	INTEGER_16	1/10	P				
	4: Heatsink temperature: Threshold - switch-off fan	INTEGER_16	1/10	P				
0x358A	Speed monitoring: Actual speed error	INTEGER_32	-		OSC		Tx	
0x35D0	Field: Values							
	1: Field: Actual field	UNSIGNED_16	-		OSC		Tx	
	2: Field: Setpoint field	UNSIGNED_16	-		OSC		Tx	
0x35D1	Motor: Currents							
	1: D current (id): Actual D current	INTEGER_16	1/100		OSC		Tx	
	2: Q current (iq): Actual Q current	INTEGER_16	1/100		OSC		Tx	
	3: D current (id): Setpoint D current	INTEGER_16	1/100		OSC		Tx	
	4: Q current (iq): Setpoint Q current	INTEGER_16	1/100		OSC		Tx	
	5: Motor current Ieff	INTEGER_16	1/100		OSC		Tx	
0x35D2	Position: Target position interpolated	INTEGER_32	-		OSC		Tx	
0x35D3	Target speeds							
	1: Speed: Setpoint speed	INTEGER_32	-		OSC		Tx	
	2: Speed: Setpoint speed 2	INTEGER_32	-		OSC		Tx	
	3: Speed: Limited setpoint speed	INTEGER_32	-		OSC		Tx	
0x35D4	Speed controller: Output signal							
	1: Speed controller: Output signal 1	INTEGER_16	1/10		OSC		Tx	
	2: Speed controller: Output signal 2	INTEGER_16	1/10		OSC		Tx	
0x35D5	Target torque	INTEGER_32	1/100		OSC		Tx	
0x35D6	Torque: Filter cascade							
	1: Torque: Filter cascade - starting value	INTEGER_16	1/10		OSC		Tx	
	2: Torque: Notch filter 1 - input value	INTEGER_16	1/10		OSC		Tx	
	3: Torque: Notch filter 2 - input value	INTEGER_16	1/10		OSC		Tx	
	4: Torque: Filtered setpoint torque	INTEGER_16	1/10		OSC		Tx	
0x35D7	Voltage values							
	1: Actual motor voltage limit: Actual voltage	INTEGER_16	1/10		OSC		Tx	
	2: D-current controller: Output signal	INTEGER_16	1/10		OSC		Tx	
	3: Q-current controller: Output signal	INTEGER_16	1/10		OSC		Tx	
	4: D-voltage (magnetisation)	INTEGER_16	1/10		OSC		Tx	
	5: Q-voltage (torque)	INTEGER_16	1/10		OSC		Tx	
0x35DC	Slip: Actual slip	INTEGER_16	1/10		OSC		Tx	
0x35DD	Device: Actual output frequency	INTEGER_16	1/10		OSC		Tx	
0x35DE	Motor: Actual position of rotor angle	INTEGER_16	-		OSC		Tx	
0x35DF	Axis: Device data							
	1: Axis: Rated current	UNSIGNED_16	1/100		OSC		Tx	
	2: Axis: Maximum current	UNSIGNED_16	1/100		OSC		Tx	
		UNSIGNED_16	$1/10^9$				Tx	
		UNSIGNED_16	$1/10^8$				Tx	
	5: Axis: Supported feedback type	UNSIGNED_8	-				Tx	
greyed out = read-only access								

Index	Name	Data type	Factor	Attributes				
0x35E0	Advanced settings							
	1: Current controller: Setting for identification	UNSIGNED_8	-					
	2: Sensorless synchronous control: Signal for test mode	UNSIGNED_8	-					
	3: Resolver: Position detection - dynamics	UNSIGNED_16	-					
	4: Resolver: 8 kHz safety signal	UNSIGNED_8	-					
	5: Torque: Internal interpolation	UNSIGNED_8	-	P				
	6: OEM service data: Serial number	UNSIGNED_32	-					
	14: Overwrite CiA control word bit 4	UNSIGNED_8	-	P				
Axis A - CiA402 profile specific area								
0x603F	Error code	UNSIGNED_16	-		OSC		Tx	
0x6040	Controlword	UNSIGNED_16	-		OSC	Rx		
0x6041	Statusword	UNSIGNED_16	-		OSC		Tx	
0x6042	vl target velocity	INTEGER_16	-		OSC	Rx		
0x6043	vl velocity demand	INTEGER_16	-		OSC		Tx	
0x6044	vl velocity actual value	INTEGER_16	-		OSC		Tx	
0x6046	vl velocity min max amount							
	1: Speed: Minimum vl	UNSIGNED_32	-		OSC	Rx		
	2: Speed: Maximum vl	UNSIGNED_32	-		OSC	Rx		
0x6048	vl velocity acceleration							
	1: Ramp: Speed interval	UNSIGNED_32	-	P	OSC	Rx		
	2: Ramp: Time interval	UNSIGNED_16	-	P	OSC	Rx		
0x6049	vl velocity deceleration							
	1: Ramp: Speed interval	UNSIGNED_32	-	P	OSC	Rx		
	2: Ramp: Time interval	UNSIGNED_16	-	P	OSC	Rx		
0x604B	vl setpoint factor							
	1: vl setpoint factor: numerator	INTEGER_16	-	P		Rx		
	2: vl setpoint factor: denominator	INTEGER_16	-	P		Rx		
0x604C	vl length factor							
	1: vl length factor: numerator	INTEGER_32	-	P		Rx		
	2: vl length factor: denominator	INTEGER_32	-	P		Rx		
0x605A	Quick stop option code	INTEGER_16	-	P				
0x605B	Behaviour in case of shutdown	INTEGER_16	-	P				
0x605E	Setting/response in the event of an error	INTEGER_16	-	P				
0x6060	Modes of operation	INTEGER_8	-	P	OSC	Rx		
0x6061	Modes of operation display	INTEGER_8	-		OSC		Tx	
0x6062	Position demand value	INTEGER_32	-		OSC			
0x6063	Position actual internal value	INTEGER_32	-		OSC		Tx	
0x6064	Position actual value	INTEGER_32	-		OSC		Tx	
0x6065	Following error window	UNSIGNED_32	-	P	OSC	Rx		
0x6066	Following error time out	UNSIGNED_16	-	P	OSC	Rx		
0x6067	Position window	UNSIGNED_32	-	P				
0x6068	Position window time	UNSIGNED_16	-	P				
0x606C	Velocity actual value	INTEGER_32	480000/ 2^{31}		OSC		Tx	
0x6071	Target torque	INTEGER_16	1/10		OSC	Rx		
0x6072	Max torque	UNSIGNED_16	1/10	P	OSC	Rx		
0x6073	Max current	UNSIGNED_16	1/10	P				
0x6074	Torque: Interpol. setpoint torque	INTEGER_16	1/10		OSC			
0x6075	Motor rated current	UNSIGNED_32	1/1000	P				CINH
0x6076	Motor rated torque	UNSIGNED_32	1/1000	P				CINH
0x6077	Torque actual value	INTEGER_16	1/10		OSC		Tx	
greyed out = read-only access								

Index	Name	Data type	Factor	Attributes				
0x6078	Current actual value	INTEGER_16	1/10		OSC		Tx	
0x6079	DC bus: Actual voltage	UNSIGNED_32	1/1000		OSC		Tx	
0x607A	Position demand value	INTEGER_32	-		OSC	Rx		
0x607E	Polarity	UNSIGNED_8	-	P				CINH
0x6080	Max motor speed	UNSIGNED_32	-	P	OSC	Rx		
0x6085	Quick stop deceleration	UNSIGNED_32	-	P				
0x608F	Position encoder resolution							
	1: Encoder increments	UNSIGNED_32	-	P				CINH
	2: Motor revolutions	UNSIGNED_32	-	P				CINH
0x6090	Velocity encoder resolution							
	1: Encoder increments per second	UNSIGNED_32	-	P				CINH
	2: Motor revolutions per second	UNSIGNED_32	-	P				CINH
0x60B1	Velocity offset	INTEGER_32	$480000/2^{31}$	P	OSC	Rx		
0x60B2	Torque offset	INTEGER_16	1/10	P	OSC	Rx		
0x60B8	Touch probe function	UNSIGNED_16	-	P	OSC	Rx		
0x60B9	Touch probe status	UNSIGNED_16	-		OSC		Tx	
0x60BA	Touch probe pos1 pos value	INTEGER_32	-		OSC		Tx	
0x60BB	Touch probe pos1 neg value	INTEGER_32	-		OSC		Tx	
0x60BC	Touch probe pos2 pos value	INTEGER_32	-		OSC		Tx	
0x60BD	Touch probe pos2 neg value	INTEGER_32	-		OSC		Tx	
0x60C0	Interpolation sub mode select	INTEGER_16	-	P		Rx		
0x60C2	Interpolation time period							
	1: Interpolation time period value	UNSIGNED_8	-	P				
	2: Interpolation time index	INTEGER_8	-	P				
0x60E0	Positive torque limit value	UNSIGNED_16	1/10	P	OSC	Rx		
0x60E1	Negative torque limit value	UNSIGNED_16	1/10	P	OSC	Rx		
0x60F4	Following error actual value	INTEGER_32	-		OSC		Tx	
0x60FA	Control effort	INTEGER_32	$480000/2^{31}$		OSC		Tx	
0x60FC	Position demand internal value	INTEGER_32	-		OSC			
0x60FD	Digital inputs	UNSIGNED_32	-		OSC		Tx	
0x60FF	Target velocity	INTEGER_32	$480000/2^{31}$		OSC	Rx		
0x6404	Motor manufacturer	STRING(50)	-	P				
0x6502	Supported drive modes	UNSIGNED_32	-					
0x67FF	ECAT: Device Profile Number	UNSIGNED_32	-					
Axis B - CiA402 profile specific area								
0x683F	Error code	UNSIGNED_16	-		OSC		Tx	
0x6840	Controlword	UNSIGNED_16	-		OSC	Rx		
0x6841	Statusword	UNSIGNED_16	-		OSC		Tx	
0x6842	vl target velocity	INTEGER_16	-		OSC	Rx		
0x6843	vl velocity demand	INTEGER_16	-		OSC		Tx	
0x6844	vl velocity actual value	INTEGER_16	-		OSC		Tx	
0x6846	vl velocity min max amount							
	1: Speed: Minimum vl	UNSIGNED_32	-		OSC	Rx		
	2: Speed: Maximum vl	UNSIGNED_32	-		OSC	Rx		
0x6848	vl velocity acceleration							
	1: Ramp: Speed interval	UNSIGNED_32	-	P	OSC	Rx		
	2: Ramp: Time interval	UNSIGNED_16	-	P	OSC	Rx		
greyed out = read-only access								

Index	Name	Data type	Factor	Attributes				
0x6849	vl velocity deceleration							
	1: Ramp: Speed interval	UNSIGNED_32	-	P	OSC	Rx		
	2: Ramp: Time interval	UNSIGNED_16	-	P	OSC	Rx		
0x684B	vl setpoint factor							
	1: vl setpoint factor: numerator	INTEGER_16	-	P		Rx		
	2: vl setpoint factor: denominator	INTEGER_16	-	P		Rx		
0x684C	vl length factor							
	1: vl length factor: numerator	INTEGER_32	-	P		Rx		
	2: vl length factor: denominator	INTEGER_32	-	P		Rx		
0x685A	Quick stop option code	INTEGER_16	-	P				
0x685B	Behaviour in case of shutdown	INTEGER_16	-	P				
0x685E	Setting/response in the event of an error	INTEGER_16	-	P				
0x6860	Modes of operation	INTEGER_8	-	P	OSC	Rx		
0x6861	Modes of operation display	INTEGER_8	-		OSC		Tx	
0x6862	Position demand value	INTEGER_32	-		OSC			
0x6863	Position actual internal value	INTEGER_32	-		OSC		Tx	
0x6864	Position actual value	INTEGER_32	-		OSC		Tx	
0x6865	Following error window	UNSIGNED_32	-	P	OSC	Rx		
0x6866	Following error time out	UNSIGNED_16	-	P	OSC	Rx		
0x6867	Position window	UNSIGNED_32	-	P				
0x6868	Position window time	UNSIGNED_16	-	P				
0x686C	Velocity actual value	INTEGER_32	$480000/2^{31}$		OSC		Tx	
0x6871	Target torque	INTEGER_16	1/10		OSC	Rx		
0x6872	Max torque	UNSIGNED_16	1/10	P	OSC	Rx		
0x6873	Max current	UNSIGNED_16	1/10	P				
0x6874	Torque: Interpol. setpoint torque	INTEGER_16	1/10		OSC			
0x6875	Motor rated current	UNSIGNED_32	1/1000	P				CINH
0x6876	Motor rated torque	UNSIGNED_32	1/1000	P				CINH
0x6877	Torque actual value	INTEGER_16	1/10		OSC		Tx	
0x6878	Current actual value	INTEGER_16	1/10		OSC		Tx	
0x6879	DC bus: Actual voltage	UNSIGNED_32	1/1000		OSC		Tx	
0x687A	Position demand value	INTEGER_32	-		OSC	Rx		
0x687E	Polarity	UNSIGNED_8	-	P				CINH
0x6880	Max motor speed	UNSIGNED_32	-	P	OSC	Rx		
0x6885	Quick stop deceleration	UNSIGNED_32	-	P				
0x688F	Position encoder resolution							
	1: Encoder increments	UNSIGNED_32	-	P				CINH
	2: Motor revolutions	UNSIGNED_32	-	P				CINH
0x6890	Velocity encoder resolution							
	1: Encoder increments per second	UNSIGNED_32	-	P				CINH
	2: Motor revolutions per second	UNSIGNED_32	-	P				CINH
0x68B1	Velocity offset	INTEGER_32	$480000/2^{31}$	P	OSC	Rx		
0x68B2	Torque offset	INTEGER_16	1/10	P	OSC	Rx		
0x68B8	Touch probe function	UNSIGNED_16	-	P	OSC	Rx		
0x68B9	Touch probe status	UNSIGNED_16	-		OSC		Tx	
0x68BA	Touch probe pos1 pos value	INTEGER_32	-		OSC		Tx	
0x68BB	Touch probe pos1 neg value	INTEGER_32	-		OSC		Tx	
0x68BC	Touch probe pos2 pos value	INTEGER_32	-		OSC		Tx	
0x68BD	Touch probe pos2 neg value	INTEGER_32	-		OSC		Tx	

greyed out = read-only access

Index	Name	Data type	Factor	Attributes				
0x68C0	Interpolation sub mode select	INTEGER_16	-	P		Rx		
0x68C2	Interpolation time period							
	1: Interpolation time period value	UNSIGNED_8	-	P				
	2: Interpolation time index	INTEGER_8	-	P				
0x68E0	Positive torque limit value	UNSIGNED_16	1/10	P	OSC	Rx		
0x68E1	Negative torque limit value	UNSIGNED_16	1/10	P	OSC	Rx		
0x68F4	Following error actual value	INTEGER_32	-		OSC		Tx	
0x68FA	Control effort	INTEGER_32	$480000/2^{31}$		OSC		Tx	
0x68FC	Position demand internal value	INTEGER_32	-		OSC			
0x68FD	Digital inputs	UNSIGNED_32	-		OSC		Tx	
0x68FF	Target velocity	INTEGER_32	$480000/2^{31}$		OSC	Rx		
0x6C04	Motor manufacturer	STRING(50)	-	P				
0x6D02	Supported drive modes	UNSIGNED_32	-					
0x6FFF	ECAT: Device Profile Number	UNSIGNED_32	-					
greyed out = read-only access								

11.2 Structure of the parameter set file

The parameter set file is designed as an extendable file. This means that it is possible to add header components later on without interrupting previous algorithms. Therefore the file is divided into three sections:

1. File header
   - The file header contains important information on the file as a whole, including its size, the checksum, the file header type, the number of data headers, and their position from the start of the file.
2. Data header
   - The data header contains information on a specific part of the data within the data area. These are for instance the data type and data interpretation and thus the type of data header, the data offset in the file, and the size of the data packet.
3. Data
   - The data is a densely packed arrangement of data, the sense of which only becomes clear when the type of data header specified is known.

11.2.1 File header

Byte offset

0	4	8	12	16	20	24	28	32
Type	Version	Size	Manufacturer	Product	Version	Data start	Data size	CRC

Byte offset	Meaning	Info
0	File type	File identification Currently only the following values are defined: • 2: Storage parameter set (the file exclusively contains parameters provided for persistence). • 3: Complete parameter set (the file contains all parameters).
4	File version	Version number of the file
8	File size	Total size of the file in [bytes]
12	Manufacturer's identification mark according to object directory	Identification of the device which this parameter set file belongs to. If the values do not comply with those of the object directory during the import, an attempt to execute the import is nevertheless carried out. Parameters which could not be imported can be identified later by means of the parameter set error report.
16	Product ID according to object directory	
20	Version identifier of the object directory	
24	Start of the data description area	Offset of the data description area in [bytes] (offset from the start of the file).
28	Size of the data description area	Size of the data description area in [bytes]
32	CRC32 (as defined in IEEE 802.3)	Cyclic Redundancy Check (CRC32) across the entire file; 0 must be entered in the checksum field at the time of calculation. ▶ Cyclic redundancy check (CRC) - parameter set comparison on the basis of the checksum (📖 24)

11.2.2 Data header

Byte offset

	0	4	8	
...	Type	Data start	Data size	...

Byte offset	Meaning	Info
0	Type of the data area	Identification of the data area Currently only the following values are defined: • 1: The data area contains parameter values. ▸ "PAR_ParamHeader" header type • 2: The data area contains error messages of the last import. ▸ "PAR_ParamErrorHeader" header type
4	Start of the data area	Offset of the data area in [bytes] (offset from the start of the file).
8	Size of the data area	Size of the data area in [bytes]

"PAR_ParamHeader" header type

If the identification "1" is given for the data area type, the parameter values are successively arranged in the following layout within the data area:

Byte offset

	0	2	4	6	
...	Data type size	Index	Subindex	Value	...

Byte offset	Meaning	Info
0	Data type size of the parameter	Number of bytes which are assigned by the parameter value.
2	Index of the parameter object	Index of the parameter in the object directory which the following parameter value belongs to.
4	Subindex of the parameter object	Subindex of the parameter in the object directory which the following parameter value belongs to.
6	Parameter value	The parameter value assigns the number of data bytes which are listed under "Data type size of the parameter". At least, however, always one byte.

"PAR_ParamErrorHeader" header type

If the identification "2" is given for the data area type, the error messages are successively arranged in the following layout within the data area:

Byte offset

	0	4	6	
...	Error code	Index	Subindex	...

Byte offset	Meaning	Info
0	Error code in accordance with CiA301 or EtherCAT	SDO abort code which was generated when the object with the following index/subindex combination was accessed. ▸ SDO abort codes (Abort codes) (📖 30)
4	Index of the parameter object	Index of the parameter in the object directory which the error code belongs to.
6	Subindex of the parameter object	Subindex of the parameter in the object directory which the error code belongs to.

11.3 Communication objects

This chapter describes the "invisible" communication objects of the i700 servo inverters. This information is important for the integration of the controller with an external control.

Objects described in this chapter

All objects are of "RECORD" data type.

Object		Name
Axis A	Axis B	
PDO mapping		
0x1600	0x1610	RPDO-->Axis A/B: Cyclic sync position mode (csp)
0x1601	0x1611	RPDO-->Axis A/B: Cyclic sync torque mode (cst)
0x1602	0x1612	RPDO-->Axis A/B: Cyclic sync velocity mode (csv)
0x1603	0x1613	RPDO-->Axis A/B: Velocity mode (vl)
0x1604	0x1614	RPDO-->Axis A/B: Touch probe (TP)
0x1605	0x1615	RPDO-->Axis A/B: Freely configurable (user)
0x1606	0x1616	RPDO-->Axis A/B: Torque limits
0x1607	0x1617	RPDO-->Axis A/B: Speed limit values
0x1A00	0x1A10	Axis A/B-->TPDO: Cyclic sync position mode (csp)
0x1A01	0x1A11	Axis A/B-->TPDO: Cyclic sync torque mode (cst)
0x1A02	0x1A12	Axis A/B-->TPDO: Cyclic sync velocity mode (csv)
0x1A03	0x1A13	Axis A/B-->TPDO: Velocity mode (vl)
0x1A04	0x1A14	Axis A/B-->TPDO: Touch probe (TP)
0x1A05	0x1A15	Axis A/B-->TPDO: Freely configurable (user)
0x1A06	0x1A16	Axis A/B-->TPDO: Additional status information
Sync manager		
0x1C00		Sync Manager: Communication type
0x1C12		Sync Manager 2 (RPDO-->Device): PDO mapping
0x1C13		Sync Manager 3 (RPDO-->device): PDO mapping
0x1C32		Sync Manager 2 (RPDO-->Device): Parameter
0x1C33		Sync Manager 3 (Device-->TPDO): Parameter

0x1600 - RPDO-->Axis A: Cyclic sync position mode (csp)

Fixed, preconfigured PDO mapping object for "Cyclic sync position mode"

Sub.	Name	Lenze setting	Data type
1	(csp) RPDO-->A: 0x6040 (Controlword)	0x60400010	UNSIGNED_32
2	(csp) RPDO-->A: 0x2830 (Lenze control word)	0x28300010	UNSIGNED_32
3	(csp) RPDO-->A: 0x6060 (Modes of operation)	0x60600008	UNSIGNED_32
4	(csp) RPDO-->A: 0x60B2 (Torque: Offset)	0x60B20010	UNSIGNED_32
5	(csp) RPDO-->A: 0x607A (Target position)	0x607A0020	UNSIGNED_32
6	(csp) RPDO-->A: 0x60B1 (Velocity offset)	0x60B10020	UNSIGNED_32
7	(csp) RPDO-->A: 0x2902 (speed controller - load starting value)	0x29020010	UNSIGNED_32
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

0x1601 - RPDO-->Axis A: Cyclic sync torque mode (cst)

Fixed, preconfigured PDO mapping object for "Cyclic sync torque mode"

Sub.	Name	Lenze setting	Data type
1	(cst) RPDO-->A: 0x6040 (Controlword)	0x60400010	UNSIGNED_32
2	(cst) RPDO-->A: 0x2830 (Lenze control word)	0x28300010	UNSIGNED_32
3	(cst) RPDO-->A: 0x6060 (Modes of operation)	0x60600008	UNSIGNED_32
4	(cst) RPDO-->A: 0x60B2 (Torque: Offset)	0x60B20010	UNSIGNED_32
5	(cst) RPDO-->A: 0x6071 (Target torque)	0x60710010	UNSIGNED_32
6	(cst) RPDO-->A: 0x2946/1 (upper speed limit)	0x29460120	UNSIGNED_32
7	(cst) RPDO-->A: 0x2946/2 (lower speed limit)	0x29460220	UNSIGNED_32
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

0x1602 - RPDO-->Axis A: Cyclic sync velocity mode (csv)

Fixed, preconfigured PDO mapping object for "Cyclic sync velocity mode"

Sub.	Name	Lenze setting	Data type
1	(csv) RPDO-->A: 0x6040 (Controlword)	0x60400010	UNSIGNED_32
2	(csv) RPDO-->A: 0x2830 (Lenze control word)	0x28300010	UNSIGNED_32
3	(csv) RPDO-->A: 0x6060 (Modes of operation)	0x60600008	UNSIGNED_32
4	(csv) RPDO-->A: 0x60B2 (Torque: Offset)	0x60B20010	UNSIGNED_32
5	(csv) RPDO-->A: 0x60FF (Target velocity)	0x60FF0020	UNSIGNED_32
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

0x1603 - RPDO-->Axis A: Velocity mode (vl)

Fixed, preconfigured PDO mapping object for "Velocity mode"

Sub.	Name	Lenze setting	Data type
1	(vl) RPDO-->A: 0x6040 (Controlword)	0x60400010	UNSIGNED_32
2	(vl) RPDO-->A: 0x2830 (Lenze control word)	0x28300010	UNSIGNED_32
3	(vl) RPDO-->A: 0x6060 (Modes of operation)	0x60600008	UNSIGNED_32
4	(vl) RPDO-->A: 0x6042 (vl target velocity)	0x60420010	UNSIGNED_32
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

0x1604 - RPDO-->Axis A: Touch probe (TP)

Fixed, preconfigured PDO mapping object for touch probe detection

Sub.	Name	Lenze setting	Data type
1	(TP) RPDO-->A: 0x60B8 (Touch probe function)	0x60B80010	UNSIGNED_32
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

0x1605 - RPDO-->Axis A: Freely configurable (user)

PDO mapping object freely configurable by the user for process data from the controller to the servo inverter

Sub.	Name	Lenze setting	Data type
1	(user) RPDO-->A: xxxx freely configurable object 1	0x00000000	UNSIGNED_32
2	(User) RPDO-->A: xxxx freely configurable object 2	0x00000000	UNSIGNED_32
3	(User) RPDO-->A: xxxx freely configurable object 3	0x00000000	UNSIGNED_32
4	(User) RPDO-->A: xxxx freely configurable object 4	0x00000000	UNSIGNED_32
5	(User) RPDO-->A: xxxx freely configurable object 5	0x00000000	UNSIGNED_32
6	(User) RPDO-->A: xxxx freely configurable object 6	0x00000000	UNSIGNED_32
7	(User) RPDO-->A: xxxx freely configurable object 7	0x00000000	UNSIGNED_32
8	(User) RPDO-->A: xxxx freely configurable object 8	0x00000000	UNSIGNED_32
☑ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

0x1606 - RPDO-->Axis A: Torque limits

Fixed, preconfigured PDO mapping object for torque limits

Sub.	Name	Lenze setting	Data type
1	(torque) RPDO-->A: 0x60E0 (Positive torque limit value)	0x60E00010	UNSIGNED_32
2	(torque) RPDO-->A: 0x60E1 (Negative torque limit value)	0x60E10010	UNSIGNED_32
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

0x1607 - RPDO --> Axis A: Speed Limits

Sub.	Name	Lenze setting	Data type
1	(Speed) RPDO-->A: 0x2946/1 (upper speed limit)	692453664	UNSIGNED_32
2	(Speed) RPDO-->A: 0x2946/2 Lower speed limit	692453920	UNSIGNED_32
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

0x1610 - RPDO-->Axis B: Cyclic sync position mode (csp)

Fixed, preconfigured PDO mapping object for "Cyclic sync position mode"

Sub.	Name	Lenze setting	Data type
1	(csp) RPDO-->B: 0x6840 (Controlword)	0x68400010	UNSIGNED_32
2	(csp) RPDO-->B: 0x3030 (Lenze control word)	0x30300010	UNSIGNED_32
3	(csp) RPDO-->B: 0x6860 (Modes of operation)	0x68600008	UNSIGNED_32
4	(csp) RPDO-->B: 0x68B2 (Torque: Offset)	0x68B20010	UNSIGNED_32
5	(csp) RPDO-->B: 0x687A (Target position)	0x687A0020	UNSIGNED_32
6	(csp) RPDO-->B: 0x68B1 (Velocity offset)	0x68B10020	UNSIGNED_32
7	(csp) RPDO-->B: 0x3102 speed controller - load starting value	0x31020010	UNSIGNED_32
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

_ _

0x1611 - RPDO-->Axis B: Cyclic sync torque mode (cst)

Fixed, preconfigured PDO mapping object for "Cyclic sync torque mode"

Sub.	Name	Lenze setting	Data type
1	(cst) RPDO-->B: 0x6840 (Controlword)	0x68400010	UNSIGNED_32
2	(cst) RPDO-->B: 0x3030 (Lenze control word)	0x30300010	UNSIGNED_32
3	(cst) RPDO-->B: 0x6860 (Modes of operation)	0x68600008	UNSIGNED_32
4	(cst) RPDO-->B: 0x68B2 (Torque: Offset)	0x68B20010	UNSIGNED_32
5	(cst) RPDO-->B: 0x6871 (Target torque)	0x68710010	UNSIGNED_32
6	(cst) RPDO-->B: 0x3146/1 upper speed limit	0x31460120	UNSIGNED_32
7	(cst) RPDO-->B: 0x3146/2 lower speed limit	0x31460220	UNSIGNED_32
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

0x1612 - RPDO-->Axis B: Cyclic sync velocity mode (csv)

Fixed, preconfigured PDO mapping object for "Cyclic sync velocity mode"

Sub.	Name	Lenze setting	Data type
1	(csv) RPDO-->B: 0x6840 (Controlword)	0x68400010	UNSIGNED_32
2	(csv) RPDO-->B: 0x3030 (Lenze control word)	0x30300010	UNSIGNED_32
3	(csv) RPDO-->B: 0x6860 (Modes of operation)	0x68600008	UNSIGNED_32
4	(csv) RPDO-->B: 0x68B2 (Torque: Offset)	0x68B20010	UNSIGNED_32
5	(csv) RPDO-->B: 0x68FF (Target velocity)	0x68FF0020	UNSIGNED_32
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

0x1613 - RPDO-->Axis B: Velocity mode (vl)

Fixed, preconfigured PDO mapping object for "Velocity mode"

Sub.	Name	Lenze setting	Data type
1	(vl) RPDO-->B: 0x6840 (Controlword)	0x68400010	UNSIGNED_32
2	(vl) RPDO-->B: 0x3030 (Lenze control word)	0x30300010	UNSIGNED_32
3	(vl) RPDO-->B: 0x6860 (Modes of operation)	0x68600008	UNSIGNED_32
4	(vl) RPDO-->B: 0x6842 (vl target velocity)	0x68420010	UNSIGNED_32
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

0x1614 - RPDO-->Axis B: Touch probe (TP)

Fixed, preconfigured PDO mapping object for touch probe detection

Sub.	Name	Lenze setting	Data type
1	(TP) RPDO-->B: 0x68B8 (Touch probe function)	0x68B80010	UNSIGNED_32
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

0x1615 - RPDO-->Axis B: Freely configurable (user)

PDO mapping object freely configurable by the user for process data from the controller to the servo inverter i700

Sub.	Name	Lenze setting	Data type
1	(user) RPDO-->B: xxxx freely configurable object 1	0x00000000	UNSIGNED_32
2	(User) RPDO-->B: xxxx freely configurable object 2	0x00000000	UNSIGNED_32
3	(User) RPDO-->B: xxxx freely configurable object 3	0x00000000	UNSIGNED_32
4	(User) RPDO-->B: xxxx freely configurable object 4	0x00000000	UNSIGNED_32
5	(User) RPDO-->B: xxxx freely configurable object 5	0x00000000	UNSIGNED_32
6	(User) RPDO-->B: xxxx freely configurable object 6	0x00000000	UNSIGNED_32
7	(User) RPDO-->B: xxxx freely configurable object 7	0x00000000	UNSIGNED_32
8	(User) RPDO-->B: xxxx freely configurable object 8	0x00000000	UNSIGNED_32
☑ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

0x1616 - RPDO-->Axis B: Torque limits

Fixed, preconfigured PDO mapping object for torque limits

Sub.	Name	Lenze setting	Data type
1	(torque) RPDO-->B: 0x68E0 (Positive torque limit value)	0x68E00010	UNSIGNED_32
2	(torque) RPDO-->B: 0x68E1 (Negative torque limit value)	0x68E10010	UNSIGNED_32
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

0x1617 - RPDO --> Axis B: Speed limits

Sub.	Name	Lenze setting	Data type
1	(Speed) RPDO-->B: 0x3146/1 (upper speed limit)	826671392	UNSIGNED_32
2	(Speed) RPDO-->B: 0x3146/2 (Lower speed limit)	826671648	UNSIGNED_32
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

0x1A00 - Axis A-->TPDO: Cyclic sync position mode (csp)

Fixed, preconfigured PDO mapping object for "Cyclic sync position mode"

Sub.	Name	Lenze setting	Data type
1	(csp) A-->TPDO: 0x6041 (Statusword)	0x60410010	UNSIGNED_32
2	(csp) A-->TPDO: 0x2831 (Lenze status word)	0x28310010	UNSIGNED_32
3	(csp) A-->TPDO: 0x6061 (Modes of operation display)	0x60610008	UNSIGNED_32
4	(csp) A-->TPDO: 0x603F (Error code)	0x603F0010	UNSIGNED_32
5	(csp) A-->TPDO: 0x606C (Velocity actual value)	0x606C0020	UNSIGNED_32
6	(csp) A-->TPDO: 0x6077 (Torque actual value)	0x60770010	UNSIGNED_32
7	(csp) A-->TPDO: 0x6064 (Position actual value)	0x60640020	UNSIGNED_32
8	(csp) A-->TPDO: 0x60F4 (Following error actual value)	0x60F40020	UNSIGNED_32
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

0x1A01 - Axis A-->TPDO: Cyclic sync torque mode (cst)

Fixed, preconfigured PDO mapping object for "Cyclic sync torque mode"

Sub.	Name	Lenze setting	Data type
1	(cst) A-->TPDO: 0x6041 (Statusword)	0x60410010	UNSIGNED_32
2	(cst) A-->TPDO: 0x2831 (Lenze status word)	0x28310010	UNSIGNED_32
3	(cst) A-->TPDO: 0x6061 (Modes of operation display)	0x60610008	UNSIGNED_32
4	(cst) A-->TPDO: 0x603F (Error code)	0x603F0010	UNSIGNED_32
5	(cst) A-->TPDO: 0x606C (Velocity actual value)	0x606C0020	UNSIGNED_32
6	(cst) A-->TPDO: 0x6077 (Torque actual value)	0x60770010	UNSIGNED_32
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

0x1A02 - Axis A-->TPDO: Cyclic sync velocity mode (csv)

Fixed, preconfigured PDO mapping object for "Cyclic sync velocity mode"

Sub.	Name	Lenze setting	Data type
1	(csv) A-->TPDO: 0x6041 (Statusword)	0x60410010	UNSIGNED_32
2	(csv) A-->TPDO: 0x2831 (Lenze status word)	0x28310010	UNSIGNED_32
3	(csv) A-->TPDO: 0x6061 (Modes of operation display)	0x60610008	UNSIGNED_32
4	(csv) A-->TPDO: 0x603F (Error code)	0x603F0010	UNSIGNED_32
5	(csv) A-->TPDO: 0x606C (Velocity actual value)	0x606C0020	UNSIGNED_32
6	(csv) A-->TPDO: 0x6077 (Torque actual value)	0x60770010	UNSIGNED_32
7	(csv) A-->TPDO: 0x6064 (Position actual value)	0x60640020	UNSIGNED_32
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

0x1A03 - Axis A-->TPDO: Velocity mode (vl)

Fixed, preconfigured PDO mapping object for "Velocity mode"

Sub.	Name	Lenze setting	Data type
1	(vl) A-->TPDO: 0x6041 (Statusword)	0x60410010	UNSIGNED_32
2	(vl) A-->TPDO: 0x2831 (Lenze status word)	0x28310010	UNSIGNED_32
3	(vl) A-->TPDO: 0x6061 (Modes of operation display)	0x60610008	UNSIGNED_32
4	(vl) A-->TPDO: 0x603F (Error code)	0x603F0010	UNSIGNED_32
5	(vl) A-->TPDO: 0x6044 (vl velocity actual value)	0x60440010	UNSIGNED_32
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

0x1A04 - Axis A-->TPDO: Touch probe (TP)

Fixed, preconfigured PDO mapping object for touch probe detection

Sub.	Name	Lenze setting	Data type
1	(TP) A-->TPDO: 0x60B9 (Touch probe status)	0x60B90010	UNSIGNED_32
2	(TP) A-->TPDO: 0x60BA (Touch probe pos1 pos value)	0x60BA0020	UNSIGNED_32
3	(TP) A-->TPDO: 0x60BB (Touch probe pos1 neg value)	0x60BB0020	UNSIGNED_32
4	(TP) A-->TPDO: 0x60BC (Touch probe pos2 pos value)	0x60BC0020	UNSIGNED_32
5	(TP) A-->TPDO: 0x60BD (Touch probe pos2 neg value)	0x60BD0020	UNSIGNED_32
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

0x1A05 - Axis A-->TPDO: Freely configurable (user)

PDO mapping object freely configurable by the user for process data from the i700 servo inverter to the controller

Sub.	Name	Lenze setting	Data type
1	(user) A-->TPDO: xxxx freely configurable object 1	0x00000000	UNSIGNED_32
2	(User) A-->TPDO: xxxx freely configurable object 2	0x00000000	UNSIGNED_32
3	(User) A-->TPDO: xxxx freely configurable object 3	0x00000000	UNSIGNED_32
4	(User) A-->TPDO: xxxx freely configurable object 4	0x00000000	UNSIGNED_32
5	(User) A-->TPDO: xxxx freely configurable object 5	0x00000000	UNSIGNED_32
6	(User) A-->TPDO: xxxx freely configurable object 6	0x00000000	UNSIGNED_32
7	(User) A-->TPDO: xxxx freely configurable object 7	0x00000000	UNSIGNED_32
8	(User) A-->TPDO: xxxx freely configurable object 8	0x00000000	UNSIGNED_32
☑ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

0x1A06 - Axis A-->TPDO: Additional status information

Sub.	Name	Lenze setting	Data type
1	(State) A-->TPDO: 0x2833 Lenze Statusword 2	674430992	UNSIGNED_32
2	(State) A-->TPDO: 0x60FD Digital inputs	1627193376	UNSIGNED_32
3	(State) A-->TPDO: xxxx Free config. object 1	0	UNSIGNED_32
4	(Status) A-->TPDO: xxxx freely configurable object 2	0	UNSIGNED_32
5	(Status) A-->TPDO: xxxx freely configurable object 3	0	UNSIGNED_32
6	(Status) A-->TPDO: xxxx freely configurable object 4	0	UNSIGNED_32
7	(Status) A-->TPDO: xxxx freely configurable object 5	0	UNSIGNED_32
8	(State) A-->TPDO: xxxx Free config. object 6	0	UNSIGNED_32
☑ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

0x1A10 - Axis B-->TPDO: Cyclic sync position mode (csp)

Fixed, preconfigured PDO mapping object for "Cyclic sync position mode"

Sub.	Name	Lenze setting	Data type
1	(csp) B-->TPDO: 0x6841 (Statusword)	0x68410010	UNSIGNED_32
2	(csp) B-->TPDO: 0x3031 (Lenze status word)	0x30310010	UNSIGNED_32
3	(csp) B-->TPDO: 0x6861 (Modes of operation display)	0x68610008	UNSIGNED_32
4	(csp) B-->TPDO: 0x683F (Error code)	0x683F0010	UNSIGNED_32
5	(csp) B-->TPDO: 0x686C (Velocity actual value)	0x686C0020	UNSIGNED_32
6	(csp) B-->TPDO: 0x6877 (Torque actual value)	0x68770010	UNSIGNED_32
7	(csp) B-->TPDO: 0x6864 (Position actual value)	0x68640020	UNSIGNED_32
8	(csp) B-->TPDO: 0x68F4 (Following error actual value)	0x68F40020	UNSIGNED_32
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

0x1A11 - Axis B-->TPDO: Cyclic sync torque mode (cst)

Fixed, preconfigured PDO mapping object for "Cyclic sync torque mode"

Sub.	Name	Lenze setting	Data type
1	(cst) B-->TPDO: 0x6841 (Statusword)	0x68410010	UNSIGNED_32
2	(cst) B-->TPDO: 0x3031 (Lenze status word)	0x30310010	UNSIGNED_32
3	(cst) B-->TPDO: 0x6861 (Modes of operation display)	0x68610008	UNSIGNED_32
4	(cst) B-->TPDO: 0x683F (Error code)	0x683F0010	UNSIGNED_32
5	(cst) B-->TPDO: 0x686C (Velocity actual value)	0x686C0020	UNSIGNED_32
6	(cst) B-->TPDO: 0x6877 (Torque actual value)	0x68770010	UNSIGNED_32
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

0x1A12 - Axis B-->TPDO: Cyclic sync velocity mode (csv)

Fixed, preconfigured PDO mapping object for "Cyclic sync velocity mode"

Sub.	Name	Lenze setting	Data type
1	(csv) B-->TPDO: 0x6841 (Statusword)	0x68410010	UNSIGNED_32
2	(csv) B-->TPDO: 0x3031 (Lenze status word)	0x30310010	UNSIGNED_32
3	(csv) B-->TPDO: 0x6861 (Modes of operation display)	0x68610008	UNSIGNED_32
4	(csv) B-->TPDO: 0x683F (Error code)	0x683F0010	UNSIGNED_32
5	(csv) B-->TPDO: 0x686C (Velocity actual value)	0x686C0020	UNSIGNED_32
6	(csv) B-->TPDO: 0x6877 (Torque actual value)	0x68770010	UNSIGNED_32
7	(csv) B-->TPDO: 0x6864 (Position actual value)	0x68640020	UNSIGNED_32
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

0x1A13 - Axis B-->TPDO: Velocity mode (vl)

Fixed, preconfigured PDO mapping object for "Velocity mode"

Sub.	Name	Lenze setting	Data type
1	(vl) B-->TPDO: 0x6841 (Statusword)	0x68410010	UNSIGNED_32
2	(vl) B-->TPDO: 0x3031 (Lenze status word)	0x30310010	UNSIGNED_32
3	(vl) B-->TPDO: 0x6861 (Modes of operation display)	0x68610008	UNSIGNED_32
4	(vl) B-->TPDO: 0x683F (Error code)	0x683F0010	UNSIGNED_32
5	(vl) B-->TPDO: 0x6844 (vl velocity actual value)	0x68440010	UNSIGNED_32
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

0x1A14 - Axis B-->TPDO: Touch probe (TP)

Fixed, preconfigured PDO mapping object for touch probe detection

Sub.	Name	Lenze setting	Data type
1	(TP) B-->TPDO: 0x68B9 (Touch probe status)	0x68B90010	UNSIGNED_32
2	(TP) B-->TPDO: 0x68BA (Touch probe pos1 pos value)	0x68BA0020	UNSIGNED_32
3	(TP) B-->TPDO: 0x68BB (Touch probe pos1 neg value)	0x68BB0020	UNSIGNED_32
4	(TP) B-->TPDO: 0x68BC (Touch probe pos2 pos value)	0x68BC0020	UNSIGNED_32
5	(TP) B-->TPDO: 0x68BD (Touch probe pos2 neg value)	0x68BD0020	UNSIGNED_32
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

0x1A15 - Axis B-->TPDO: Freely configurable (user)

PDO mapping object freely configurable by the user for process data from the i700 servo inverter to the controller

Sub.	Name	Lenze setting	Data type
1	(user) B-->TPDO: xxxx freely configurable object 1	0x00000000	UNSIGNED_32
2	(User) B-->TPDO: xxxx freely configurable object 2	0x00000000	UNSIGNED_32
3	(User) B-->TPDO: xxxx freely configurable object 3	0x00000000	UNSIGNED_32
4	(User) B-->TPDO: xxxx freely configurable object 4	0x00000000	UNSIGNED_32
5	(User) B-->TPDO: xxxx freely configurable object 5	0x00000000	UNSIGNED_32
6	(User) B-->TPDO: xxxx freely configurable object 6	0x00000000	UNSIGNED_32
7	(User) B-->TPDO: xxxx freely configurable object 7	0x00000000	UNSIGNED_32
8	(User) B-->TPDO: xxxx freely configurable object 8	0x00000000	UNSIGNED_32
☑ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

0x1A16 - Axis B-->TPDO: Additional status information

Sub.	Name	Lenze setting	Data type
1	(State) B-->TPDO: 0x3033 Lenze Statusword 2	808648720	UNSIGNED_32
2	(State) B-->TPDO: 0x68FD Digital inputs	1761411104	UNSIGNED_32
3	(State) B-->TPDO: xxxx Free config. object 1	0	UNSIGNED_32
4	(Status) B-->TPDO: xxxx freely configurable object 2	0	UNSIGNED_32
5	(Status) B-->TPDO: xxxx freely configurable object 3	0	UNSIGNED_32
6	(Status) B-->TPDO: xxxx freely configurable object 4	0	UNSIGNED_32
7	(State) B-->TPDO: xxxx Free config. object 5	0	UNSIGNED_32
8	(State) B-->TPDO: xxxx Free config. object 6	0	UNSIGNED_32
☑ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

0x1C00 - Sync Manager: Communication type

Sub.	Name	Lenze setting	Data type
1	SM1: Communication type	1: Receive mailbox	UNSIGNED_8
2	SM2: Communication type	2: Transmit mailbox	UNSIGNED_8
3	SM3: Communication type	3: Transit process data	UNSIGNED_8
4	SM4: Communication type	4: Receive process data	UNSIGNED_8
☐ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

0x1C12 - Sync Manager 2 (RPDO-->Device): PDO mapping

Sub.	Name	Lenze setting	Data type
1	SM2 (RPDO-->Device): Assignment PDO 1	0x1600	UNSIGNED_16
2	SM2 (RPDO-->device): PDO assignment 2	0x1604	UNSIGNED_16
3	SM2 (RPDO-->device): PDO assignment 3	0x1606	UNSIGNED_16
4	SM2 (RPDO-->device): PDO assignment 4	0x1607	UNSIGNED_16
5	SM2 (RPDO-->device): PDO assignment 5	0x1610	UNSIGNED_16
6	SM2 (RPDO-->device): PDO assignment 6	0x1614	UNSIGNED_16
7	SM2 (RPDO-->device): PDO assignment 7	0x1616	UNSIGNED_16
8	SM2 (RPDO-->device): PDO assignment 8	0x1617	UNSIGNED_16
9	SM2 (RPDO-->Device): PDO assignment 9	0x0000	UNSIGNED_16
10	SM2 (RPDO-->Device): PDO assignment 10	0x0000	UNSIGNED_16
☑ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

0x1C13 - Sync Manager 3 (RPDO-->Device): PDO mapping

Sub.	Name	Lenze setting	Data type
1	SM3 (Device-->TPDO): Assignment PDO 1	0x1A00	UNSIGNED_16
2	SM3 (device-->TPDO): PDO assignment 2	0x1A04	UNSIGNED_16
3	SM3 (device-->TPDO): PDO assignment 3	0x1A06	UNSIGNED_16
4	SM3 (device-->TPDO): PDO assignment 4	0x1A10	UNSIGNED_16
5	SM3 (device-->TPDO): PDO assignment 5	0x1A14	UNSIGNED_16
6	SM3 (Device-->TPDO): Assignment PDO 6	0x1A16	UNSIGNED_16
7	SM3 (Device-->TPDO): PDO assignment 7	0x0000	UNSIGNED_16
8	SM3 (Device-->TPDO): PDO assignment 8	0x0000	UNSIGNED_16
☑ Write access ☐ CINH ☐ OSC ☐ P ☐ RX ☐ TX			

0x1C32 - Sync Manager 2 (RPDO-->Device): Parameter

Sub.	Name	Lenze setting	Data type
1	SM2 (RPDO-->Device): Sync type	0: Free Run	UNSIGNED_16
2	SM2 (RPDO-->Device): Cycle time	250000 ns	UNSIGNED_32
3	SM2 (RPDO-->Device): Shift time	0 ns	UNSIGNED_32
4	SM2 (RPDO-->Device): Supported sync types	0x0825	UNSIGNED_16
5	SM2 (RPDO-->Device): Min. cycle time	125000 ns	UNSIGNED_32
6	SM2 (RPDO-->Device): Signal processing time	0 ns	UNSIGNED_32
9	SM2 (RPDO-->Device): Delay time	0 ns	UNSIGNED_32
11	SM2 (RPDO-->Device): Counter - missing events		UNSIGNED_16
12	SM2 (RPDO-->Device): Cycle time too short		UNSIGNED_16
32	SM2 (RPDO-->Device): Sync error		BOOL

0x1C33 - Sync Manager 3 (Device-->TPDO): Parameter

Sub.	Name	Lenze setting	Data type
1	SM3 (Device-->TPDO): Sync type	0: Free Run	UNSIGNED_16
2	SM3 (Device-->TPDO): Cycle time	250000 ns	UNSIGNED_32
3	SM3 (Device-->TPDO): Shift time	62500 ns	UNSIGNED_32
4	SM3 (Device-->TPDO): Supported sync types	0x0825	UNSIGNED_16
5	SM3 (Device-->TPDO): Min. cycle time	125000 ns	UNSIGNED_32
6	SM3 (Device-->TPDO): Signal processing time	0 ns	UNSIGNED_32
9	SM3 (Device-->TPDO): Delay time	0 ns	UNSIGNED_32
11	SM3 (Device-->TPDO): Counter - missing events		UNSIGNED_16
12	SM3 (Device-->TPDO): Cycle time too short		UNSIGNED_16
32	SM3 (Device-->TPDO): Sync error		BOOL

11.4 Licences

lwIP - TCP/IP stack

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Figures

Index

B

C

D

E

L

M

N

FEEDBACK

Your opinion is important to us

These instructions were created to the best of our knowledge and belief to give you the best possible support for handling our product.

If you have suggestions for improvement, please e-mail us to:

feedback-docu@Lenze.de

Thank you for your support.

Your Lenze documentation team

i700 servo inverters · Reference manual · EDSE70ACMxx · 13567282 · DMS 7.0 EN · 05/2019 · TD06

Lenze Automation GmbH
Hans-Lenze-Str. 1
D-31855 Aerzen
Germany
HR Hannover B 205381
+49 (0)51 54 / 82-0
+49 (0)51 54 / 82-28 00
sales.de@lenze.de
www.Lenze.com

Service
Lenze Service GmbH
Breslauer Straße 3
D-32699 Extertal
Germany
00 80 00 / 24 4 68 77 (24 h helpline)
+49 (0)51 54 / 82-11 12
service.de@lenze.de

Automation Systems

Controller-based Automation

EtherCAT®________________________________

Communication Manual EN

13534143

Lenze

Contents

1 About this documentation

This documentation ...

- contains detailed information about the commissioning, configuration, and diagnostics of the EtherCAT® bus system as part of the Lenze automation system "Controller-based Automation".
- is part of the "Controller-based Automation" manual collection. It consists of the following sets of documentation:

Documentation type	Subject
Product catalogue	Controller-based Automation (system overview, sample topologies) Lenze Controller (product information, technical data)
System manuals	Visualisation (system overview/sample topologies)
Communication manuals **Online helps**	Bus systems • Controller-based Automation EtherCAT® • Controller-based Automation CANopen® • Controller-based Automation PROFIBUS® • Controller-based Automation PROFINET®
Reference manuals **Online helps**	Lenze Controllers: • Controller 3200 C • Controller c300 • Controller p300 • Controller p500
software manuals **Online helps**	Lenze Engineering Tools: • »PLC Designer« (programming) • »Engineer« (parameter setting, configuration, diagnostics) • »VisiWinNET® Smart« (visualisation) • »Backup & Restore« (backup, restore, update)

More technical documentation for Lenze components

Further information on Lenze products which can be used in conjunction with Controller-based Automation can be found in the following sets of documentation:

	Design / configuration / technical data
🗆	**Product catalogues** • Controller-based Automation • Controllers • Inverter Drives/Servo Drives
	Installation and wiring
🕮	**Mounting instructions** • Controllers • Communication cards (MC-xxx) • I/O system 1000 (EPM-Sxxx) • Inverter Drives/Servo Drives • Communication modules
🗆	**Hardware manuals** • Inverter Drives/Servo Drives
	Parameterisation / configuration / commissioning
🗆	**Online help/reference manuals** • Controllers • Inverter Drives/Servo Drives • I/O system 1000 (EPM-Sxxx)
🗆	**Online help/communication manuals** • Bus systems • Communication modules
	Sample applications and templates
🗆	**Online help / software manuals and reference manuals** • i700 application sample • Application Samples 8400/9400 • FAST application template • FAST technology modules

Symbols:

🕮 Printed documentation

🗆 PDF file / online help in the Lenze Engineering tool

Tip!

Current documentation and software updates with regard to Lenze products can be found in the download area at:

www.lenze.com

Target group

This documentation is intended for persons who commission and maintain a Controller-based automation system by means of a Lenze Controller and the »PLC Designer« engineering tool.

Information regarding the validity

The information provided in this documentation is valid for the Lenze automation system "Controller-based Automation" from release 3.0.

Screenshots/application examples

All screenshots in this documentation are application examples. Depending on the firmware version of the field devices and the software version of the Engineering tools installed (e.g. »PLC Designer«), screenshots in this documentation may differ from the representation on the screen.

1.1 Document history

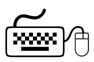
Version			Description
7.4	05/2017	TD17	New: Start parameters of the Inverter Drives 8400 motec (📖 105)
7.3	11/2016	TD17	Update for the Lenze automation system "Controller-based Automation" 3.14 • Chapter Controller-based Automation: Central motion control (📖 15)
7.2	05/2016	TD17	Update for the Lenze automation system "Controller-based Automation" 3.13 • Chapter Restarting the EtherCAT fieldbus (📖 179): Notes added.
7.1	10/2015	TD17	Update for the Lenze automation system "Controller-based Automation" 3.12 • Corrections in chapter Modular machine configuration (📖 108) • Messages: Invalid Input Configuration / Invalid Output Configuration (📖 198) in chapter Error scenarios (📖 194) supplemented.
7.0	05/2015	TD17	Update for the "Controller-based Automation" 3.10 Lenze automation system • Chapter Modular machine configuration (📖 108) new. • Chapter Error messages for modular machine configuration (📖 211) new
6.5	01/2015	TD17	Update for the Lenze automation system "Controller-based Automation" 3.9 • L_ETC_STATE (📖 178) supplemented. • Message: Invalid SyncManager Configuration (📖 197) in chapter Error scenarios (📖 194) supplemented.
6.4	04/2014	TD17	Update for the "Controller-based Automation" 3.8 Lenze automation system
6.3	11/2013	TD17	Update for the Lenze automation system "Controller-based Automation" 3.6
6.2	03/2013	TD17	Update for the Lenze automation system "Controller-based Automation" 3.5
6.1	12/2012	TD17	New function block L_ETC_GetEmergency (FB) (📖 153) in the "Controller-based Automation" 3.4 Lenze automation system
6.0	11/2012	TD17	• General corrections • New layout
5.0	08/2012	TD17	• Revision on the Lenze automation system "Controller-based Automation" 3.3 • Information on the Servo-Inverter i700 added. • Information on the ECS servo system and »GDC« removed.
4.2	12/2011	TD17	• Update for the Lenze automation system "Controller-based Automation" 3.2 • Chapter L_IODrvEtherCAT function library (📖 119) updated.
4.1	03/2011	TD17	• Special features for the ECS servo system added. • References to Lenze sample projects for EtherCAT logic field devices (device application + PLC program) added. ▸ Commissioning of the system (📖 40)
4.0	10/2010	TD17	Commissioning and configuration with the Lenze »PLC Designer« V3.x
3.0	10/2009	TD17	General update
2.0	05/2009	TD17	General update
1.0	09/2008	TD17	First edition

1.2 Conventions used

This documentation uses the following conventions to distinguish between different types of information:

Type of information	Highlighting	Examples/notes
Spelling of numbers		
Decimal	Normal spelling	Example: 1234
Decimal separator	Point	The decimal point is always used. For example: 1234.56
Hexadecimal	0x[0 ... 9, A ... F]	Example: 0x60F4
Binary • Nibble	0b[0, 1]	Example: '0b0110' Example: '0b0110.0100'
Text		
Program name	» «	PC software Example: Lenze »Engineer«
Window	*italics*	The *message window*... / The *Options* dialog box ...
Variable names		Setting *bEnable* to TRUE...
Control element	**bold**	The **OK** button ... / The **Copy** command ... / The **Properties** tab ... / The **Name** input field ...
Sequence of menu commands		If several successive commands are required for executing a function, the individual commands are separated from each other by an arrow: Select the command **File** ➜ **Open** to...
Shortcut	**<bold>**	Use **<F1>** to open the online help.
		If a shortcut is required for a command to be executed, a "+" has been put between the key identifiers: With **<Shift>+<ESC>** ...

Program code	`Courier`	`IF var1 < var2 THEN`
		`a = a + 1`
		`END IF`

Keyword	**`Courier bold`**	
Hyperlink	<u>underlined</u>	Optically highlighted reference to another topic. Can be activated with a mouse-click in this documentation.
Icons		
Page reference	(📖 9)	Optically highlighted reference to another page. Can be activated with a mouse-click in this documentation.
Step-by-step instructions		Step-by-step instructions are indicated by a pictograph.

1.3 Terminology used

Term	Meaning
Code	Parameter for parameterising or monitoring the field device. The term is also referred to as "index" in common usage.
CoE	CANopen over EtherCAT
Controllers	The Controller is the central component of the Lenze automation system which controls the motion sequences by means of the operating system. The Controller communicates with the field devices (inverters) via the fieldbus.
DC	Distributed clocks (distributed synchronised clocks)
Engineering PC	The Engineering PC and the Engineering tools installed serve to configure and parameterise the system "Controller-based Automation". The Engineering PC communicates with the controller via Ethernet.
Engineering tools	Software solutions for easy engineering in all phases which serve to commission, configure, parameterise and diagnose the Lenze automation system. ▶ Lenze Engineering tools (📖 30)
ETG	EtherCAT Technology Group (EtherCAT user organisation)
FAST	By default, the **Lenze FAST** application software is installed on the Lenze Controller in the **"FAST Runtime"** mode with **"FAST Motion"** for central control of PLC applications.
FB	Function block (contained in a function library)
Fieldbus node	Devices integrated in the bus system as, for instance, Controller and inverter
Field device	
FoE	File Access over EtherCAT
FUN	Function (contained in a function library)
Index	Each object has a unique index for addressing purposes. In this documentation the index is represented as a hexadecimal value and is identified by a prefixed "0x", e.g. "0x1000".
Inverter	Generic term for Lenze frequency inverters, servo inverters
PLC	Programmable Logic Controller
Object	"Container" for one or more parameters with which you can parameterise or monitor the fieldbus node.
PLC	Programmable Logic Controller (PLC)
Subcode	If a code contains several parameters, they are stored in "subcodes". This manual uses a slash "/" as a separator between code and subcode (e.g. "C00118/3"). Is usually referred to as "subindex".
Subindex	If a code contains several parameters, they are stored in so-called "subindexes". In this documentation, the slash "/" is used as a separator between an index and subindex, e.g. "0x1018/1".
Bus systems	
CAN	CAN (Controller Area Network) is an asynchronous, serial fieldbus system.
CANopen	CANopen® is a communication protocol based on CAN. The Lenze system bus (CAN on board) operates with a subset of this communication protocol. CANopen® is a registered community trademark of the CAN user organisation CiA® (CAN in Automation e. V.).
EtherCAT	EtherCAT® (**Ether**net for **C**ontroller and **A**utomation **T**echnology) is an Ethernet-based fieldbus system which fulfils the application profile for industrial real-time systems. EtherCAT® is a registered trademark and patented technology, licenced by Beckhoff Automation GmbH, Germany.
ETHERNET	Ethernet specifies the software (protocols) and hardware (cables, plugs, etc.) for wired data networks. In the form of "Industrial Ethernet", the Ethernet standard is used in industrial production systems. On the basis of IEEE 802.3, standard Ethernet is specified by the Institute of Electrical and Electronics Engineers (IEEE), USA.

Term	Meaning
EtherNet/IP	EtherNet/IP™ (EtherNet Industrial Protocol) is an Ethernet-based fieldbus system that uses Common Industrial Protocol™ (CIP™) to exchange data. EtherNet/IP™ and Common Industrial Protocol™ (CIP™) are brand labels and patented technologies, licensed by the ODVA user organisation (Open DeviceNet Vendor Association), USA.
PROFIBUS®	PROFIBUS® (Process Field Bus) is a widely used fieldbus system for the automation of machines and production lines. PROFIBUS® is a registered trademark and patented technology licensed by the PROFIBUS & PROFINET International (PI) user organisation.
PROFINET®	PROFINET® (Process Field Network) is a real-time capable fieldbus system based on Ethernet. PROFINET® is a registered trademark and patented technology licensed by the PROFIBUS & PROFINET International user organisation (PI).

1.4 Definition of the notes used

The following signal words and symbols are used in this documentation to indicate dangers and important information:

Safety instructions

Layout of the safety instructions:

Pictograph and signal word!

(characterise the type and severity of danger)

Note

(describes the danger and gives information about how to prevent dangerous situations)

Pictograph	Signal word	Meaning
	Danger!	**Danger of personal injury through dangerous electrical voltage** Reference to an imminent danger that may result in death or serious personal injury if the corresponding measures are not taken.
	Danger!	**Danger of personal injury through a general source of danger** Reference to an imminent danger that may result in death or serious personal injury if the corresponding measures are not taken.
STOP	**Stop!**	**Danger of property damage** Reference to a possible danger that may result in property damage if the corresponding measures are not taken.

Application notes

Pictograph	Signal word	Meaning
	Note!	Important note to ensure trouble-free operation
	Tip!	Useful tip for easy handling
		Reference to another document

2 Safety instructions

Please observe the safety instructions in this documentation when you want to commission an automation system or a plant with a Lenze Controller.

The device documentation contains safety instructions which must be observed!

Read the documentation supplied with the components of the automation system carefully before you start commissioning the Controller and the connected devices.

Danger!

High electrical voltage

Injury to persons caused by dangerous electrical voltage

Possible consequences

Death or severe injuries

Protective measures

Switch off the voltage supply before working on the components of the automation system.

After switching off the voltage supply, do not touch live device parts and power terminals immediately because capacitors may be charged.

Observe the corresponding information plates on the device.

Danger!

Injury to persons

Risk of injury is caused by ...

- unpredictable motor movements (e.g. unintended direction of rotation, too high velocities or jerky movement);
- impermissible operating states during the parameterisation while there is an active online connection to the device.

Possible consequences

Death or severe injuries

Protective measures

- If required, provide systems with installed inverters with additional monitoring and protective devices according to the safety regulations valid in each case (e.g. law on technical equipment, regulations for the prevention of accidents).
- During commissioning, maintain an adequate safety distance to the motor or the machine parts driven by the motor.

Stop!

Damage or destruction of machine parts

Damage or destruction of machine parts can be caused by ...

- Short circuit or static discharges (ESD);
- unpredictable motor movements (e.g. unintended direction of rotation, too high velocities or jerky movement);
- impermissible operating states during the parameterisation while there is an active online connection to the device.

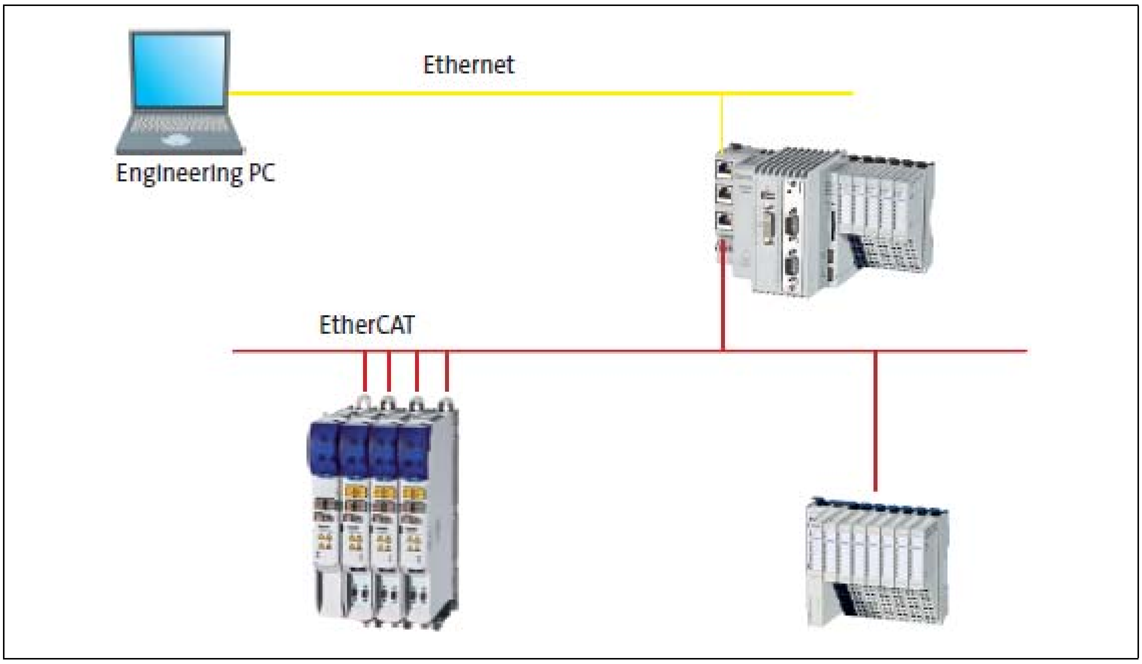
Protective measures

- Always switch off the voltage supply before working on the components of the automation system.
- Do not touch electronic components and contacts unless ESD measures were taken beforehand.
- If required, provide systems with installed inverters with additional monitoring and protective devices according to the safety regulations valid in each case (e.g. law on technical equipment, regulations for the prevention of accidents).

3 Controller-based Automation: Central motion control

The Lenze "Controller-based Automation" system serves to create complex automation solutions with central motion control. Here, the Controller is the control centre of the system.

System structure of Controller-based Automation



[3-1] **Example:** EtherCAT with Controller 3231 C, I/O system 1000 and Servo-Inverter i700

Lenze provides especially coordinated system components:

- Engineering software

  The Lenze Engineering tools (📖 30) on your Engineering PC (Windows® operating system) serve to parameterise, configure and diagnose the system. The Engineering PC communicates with the Controller via Ethernet.

  The Lenze engineering tools are available for download at:

  www.lenze.com → **Download → Software Downloads**
- Controllers

  The Lenze Controller is available as Panel Controller with integrated touch display and as Cabinet Controller in control cabinet design.

  Cabinet Controllers provide a direct coupling of the I/O system 1000 via the integrated backplane bus.
- Bus systems

  EtherCAT is the standard "on-board" bus system of the Controller-based Automation. EtherCAT enables the control of all nodes on one common fieldbus.

  Optionally, CANopen, PROFIBUS and PROFINET can be used as extended topologies.

  With Controllers 3200 C and p500 it is also possible to use EtherNet/IP via the Ethernet interfaces.

  Controllers c300 and p300 are provided with an "on board" CANopen interface (in addition to EtherCAT).
- Inverter (e.g. Servo-Inverter i700)

"Application software" of the Lenze Controllers

The "application software" of the Lenze Controllers enables the control and/or visualisation of motion sequences.

FAST technology modules provide for an easy development of a modular machine control in the »PLC Designer«.

The following "Application Software" versions are available:

- "FAST Runtime"

  Sequence control takes place in the Controller (by means of logically linked control signals).

  Motion control takes place in the inverter.
- "FAST Motion"

  Sequence control and motion control take place in the Controller.

  The inverter solely serves as an actuating drive.

  Motion applications make special demands on the cycle time and real-time capability of the bus system between the Controller and the subordinate fieldbus nodes. This is the case, for instance, if the nodes are to be traversed in a synchronised way or position setpoints are to be transferred.
- "Visualisation"

  The optional visualisation of the automation system can be used separately or additionally to "FAST Runtime" or "FAST Motion".

  For this purpose, an external monitor panel/display can be connected to the Cabinet Controller 3231 C/3241 C/3251 C.

Fieldbus communication

The Lenze controllers have different interfaces for fieldbus communication:

Range	Cabinet Controller		Panel Controller	
	c300	3200 C series	p300	p500
Interfaces (on board)				
Ethernet	1	2	1	2
EtherNet/IP	-		-	
EtherCAT	1 1)	1	1 1)	1
CANopen	1	-	1 2)	-
Optional interfaces (communication cards)				
CANopen MC-CAN2	-	●	-	● 2)
PROFIBUS master MC-PBM	-	●	-	●
PROFIBUS slave MC-PBS	-	●	-	●
PROFINET device MC-PND	●	●	●	●
Ethernet MC-ETH	-	●	-	●
Serial interfaces MC-ISI	-	●	-	●

1) Only the master functionality is supported.

2) Up to release 3.9: "EL 100 CAN" driver / from release 3.10: "Lenze CAN driver"

Ethernet interface

The Ethernet interface serves to connect the Engineering PC or to create line topologies (no integrated switch for Controller c300/p300).

With Controllers 3200 C and p500, the Ethernet interfaces also provide for EtherNet/IP communication.

4 The Lenze automation system with EtherCAT

This chapter provides basic information about ...

- the structure of the Lenze automation system using the EtherCAT bus system;
- the Lenze Engineering tools required for commissioning;
- the interaction of the components.

4.1 Brief description of EtherCAT

Tip!

Detailed information on EtherCAT is provided on the Internet website of the EtherCAT Technology Group:

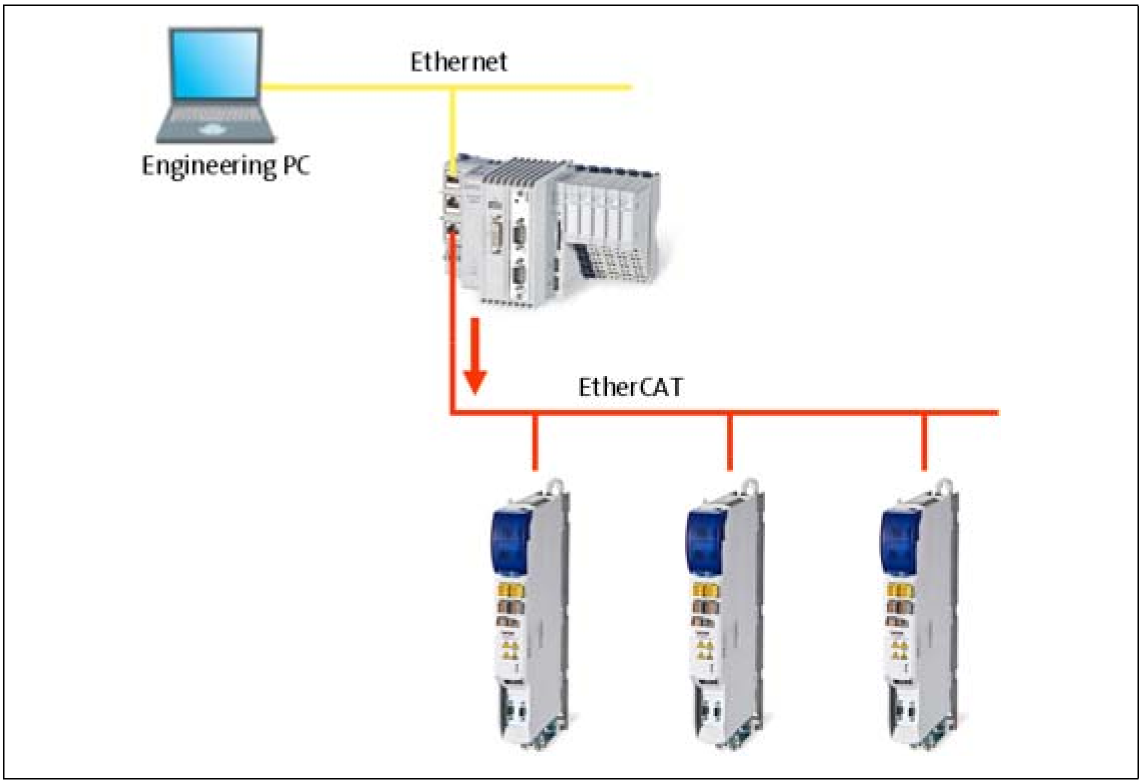
www.ethercat.org

Product features

- EtherCAT is a powerful bus system which is based on Ethernet.
- Due to its integrated synchronisation mechanisms based on "distributed clocks", EtherCAT possesses outstanding real time characteristics.
  Synchronisation with "Distributed clocks" (DC) (37)
- EtherCAT provides a higher bandwidth compared to CANopen:
  - This enables motion and logic applications to be operated by the same fieldbus.
  - The number of the nodes to be controlled is higher.
  - The maximally possible bus length is longer.
- EtherCAT can access all field devices via a common interface. Therefore, unlike for the Lenze CANopen control technology, a division into Logic bus and Motion bus is not required.
- The "Modular Device Profile for IPC" (MDP) is based on the "Modular Device Profile Specification" of the EtherCAT Technology Group. All (software and hardware) components of the Lenze Controller or Embedded PC are divided into modules. The list of the modules available is generated dynamically, depending on the physically available components. The Lenze I/O system 1000 with the EPM-S130 head end supports the "Modular Device Profile".

4.1.1 Structure of the EtherCAT bus system

Basic structure



[4-1] **Example:** EtherCAT bus system with 3231 C controller and i700 servo inverter

Physical structure

An EtherCAT master can communicate with one or more nodes (slaves).

Internally, the EtherCAT bus has a ring topology. Since Ethernet cables are provided with a feed conductor and a return conductor within one cable, for the installer the topology seems to be a line. The last slave closes the ring.

Switches, hubs, or other infrastructure components known from the Ethernet standard must not be used because they impair the real-time performance.

4.1.2 communication

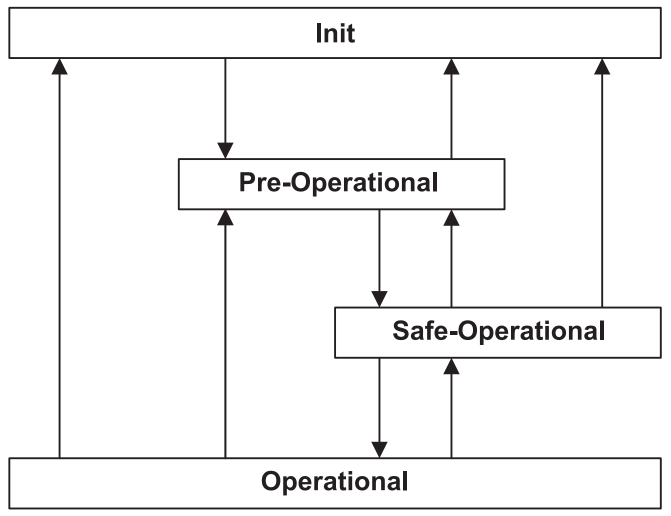
Compared with conventional Ethernet, the collision-free transfer of telegrams on the fieldbus makes EtherCAT a real-time capable bus system.

Communication is always initiated by the EtherCAT master, i.e. the Lenze Controller. A telegram sent by the master passes through all EtherCAT slaves. The last slave of the communication chain sends the telegram back to the EtherCAT master. On the way back, the telegram is directly sent to the EtherCAT master, without being processed in the slaves.

When EtherCAT is used, data are transferred in so-called "EtherCAT frames". The fieldbus nodes only remove the data intended for them and do so while the EtherCAT frame is passing through the device. Output data are inserted into the frame at the same time. Read and write access is always carried out on a small section of the overall EtherCAT frame – the datagrams. This means that a frame does not have to be received completely before it is processed. Each datagram is passed on with minimal delay.

4.1.2.1 The EtherCAT state machine

Before communication via EtherCAT is possible, the fieldbus is scanned by the EtherCAT state machine when the installation is being powered up. The following illustration shows the possible status changes from the point of view of an EtherCAT slave.



[4-2] EtherCAT state machine

Status	Description
Init	• Initialisation phase • No SDO/PDO communication with the slaves • Device can be detected by fieldbus scan
Pre-Operational	• The fieldbus is active. • SDO communication (mailbox communication) is possible. • No PDO communication

Safe-operational	• SDO communication (mailbox communication) is possible.
	• PDO communication:
	  • The input data in the process image are updated.
	  • The output data from the process image are not transferred to the slaves.

Operational	Normal operation • SDO communication • PDO communication • Fieldbus synchronisation has been successful (if used)

Note!

- A fieldbus scan is possible in any EtherCAT state.
  ▶ Determining the physical EtherCAT configuration (fieldbus scan) (📖 76)
- The SDO communication via the EtherCAT bus is only possible if at least the "Pre-Operational" state has been reached.

AL Status Code

Possible errors during transitions between states are entered in the EtherCAT register "AL Status Code" (address 0x0134:0x0135).

AL Status Code [hex]	Description
0x0000	No error
0x0011	Invalid status change requested
0x0012	Unknown status requested
0x0013	"Bootstrap" status is not supported
0x0016	Invalid mailbox configuration "Pre-operational"
0x001A	Synchronisation error
0x001B	Sync manager watchdog
0x001D	Invalid output data configuration
0x001E	Invalid input data configuration
0x002B	Invalid input and output data
0x0030	Invalid configuration of DC synchronisation
0x9001	Firmware watchdog error
0x9002	Mapping error

4.1.2.2 Addressing of the slaves

The EtherCAT system uses two types of addressing for the slaves:

Auto-increment addressing

The auto-increment addressing is used by the master during the initialisation phase of the fieldbus. When the "Pre-Operational" state has been reached, the master uses the Fixed-Address addressing.

Fixed-address addressing

With the fixed-address addressing, the slaves are addressed via the station address distributed by the master during the start-up phase.

In the EtherCAT bus topology in »PLC Designer«, the first slave is given the address ’1001’, the second the address ’1002’ and so on. The EtherCAT addresses cannot be changed.

The EtherCAT address of the master is ’0’. Access to master objects with the address ’0’ is possible.

Example

The first slave of a configuration is given the following addresses ...

- ’0’ due to the automatic incrementation procedure;
- ’1001’ due to the fixed addressing procedure.

4.1.2.3 Working counter

Each EtherCAT datagram contains a working counter (WKC) which is incremented by each slave after the data have been processed successfully.

The working counter (WKC) can be used as a diagnostics option to check the processing of the EtherCAT telegrams by the slaves.

In each cycle, the Lenze Controller compares the expected value of the working counter with the value read back via the fieldbus. If the read-back value is smaller than the expected value, not all addressed slaves have been reached. The controller detects this and reports an error.

Messages: WKC Error / Not all slaves "Operational" / SyncManager Watchdog (197)

Example

- 10 slaves read/write process data in the "Operational" state
  Expected value of the WKC: 10
- A cable break between the 8th and 9th slave causes the master to be unable to access slave 9 and slave 10:
  - Value of the restored WKC: 8
  - An error response is initiated in the Lenze Controller.
  - The EtherCAT bus changes to the state "Pre-Operational".

4.2 Required hardware components

4.2.1 Field devices

The Lenze automation system supports the following EtherCAT-capable logic and motion components:

Field devices		EtherCAT	
		Logic	Motion
Controllers	Controller 32xx C	●	●
	Controller c300	●	●
	Controller p300	●	●
	Controller p500	●	●
i700 servo inverter	Single axis		●
	Double axis		●
Servo Drives 9400 *1)*	HighLine	●	
	HighLine with CiA402	●	●
	PLC	●	
	regenerative power supply module commissioning guidelines	●	
Inverter Drives 8400 *2)*	StateLine	●	
	HighLine	●	
	TopLine	●	
I/O system 1000	EPM-Sxxx	●	

1) With EtherCAT E94AYCET communication module

2) With EtherCAT E84AYCET communication module

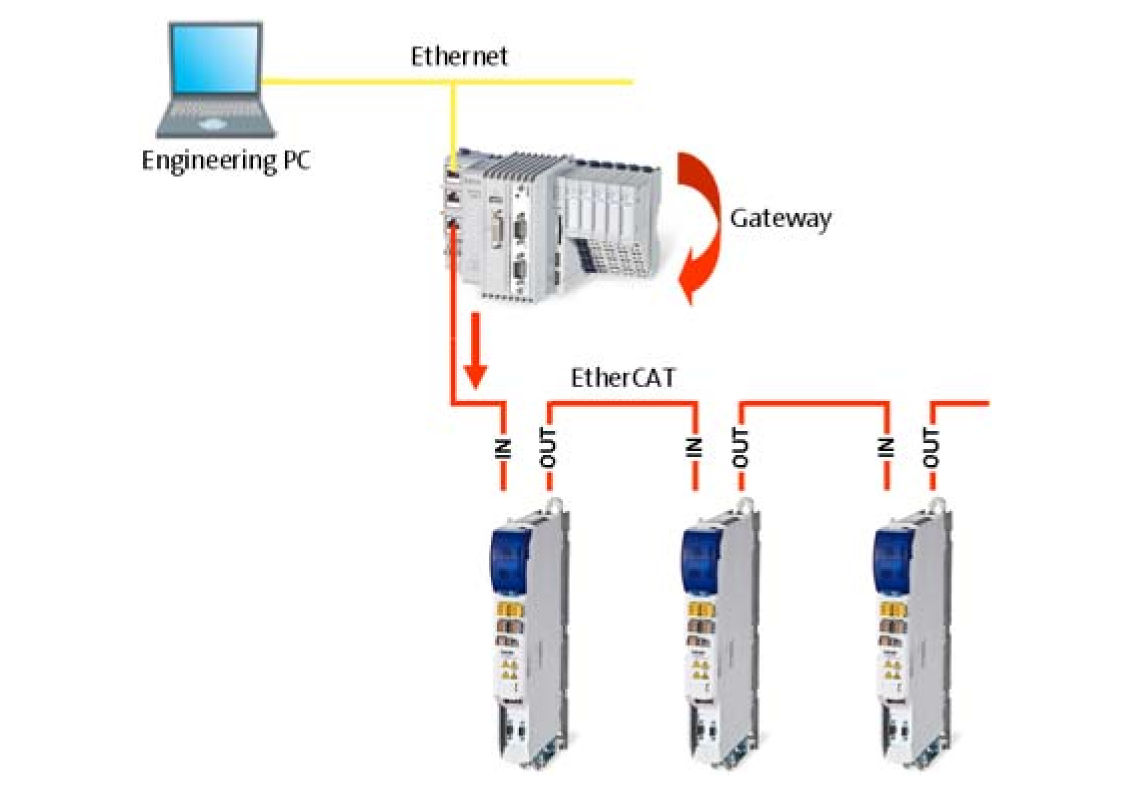
Note!

Bus coupler module EPM-S130 (EtherCAT)

Only **EPM-Sxxx I/O compound modules from hardware version 1B onwards** are supported.

Field devices of other manufacturers can be integrated as Logic nodes if they provide a standard-compliant EtherCAT device description.

4.2.2 The Lenze Controller - the central component



[4-3] **Example:** EtherCAT bus system with 3231 C controller as gateway and i700 servo inverter

The Lenze Controller is the central component in the EtherCAT bus system:

- The controller is the EtherCAT master.
- The Lenze Controllers have an EtherCAT interface "on-board".
- The controller acts as an EtherCAT gateway in order to enable access to the field devices from the Engineering PC via Ethernet and EtherCAT.
- The devices are interconnected successively in line. For correct operation, it is necessary that the physical sequence of the EtherCAT field devices matches the bus topology created in »PLC Designer«.
  Otherwise the system will not become "Operational". (An error message indicates which slave (product code) is expected at what position.)
- Each EtherCAT slave has two EtherCAT ports.
  In contrast to Ethernet, one port is assigned as input, the other one as output.
  The inputs (IN) and outputs (OUT) must be correctly wired to each other
- A bus termination at the last slave is not required since the bus system at the last slave is terminated automatically.

4.2.3 EtherCAT product codes

Device descriptions can be assigned to the corresponding devices with the help of the product codes. In »PLC Designer«, you can install device descriptions with the menu command **Tools → Device repository....**

▸ Importing missing devices / device description files (📖 79)

Structure of the device ID: <Manufacturer ID>_<Productcode><Revision number

Identification	Meaning
Manufacturer ID	Unique identification for the manufacturer, for Lenze devices: 0x3B (59)
Product code	Product code of the product range/the device
Revision number	Revision number

If, for instance, a device available at the fieldbus without an installed device description is detected during a Determining the physical EtherCAT configuration (fieldbus scan) (📖 76) a message with the device ID as hexadecimal value is displayed:

In this example, the device description for a Lenze Servo Drive 9400 HighLine, actuating drive - speed, is not installed (0x38079CD9 = 940023001).

Product codes for Servo Drives 9400

Product code [dec]									Meaning
9	4	0	0	2	1	x	x	x	Servo Drive 9400 in general
9	4	0	0	2	2	x	x	x	Servo Drive 9400 StateLine
9	4	0	0	2	3	x	x	x	Servo Drive 9400 HighLine
9	4	0	0	2	4	x	x	x	Servo Drive 9400 TopLine
9	4	0	0	2	5	x	x	x	Servo Drive 9400 PLC
9	4	0	0	2	6	x	x	x	Servo Drive 9400 V/R (regenerative power supply module)
									Applications:
						0	0	0	Empty application
						0	0	1	Actuating drive speed
						0	0	2	Actuating drive - torque
						0	0	3	Electronic gearbox
						0	0	4	Synchronism with mark synchronisation
						0	0	5	Table positioning
						0	0	6	Positioning sequence control
						0	0	7	PLC application
						0 0	0 ... 9	8 9	Reserved ... Reserved
						1	x	x	Reserved for device profiles
						1	0	1	CiA402
						2	x	x	Reserved for Lenze applications
						2	0	1	Regenerative power supply module application

Product codes for Inverter Drives 8400

Product code [dec]						Meaning
8	4	0	0	2	2	Inverter Drive 8400 StateLine
8	4	0	0	2	3	Inverter Drive 8400 HighLine
8	4	0	0	2	4	Inverter Drive 8400 TopLine

Product codes for i700 servo inverter

Product code [hex]								Meaning
6	9	0	7	0	0	0	1	Single axis
6	9	0	7	0	0	0	2	Double axis

Product codes for the I/O system 1000

Product code [dec]				Meaning
1	3	0	0	I/O system EPM-S130

4.2.4 The EtherCAT interface of the Lenze Controller

The EtherCAT interface links the controller to an EtherCAT network.

Note!

In the case of a correct connection to the field devices, the LEDs of the EtherCAT interface are lit.

▶ EtherCAT interface of the Lenze Controller (📖 35)

Example

ETC1: EtherCAT network connection

[4-4] EtherCAT interface at the 3231 C controller

4.3 Lenze Engineering tools

The Lenze Engineering tools enable the configuration and operation of controller-based Lenze automation systems according to individual requirements.

Use the corresponding Engineering tool applicable to the field device.

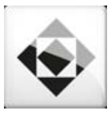
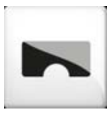
 **»EASY Navigator«: Starting the suitable Engineering tool**

The Lenze Engineering software consists of the Engineering tools optimised for the respective Engineering stage.

The »EASY Navigator« represents the Lenze Engineering tools installed on the Engineering PC. Start the desired Engineering tool via the corresponding button:

The »EASY Navigator« ...

- simplifies orientation for selecting the suitable Engineering tool;
- allows for the simple start of the required Engineering tool (depending on the application):

What would you like to do?	Button	Engineering tool
Programming • Program the controller • Parameterise/commission the Servo-Inverter i700 • Parameterise the I/O system 1000		»PLC Designer«
Parameterising/configuring the inverter • Parameterising and configuring the automation/drive system • Parameterising Inverter Drives 8400/Servo Drives 9400		»Engineer«
Visualisation • Visualising the applications of the automation system • Creating the visualisation/user interfaces		»VisiWinNET«
Online diagnostics • Easy online diagnostics of the controllers (from »EASY Starter« V1.2) and other Lenze devices		»EASY Starter« (reading parameters)
Online parameterisation • Online parameterisation/commissioning of Lenze devices • Direct online parameterisation when the online connection to the Lenze devices is active.		»EASY Starter« (reading/writing parameters)

Further Engineering tools that are not called via the »EASY Navigator« are:

- »WebConfig« (web-based parameterisation, configuration, and online diagnostics)
- »Backup & Restore« (data backup/recovery, software update).

4.4 Interaction of the components

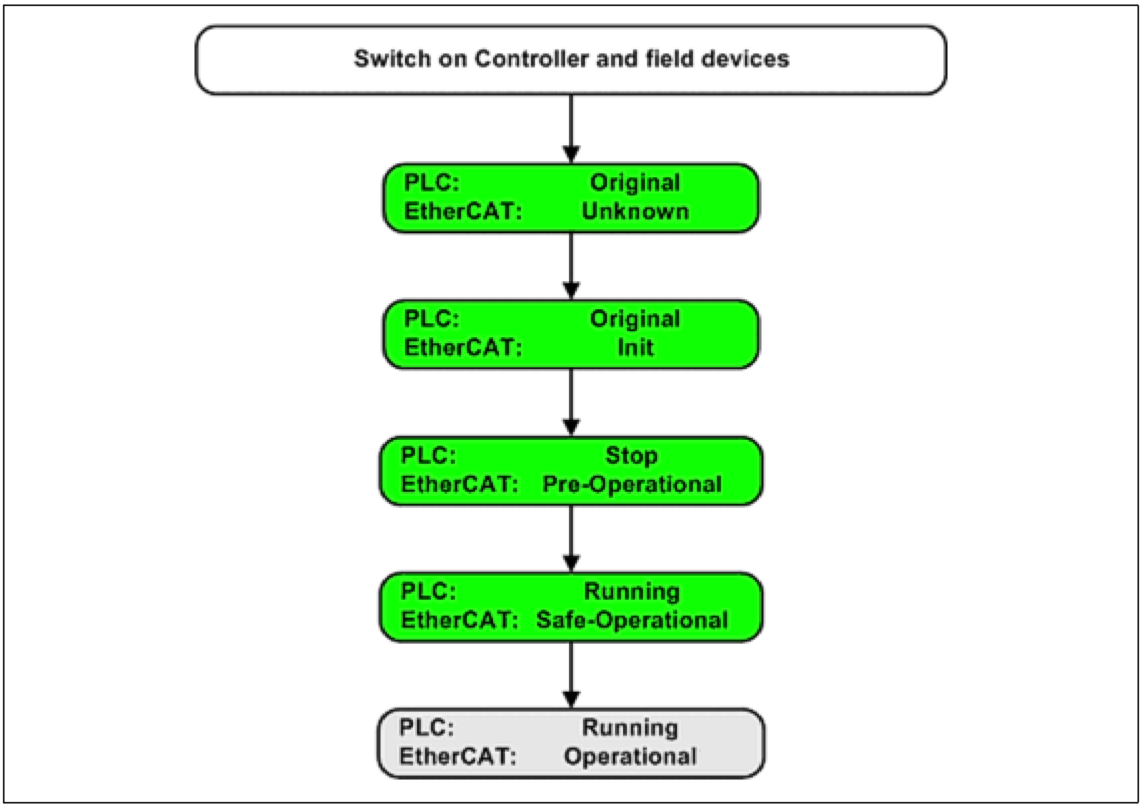
4.4.1 The state machine of the Lenze control technology

In the Lenze control technology, the states of the PLC and the EtherCAT bus are coupled. The PLC controls the fieldbus.

After switch-on, the system automatically powers up if the following conditions are fulfilled:

- There is an executable PLC boot project on the controller (»PLC Designer« project).
- The slaves that are on the fieldbus and have been configured in »PLC Designer« are accessible.

The following illustration shows the linkage of the states in the state machine of the Lenze control technology when the conditions for the automatic acceleration of the system are fulfilled (boot project with EtherCAT configuration):



[4-5] States in the Lenze control technology

Legend	
	Transitional state, automatic change to next state
	Stationary state, change to next state by external actions
PLC	State of the PLC
EtherCAT	State of the EtherCAT bus

Explanation of the transitions during system start

While a state is passed through, different tests are carried out (e.g. it is tested whether the physical topology complies with the configuration). If the tests are successful, the system automatically changes to the next state.

Status		What happens?	What is tested?
PLC	EtherCAT		
Source	Unknown	The system starts.	EtherCAT configuration available?
Source	Init	EtherCAT is initialised (a fieldbus scan is carried out).	Does the EtherCAT configuration match the results of the fieldbus scan?
Stop	Pre-Operational	• EtherCAT is active. • SDO communication is possible.	
Running	Safe-operational	• The PLC program is being loaded. • The PLC is running. • The input data in the process image are updated. • The output data from the process image are not transferred to the slaves.	
Running	Operational	The system is running.	

If the tests are not successful, the fieldbus changes to the next state. Corresponding remedial measures are described in the chapter "Error scenarios" (📖 194).

4.4.2 Communication between the Engineering PC and the field devices

For commissioning of the field devices, an online connection is required between the Engineering PC and the corresponding field device. Depending on the state of the EtherCAT bus, there are two options:

- EtherCAT bus not in operation (📖 33)
- EtherCAT bus in operation (gateway function) (📖 34)

4.4.2.1 EtherCAT bus not in operation

You can communicate serially or via CANopen.

Note!

For the **i700 servo inverter**, there is no possibility of communication in this case.

Precondition:

Serial communication:

- You require the E94AZCUS diagnostic adapter.
- The field device and Engineering PC (USB interface) must be connected via the diagnostic adapter.

Communication via CANopen

- You required the EMF2177IB USB system bus adapter .
- The field devices and the Engineering PC are connected via the system bus adapter - either via a point-to-point connection or via the bus system.

Advantage:

Quick option of communication without commissioning of the EtherCAT bus.

Disadvantage:

You require additional hardware.

Tip!

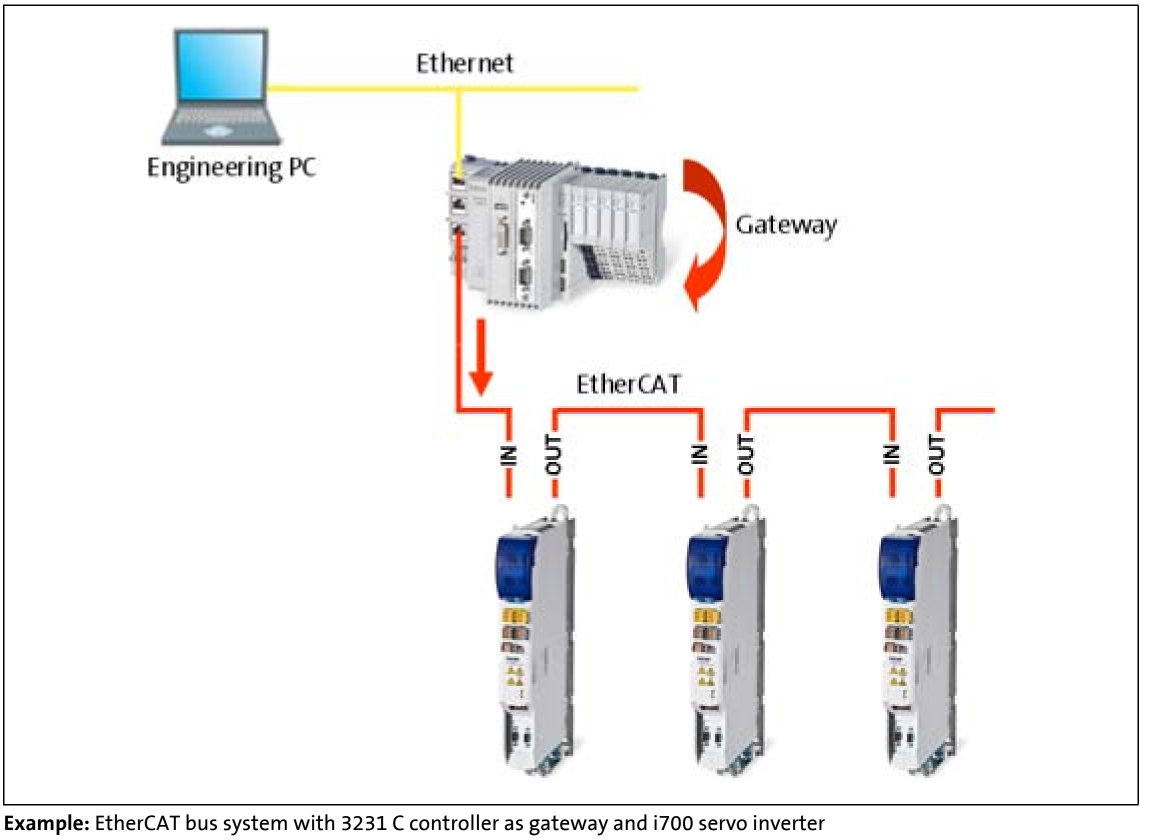
As soon as the fieldbus has been commissioned and is at least in the "Pre-Operational" state, this communication channel is of secondary importance. We recommend that the EtherCAT bus be commissioned as early as possible in order to enable use of the gateway function.

4.4.2.2 EtherCAT bus in operation (gateway function)

You communicate directly via EtherCAT and use the controller as a gateway.

Note!

A PLC program does not need to run to be able to use the gateway function.



[4-6] **Example:** EtherCAT bus system with 3231 C controller as gateway and i700 servo inverter

Precondition:

- The EtherCAT configuration has been created with »PLC Designer« to match the hardware configuration.
- The EtherCAT configuration has been loaded and activated on the controller with »PLC Designer«.
- The EtherCAT bus is at least in the "Pre-Operational" state.

Advantage:

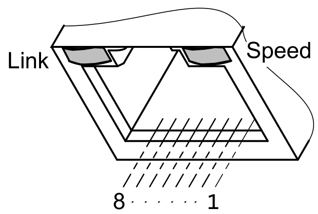
- You do not require any additional hardware.
- The process data, parameter data and diagnostic data are transferred via a single bus connection at the same time.

5 Technical data

5.1 General data

Range	Values
Higher-level network protocol	EtherCAT device protocol
Communication medium / cable type	S/FTP (Screened Foiled Twisted Pair, ISO/IEC 11801 or EN 50173), CAT5e Standard Ethernet (in accordance with IEEE 802.3), 100Base-TX (Fast Ethernet)
Network topology	Line
Type within the network	master
Number of nodes	Max. 65535 (in the entire network)
Max. cable length	100 m between two stations
Baud rate	100 Mbps
Supported communication profiles	CoE (CANopen over EtherCAT) FoE (File Access over EtherCAT)
Synchronisation	Distributed clocks

5.2 EtherCAT interface of the Lenze Controller

RJ45 socket	Pin	Assignment	LEDs
 E94YCEP018	1	Tx +	"Link" LED (green): On: connection ok
	2	Tx -	
	3	Rx +	"Speed" LED (yellow): Blinking: active data exchange
	4	-	
	5	-	
	6	Rx -	
	7	-	
	8	-	

5.3 Communication times and drive-specific data

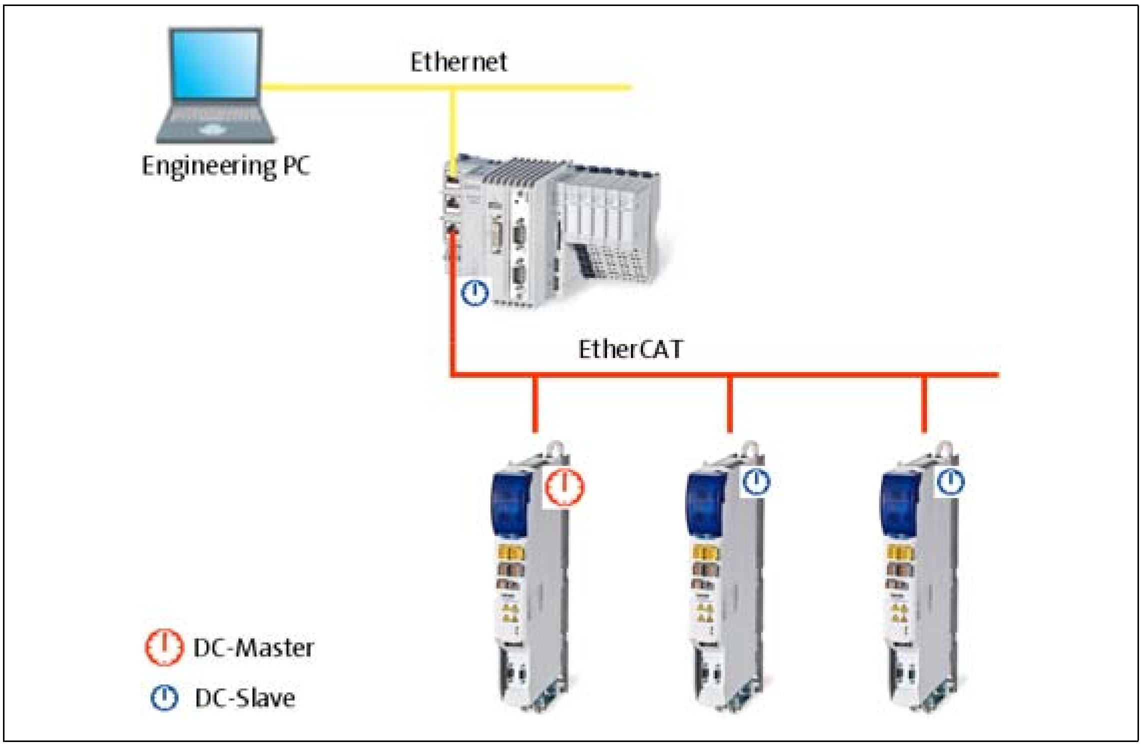
Range	Values	
User data per frame	1344 bytes	
Process data words (PCD)	Depending on the inverter used (see documentation of the inverter)	
Parameter data (SDO) transfer	Max. 128 bytes	
Permissible EtherCAT cycle times	1 ... 10 ms	
Max. number of drives per frame	User data of the frame (1344 bytes) divided by the process data length resulting from setpoints and actual values of the drives: • for 32 Tx/Rx bytes: 1344 bytes / 64 bytes = 21 drives • for 16 Tx/Rx bytes: 1344 bytes / 32 bytes = 42 drives	
Cross communication	Not possible	
Cycle synchronisation with locked PLL (Jitter)	+/-1 µs	
Instant of transmission for the EtherCAT frame (for setting, see Optimising the task utilisation (📖 106))	The EtherCAT frame is sent at the beginning of the bus cycle task.	The EtherCAT frame is sent at the end of the bus cycle task.
Total signal runtime for a cycle time of 1 ms Drive → controller → drive	4 ms	3 ms
Runtime of the setpoints Controller → drive	2 ms	1 ms
Runtime of the actual values Drive → controller	1 ms	1 ms

6 Synchronisation with "Distributed clocks" (DC)

The "Distributed clocks" (DC) functionality enables exact time synchronisation for applications in which several axes perform a coordinated movement simultaneously. Data are incorporated synchronously with the PLC program. During DC synchronisation, all slaves are synchronised with a reference clock, the so-called "DC master".

Note!

- DC synchronisation is absolutely required for Motion applications.
- DC synchronisation can also be used for Logic applications.
- Not all slaves support the DC functionality.
- In order to be able to use the DC functionality, the first slave connected to the EtherCAT master (Lenze Controller) must have **DC master capability**. The other slaves can be connected in a mixed arrangement, either DC capable or non-DC-capable.
- The first EtherCAT slave after the Lenze Controller must be the **DC master** that supplies the other EtherCAT nodes (incl. controller) with the exact time.



[6-1] **Example:** EtherCAT bus system with 3231 C controller and i700 servo inverter

The settings for the DC synchronisation are made with the »PLC Designer«.

▶ Setting a DC synchronisation (📖 88)

6.1 Synchronous communication

The DC synchronisation provides for a phase-synchronous operation of master and slaves:
Within one bus cycle the setpoints are accepted and the actual values are detected in the fieldbus at exactly the same time.

If the Lenze Controller (master) is in synch with the distributed clocks, the data (actual values) acquired by the slave are sent to the master at the end of the bus cycle and data from the master (setpoint values) are sent to the slaves for processing. .

When the next DC synchronisation event occurs, the data are accepted.

Note!

- The DC synchronisation is only carried out in the "Operational" state.
- After an "Out-of-sync" occurrence, the EtherCAT master synchronises the slaves. Successful synchronisation is indicated by means of the "In-Sync" message (see Diagnostic tabs of the EtherCAT master (184)).

6.2 Test of DC synchronicity

DC synchronicity is only available in the "Operational" state.

Test of DC synchronicity in »PLC Designer«

- EtherCAT master: **Diagnostic Master** tab
  "DC In-Sync" is set (TRUE) if the DC master and all DC slaves have been synchronised.
  ▸ Diagnostic tabs of the EtherCAT master (📖 184)
- Function block L_ETC_GetMasterDiagnostic (FB) (📖 156) / Visualisation of the function block L_ETC_GetMasterDiagnostic (📖 186)
  "DC In-Sync" is set (TRUE) at the *oDiagnostic.xDC_InSync* output if the DC master **AND** all DC slaves are synchronised.
- Function block L_IODrvEtherCAT (FB) (📖 151)
  "DC In-Sync" is set (TRUE) at the *xDistributedClockInSync* output if all DC slaves are synchronised.

Test of DC synchronicity in »WebConfig«:

In code C281/5 "DC In-Sync" is set (TRUE) if all DC slaves are synchronised.

▸ Diagnostic codes in the »WebConfig« (📖 189)

7 Commissioning of the system

This chapter provides information on how to commission the Lenze automation system with EtherCAT.

Depending on the field devices used, the following Lenze Engineering tools (📖 30) are required:

- »EASY Starter«
- »Engineer«
- »PLC Designer«

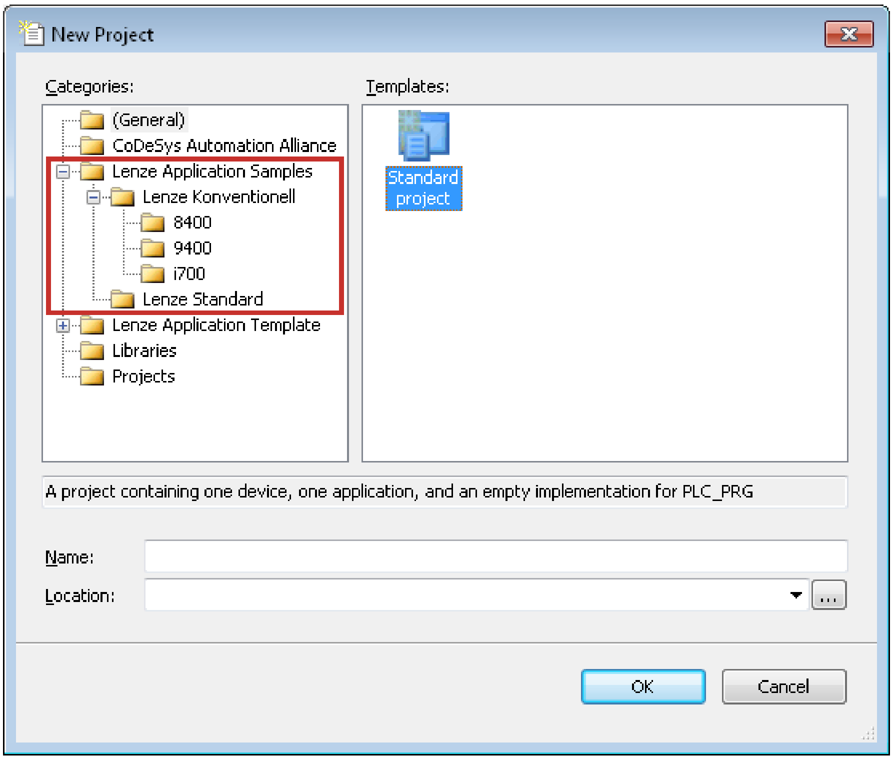
7.1 Sample projects (Application Samples)

There already exist sample projects (device application + PLC program) for commissioning of Lenze inverters.

The Lenze sample projects can be found in the **MS Windows start menu** under:

Start → All programs → Lenze → AppSamples → ...

The Lenze sample projects can also be opened in the »PLC Designer« via the menu command **File → New project...**, or using **<Ctrl>+<N>**.

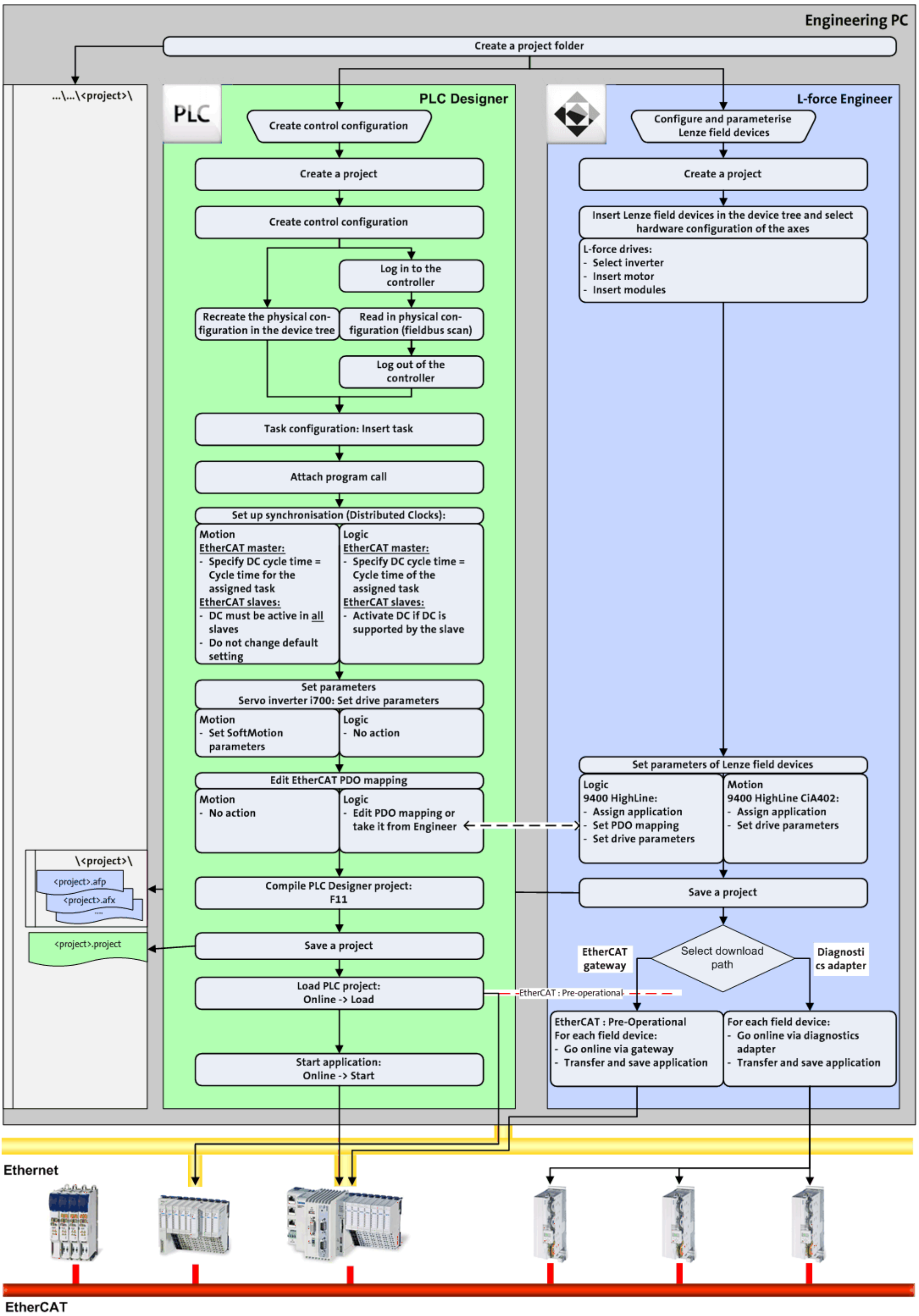


Detailed information on the sample projects can be found in the following documentation:

- SW_ApplicationSample_i700_(PLC Designer V3)_Vx-y_DE/EN.pdf
- SW_ApplicationSamples_(Controller-based)_Vx-y_DE/EN.pdf

7.2 Overview of the commissioning steps

In the following illustration, the individual commissioning steps and their processing order are summarised. Detailed information on the individual processing steps can be found in the chapter Detailed description of the commissioning steps (🕮 43).



The main commissioning steps are listed in the following table:

Step	Activity	Lenze software to be used
1.	Installing field devices (🕮 44)	
2.	Create a project folder (🕮 44)	
3.	Commissioning the i700 servo inverter (🕮 45)	»PLC Designer«
	Commissioning other Lenze field devices (🕮 71)	»Engineer« / »EASY Starter«
4.	Creating a PLC program with a target system (Logic/Motion) (🕮 72)	»PLC Designer«
5.	Configuring the communication parameters (🕮 74)	
6.	Determining the physical EtherCAT configuration (fieldbus scan) (🕮 76)	
	If necessary Importing missing devices / device description files (🕮 79)	
6.	Creating a control configuration (adding field devices) (🕮 80)	
7.	Creating a task (🕮 83)	
8.	Setting a DC synchronisation (🕮 88)	
10.	Setting SoftMotion parameters (🕮 93) Only required for drives with Motion functionality.	
11.	Processing EtherCAT I/O mapping (🕮 96) Only required for drives that solely have the master functionality (logic bus).	
12.	Compiling the PLC program code (🕮 104)	
13.	Logging in on the controller with the »PLC Designer« (🕮 104) With the log-in, the fieldbus configuration and the PLC program are loaded to the controller.	
14.	Starting the PLC program (🕮 104)	

7.3 Detailed description of the commissioning steps

In the following sections, the individual commissioning steps are described.

Follow the instructions of these sections step by step in order to commission your system.

For more details on how to use the Lenze engineering tools, please refer to the corresponding software manuals and the online help systems.

7.3.1 Planning the bus topology

Before you set up an EtherCAT network, draw up a plan of the network.

How to plan the bus topology for your configuration

1. Create an overview screen of the planned EtherCAT network with all the field devices to be integrated into the network.
2. Start with the Lenze Controller (master).
3. Add the other field devices (slaves) below.

   The following cases are distinguished:
   - Operation without synchronisation via distributed clocks:
     DC synchronisation is mostly not required if exclusively Logic field devices are to be operated on the network. The sequence of the field devices on the fieldbus can be freely selected.
   - Operation with synchronisation via distributed clocks:
     DC synchronisation is required if Motion and Logic field devices are to be operated on the network. The first node connected to the Lenze Controller must be capable of being a DC master. The sequence of the other Logic and Motion field device interface connections at the fieldbus can be freely selected.

7.3.2 Installing field devices

For the installation of a field device, follow the mounting instructions for the respective device.

Note!

- In the case of all field devices, the EtherCAT interfaces must be wired in accordance with the preceding topology planning. Make sure that the inputs (IN) and outputs (OUT) are not mixed up with each other; otherwise, the topology changes.
  ▶ communication (📖 20)
- The physical sequence of the field devices in the EtherCAT network must match the EtherCAT configuration created in »PLC Designer«.
- The master automatically assigns the node addresses to the slaves. Therefore, a manual address assignment is not required.

7.3.3 Create a project folder

Create a project folder on the Engineering PC.

Use this project folder to store the data generated in the following different project configuration steps:

- Project data created in the »Engineer« or »EASY Starter«
- The project file created in the »PLC Designer«

Tip!

Create a separate project folder for every EtherCAT configuration and store the project files.

7.3.4 Commissioning the i700 servo inverter

This chapter tells you how to commission the Servo-Inverter i700 in the Lenze automation system with the help of »PLC Designer«.

For speed commissioning, the Servo-Inverter i700 provides diverse functions for automatic calculation and setting of parameters.

Danger!

Severe personal injury and damage to the machine/installation

Activate the controller inhibit before you set the parameters for the Servo-Inverter i700 as uncontrolled movements can lead to severe personal injury and damage to the machine/installation.

Note!

You can also set the parameters for the Servo-Inverter i700 online. To do this, you must first configure EtherCAT communication.

Setting parameters online is possible from the EtherCAT state "Pre-Operational" onwards.

Reference manual/online help for the i700 servo inverter

Here, you can find detailed information on all parameters/Objects (object dictionary), functions and error messages of the i700 servo inverter.

Tip!

There already exist sample projects (device application + PLC program) for commissioning of the i700 servo inverter.

▸ Sample projects (Application Samples) (📖 40)

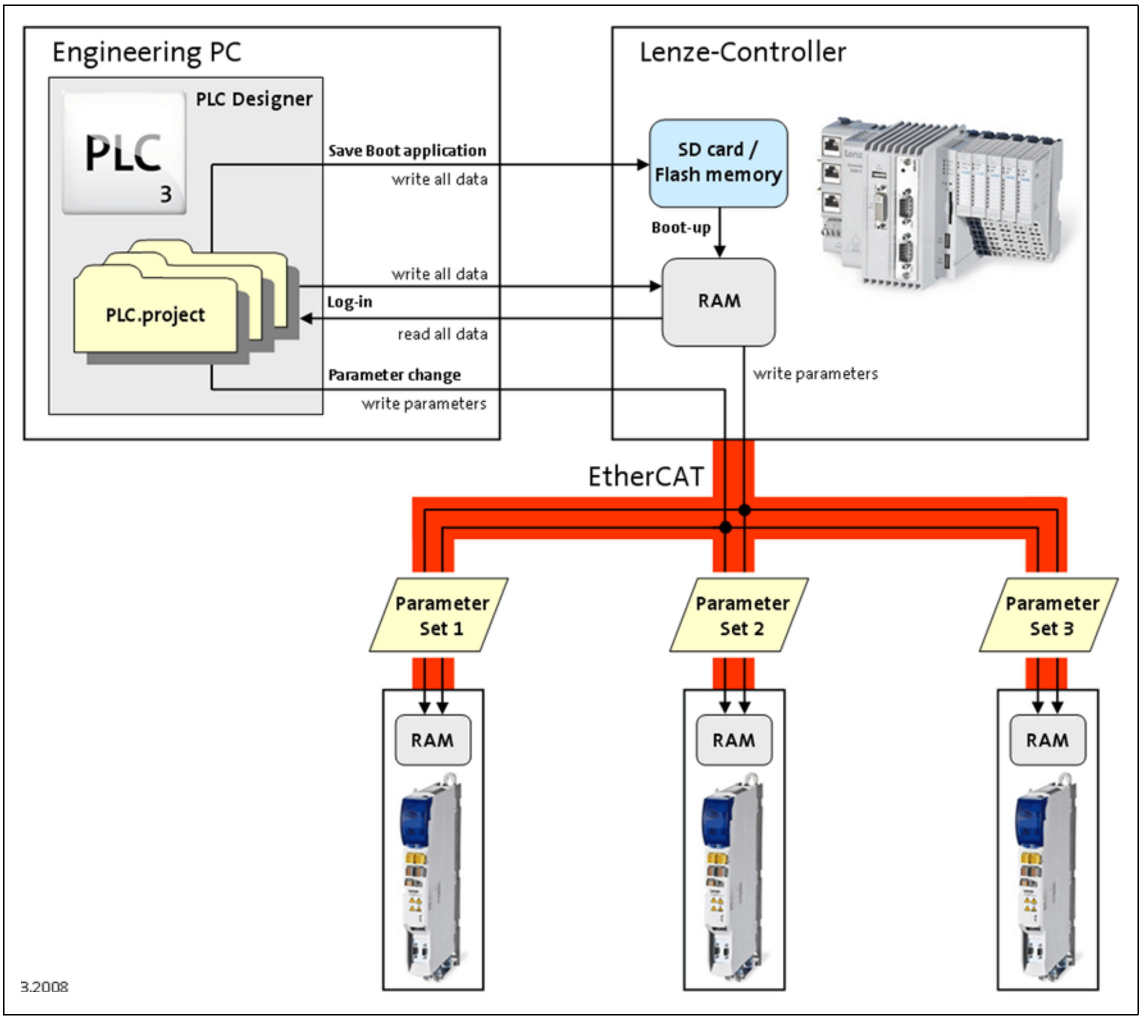
7.3.4.1 i700 parameter management in the Controller-based Automation system

Parameter download

The Servo-Inverter i700 itself does not store parameter settings safe against mains failure. All servo inverter settings deviating from the Lenze standard setting are maintained centrally in the Lenze Controller and stored there permanently (persistently). During the initialisation at run-up, only these deviations are transferred to the servo inverter by the controller. Like this it is ensured that the servo inverter works with the parameter settings provided for it.

Firmware download (optional)

The firmware of the i700 servo inverter can – if so desired – be stored together with the »PLC Designer« project. When the system is being powered up, the Lenze controller then checks whether the firmware version in the servo inverter matches the firmware version stored in the project for this device. If this is not the case, the controller loads the firmware version stored in the project into the servo inverter. In the event of service whereby a device has to be replaced, it can thus be ensured that the replacement device works with the same firmware version stored in the project for the original device.



[7-1] Parameter set transfer from »PLC Designer« to the i700 servo inverter via the 3231 C controller

The parameters of the i700 servo inverter are managed within the »PLC Designer« project. With the help of the storage function of »PLC Designer«, the »PLC Designer« project, including the i700 parameters, are stored on the Engineering PC.

During login to the Lenze Controller,»PLC Designer« transfers the i700 parameters to the controller. The latter, in turn, transfers the parameter sets to the lower-level Servo-Inverter i700.

Whenever the system is booted, the parameter sets are written into the Servo-Inverter i700 again by the Lenze Controller.

There are three cases of application for the management and alteration of parameters:

A. Changing parameters of an Servo-Inverter i700 online:

If a parameter is changed online, »PLC Designer« writes the parameter directly into the corresponding servo inverter and, at the same time, changes the parameter in the»PLC Designer« project.

N.B.:

The parameter change is not recorded in the Lenze Controller. In the event of a "cold reset" of the controller, the parameter change in the servo inverter is lost.

In order to avoid this, log out with the »PLC Designer« and then log in again. (Menu commands: **Online → Logout / Online → Login**)

Like this, the complete parameter set is written to the controller and is transferred to the servo inverter.

After the transfer by log-in, the parameter set is only available in the servo inverter and in the controller until the next switch-off of the system (it is not saved with mains failure protection).

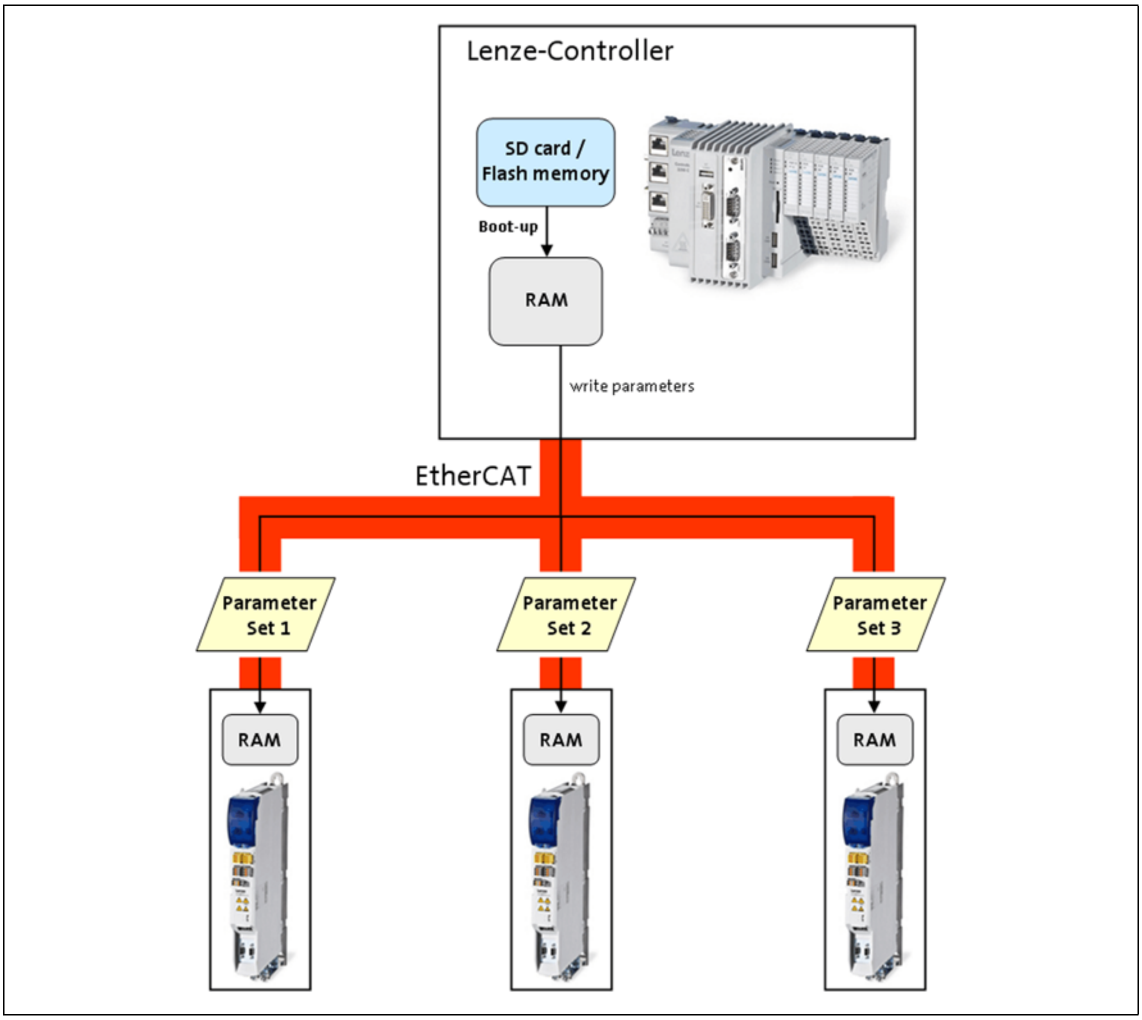
B. Changing parameters of a Servo-Inverter i700 offline:

If a parameter is changed offline, the »PLC Designer« changes the value directly in the »PLC Designer« project. When logging in on the Lenze Controller, the complete parameter set is written to the controller, which, in turn, then transfers the parameter set to the servo inverter.

After the transfer by log-in, the parameter set is only available in the servo inverter and in the controller until the next switch-off of the system (it is not saved with mains failure protection).

C. Store the parameters of the Servo-Inverter i700 in a non-volatile memory so that they are retained in the device when the power is switched off:

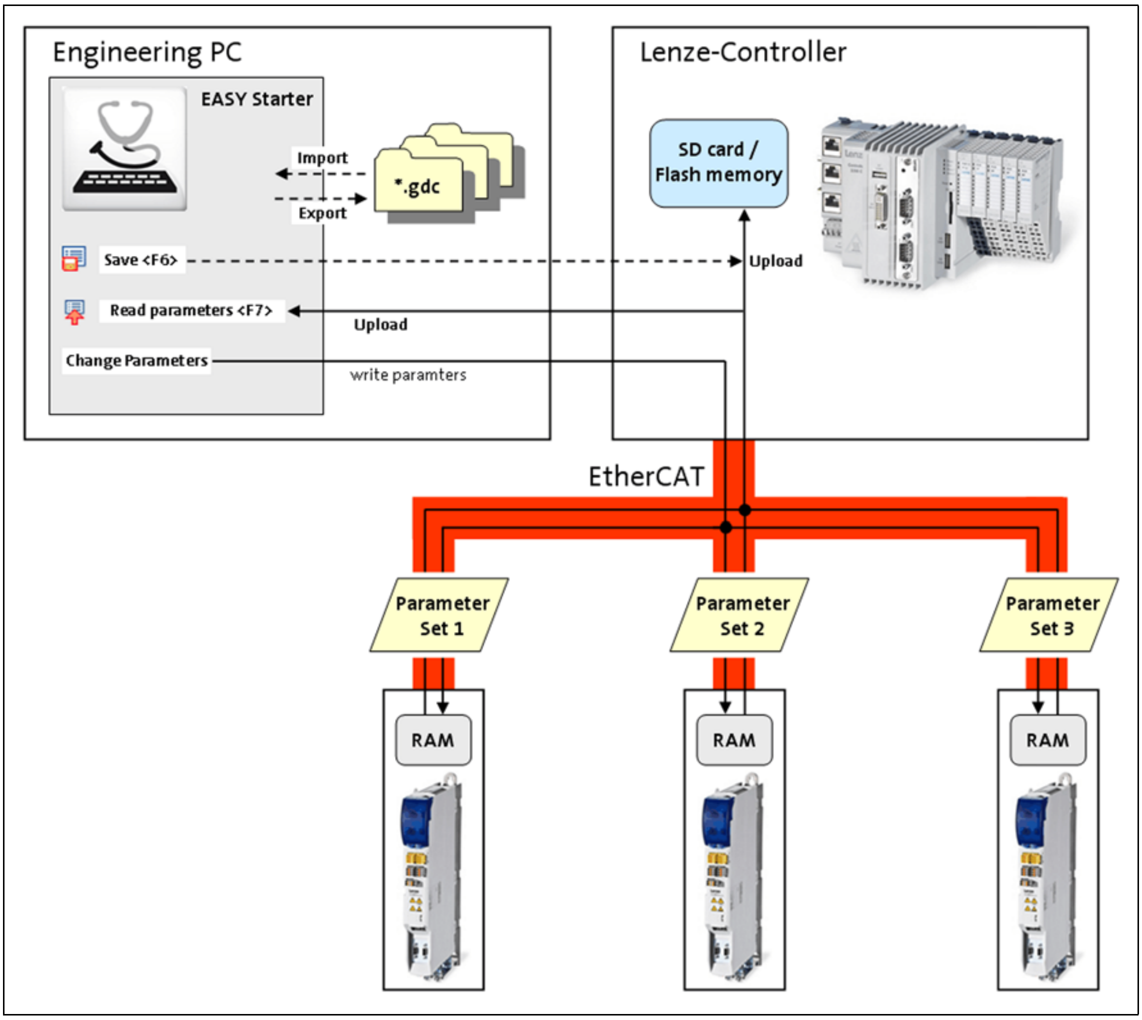
   In the boot project of the Lenze Controller, there is a separate parameter set for the lower-level Servo-Inverters i700. The latter is then only updated when the "boot project" function is executed via the »PLC Designer«. The current parameterisation is then saved and is available after mains switching. During boot-up after mains connection, the saved parameter set is automatically written from the controller into the servo inverter.



[7-2] Parameter set transfer from the 3231 C controller to the Servo-Inverter i700 during a boot-up

7.3.4.2 i700 parameter management in »EASY Starter«

For parameter management in »EASY Starter«, you must insert the Lenze Controller and the Servo-Inverter i700 into the device list with the help of the button.



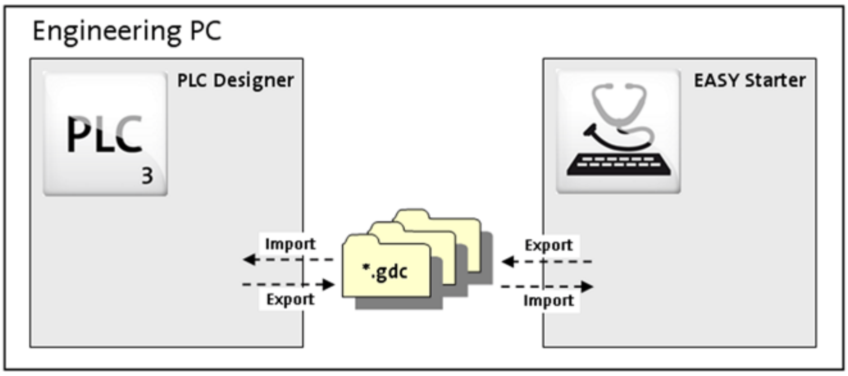
[7-3] Parameter set transfer with »EASY Starter«

With »EASY Starter«, there are two application cases for parameter management:

A. Storing the current Servo-Inverter i700 parameter setting in the Lenze Controller with mains failure protection:
   - Select the Lenze Controller from the device list.
   - With the button or the **<F6>** function key, store the parameter set in the controller so that it is saved there even in the event of a power failure. (To this end, the controller uploads all the parameter sets of the lower-level i700 servo inverter.)
   - After the system has been switched off and then switched on again. the controller transfers the parameter sets to the i700 inverter again.

B. Upload the parameters currently set in a Servo-Inverter i700 to the Engineering PC and store them there:
   - Select a Servo-Inverter i700 in the device list
   - With the button or the **<F7>** function key, upload the parameter set of the servo inverter. (The parameter set of the servo inverter is transferred directly to »EASY Starter« without the parameter set being stored on the controller.)
   - With the **Save** button, store the parameter as a **GDC file** on the Engineering PC.

7.3.4.3 Exchanging i700 parameter sets between »PLC Designer« and »EASY Starter«



[7-4] Parameter set transfer between »PLC Designer« and »EASY Starter«

The **GDC files** that have been stored on the Engineering PC with the help of »EASY Starter« can be imported in »PLC Designer«.

It is also possible to export GDC files with »PLC Designer«; they can then be imported in »EASY Starter«.

In the framework of this import/export functionality, the parameter settings of a single axis can be imported into a double axis, for example.

How to import parameters in »PLC Designer«

1. Log out of the Lenze Controller with the menu command **Online → Logout** or **<Ctrl>+<F8>**.
2. Select the corresponding Servo-Inverter i700 in the project tree.
3. Select and import the corresponding GDC file with the menu command **Project → Device parameters → Import device parameters**.
4. Use the menu command **Online → Login** or **<Alt>+<F8>** to log in on the Lenze Controller.
   - For this, the PLC program must be error-free.
   - The complete parameter set is written into the controller. The latter than transfers the parameter set to the i700 servo inverter.

Parameters are exported in »PLC Designer« in a similar way with the help of the menu command **Project → Device parameters → Export device parameters**.

7.3.4.4 Overview of the commissioning steps

The main commissioning steps are listed in the following table:

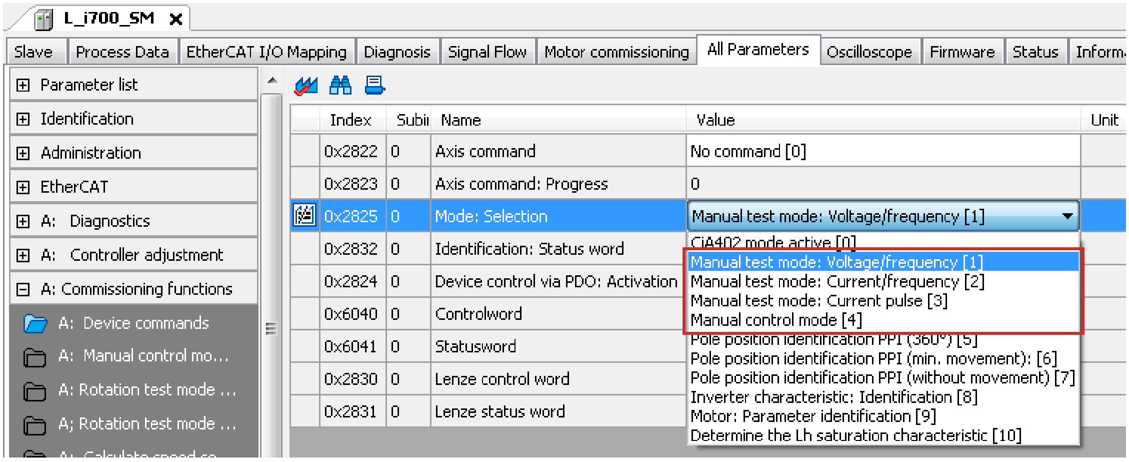
Step	Activity
1.	Create a project folder (🕮 44)
2nd	Creating a PLC program with a target system (Logic/Motion) (🕮 72)
3.	Configuring the communication parameters (🕮 74)
4.	Determining the physical EtherCAT configuration (fieldbus scan) (🕮 76) or Creating a control configuration (adding field devices) (🕮 80)
5th	Creating a task (🕮 83)
6.	Setting a DC synchronisation (🕮 88)
6.	Checking the wiring (🕮 55)
7.	Entering motor and controller settings (🕮 55)
8.	Setting the feedback system for servo control (🕮 58)
10.	Integrating the L_SMC_AxisBasicControl function block (🕮 60)
11.	Setting SoftMotion parameters (🕮 93) Only required for drives with Motion functionality.
12.	Processing EtherCAT I/O mapping (🕮 96) Only required for drives that solely have the master functionality (logic bus).
13.	Compiling the PLC program code (🕮 104)
14.	Logging in on the controller with the »PLC Designer« (🕮 104) With the log-in, the fieldbus configuration and the PLC program are loaded to the controller.
15.	Starting the PLC program (🕮 104)
16.	Executing manual control (🕮 66)
17.	Optimising control (🕮 68)

7.3.4.5 Checking the wiring

Before you start to set the parameters of the drive control system, check the wiring of the motor (power and encoder connections) for faults.

For this purpose, you can activate the following test modes with the controller command **Operating mode (0x2825 / 0x3025)**:

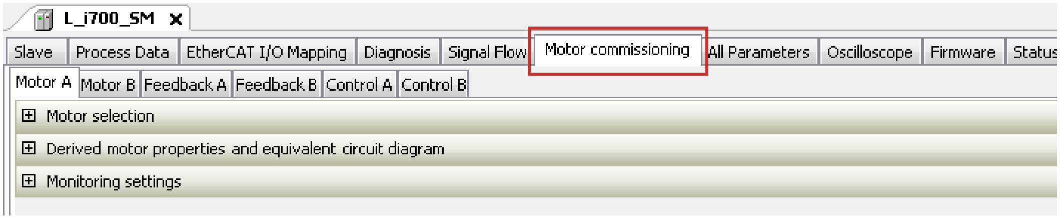
- Test mode: Voltage/frequency
- Test mode: Current/frequency
- Test mode: Current pulse
- Executing manual control (📖 66)



[7-5] Example: operating modes for i700 servo inverter, double axis

7.3.4.6 Entering motor and controller settings

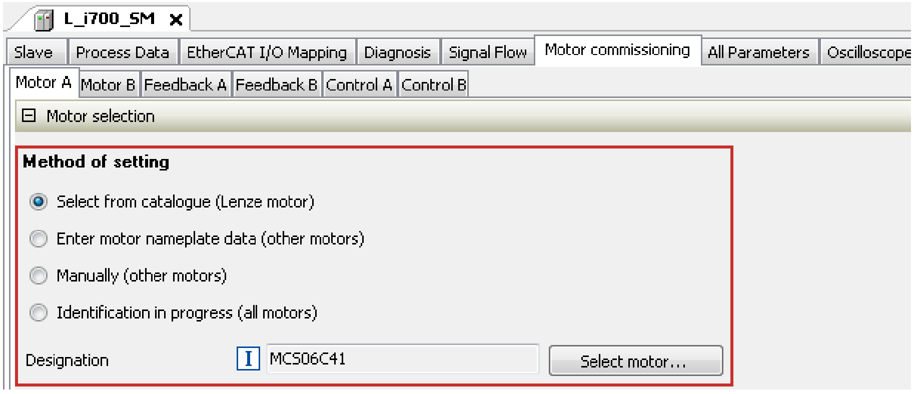
You can enter the motor and controller settings on the **Motor commissioning** tab of the i700 servo inverter.



[7-6] Example: i700 servo inverter, double axis

How to enter the motor and controller settings

1. Go to the **Motor selection** section and specify the data of the motor to be operated in conjunction with the i700 servo inverter.



A. Select the Lenze motor from the motor catalogue by clicking the **Select motor...** button.

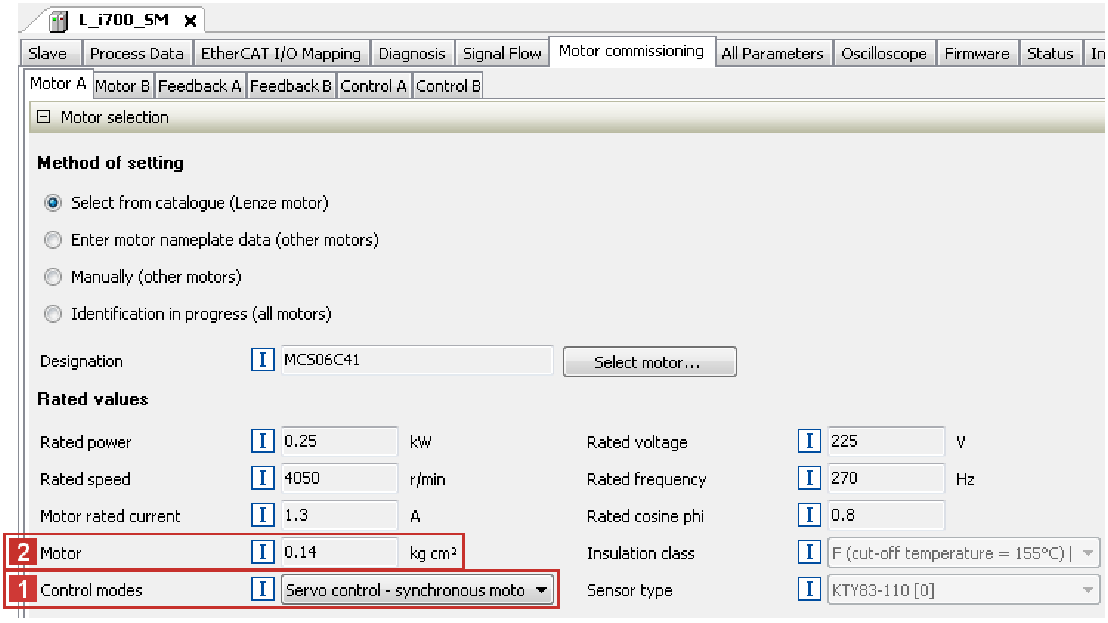
- The corresponding motor data will then be incorporated in the »PLC Designer« project.
- The controller parameters are calculated automatically.

or

B. Enter the motor data manually or specify them by means of an identification run (e.g. in the case of motors from other manufacturers).

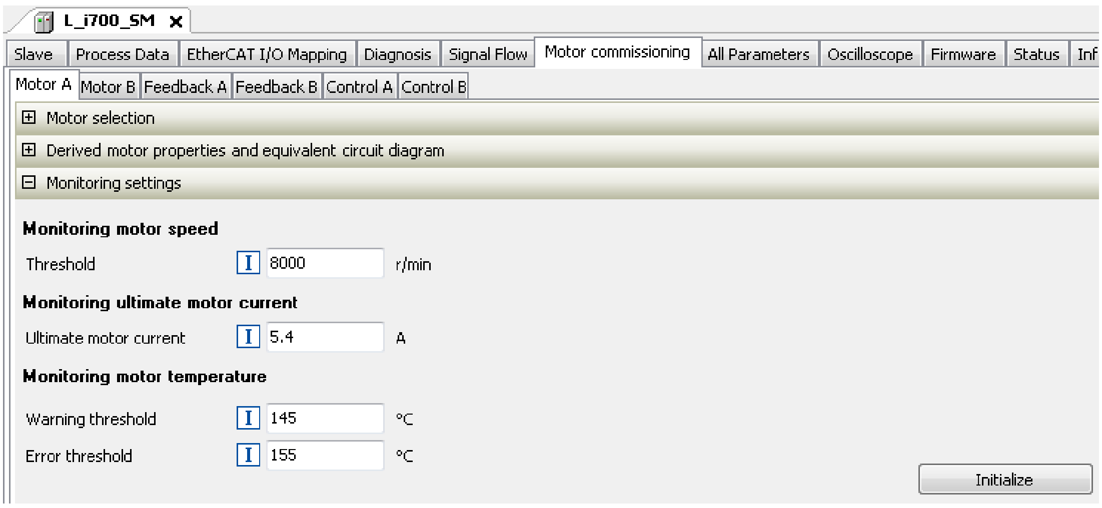
The controller parameters are calculated automatically.

2. 1 Select **Control method (0x2C00 / 0x3400)**:
   - Servo control for synchronous motor (SM)
   - Servo control for asynchronous motor (ASM)
   - VFC: V/f characteristic control

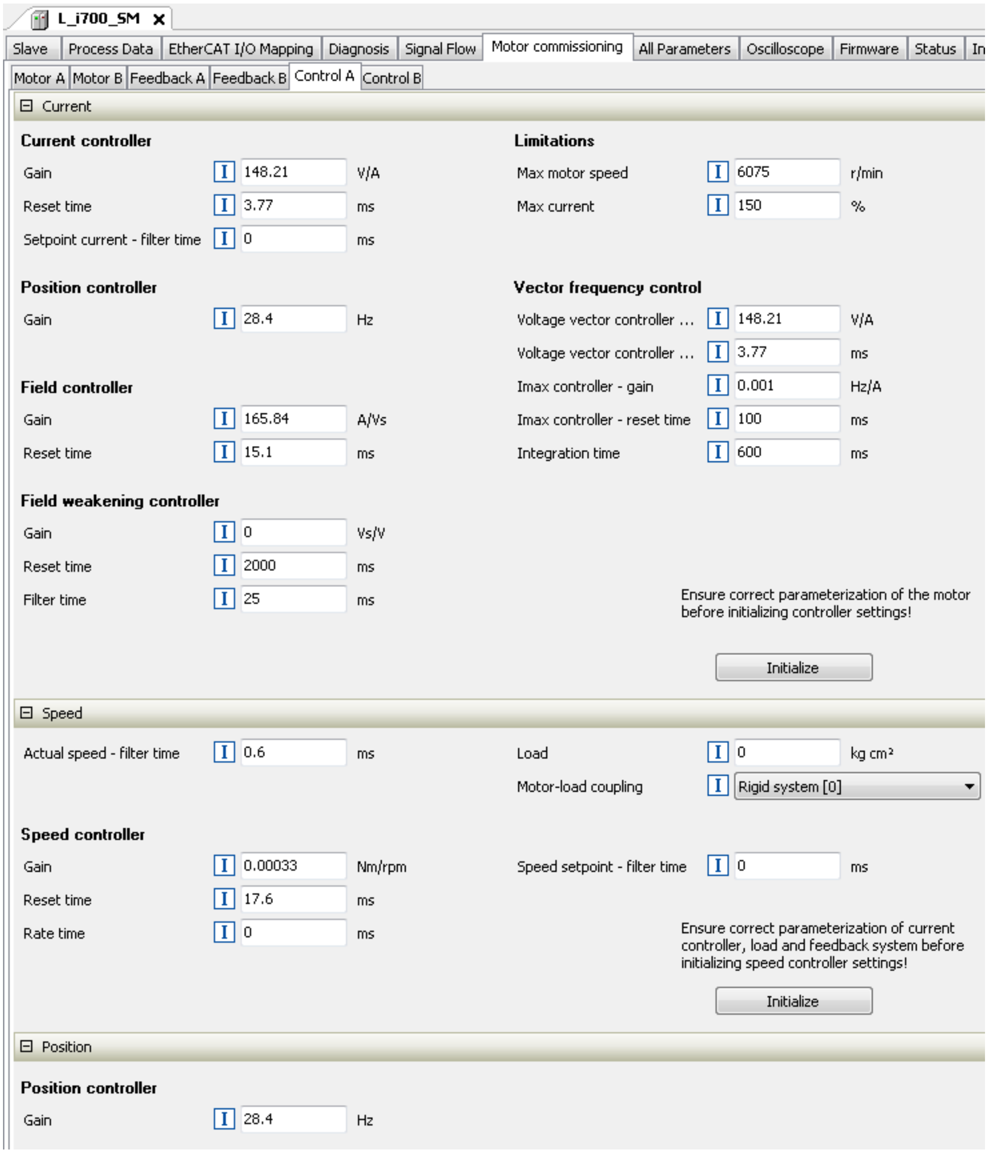


3. If you do not obtain the motor data from the catalog, you have to set the 2 **moment of inertia (0x2910/1 / 0x3110/1)** to a non-zero value.

   The controller parameters are calculated automatically.
4. You can select the parameters for the monitoring functions in the **Monitoring settings** section.



5. Retain or alter the automatically calculated controller parameters under **Control**.



7.3.4.7 Setting the feedback system for servo control

Danger!

Using the encoder/resolver as a motor encoder

In the Lenze setting, the resolver/encoder cable is monitored for open circuit.

In the event of a fault, safe operation of the motor is no longer guaranteed!

- For safety reasons, always select "Fault" (Lenze setting) as a response for the (open-circuit) monitoring of the encoder/resolver.
- To avoid interference injections when an encoder is used, only use shielded motor and encoder cables.

Determining the pole position of the synchronous motor

Pole position identification (angle between the motor phase U and the field axis of the rotor) is necessary ...

- for servo control with a synchronous motor of another manufacturer;
- for servo control with a synchronous motor and use of incremental encoders (TTL or sin/cos encoder and resolver);
- after changes to the feedback system (e.g. replacement of the encoder).

For Lenze motors with an absolute encoder or resolver, the pole position has already been set correctly.

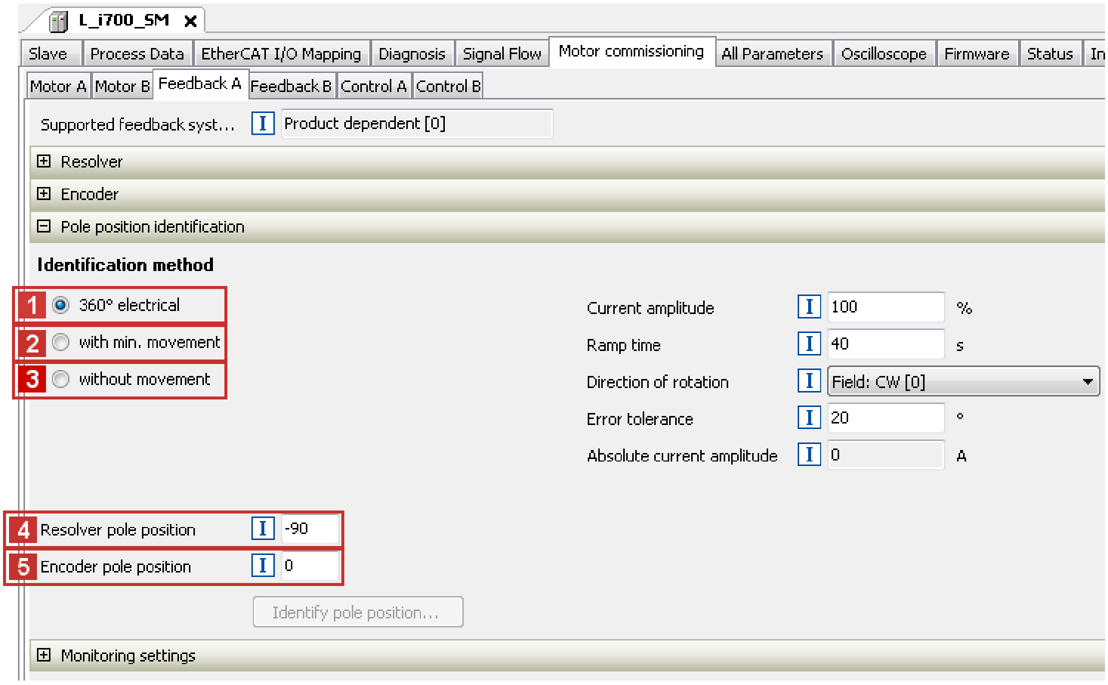
Danger!

- During pole position identification, it must be ensured that the motor is not braked or blocked. Pole position identification is therefore not permissible in the case of hanging loads.
- The rotor will align itself during pole position identification. The motor shaft will make a maximum of one electrical rotation, as a result of which the connected machine will move accordingly.
- For firmly braked motors, the function "Pole position identification PPI (without movement)" must be used.

Stop!

Before carrying out pole position identification, check that the parameters for maximum current monitoring have been set correctly so that the motor is not permanently damaged in the event of a fault.

Either of the functions 1 **Pole position identification for 360°**, 2 **Pole position identification with min. movement** and 3 **Pole position identification without movement** can be selected in order to determine the pole position for the currently activated motor encoder:



The functions should deliver approximately the same result. Due to e.g. friction, bearing forces, and a trapezoidal field curve, however, the results can differ from each other. Here the method with one full revolution (360°) will provide the most precise results and the method without any movement will provide the most inaccurate results. The precision of the results can be increased by increasing the percentage of the current amplitude.

After successful completion of pole position identification ...

... the controller inhibit is set automatically and the 4 **Resolver pole position (0x2C03/2 / 0x3403/2)** or the 5 **Encoder pole position (0x2C03/4 / 0x3403/4)** determined for the activated feedback system is set.

- For permanent storage, the changed settings must be uploaded from the Servo-Inverter i700 to the Lenze Controller.
  The »EASY Starter« can be used to upload the parameters of the servo inverter and to save them as a file. This file can then be imported to the »PLC Designer«.
- The controller inhibit automatically set by the procedure can be deactivated via the **CiA402 control word (0x6040 / 0x6840)**.

Fault

If an error occurs during pole position identification or if pulse inhibit becomes active (e.g. due to short-term undervoltage), the procedure is ended with controller inhibit without the settings being changed.

If the motor was braked or blocked during pole position identification, this is detected at the end of a measurement and no changes are made.

If pole position identification is aborted, the reaction set in the object, namely **0x2C60/0x3460 (monitoring of pole position identification: reaction)** is triggered (Lenze setting: Fault).

7.3.4.8 Integrating the L_SMC_AxisBasicControl function block

For operating the Servo-Inverter i700, the **L_SMC_AxisBasicControl** function block has to be integrated into the »PLC Designer« project.

The function block ...

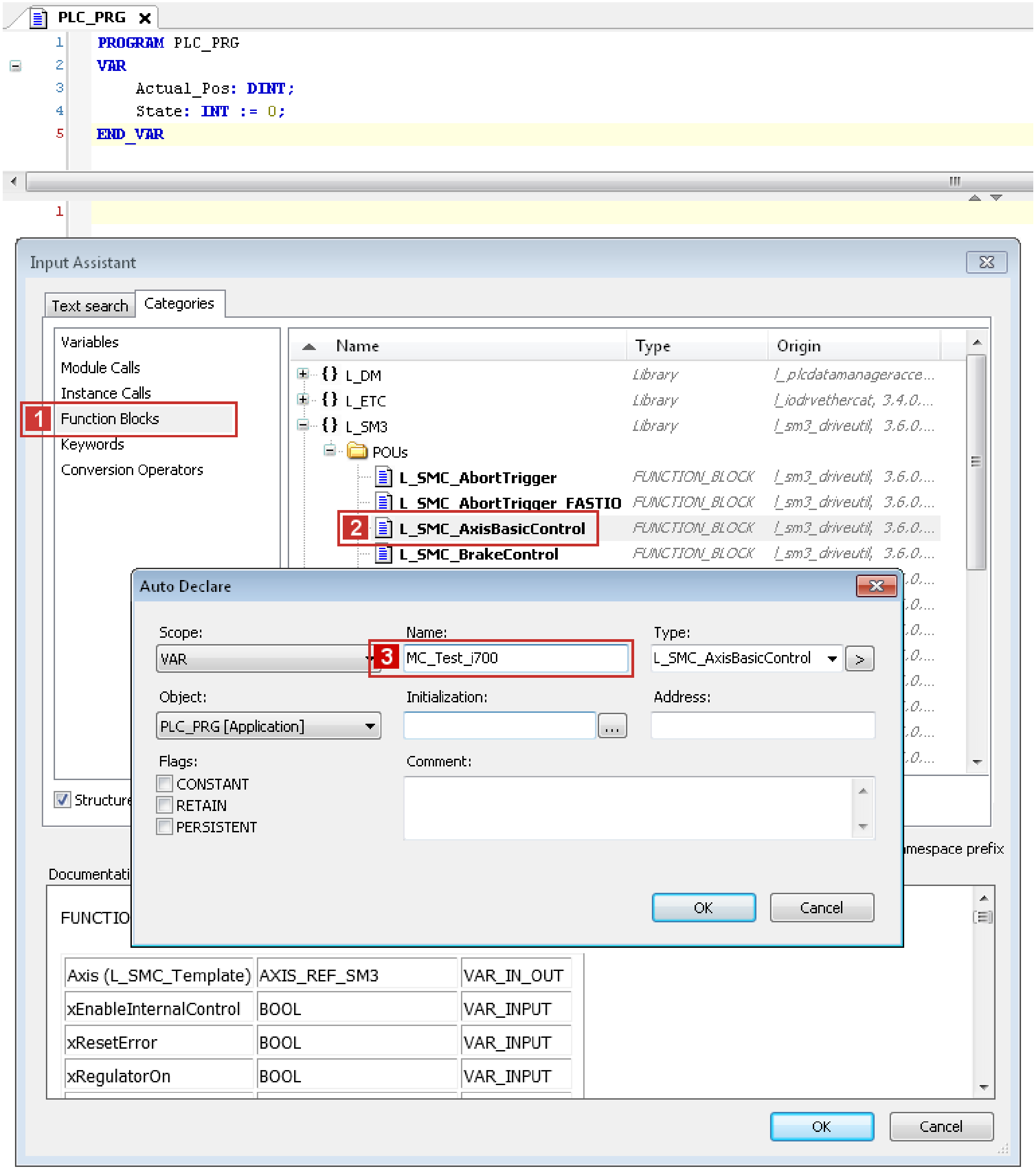
- contains various variables for drive control (e.g. for quick stop function (QSP), following error monitoring, etc.);
- is part of the **L_SM3_DriveUtil** function library.

»PLC Designer« Online help

Here you can find detailed information on the function block.

How to integrate the L_SMC_AxisBasicControl function block into the »PLC Designer« project:

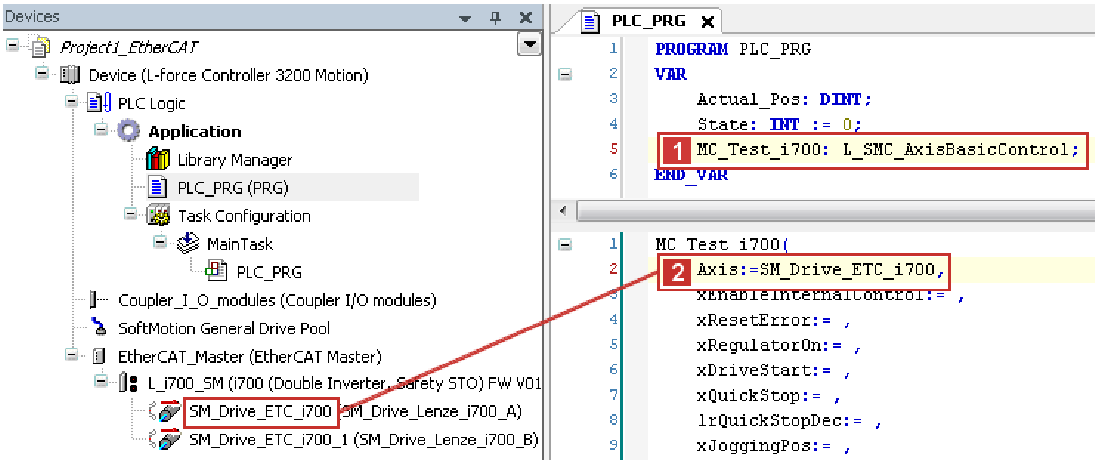
1. Open the PLC program code (PLC_PRG).
2. Open **Input assistance** in the lower input area with a right mouse click via the context menu.



3. **1** Open the category **Function blocks**.
4. **2** In the element **L_SM3**, select **POUs** and then the **L_SMC_AxisBasicControl** function block.
5. **3** In the "Declare variables" dialog box, enter a variable name (in the example above: "MC_Test_i700").

6. Close the variable declaration by clicking the **OK** button.

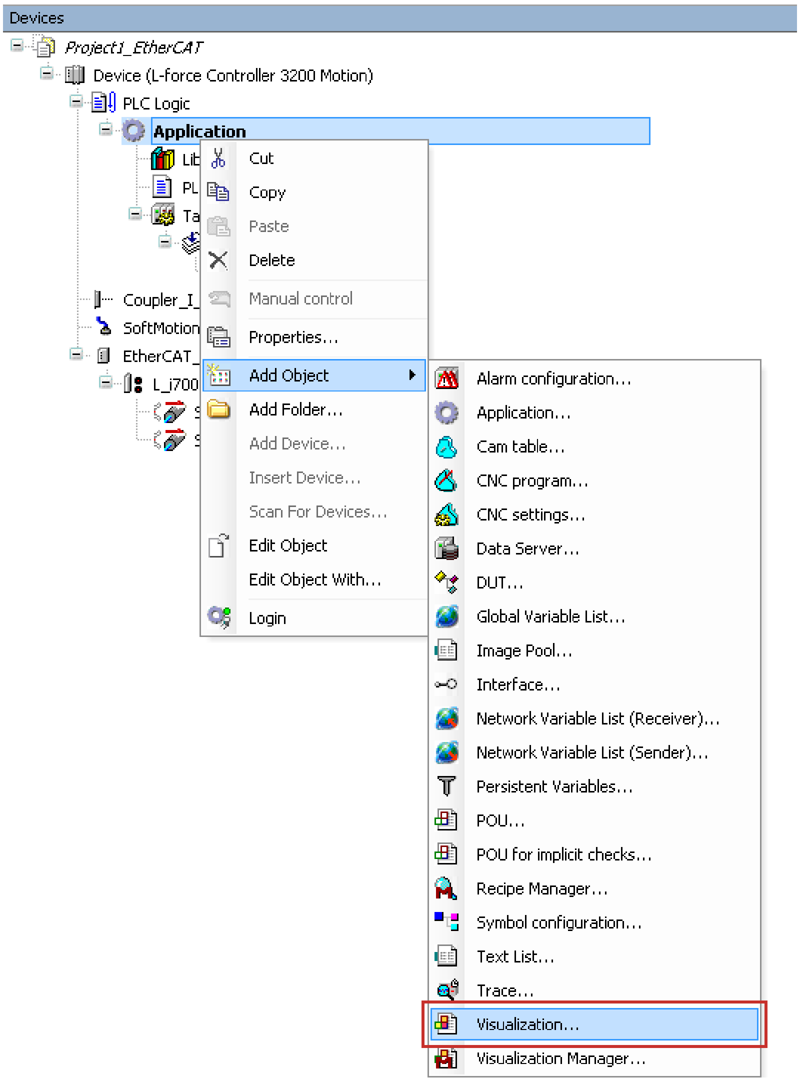
   1 The **L_SMC_AxisBasicControl** function block, together with its data structure, is integrated into the PLC program code.



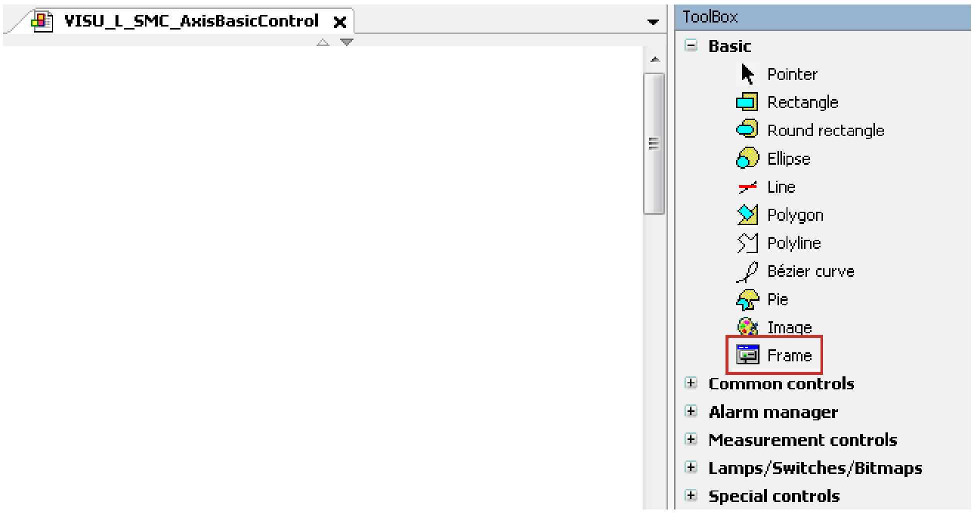
7. 2 Open the **L_SMC_AxisBasicControl** function block and set the reference to the axis data structure (in the example "Axis:= SM_Drive_ETC_i700").

8. Open the context menu for **Application**, select the command **Add object → Visualisation...** and then insert the visualisation of the function block.

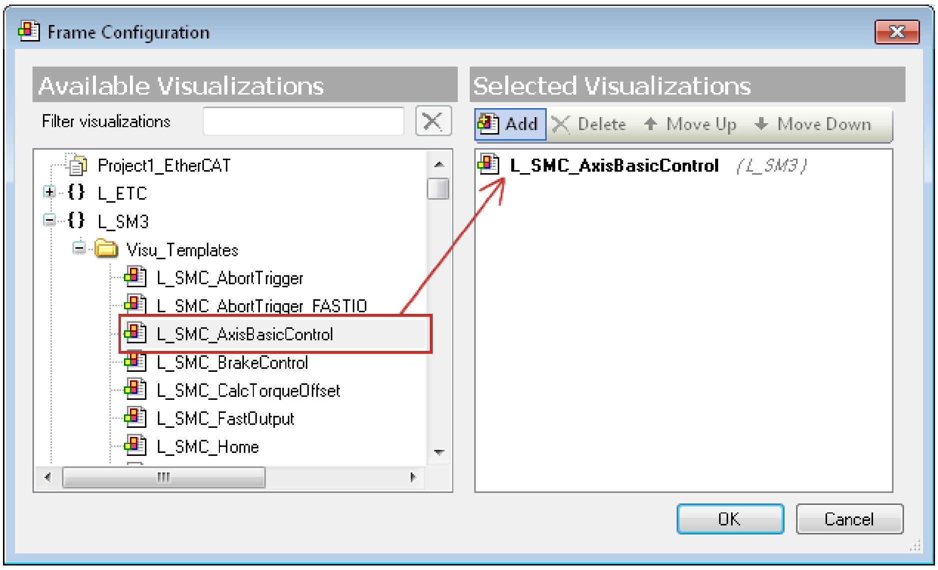
   Enter an appropriate name (e.g. "VISU_L_SMC_AxisBasicControl").



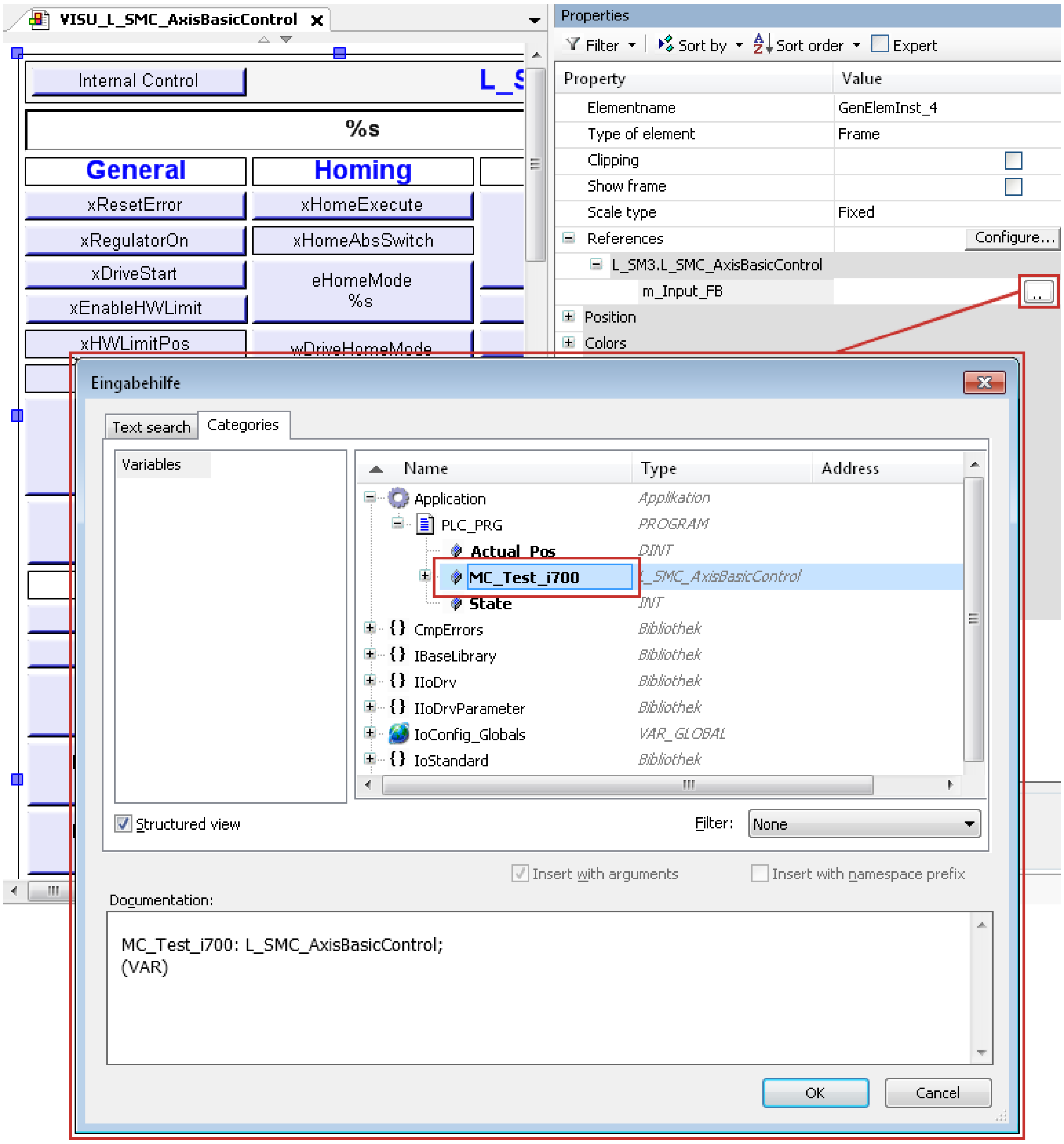
9. Insert a frame in the visualisation with the help of the **frame tool**.



10. Add the frame visualisation of the function block **L_SMC_AxisBasicControl** and close the dialog box by clicking the **OK** button.



11. Under "Properties", select the reference of the function block with which the visualisation is to be connected (in the example "MC_Test_i700").



12. Confirm your selection by clicking the **OK** button.
13. Translate the PLC program code.

    Menu command **Build→ Build** or **<F11>** function key
14. Store the »PLC Designer« project in the project folder.

    Menu command **File → Save project / Save project as ...**

7.3.4.9 Executing manual control

The purpose of manual control is to check the wiring (test mode) and to carry out a traversing movement.

Note!

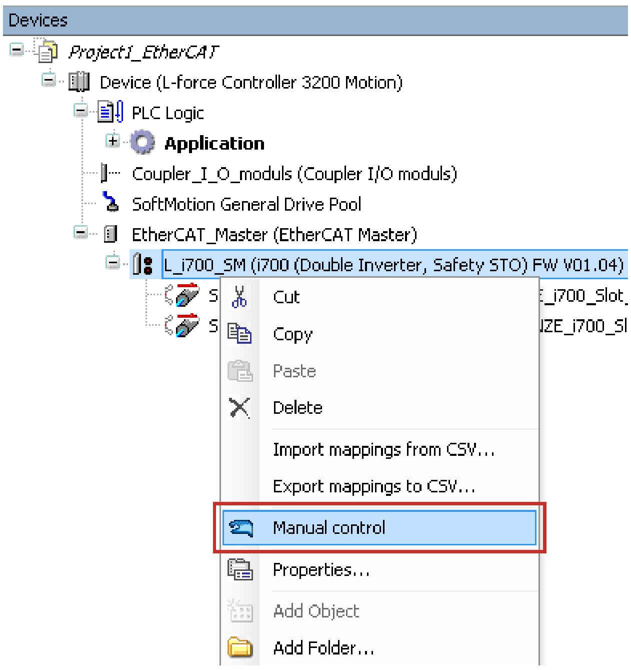
For problem-free manual control, the machine parameters – at least the gearbox factor and feed constant – must be set correctly.

Preconditions for manual control

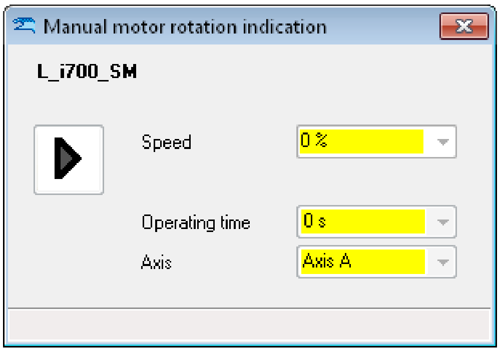
- No trouble is active.
- The mains voltage is switched on.
- The STO function (Safe Torque Off) is not active.
- The inverter drive has been inhibited by means of the software.

How to activate manual control

1. Use the menu command **Online → Login**, or log in on the Lenze Controller with **<Alt>+<F8>**.
   - For this, the PLC program must be error-free.
   - By means of the log-in, the PLC program is loaded into the controller. In doing so, any existing program is overwritten.
2. If the inverter has been enabled, inhibit it via »PLC Designer«.
3. Open the context menu of the Servo-Inverter i700 to be traversed and execute the command **Manual control**.

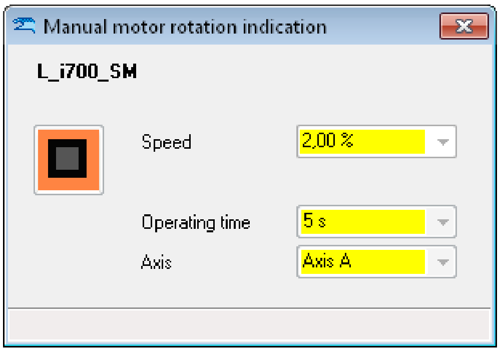


The dialog box for manual control appears:



4. Enable the inverter drive via the »PLC Designer«.
5. Enter the traversing speed (speed) in the dialog box.

   Value as a percentage of maximum motor speed (0x6080):
   - Positive % value: clockwise movement
   - Negative % value: counterclockwise movement
6. Click the button to start manual mode.



Manual mode can be stopped by clicking the button.

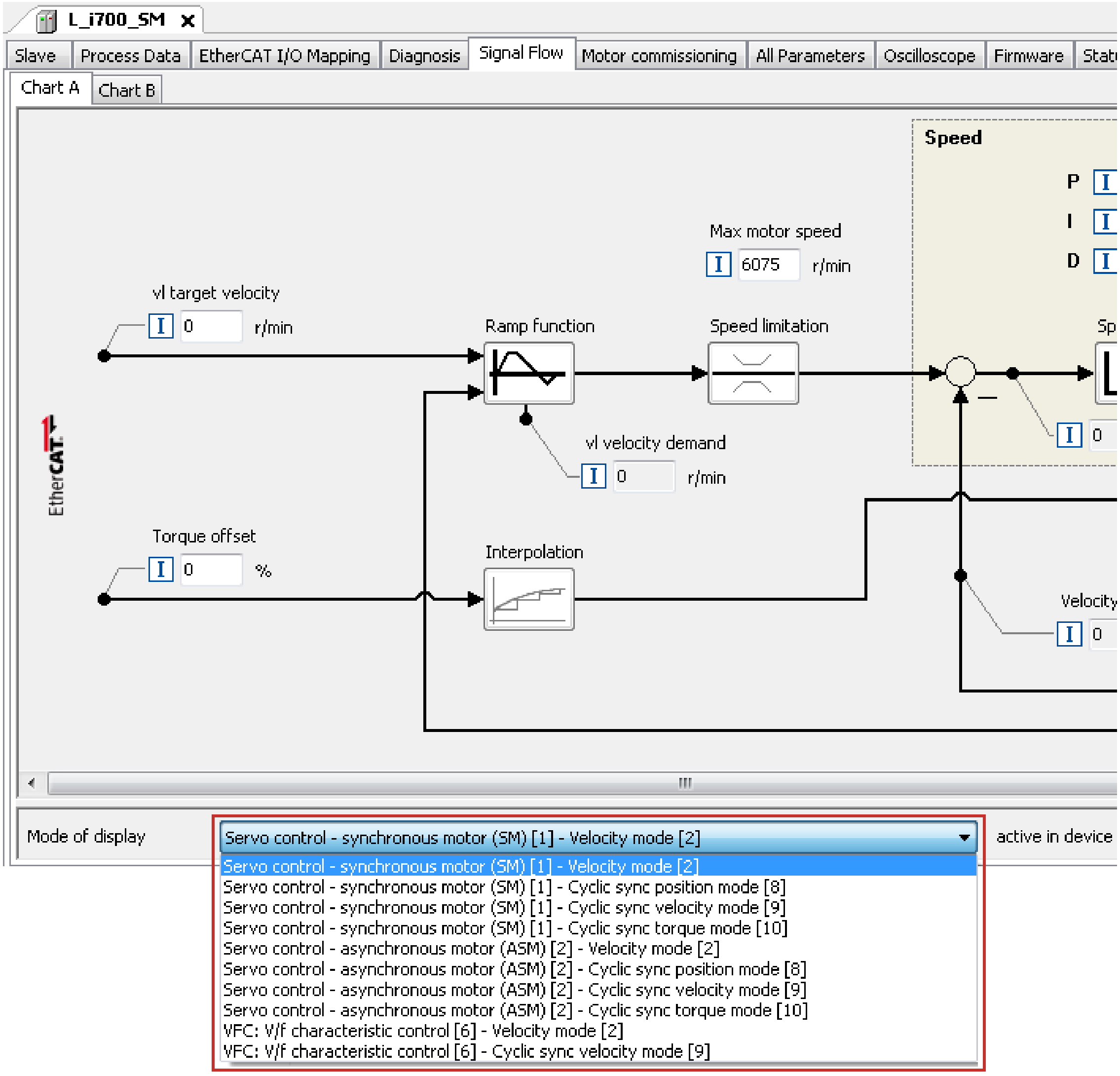
7.3.4.10 Optimising control

The final controller settings are carried out online during commissioning with load on the real machine.

How to optimise the control system

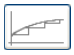
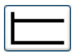
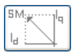
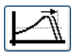
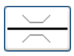
1. Use the menu command **Online → Login**, or log in on the Lenze Controller with **<Alt>+<F8>**.
   - For this, the PLC program must be error-free.
   - By means of the log-in, the PLC program is loaded into the controller. In doing so, any existing program is overwritten.
2. Press the **<F5>** function key to start the PLC program.
3. Open the tab **Signal flow → Chart A** and select the display mode (operating mode).

   The corresponding signal flow diagram is shown with the current parameter values.



4. Adapt parameter values in order to optimise the control system.
    - In the signal flow diagrams, you can adapt some parameter values directly in the corresponding input fields.
    - In addition, you can adapt parameter values for particular functions by means of function buttons, depending on the operating mode being used (see table below).

Functions and buttons in the signal flow diagrams

Function	Button	Operating mode					
		Servo control SM/ASM				V/f control	
		Velocity mode	Cyclical synchr. position mode	Cyclical synchr. velocity mode	Cyclical synchr. torque mode	Velocity mode	Cyclical synchr. velocity mode
Ramp function		●				●	
Speed limitation		●	●	●		●	●
Speed controller		●	●	●			
Speed restriction					●		
Torque limitation		●	●	●	●		
Interpolation		●	●	●	●		●
Angle/position controller			●				
Field-oriented control		●	●	●	●		
Slip compensation						●	●
Oscillation damping						●	●
Load adjustment						●	●
V/f characteristic						●	●
PWM control DC braking Flying restart function						●	●

7.3.5 Commissioning other Lenze field devices

Parameterise the Servo Drives 9400, Inverter Drives 8400 and the I/O system 1000 (EPM-Sxxx) connected to the EtherCAT network) using the »Engineer« or »EASY Starter«.

EtherCAT is exclusively configured by means of the »PLC Designer«.

EtherCAT settings of the field devices which have possibly been carried out with the »Engineer«/ »EASY Starter« are overwritten.

Documentation of the Lenze field devices

Here you are provided with some detailed information relating to the commissioning of the Lenze field devices.

Tip!

We recommend to commission each field device individually and then integrate them into the PLC program.

There already exist sample projects (device application + PLC program) for commissioning of Lenze inverters.

▸ Sample projects (Application Samples) (📖 40)

7.3.6 Creating a PLC program with a target system (Logic/Motion)

The »PLC Designer« serves to model the network topology in the control configuration.

Tip!

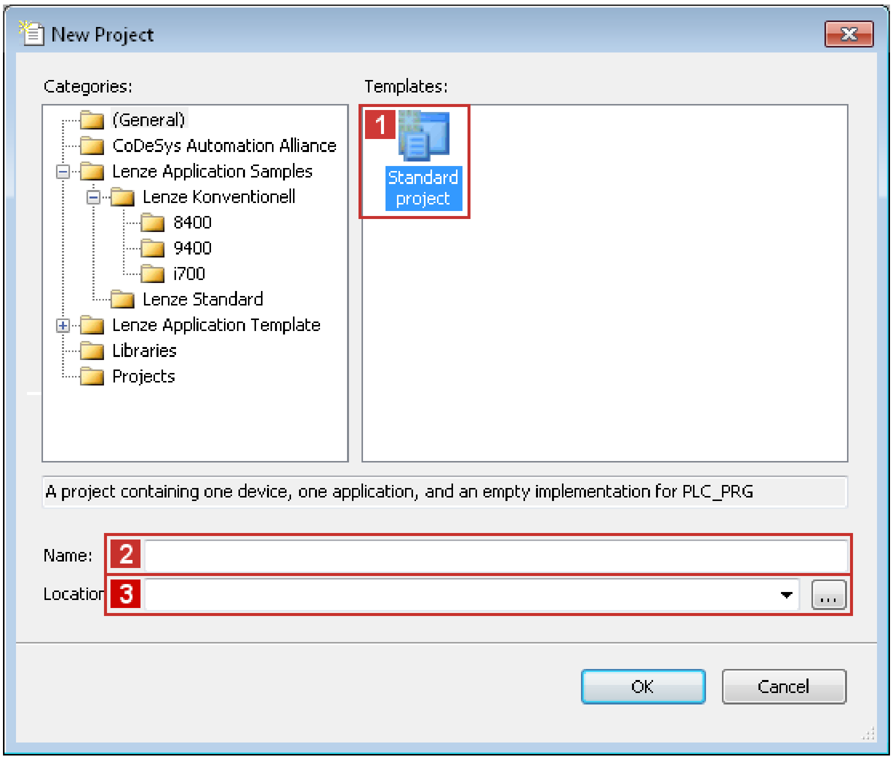
In »PLC Designer«, EtherCAT nodes as well as nodes of other fieldbus systems can be configured.

▶ Mixed operation - EtherCAT with other bus systems (📖 115)

How to create a PLC program in the »PLC Designer«:

1. Use the menu command **File → New project** to create a new »PLC Designer« project.
2. Select "Standard project" in the New project 1 dialog box.

   A "Standard project" simplifies the structure of a project in the »PLC Designer«; for instance, a device tree structure with a target system, PLC logic, etc. is provided.

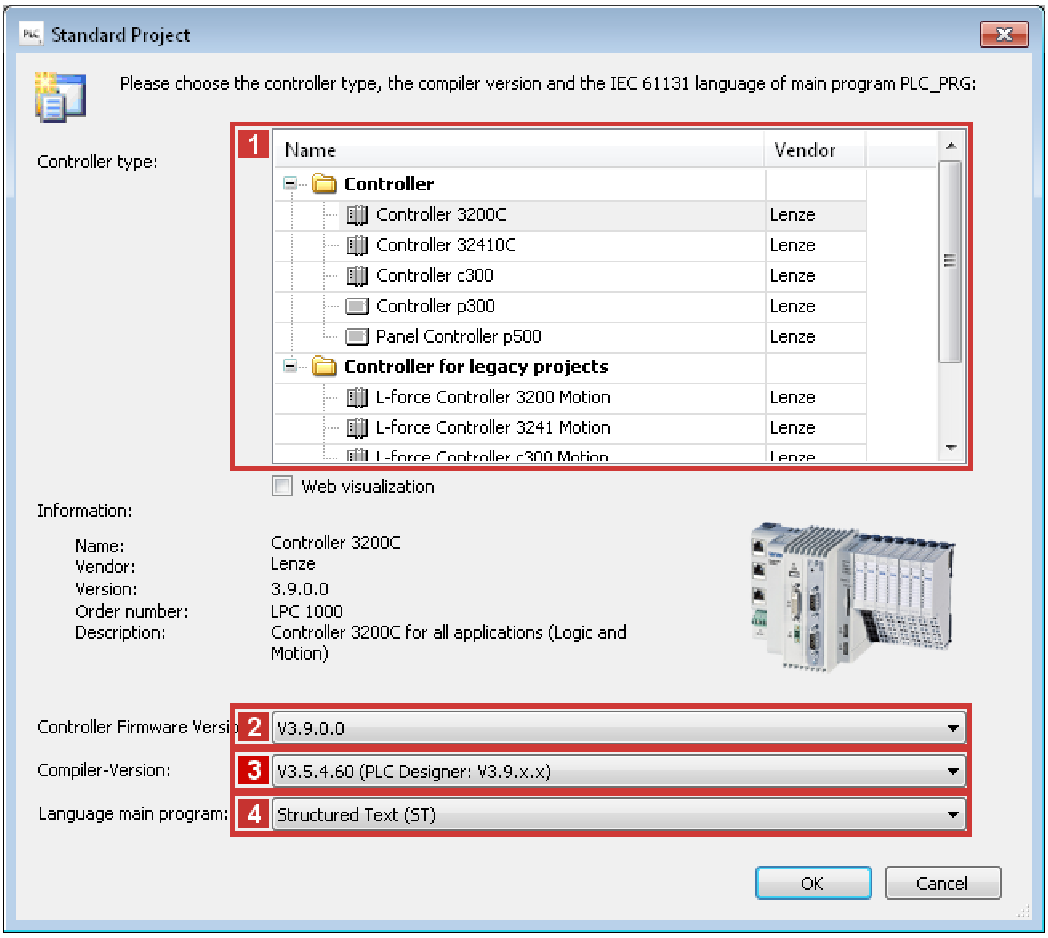


   - Enter a name for your project in the 2 **Name** »PLC Designer«input field.
   - Select the previously created project folder as storage location in the 3 **Location** selection field.

   ▶ Create a project folder (📖 44)

3. Confirm the entries with **OK**.

4. Go to the "Standard Project" dialog box and select the target system in the 1 **Controller type** selection field:



Further optional project settings

2 Selection of the Controller firmware version

3 Selection of the compiler version

4 Selection of the programming language:
- Sequential function chart (AS)
- Instruction list (AWL)
- Continuous Function Chart (CFC)
- Function block diagram (FUP)
- Ladder diagram (KOP)
- Structured text (ST)

5. Confirm the selection by clicking **OK**.

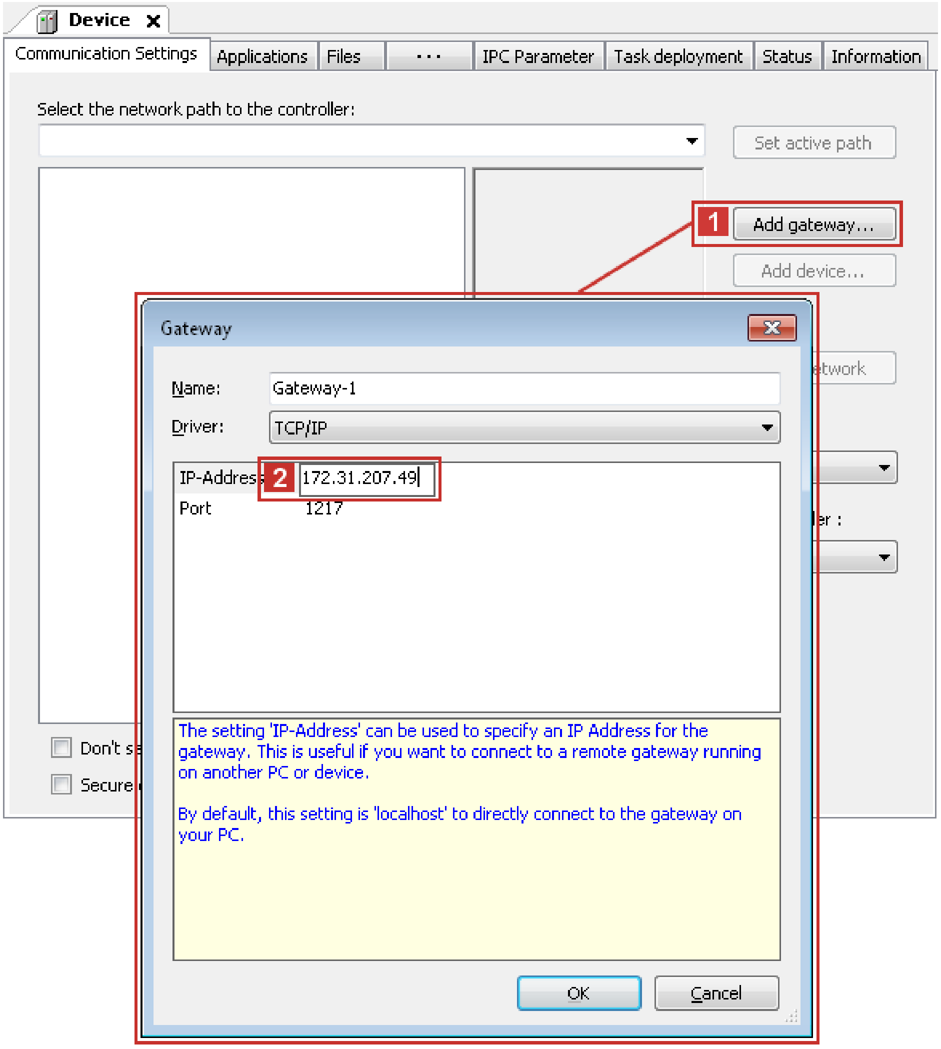
7.3.7 Configuring the communication parameters

Set the communication parameters in order to be able to carry out a fieldbus scan at a later time or in order to be able to establish an online connection to the Lenze Controller .

How to configure the communication parameters

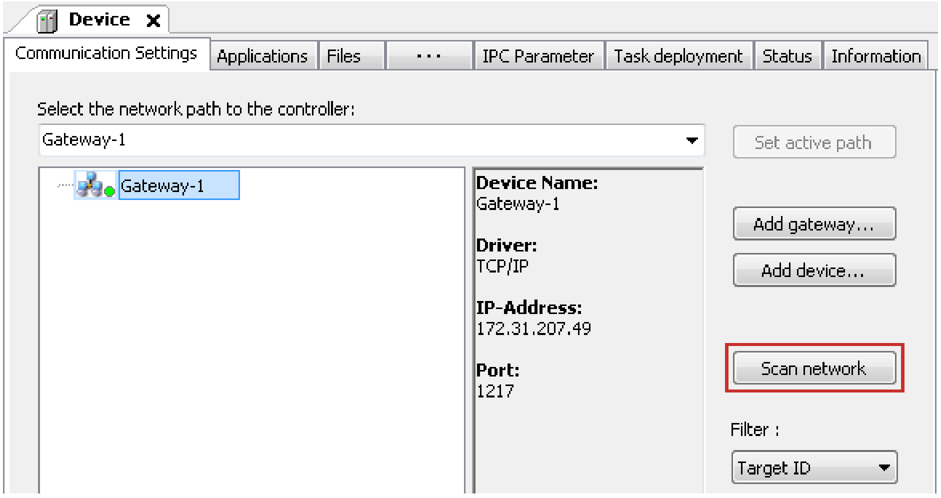
1. Go to the **Communication settings** tab of the target system (device) and click the 1 **Add gateway** button.

   Then go to the Gateway dialog box and enter the 2 **IP address** of the controller. (By double-clicking the predefined value it can be overwritten.)

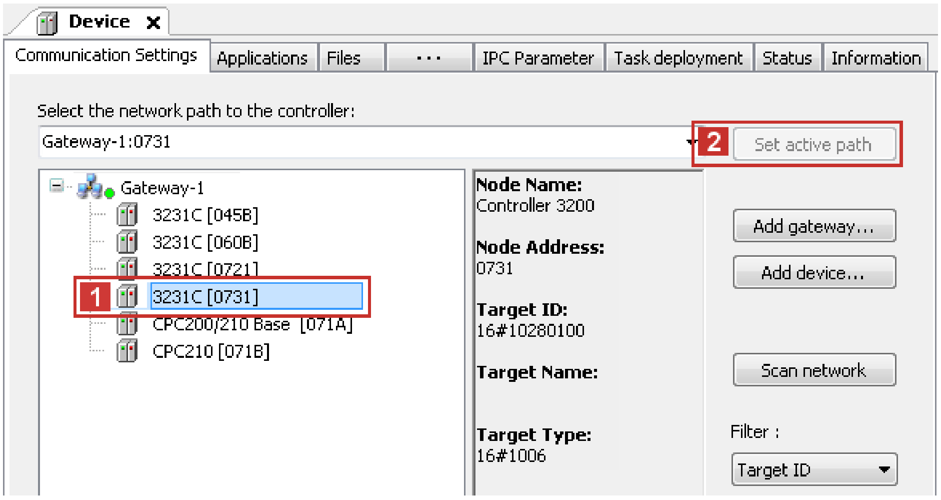


2. Confirm the entry by clicking **OK**.

3. Click the **Scan network** button.



4. Select the suitable 1 controller for the IP address entered under 2. and activate it by means of the 2 **Set active path** button (or by double-click).



5. Now you can carry out the following actions using the »PLC Designer«:
   - Determining the physical EtherCAT configuration (fieldbus scan) (📖 76)
   - Logging in on the controller with the »PLC Designer« (📖 104)

7.3.8 Determining the physical EtherCAT configuration (fieldbus scan)

In order to check the physical EtherCAT configuration, you can use »PLC Designer« to carry out a fieldbus scan on the Lenze Controller online.

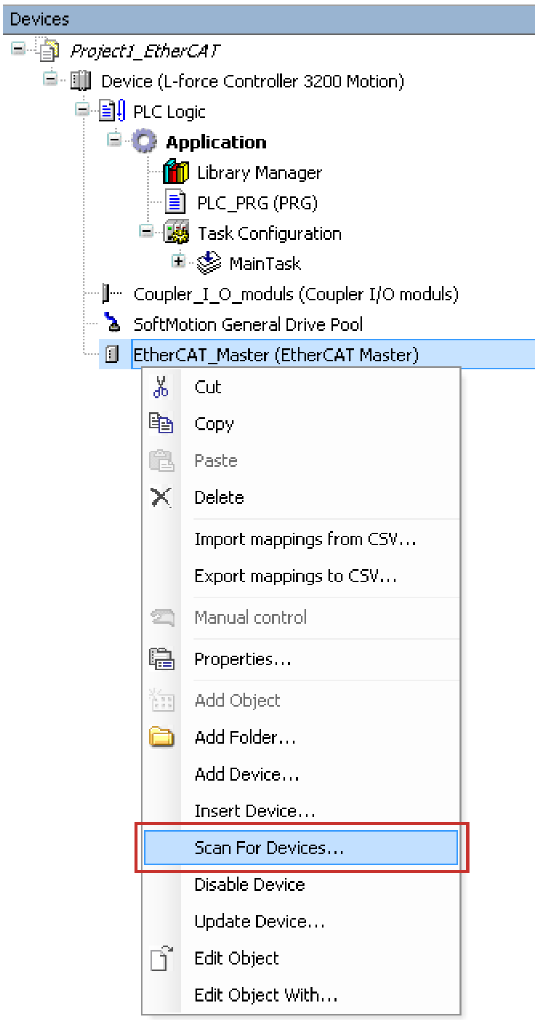
Conditions

In order to execute a fieldbus scan, you must first configure the ...

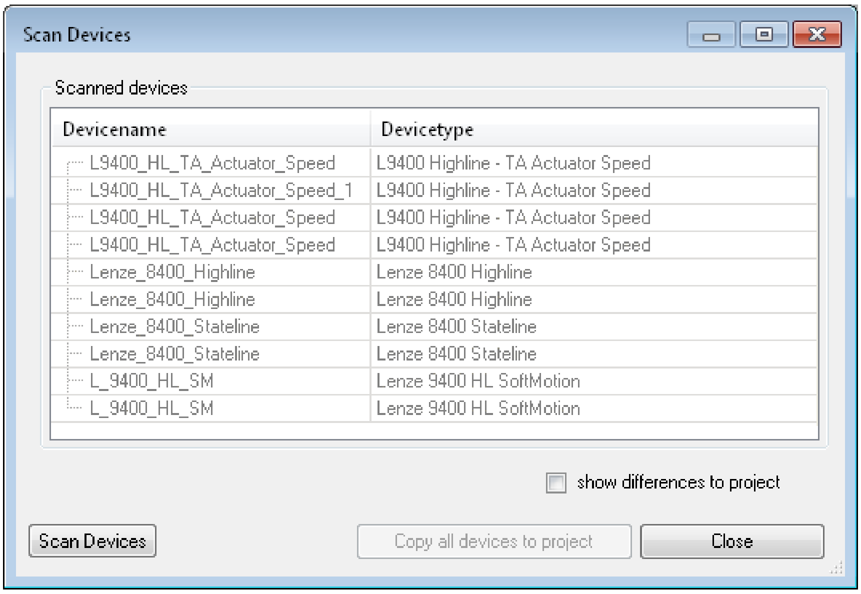
- Configuring the communication parameters (📖 74) and ...
- then use the menu command **Online → Login** or **<Alt>+<F8>** to log in on the Lenze Controller.
  - For this, the PLC program must be error-free.
  - By means of the log-in, the PLC program is loaded into the controller. In doing so, any existing program is overwritten.

How to carry out a fieldbus scan with »PLC Designer«

1. Execute the **Scan For Devices** command in the context menu of the EtherCAT master.



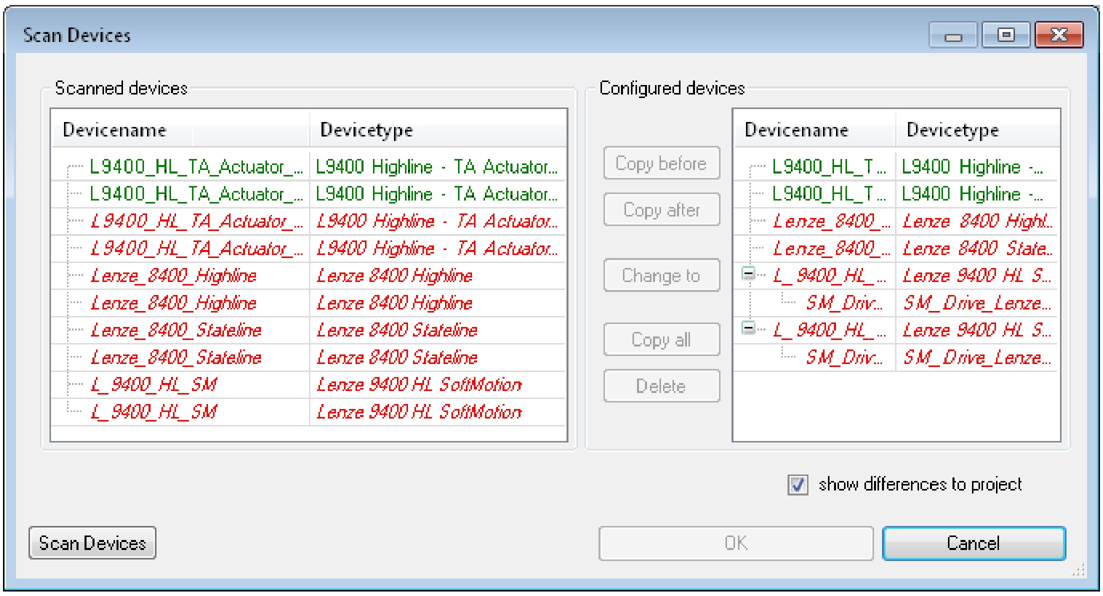
2. »PLC Designer« determines the EtherCAT nodes available on the fieldbus. In the "Scan Devices" dialog box, the devices are listed according to their physical sequence on the fieldbus.



Now you can ...

- Click the **Copy all devices to project** button to copy all available devices into your »PLC Designer« project.
- Go to the "Device name" column and select individual devices, then use the **Copy to project** button to copy them into your »PLC Designer« project.
  (The text of the **Copy all devices to project** button changes into **"Copy to project"** if one or more devices have been selected.)

By setting the checkbox "Show differences to project", the dialog is extended. Here, the physical bus structure and the configured bus structure can be adjusted.



Note!

Set up the sequence of devices in the »PLC Designer« project so that it is identical to the physical sequence in the network. Otherwise, a "bus mismatch" occurs during downloading.

In the case of field devices shown in **green**, their position in the physical network matches the position within the »PLC Designer« configuration. If field devices are shown in **red**, this is not the case.

You now have two ways of adapting the »PLC Designer« configuration:

- If you click the **Copy all** button and then confirm by clicking the **OK** button, all devices will be incorporated into the »PLC Designer« configuration.
- You can also incorporate individual devices into the »PLC Designer« configuration or replace ones already there:
  1. Select field device under "Searched devices".
  2. Select a device under "Configured devices".
  3. Click one of the buttons that are now active: **Copy (before)**, **Copy (after)** or **Replace with**.

Note!

The change in the device configuration only comes into effect after renewed translation of the »PLC Designer«-project:

1. Log out: menu command **Online → Logout** or **<Ctrl>+<F8>**

2. Translate: menu command **Build→ Build** or **<F11>**

3. Log in: menu command **Online → Login** or **<Alt>+<F8>**

Only then will all EtherCAT slaves be initialised.

Missing device descriptions

If a device available at the fieldbus is not present, an error message within the "Scan Devices" dialog box will inform you about it:

- The device cannot be interpolated into the project as the corresponding device description has not been installed.
- In order to install the device in the »PLC Designer«, the suitable device description file is required. The device identification (Vendor ID, Product Code, Revision) can be helpful to identify the device (see also Lenze EtherCAT product codes (📖 27)).

▸ Importing missing devices / device description files (📖 79)

7.3.9 Importing missing devices / device description files

The device description file contains the data of the fieldbus peripherals required for the master control. This file is required to program the control system.

With the »PLC Designer«, device descriptions for the following Lenze device series are installed as well:

- i700 servo inverter
- Servo Drives 9400
- Inverter Drives 8400
- I/O system 1000 (EPM-Sxxx)
- Fieldbus communication cards for Lenze Controllers (EtherCAT, CANopen, PROFIBUS, PROFINET)

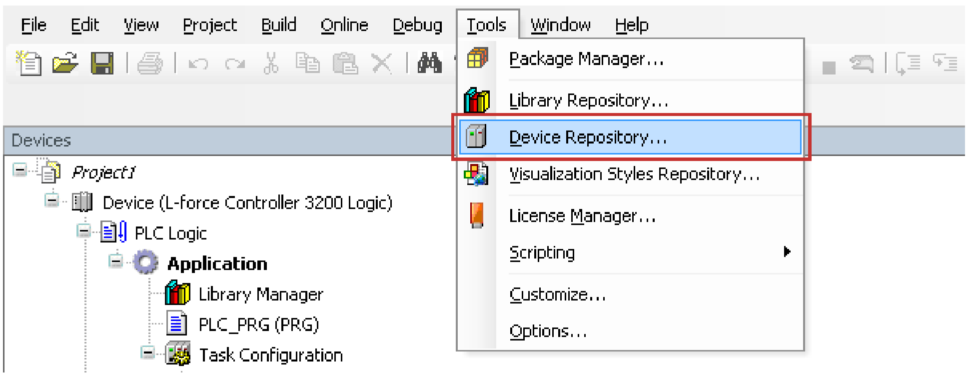
Note!

We recommend that the additionally installed EtherCAT device descriptions be retained and not replaced with the XML device descriptions for the download section at www.Lenze.com.

The installed device descriptions contain additional information on how to improve usability (pictograms etc.); this information is not contained in the XML files.

In order to furthermore integrate missing devices or devices of other manufacturers, the corresponding device description files of the manufacturer are required.

In the »PLC Designer« you can import device description files of the *.XML, *.devdesc.XML, *.EDS, *.DCF, and *.GSx type via the menu command **Tools → Device Repository...**.



For EtherCAT slaves, select the file type "EtherCAT XML Device Description Configuration".

7.3.10 Creating a control configuration (adding field devices)

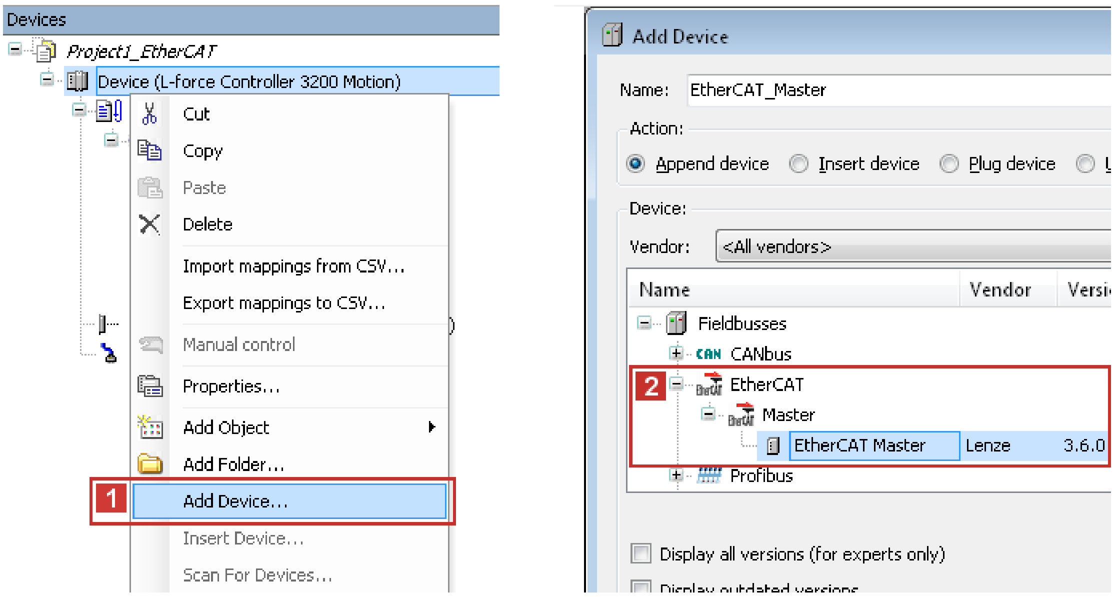
Note!

Before creating an EtherCAT configuration in »PLC Designer«, ensure that the following conditions have been met:

- The sequence of EtherCAT slaves in the device tree must correspond to the physical arrangement of the EtherCAT topology.
- SoftMotion operation is only possible with EtherCAT slaves that use the CiA402 application (SM_Drives, e.g. Servo Drives 9400 Highline CiA402).
- Select the cycle times according to the technical data, from 1 ... 10 ms.

How to create a control configuration in »PLC Designer«

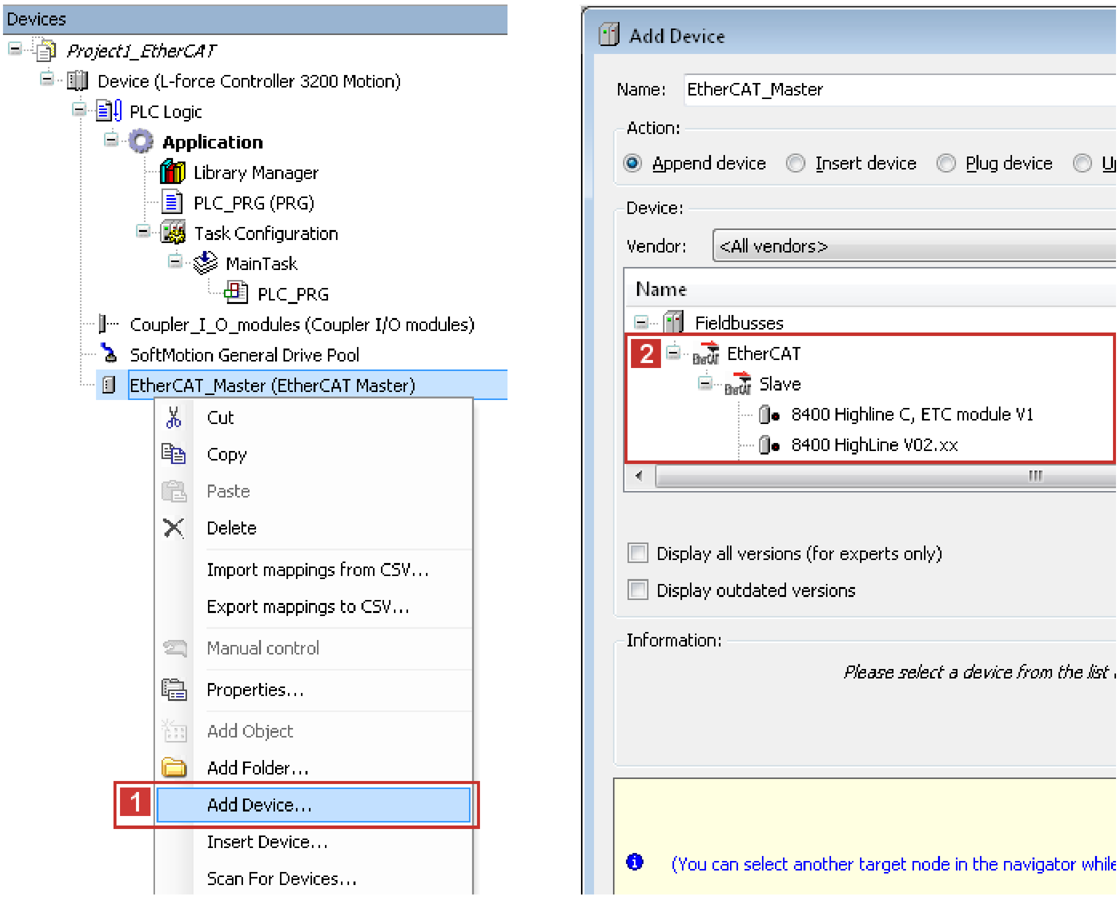
1. Open the context menu of the target system and execute the command 1 **Add device** in order to extend the control configuration with 2 "EtherCAT Master".



2. Append the EtherCAT slaves under the EtherCAT master.

   You have two options:

   - **Automatically** Determining the physical EtherCAT configuration (fieldbus scan) (📖 76) (before: Configuring the communication parameters (📖 74).)
   - **Manually** with the command 1 **Add device** in the context menu of the EtherCAT master



Select a field device from the 2 selection list. Only those devices can be selected whose EtherCAT device descriptions have been imported in »PLC Designer«.

▸ Importing missing devices / device description files (📖 79)

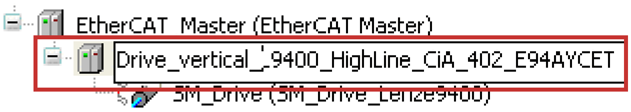
Repeat the command 1 **Add device** until all participating slaves on the field bus have been incorporated in the EtherCAT configuration.

3. Give the inserted slaves suitable names (e.g. "Drive_vertical").

   The names must ...
   - only contain the characters "A ... Z", "a ... z", "0 ... 9" or "_";
   - must not begin with a digit.

   With a mouse-click on the element or by pressing the space key, the name is enabled for entry.

   Example :



Tip!

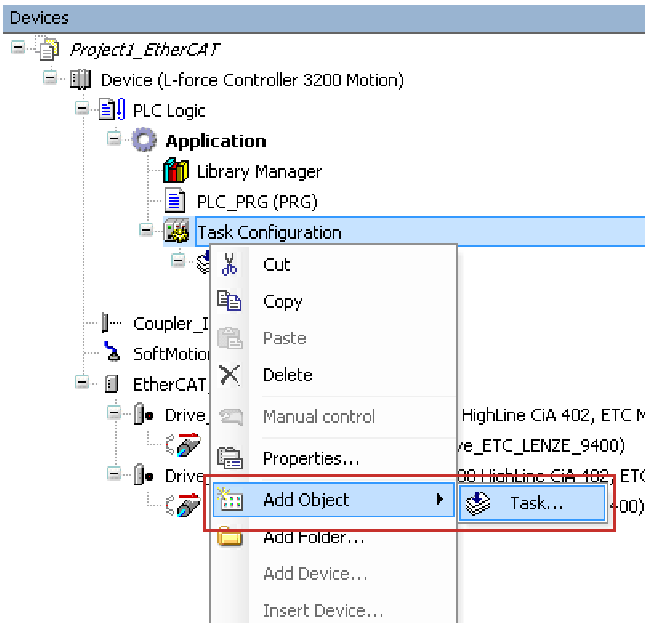
We recommend that 9400 servo drives and 8400 inverter drives be given the same designation that has been entered under the device code **C00199**.

7.3.11 Creating a task

How to create a task in »PLC Designer«

1. Open the context menu of the **Task configuration** and execute the command **Add object** in order to create a new task.

   Assign a reasonable task name, e.g. "MotionTask".

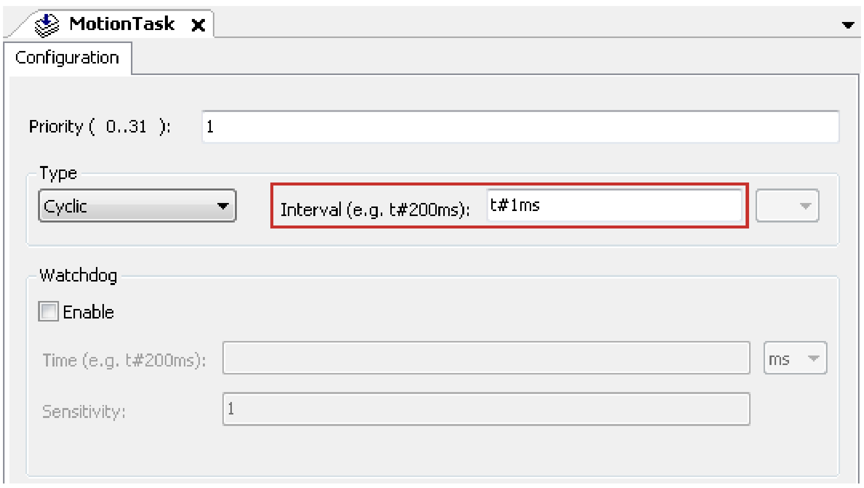


2. Enter a reasonable cycle time in milliseconds for the created task in the **Interval** input field.

Note!

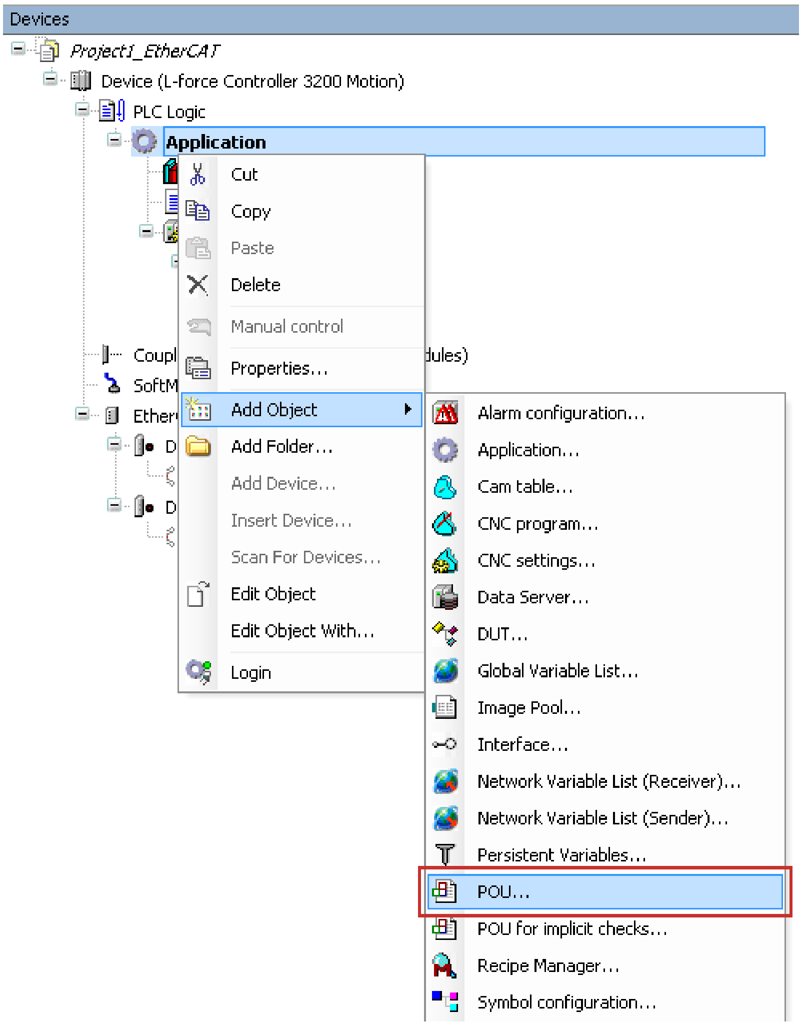
- Select a cycle time, according to the technical data, from 1 ... 10 ms.
- If "distributed clocks" (DC) are used, the task cycle time to be set must comply with the set DC cycle time.

▶ Setting a DC synchronisation (📖 88)



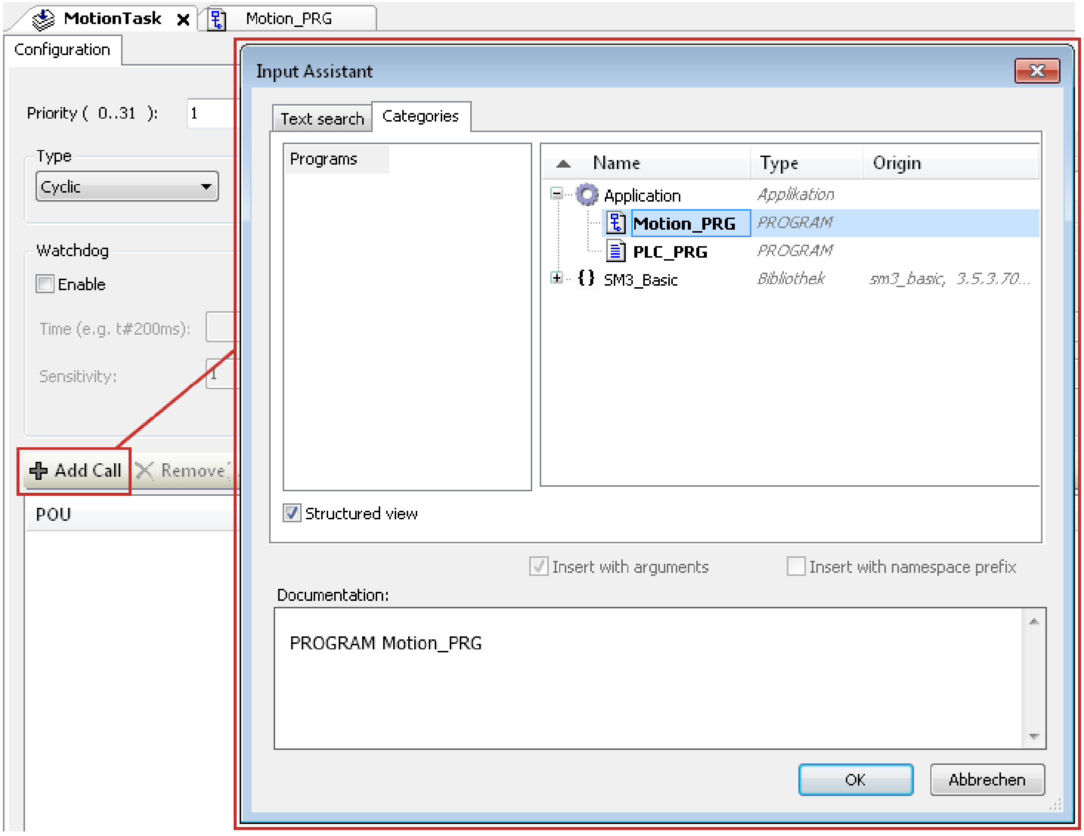
3. Open the context menu for **Application** and execute the command **Add object → POU...** in order to create a new program block (POU) in the application.

   Assign a reasonable POU name (e.g. "Motion_PRG").

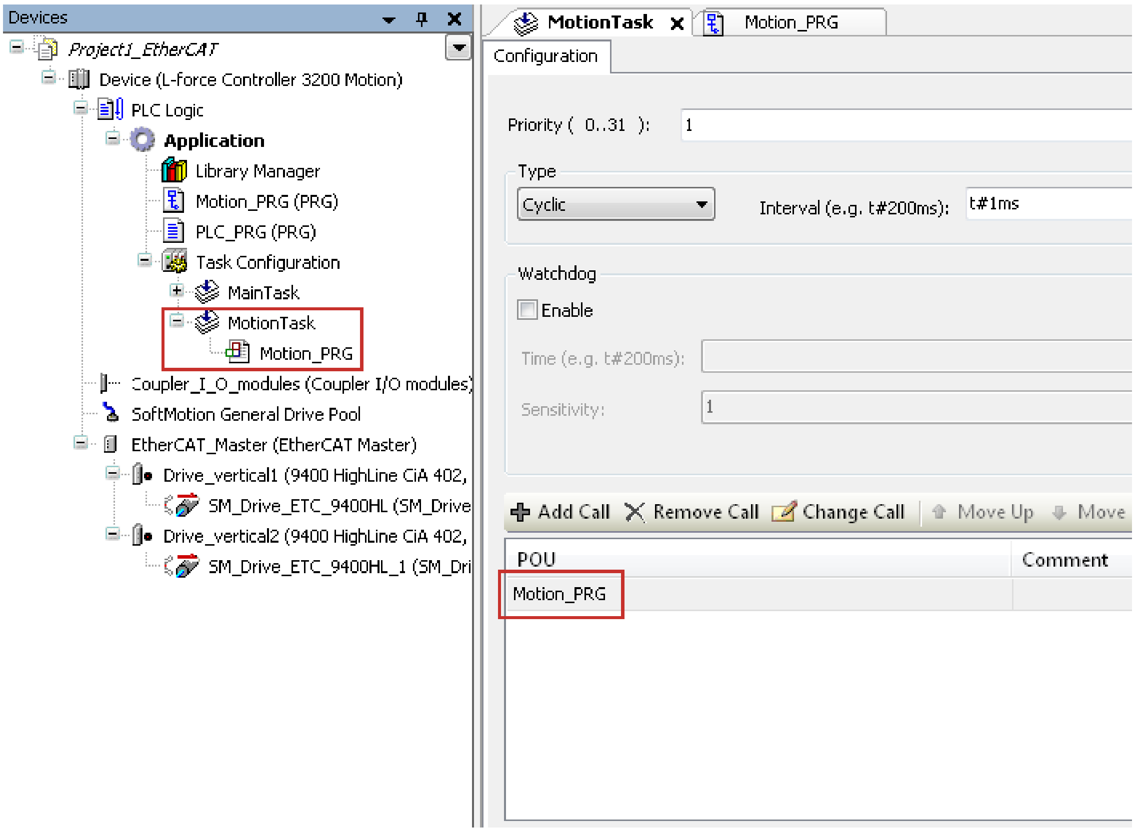


4. Click the **Add call** button to open the input assistant.

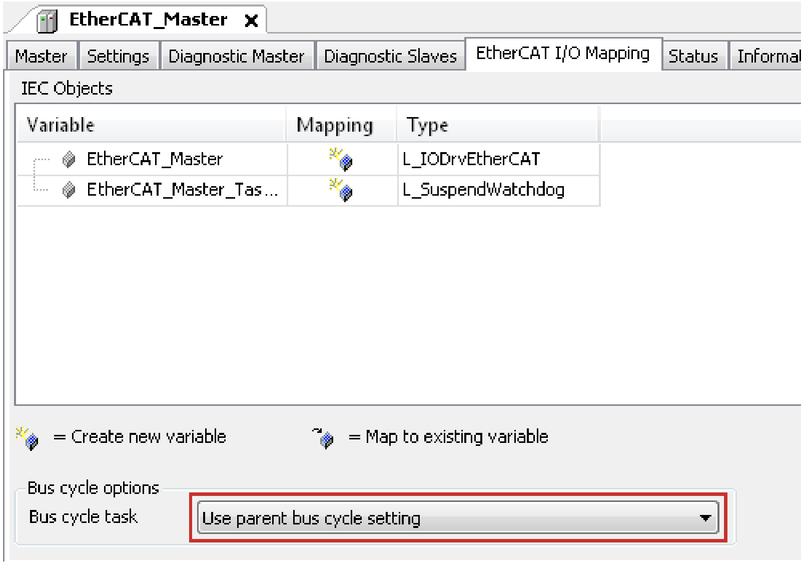
   Select the program call under "Application" and add it to the task by clicking the **OK** button.



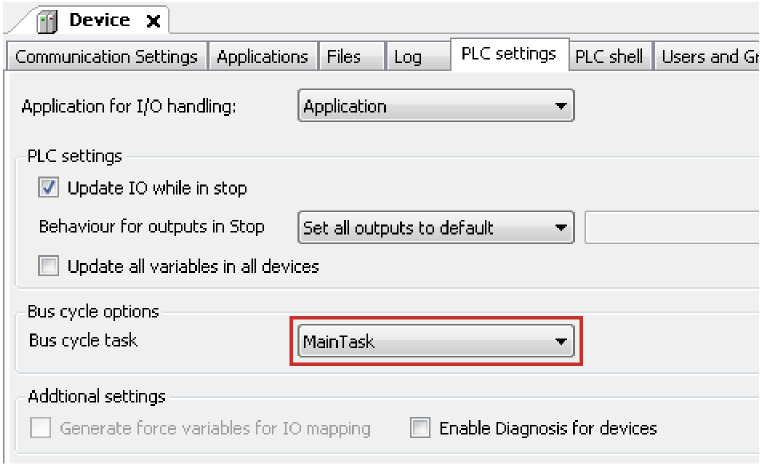
The following task configuration is caused:



5. Open the **EtherCAT I/O image** tab of the EtherCAT master and select the bus cycle task for the master (Lenze Controller).



The "Cycle settings of the higher-level bus" serve to use the bus cycle task set via the **PLC settings** tab of the Lenze Controller (device):



7.3.12 Setting a DC synchronisation

Note!

The manual configuration of the slave DC features requires detailed knowledge regarding EtherCAT and the field device. Thus, DC settings should only be made by experts.

We recommend that the basic DC settings be retained in the case of Lenze field devices in order to ensure correct DC synchronisation.

- DC synchronisation is absolutely required for Motion applications.
- DC synchronisation can also be used for Logic applications.
- Not all slaves support the DC functionality.
- In order to be able to use the DC functionality, the first slave connected to the EtherCAT master (Lenze Controller) must have **DC master capability**.
  When additional slaves are added, devices with and without DC capability can be mixed.
- The first EtherCAT slave after the Lenze Controller must be the **DC master** that supplies the other EtherCAT nodes (incl. controller) with the exact time.

Adjusting the task cycle time and DC cycle time

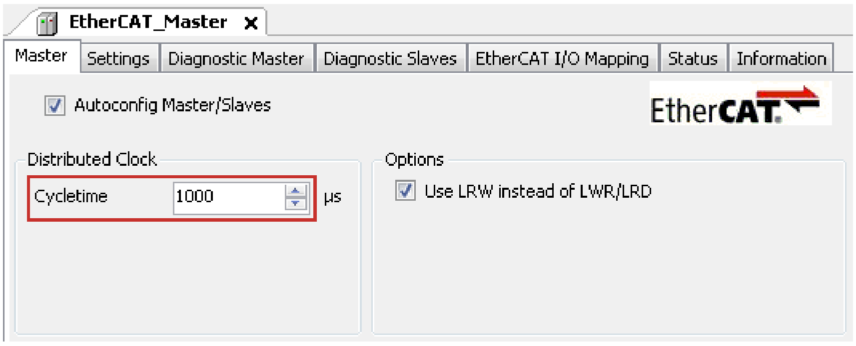
The Lenze Controller is the EtherCAT master. The clock pulse of the bus system is determined by the cycle time of the task that is assigned to the drives (slaves) integrated in the »PLC Designer«.

The task settings in the »PLC Designer« only support integer millisecond cycles and the smallest possible bus cycle is 1 millisecond. This cycle time can be defined via the »PLC Designer« in the **Task Configuration**.

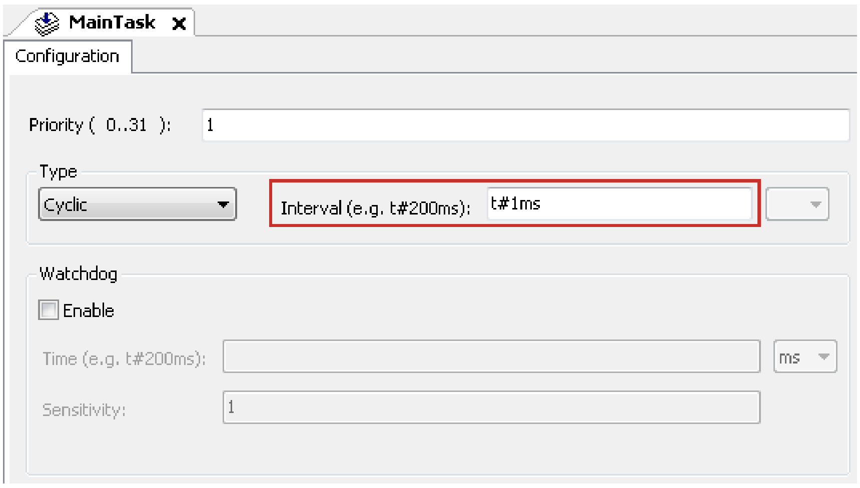
Note!

- The DC cycle time to be set must match the set task cycle time.
- Select the cycle times according to the technical data, from 1 ... 10 ms.

The DC cycle time on the tab of the EtherCAT **master**:

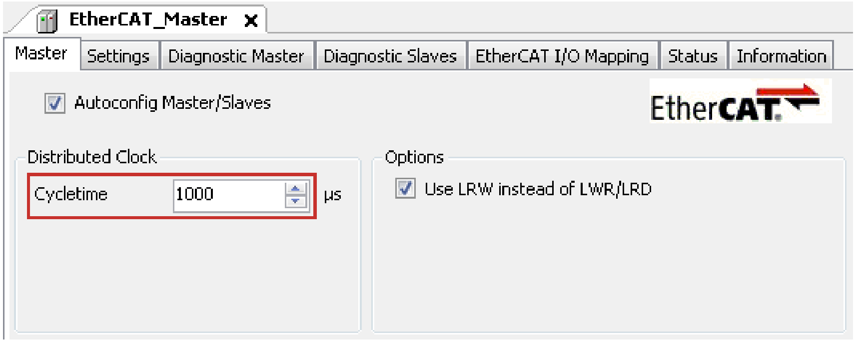


The task cycle time on the **Configuration** tab of the "MainTask":



How to set DC synchronisation:

1. Set the DC cycle time at the master (Lenze Controller) in the **Master** tab of the EtherCAT master.

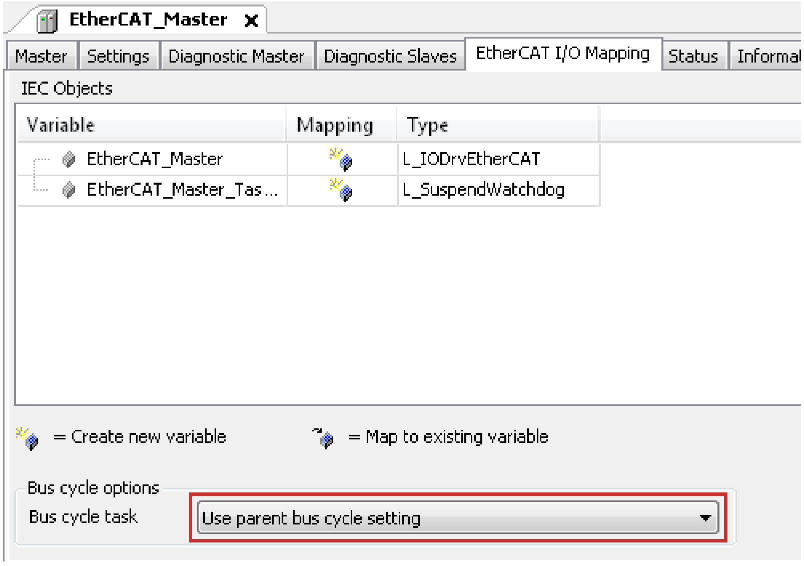


- Select the cycle times according to the technical data, from 1 ... 10 ms.
- The (basic) cycle time set here is valid for all Logic and Motion nodes synchronised by distributed clocks.
- For the Lenze field devices listed in the following table, the indicated settings must be made by means of the **»Engineer«**. The Lenze Controller does not write the values into the slave field devices.

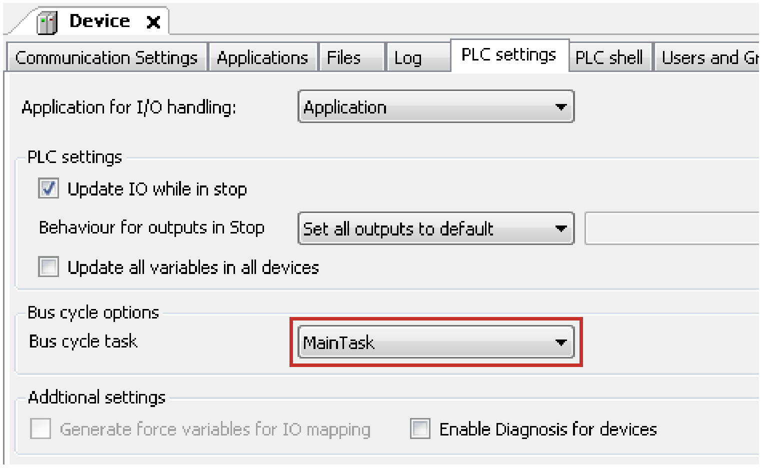
Field devices	Settings in »Engineer«
Inverter Drives 8400	• C01120 = 4 (sync source: EtherCAT module in MCI)
Servo Drives 9400	• C01120 = 4 or 5 (sync source: EtherCAT module in MXI1 or MXI2) • C013892/C14892 = 1 (process data mode = "deterministic mode") **Note!** For **Servo Drives 9400 CiA402**, C013892/C14892 (process data mode) is defined via the »PLC Designer«. See Start parameters of the Servo Drives 9400 HighLine CiA 402 (📖 104).

- In the case of the **Servo-Inverter i700**, all the parameters needed for operation are specified via the Lenze Controller (settings in the »PLC Designer«).
  ▶ Commissioning the i700 servo inverter (📖 45)
- If the DC setting and the selection of the sync source are inconsistent with each other (e.g. Servo Drives 9400: C01120 = MXI1 and "DC unused"), the devices cannot be set to the "Operational" state.
- The settings of the parameters sync cycle time (C01121), sync phase position (C01122), sync tolerance (C01123) and sync PLL increment (C01124) common for the Lenze system bus (CAN) cannot be made for EtherCAT. These values are automatically calculated by the EtherCAT communication module and set internally in the inverter.

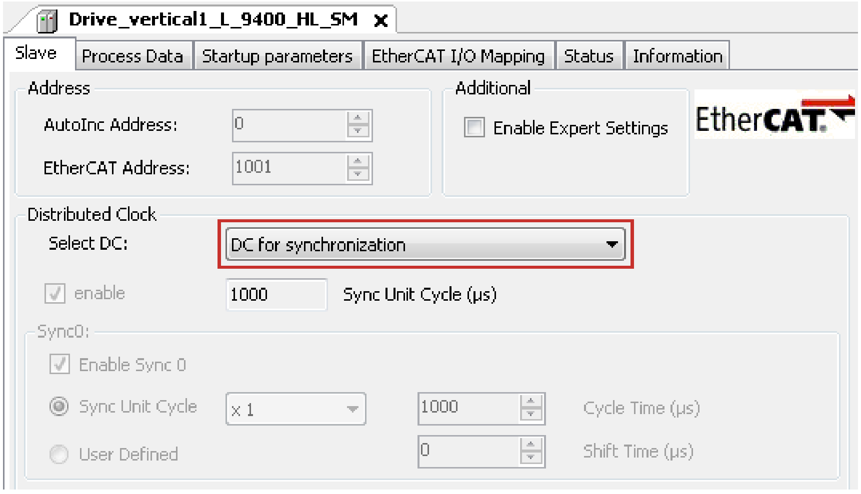
2. Open the **EtherCAT I/O image** tab and select the bus cycle task for the master (in so far as this has not yet happened in the task configuration).



The "Cycle settings of the higher-level bus" serve to use the bus cycle task set via the **PLC settings** tab of the Lenze Controller (device):



3. Select the DC functionality "DC for synchronization" in the device tree for the **first slave** (DC master) under the master (Lenze Controller).
   - The first slave connected to the master must have DC capability.
   - If a slave does not support any distributed clocks, only "DC unused" can be selected here.



4. Also select the DC functionality "DC for synchronization" for all other slave devices which are to use the DC synchronisation.

7.3.13 Setting SoftMotion parameters

Note!

In »PLC Designer«, the SoftMotion tabs are only available in the case of field devices that use a Motion application.
- i700 servo inverter
- Servo Drive 9400 Highline CiA402

The SoftMotion parameters must be set application-dependent.

In the case of the **Servo-Inverter i700**, all the parameters needed for operation are stipulated by means of the Lenze controller (settings via the »PLC Designer«).

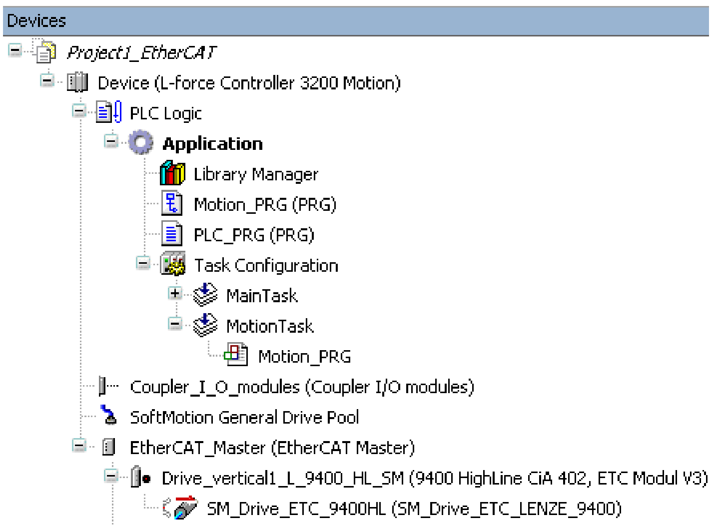
▸ Commissioning the i700 servo inverter (📖 45)

In the case of the **Servo Drive 9400 Highline CiA402** in contrast, the following parameters must be set manually via »Engineer«:

- Homing mode (C02640, set in application-dependent manner)
- Touch-probe interface (set in application-dependent manner)
- Control of the holding brake (0x60FB/2 | Brake control)
  Depending on the setting of this parameter, the holding brake is applied for a short time after the conclusion of the home position path. In order to avoid this, set bit 2 in this parameter ("disable stop": does not apply the brake at standstill).

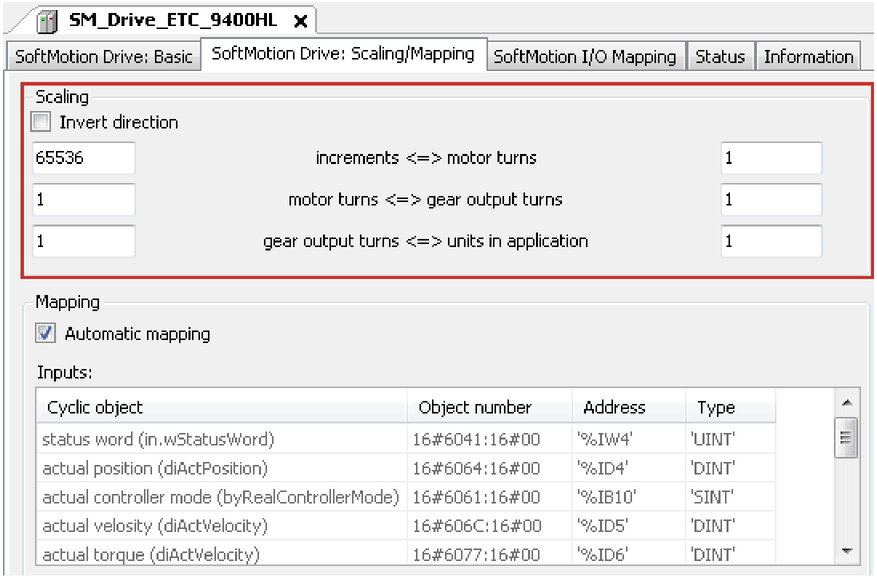
These parameters are not set via the Lenze Controller.

Example of a minimum configuration with a Motion device (Servo Drive 9400 HighLine CiA 402)



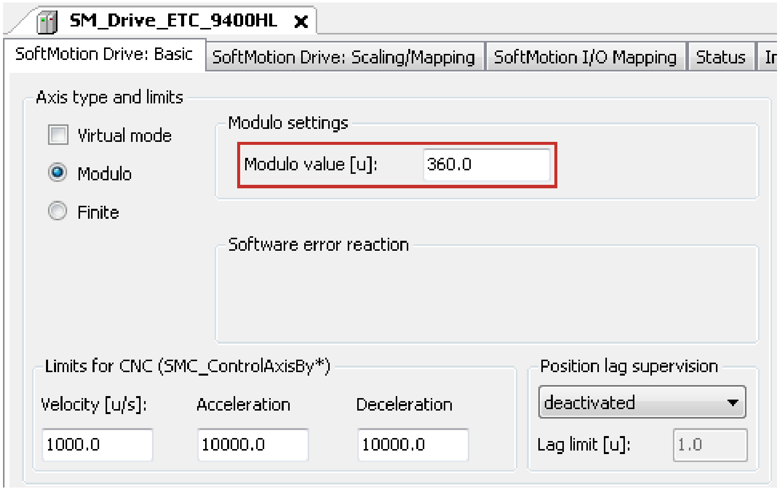
How to set the SoftMotion parameters

1. Open the tab **SoftMotion drive: Scaling/Mapping** and adapt the conversion factors in the "Scaling" area.

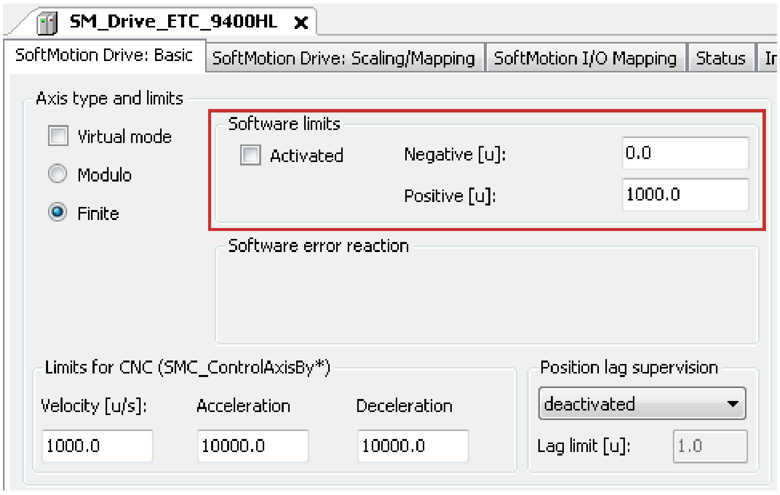


2. Open the tab **SoftMotion drive: basic parameters** and set the axis types and limitations.
   - Do not use the "virtual mode" setting.
   - Virtual axes are located in the "SoftMotion General Drive Pool".

   Configuration of a Motion device **Rotary axis**
   (type: Modulo, 360°/revolution, ratio 1:1):



   For configuring a **linear axis** Motion device (type: Finite), you can activate and determine the software limit switches:



3. Repeat steps 1 and 2 for all Motion devices connected to the field bus.

Online help of »PLC Designer«

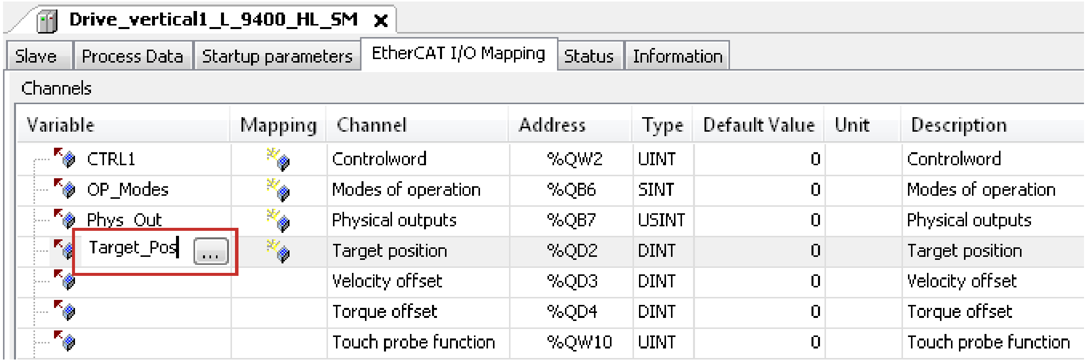
Here, you can find detailed descriptions of the **SoftMotion tabs**.

7.3.14 Processing EtherCAT I/O mapping

Note!

- If you insert more field devices in the control configuration or change the PDO mapping, the object addresses change as well (%Qxx, %Ixx). Hence, the input and output objects must be accessed via individual, **unambiguous** variables. The variable names must comply with the IEC 61131 syntax (no space and leading digits in the variable name).
- The manual assignment of object addresses in the "Address" column is not supported. Hence, only use the automatically assigned addresses of the process image. A manual assignment causes malfunctions.

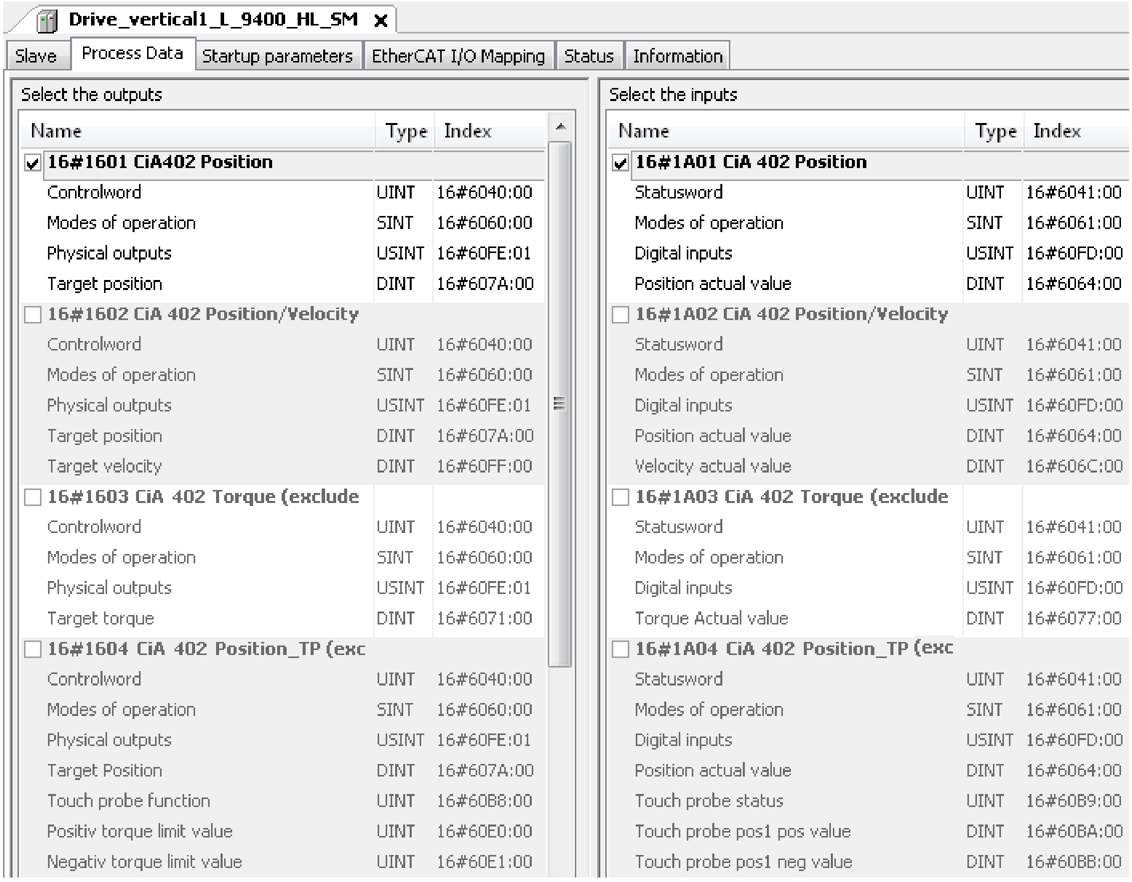
On the **EtherCAT I/O image** tab, you can enter variable names by double-clicking the variable fields or pressing the space key:



By clicking the ... button, you can reference already existing variables (e.g. global variables from function libraries) or you can enter variable names directly in the input field and thus create system variables. The corresponding system variables for the PLC program are available.

7.3.14.1 Entering the settings for PDO mapping

You set PDO mapping by means of the **Process data** tab:



PDO mapping for Lenze inverters can be composed of three parts:

- The static part is permanent (cannot be changed) and cannot be deactivated either.
- The dynamic part contains PDOs that have been preconfigured for the different CiA402 operating modes. Depending on the CiA402 operating modes, one of these PDOs can be activated in »PLC Designer«.
- The freely configurable part can be activated in »PLC Designer« if necessary and enables individual mapping.
  - Configuring individual PDO mapping (98)

Some inverters only allow for one PDO mapping per direction.

The input and output PDO mappings selected must be identical (see illustration).

In order to change the setting, you must first deselect the current setting (remove checkmark in the checkbox). After that you can freely select the desired setting (set checkmark).

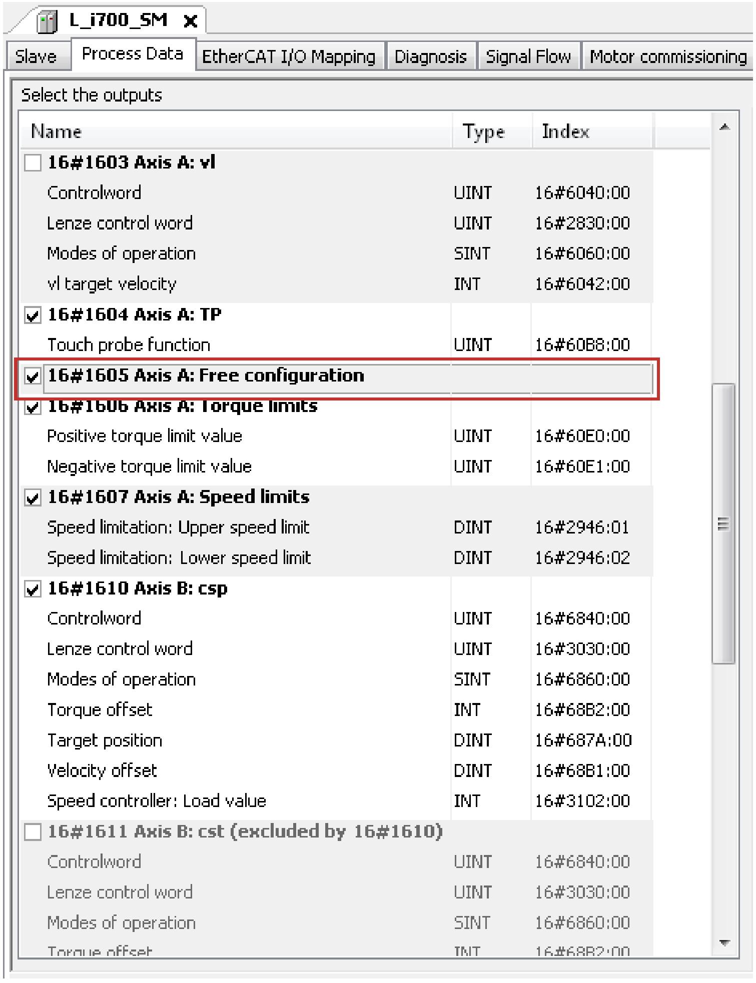
7.3.14.2 Configuring individual PDO mapping

In addition to the static and preconfigured PDO mappings, you can also configure an individual PDO mapping.

How to configure an individual PDO mapping in »PLC Designer«

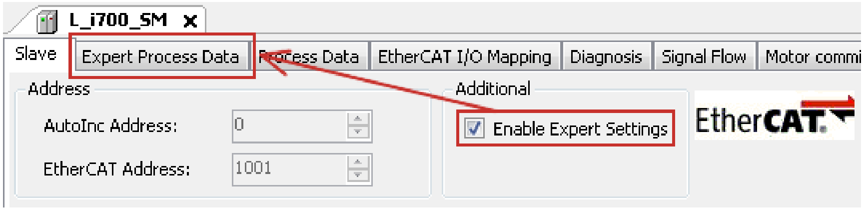
(example of an individual output PDO for the i700 servo inverter)

1. Open the **Process data** tab and set a checkmark in the checkbox of the free PDO mapping (16#1605).

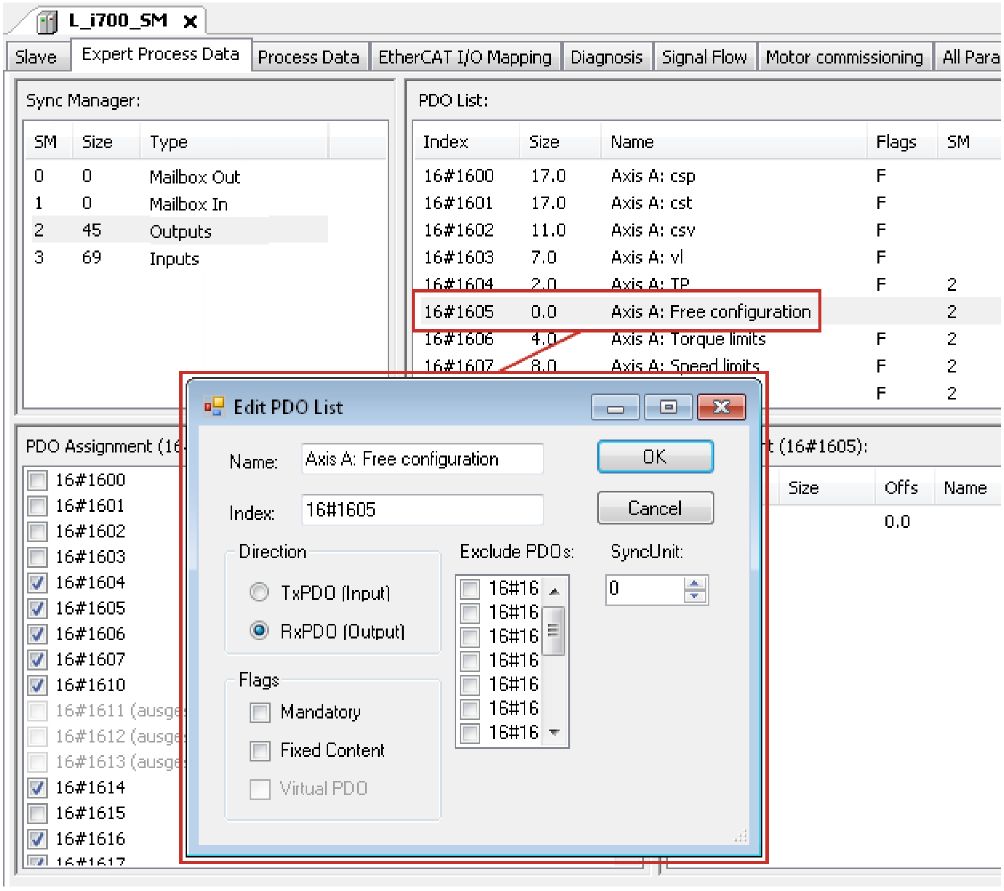


2. Activate the expert settings on the **Slave** tab of the slave drive.

   The **Expert Process Data** tab appears next to the **Slave** tab:

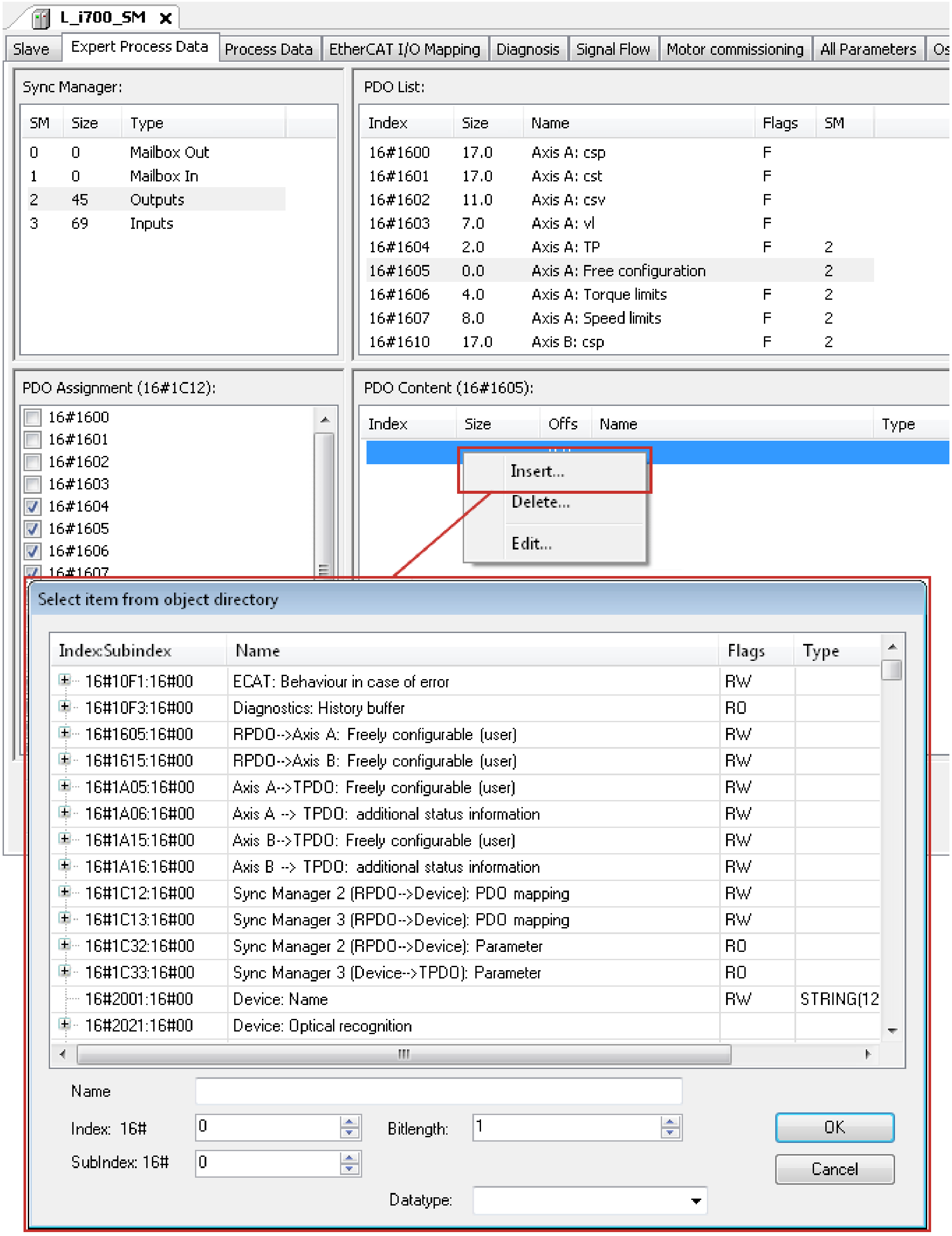


3. Open the **Expert Process Date** tab.
4. Double-click the free PDO mapping (16#1605) in the **PDO list**.



5. In the dialog box that appears, process the free PDO mapping and close the dialog box by clicking the **OK** button.

6. Open the context menu by right-clicking the **PDO Content (16#1605)** and execute the **Insert** menu command.

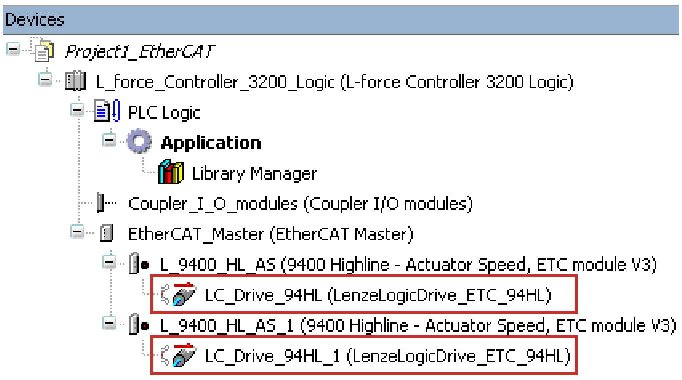


7. In the dialog box that appears, you can select a single index (parameter) or a group of indices and insert them into the **PDO Content (16#1605)** via the **OK** button.

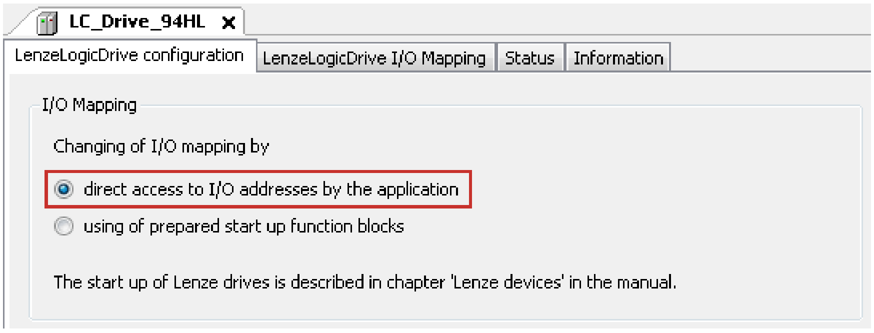
   The inserted indices (parameters) are also displayed in the free PDO mapping (16#1605) under the **process data** tab.

7.3.14.3 PDO mapping for logic devices

If the device descriptions for logic devices that are supplied with the »PLC Designer« are used, the process data are copied to the subordinate logic drive node automatically.



If the process data are still to be linked manually, activate the "direct access to I/O addresses by the application" option.



In this setting, the prepared function blocks are not usable.

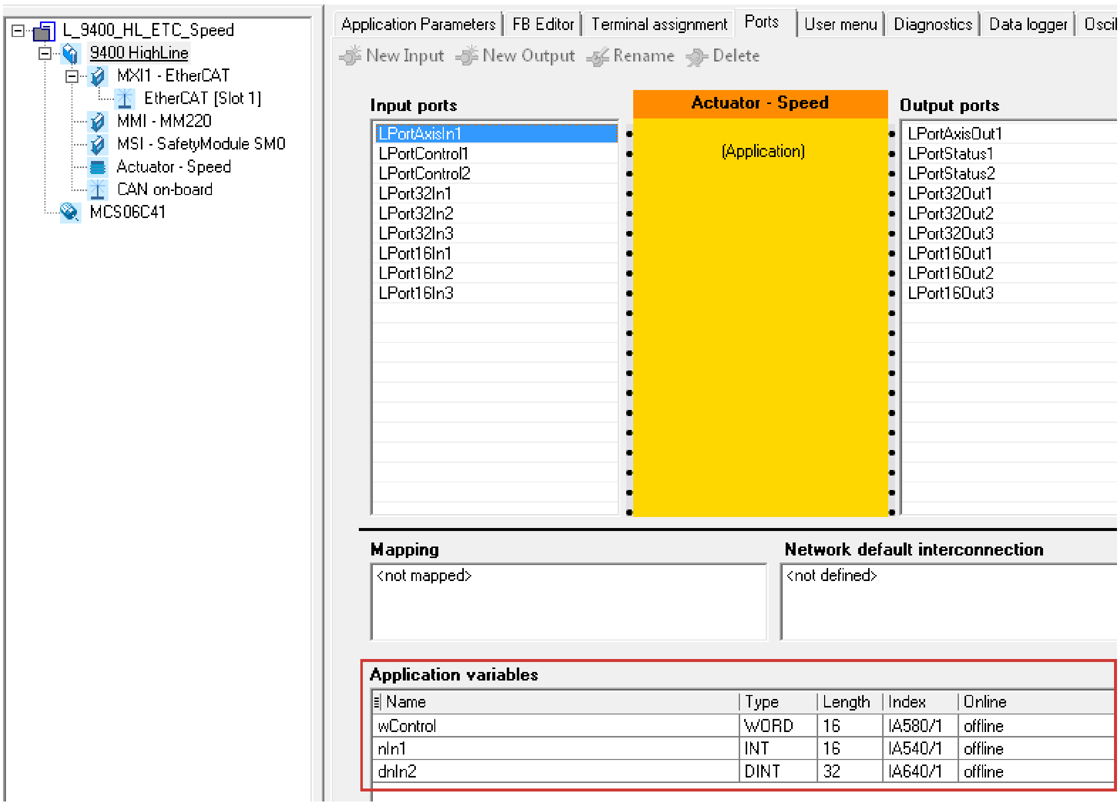
The process data (I/O addresses) must be linked manually.

7.3.14.4 Using PDO mapping settings from »Engineer«

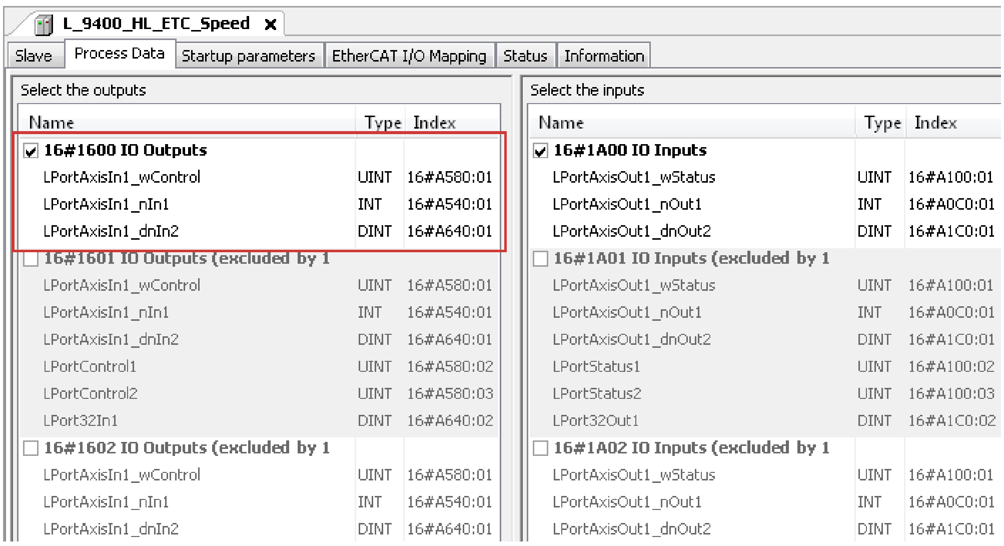
If the PDO mapping was set via the »Engineer«, the same settings must be made in the »PLC Designer« project. During the boot-up of the network, the mapping is written back to the inverter (slave). That way, it is ensured that the mapping indices in the EtherCAT master and in the slave are identical.

The illustrations below show examples of the PDO mappings of ports **LPortAxisIn1** and **LPortAxisOut1** (application: "Speed actuating drive") in the »Engineer« and in the »PLC Designer«, respectively.

PDO mapping in the »Engineer«:



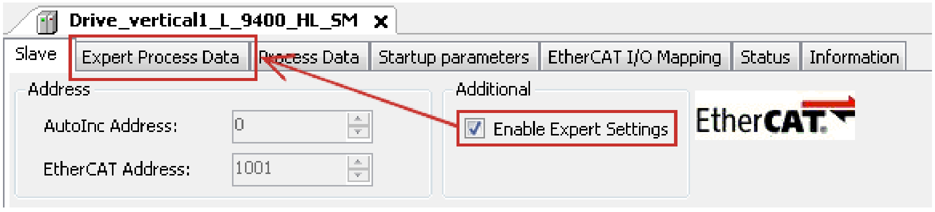
PDO mapping in the »PLC Designer«:



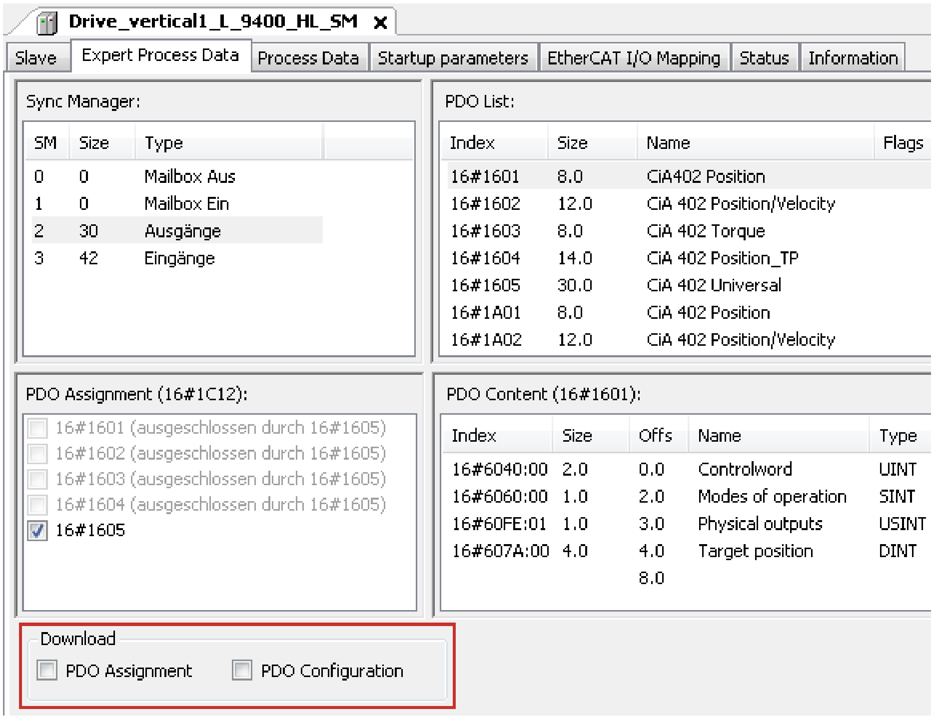
In »PLC Designer«, how to stipulate that the PDO mapping settings from »Engineer« are used for a slave drive

1. Activate the expert settings on the **Slave** tab of the slave drive.

   The **Expert Process Data** tab appears next to the **Slave** tab:



2. Remove the checkmarks in the checkboxes on the **Expert Process Data** tab in the **Download** area.



   If the checkboxes are empty, the complete PDO mapping from »Engineer« is applicable to the slave drive.

   The mapping settings carried out in »PLC Designer« are not written into the slave drive.

7.3.15 Compiling the PLC program code

In order to compile the PLC program code, select the menu command **Build → Build**, or press the function key **<F11>**.

- If errors occur during translation, they can be located and corrected on the basis of the »PLC Designer« error messages.
  Then re-translate the program code.
- If no errors occur during translation, save the »PLC Designer« project in the project folder with the menu command **File → Save project / Save project as ...**

7.3.16 Logging in on the controller with the »PLC Designer«

Use the menu command **Online → Login** or **<Alt>+<F8>** to log in on the Lenze Controller.

- For this, the PLC program must be error-free.
- With the log-in, the fieldbus configuration and the PLC program are loaded to the controller. Any configuration or a PLC program that is possibly available is overwritten.

7.3.17 Starting the PLC program

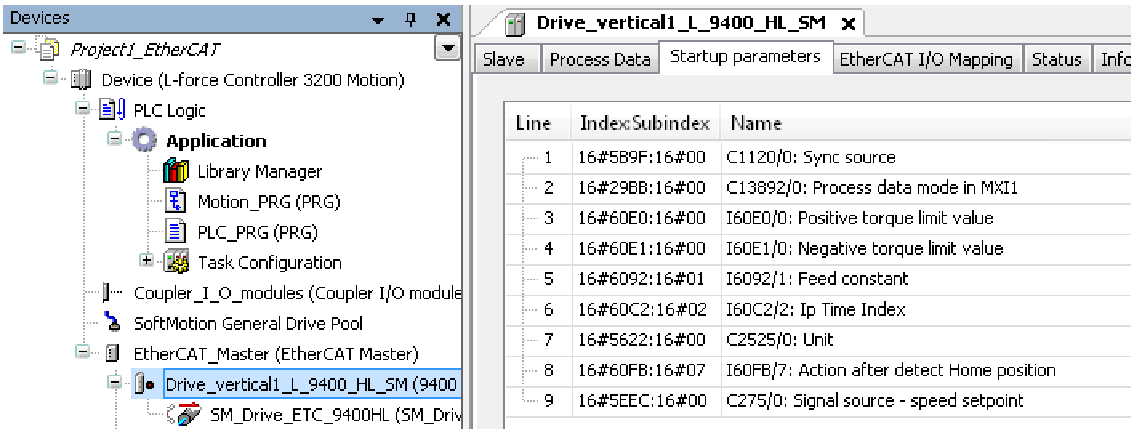
Before the start, the PLC program must be loaded to the Lenze Controller using the menu command **Online → Login**.

Use the menu command **Debug → Start** or the function key **<F5>** to start the PLC program.

7.3.18 Start parameters of the Servo Drives 9400 HighLine CiA 402

When the Lenze Controller is being powered up, some "start parameters" are automatically loaded into the servo drives 9400 HighLine CiA 402.

These parameters are shown on the **Startup parameters** tab.



7.3.19 Start parameters of the Inverter Drives 8400 motec

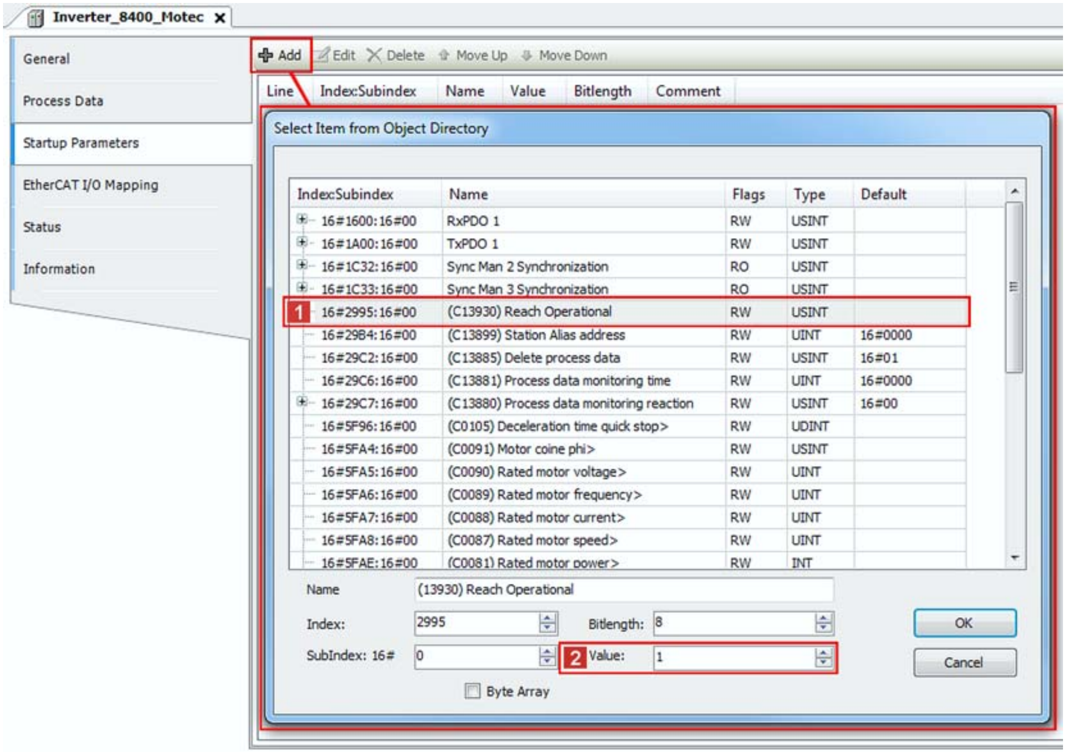
Communication manual EtherCAT – Inverter Drives 8400 motec

Please note the detailed information on the state change after "Operational" in the stay-alive operation.

Configuring the state change after "Operational" in the stay-alive operation:

To be used from Communication Unit SW version V01.02!

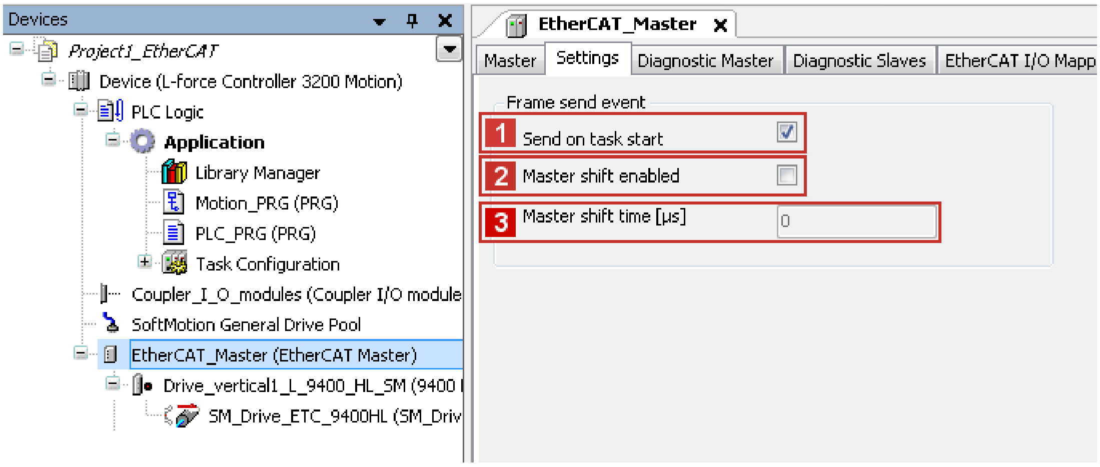
1. Add the 1 **0x2995** parameter (C13930) to the list of the start parameters of the inverter.
2. Set the 2 parameter value to **'1'**.
3. Complete the entry with **OK**.



7.3.20 Optimising the task utilisation

Optimise the task utilisation to obtain a lower jitter of the process data frames.

For this, you can enter the following settings on the **Settings** tab of the EtherCAT master:



Pos.	Setting
1	Instant of transmission for the EtherCAT bus cycle frame
	☑: The EtherCAT frame is sent at the beginning of the bus cycle task.
	☐: The EtherCAT frame is sent at the beginning of the bus cycle task.
2	Activation of "Master shift" The input field is only active if the checkbox 1 "Send at task start" is set.
	☑: "Master shift" active
	☐: No "Master shift"
3	"Master shift time" in µs Time by which the PLC system clock is placed before the SYNC0 event. Since the PLC system clock is 1 ms, only a value of 0 ... 1000 µs will be useful. The input field is only active if the checkbox 2 "Enable master shift" is set.

Note!

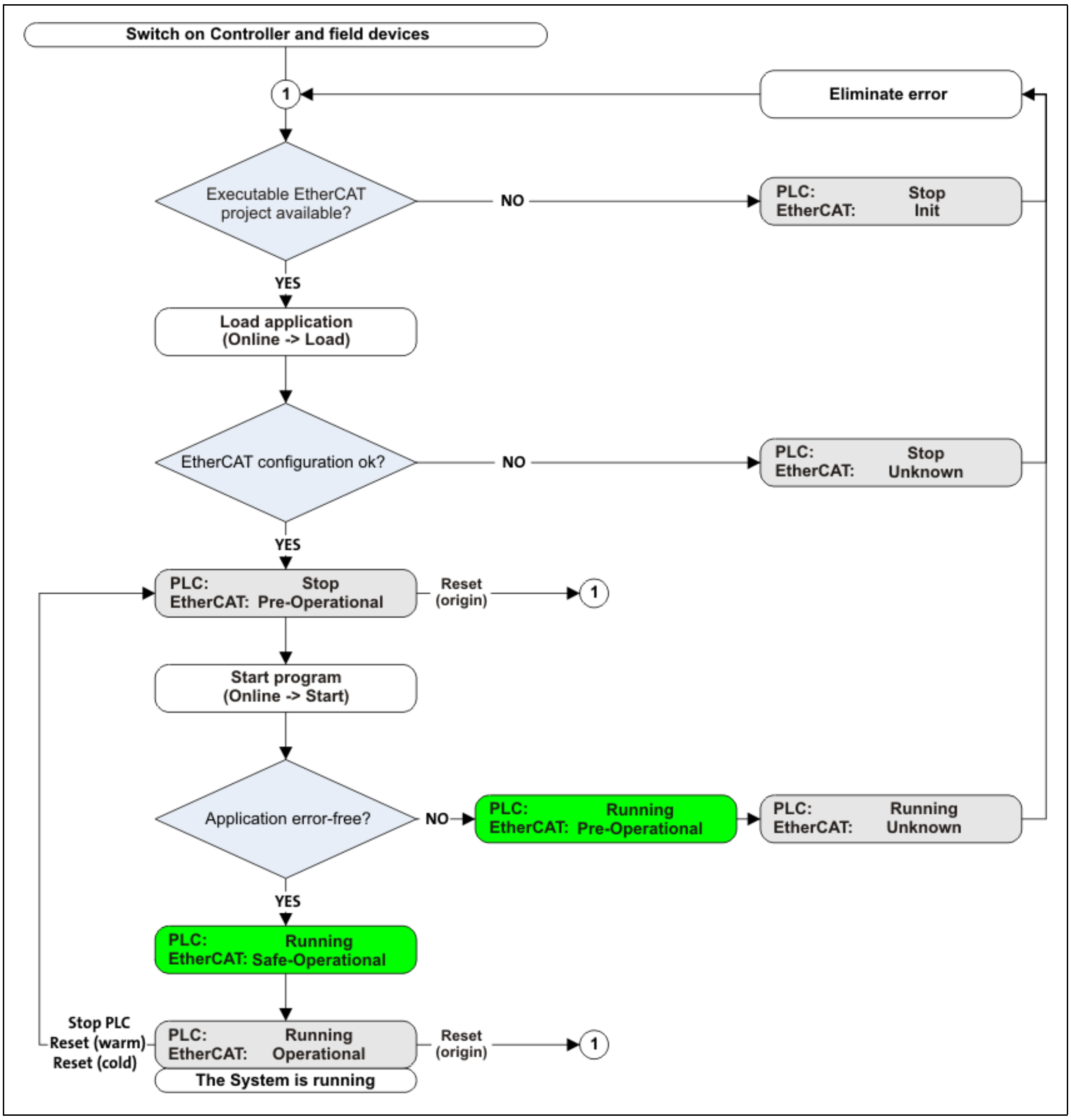
For projects up to and including release 3.5 and for updates to release 3.6, in the EtherCAT master, the 1 "Send at task start" option is not set. The EtherCAT frame is sent at the end of the bus cycle task.

For new projects of release 3.6, the "Send at task start" option is set in the EtherCAT master.

7.4 State diagram for commissioning

The state diagram displays the system behaviour. On the basis of the state diagram, you can locate errors.

You can find further information in the chapter entitled Error scenarios (📖 194).



[7-7] Status diagram for system behaviour

8 Modular machine configuration

The modular machine configuration can be used from release 3.10 onwards!

The modular machine configuration enables only one project to be used for all machine variants (maximum configuration).

Main features of the modular machine configuration are:

- Machine parts can be selected per configuration – without changing the code.
- The order of the EtherCAT node in the »PLC Designer« configuration does not have to comply with the physical order at the fieldbus.
- Changes (add/remove node) are reported to the application.
- EtherCAT nodes (change-over of the configuration) can be activated/deactivated without ...
  - stop/(re-)start of the PLC application;
  - Reset of the control.
- The application can detect whether an EtherCAT node is activated or not.
- In the event of an error in one or several nodes, the machine can continue to run in a defined constellation.

8.1 Behaviour of the EtherCAT master

When the modular machine configuration is used, the EtherCAT master behaviour derives from the behaviour known so far.

As soon as one of the function blocks L_ETC_MMCController (📖 164) or L_ETC_MMCControllerBus (📖 166) is instanced in the PLC application, the EtherCAT master waits with booting the bus.

Via the function blocks **L_ETC_MMCController** and **L_ETC_MMCControllerBus**, a service has to be defined that determines the operating mode of the master. Based on a configuration, the EtherCAT bus can be set to the "Operational" state. A Second Station Address (alias address) can be assigned to the EtherCAT slaves.

Note!

- The function blocks **L_ETC_MMCController** and **L_ETC_MMCControllerBus** may only be instanced once within the PLC application.
- The configuration is only checked while the EtherCAT master is booting. If slaves are removed or added during operation, respective checks have to be carried out by the PLC application.

8.2 Mandatory slaves / Optional slaves

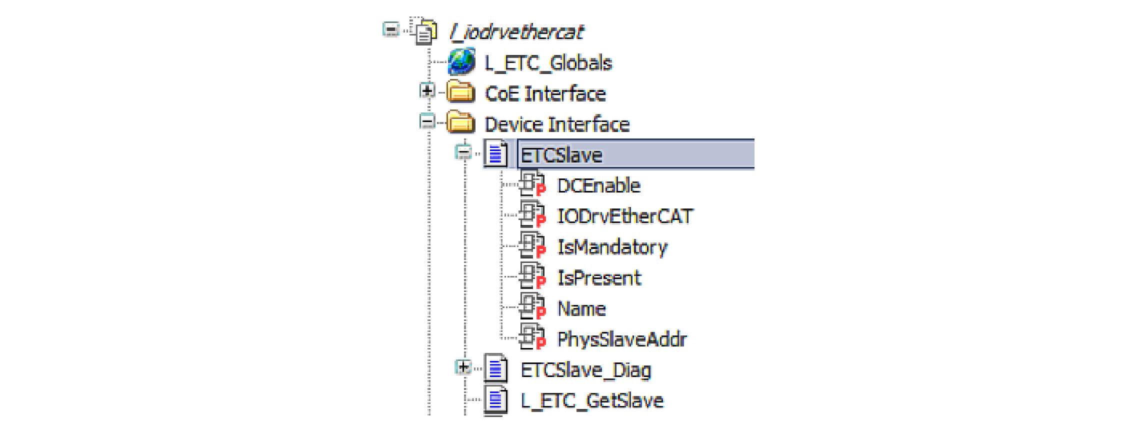
The concept of the modular machine configuration is based on the fact that, depending on the selected configuration within a project, certain EtherCAT slaves have to exist physically at the bus (Mandatory Slaves) or do not have to exist (Optional Slaves). The problem is the "clear" identification of identically constructed devices with the EtherCAT standard mechanisms. For this purpose, the "Second Station Address" (alias address) described the ETG is used which is saved in the EEPROM of the EtherCAT slave.

All EtherCAT slaves have to be defined in the PLC application. The order of the definition determines the EtherCAT address by assigning the addresses continuously from ’1001’. The address is the biunique identifying feature of a device in the network.

If, for instance, a slave shall contain the application A or the application B, a device has to be created in the project for each application. This way, the applications are identified by the different EtherCAT addresses. The same applies to the process data mapping, terminal configuration etc.)

Configurations (Configuration files (🕮 111)) serve to indicate whether an EtherCAT slave is mandatory or optional. The configurations are summarised in the **mmc-0-conf.csv** text file on the Lenze Controller. More identifying features of the slaves are included in the **mmc-0-ident.csv** text file.

If the device is a "Mandatory Slave" or an "Optional Slave" and whether the slave is available at the EtherCAT bus, is displayed via the properties *IsMandatory* and *IsPresent* of the ETCSlave (FB) (📖 146) function block:



[8-1] Properties of the **ETCSlave** function block

Identifier/data type	Meaning/possible settings
IsMandatory (GET) BOOL	Depending on the currently selected modular machine configuration, the feature is set or not set for the EtherCAT slave. If the modular machine configuration is not used, the value 'TRUE' is automatically returned. • TRUE: Slave is "mandatory". • FALSE: Slave is "optional".
IsPresent (GET) BOOL	• TRUE: The slave is currently available at the bus. • TRUE: The slave is currently not available at the bus.

Sample calls in the program:

```
// Mandatory:
xMandatory_94 := L_9400_HL_SM.IsMandatory;
xMandatory_i700_S := L_i700_SM.IsMandatory;
xMandatory_i700_D := L_i700_SM_1.IsMandatory;

// Present:
xPresent_94 := L_9400_HL_SM.IsPresent;
xPresent_i700_S := L_i700_SM.IsPresent;
xPresent_i700_D := L_i700_SM_1.IsPresent;
```

8.3 Configuration files

The configurations if certain EtherCAT slaves are mandatory or optional, are summarised in the **mmc-0-conf.csv** text file on the Lenze Controller.

More identifying features of the slaves are included in the **mmc-0-ident.csv** text file.

Storage directory: `/SDCard/IPC/PLC` or `/USBStorage/IPC/PLC`

Note!

- The machine configuration **mmc-0-conf.csv** and the identifying features **mmc-0-ident.csv** are only loaded while a project (application download boot project) is loaded.
- If an error occurs during the initialisation process, a corresponding error message including the line number is output in the Logbook of the Lenze Controller in the »WebConfig« (190).
  See:
  - L_ETC_MMCSERVICEERROR (176)
  - Error messages for modular machine configuration (211)

8.3.1 Machine configuration

Basically, the machine configuration file **mmc-0-conf.csv** consists of a matrix.

All EtherCAT slaves are given horizontally in ascending order, the single configurations are given vertically.

The EtherCAT slaves are referenced based on the EtherCAT address (Fixed Address), the configuration is based on the designator (STRING).

If a slave has to be contained in a configuration (Mandatory Slave), it is marked by an 'X' in the following sample table.

Address	1001	1002	1003
Inverter	i700	9400 CiA	9400 AS
Configuration 1 (all inverters)	X	X	X
Configuration 2 (1st and 2nd inverter)	X	X	
Configuration 3 (1st and 3rd inverter)	X		X

Contents of the configuration file as text:

```
version;1;0;;
address;1001;1002;1003;comment
name;i700;9400cia;9400as;
conf-1;x;x;x;all drives
conf-2;x;x;;first and second drive
conf-3;x;;x;first and last drive
```

The configuration file does not necessarily have to be available. Preconditions for this are:

- Manual address assignment via the L_ETC_MMCAssignAddress (FB) (163) function block and the ADDR_ASSIGNMENT_EXTERNALLY service
- Operation of the EtherCAT master in the RUN_WITHOUT_CHECK mode

8.3.2 Further identification features of the EtherCAT slaves

An EtherCAT slave is identified by the following features:

- VendorID/ProductCode/Revision
- "Second Station Address" (alias address)
- ID selector (is not supported)

Moreover, the modular machine configuration makes it possible to use CoE objects for further identification of the EtherCAT slaves. These additional identification features are contained in the **mmc-0-ident.csv** text file.

If the identification file is not available on the Lenze Controller, the EtherCAT master assumes that no additional identification features are to be used.

The structure of the file is strictly defined. All columns have to be arranged in the given order or can contain an empty string. In the following sample table, the EtherCAT slaves are arranged in ascending order according to their addresses.

Name	Address	Index	Subindex	Type	Data
L_i700_SM	1001	16#1018	1	DWORD	03 B0 00 00
	1001	16#1019	2	DWORD	02 00 07 69
L_9400_HL_S	1002	16#1020	1	DWORD	3B 00 00 00
	1002	16#1021	2	DWORD	3D 9D 07 38
L_9400_HL_S	1003	16#1022	1	DWORD	3B 00 00 00
	1003	16#1023	2	DWORD	3D 9D 07 38

Column	Description	Notation
Name	Device name of the EtherCAT slave (optional)	STRING
Address	Address of the EtherCAT slave (required) • If no additional identification features are required for a slave, the entire line must be omitted. • If more than one identification feature is to be requested for a slave, a completely new line has to be created (see example below).	INT (decimal)
Index	Index of the CoE object to be requested (required)	INT (hexadecimal with prefix " 16# ")
Subindex	Subindex of the CoE object to be requested (required)	SINT (decimal)
Type	Type of the CoE object to be requested (required) • Only BYTE, WORD and DWORD are supported.	IEC basic types (STRING)
Data	Data the given CoE object has to contain (required) • The number of bytes has to comply with the type.	Octet stream (hexadecimal with spaces between the bytes, Little Endian) Example : The hexadecimal value ’0x00001234’ (’4660’ decimal) is given here with ’34 12 00 00’.
Comment	Comment field for the user (optional)	STRING

8.3.3 Dependencies between configuration files and services

Depending on the selected service (see L_ETC_MMCSERVICE (175)), information from the machine configuration file (mmc-0-conf.csv) and the file for further identification features of the slaves (mmc-0-ident.csv) is required. If the files are not available or if a "Parsing Error" exists, the error message CONFIG_FILE_ERROR or IDENT_FILE_ERROR is caused as soon as a service is activated (see L_ETC_MMCSERVICEERROR (176)).

Service	Machine configuration file mmc-0-conf.csv		Identification file mmc-0-ident.csv	
	Nonexistent	Parsing Error	Nonexistent	Parsing Error
RUN_WITHOUT_CHECK	OK	OK	OK	OK
RUN_OPTIONAL_SLAVES_ALLOWED	rejected 1)	rejected 1)	OK 2)	rejected 3)
RUN_OPTIONAL_SLAVES_PROHIBITED	rejected 1)	rejected 1)	OK 2)	rejected 3)
ADDR_ASSIGNMENT_EXTERNALLY	OK	OK	OK	OK
ADDR_ASSIGNMENT_CONFIG_SLAVEORDER	rejected 1)	rejected 1)	OK	OK
ADDR_ASSIGNMENT_CONFIG_PARAMETER	OK	OK	rejected 3)	rejected 3)

OK File is not required.

1) CONFIG_FILE_ERROR is set (see L_ETC_MMCSERVICEERROR (176)).

2) File is not available. It is assumed that no additional identification features are required.

3) IDENT_FILE_ERROR is set (see L_ETC_MMCSERVICEERROR (176)).

8.4 Address assignment

Before the EtherCAT master in the modular machine configuration can switch to the operating mode (RUN_[...]), each EtherCAT slave has to be assigned to a "Second Station Address" (alias address).

For this purpose, the L_ETC_MMCController (📖 164) function block provides the services **ADDR_ASSIGNMENT_EXTERNALLY** and **ADDR_ASSIGNMENT_CONFIG_SLAVEORDER**.

In case of all services for address assignment, the EtherCAT master reads out the relevant information of the slaves, creates a temporary configuration and sets the slaves into the "Pre-Operational" state. The master assigns the addresses for the slaves in ascending order starting with '1'. Thus, the address is identical to the position of the slave in the network (logical ring: Master – 1 – 2 – 3 – 4 – 5 – ...).

Service ADDR_ASSIGNMENT_EXTERNALLY

In case of this service, the "Second Station Address" (alias address) can be assigned manually out of the PLC application (e.g. via a visualisation).

This service provides the CoE function blocks. Parameters such as vendor-ID, product code, revision, serial number can be read out of the slave or parameters or parameters can be described for "Optical Tracking".

If a slave has been clearly identified and assigned, the L_ETC_MMCAssignAddress (FB) (📖 163) function block can be used for writing the address to the slave.

Service ADDR_ASSIGNMENT_CONFIG_SLAVEORDER

In case of this service, the "Second Station Address" (alias address) is automatically assigned.

For this purpose, a configuration has to be specified, e.g. via the **mmc-0-conf.csv** machine configuration file. Based on the configuration, the EtherCAT master recieves information on the type of slaves to be expected on the bus. If the slaves are actually available, the "Second Station Address" is written to the slaves. Here, the current (ascending) order of the slaves at the bus is important (besides vendor ID and product code).

Example :

Address	1001	1002	1003
Inverter	i700	9400 CiA	9400 AS
Configuration 1 (all inverters)	X	X	X
Configuration 2 (1st and 2nd inverter)	X	X	
Configuration 3 (1st and 3rd inverter)	X		X

For the **configuration 3**, the following slaves at the bus have to be switched on in the following order:

Master – i700 (address '1001') **– 9400as** (address '1003')

If a node is missing, the ADDR_LESS_CONNECTED error message is output.

If more nodes are available, the ADDR_MORE_CONNECTED error message is output.

(See L_ETC_MMCSERVICEERROR (📖 176).)

9 Mixed operation - EtherCAT with other bus systems

Within the Lenze Controller-based Automation arrangement, the EtherCAT bus system can be combined with CANopen, PROFIBUS or PROFINET. This makes senses if only some of the field devices are available for the same bus system or if a Motion bus is needed parallel to the logic bus (CANopen, PROFIBUS, PROFINET).

Note!

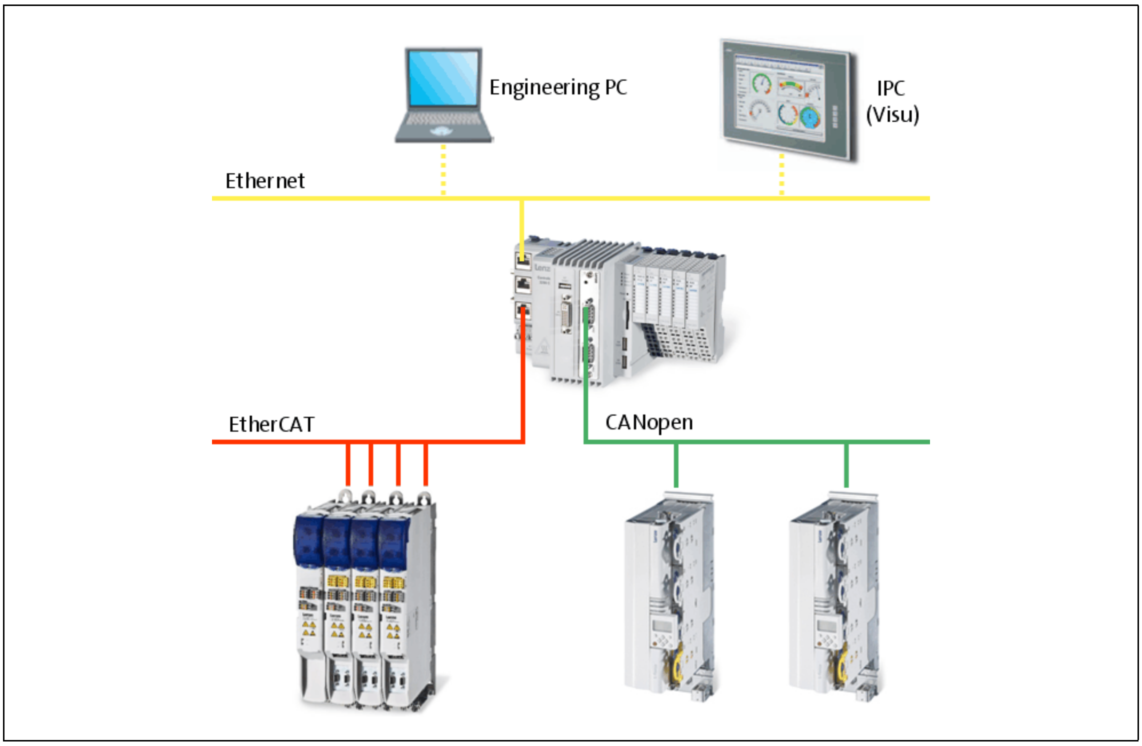
Lenze i700 servo inverter

In the case of the i700 servo inverter, fieldbus communication only takes place via EtherCAT. The servo inverter does not have any CANopen, PROFIBUS or PROFINET interfaces.

Mixed operation - EtherCAT with CANopen

- Due to the demands on the real-time behaviour of the fieldbus system and the limited transfer capacity, for CANopen it is useful to operate Logic and Motion devices on separate fieldbus lines – on a Logic bus and a Motion bus.
- In mixed operation, ensure that the CAN Motion task has the highest priority. The task assigned to the EtherCAT bus should have the second-highest priority. The tasks assigned to the Logic bus systems should be configured with a lower priority.

9.1 EtherCAT and CANopen



[9-1] **Example:** Mixed operation of EtherCAT with CANopen connected to a 3231 C controller with Servo-Inverter i700 and Servo Drives 9400

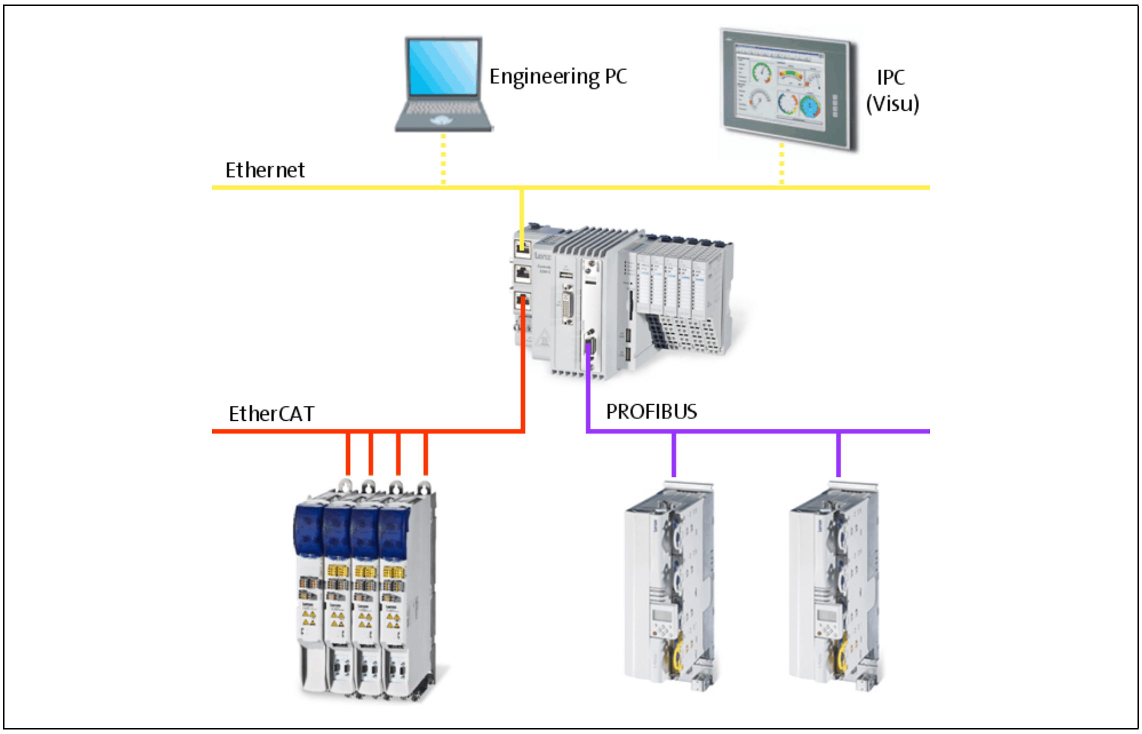
Communication manual for Controller-based Automation with CANopen
Here you can find detailed information on how to commission CANopen components.

9.2 PROFIBUS as the logic bus and EtherCAT as a logic bus or motion bus

Note!

In the Lenze automation system, only the PROFIBUS master functionality (Logic Bus) is supported.

The Motion functionality is not supported when PROFIBUS is used. Always use EtherCAT to connect inverters to be controlled via the central motion functionality.



[9-2] **Example:** Mixed operation of EtherCAT with PROFIBUS connected to a 3231 C controller with Servo-Inverter i700 and Servo Drives 9400

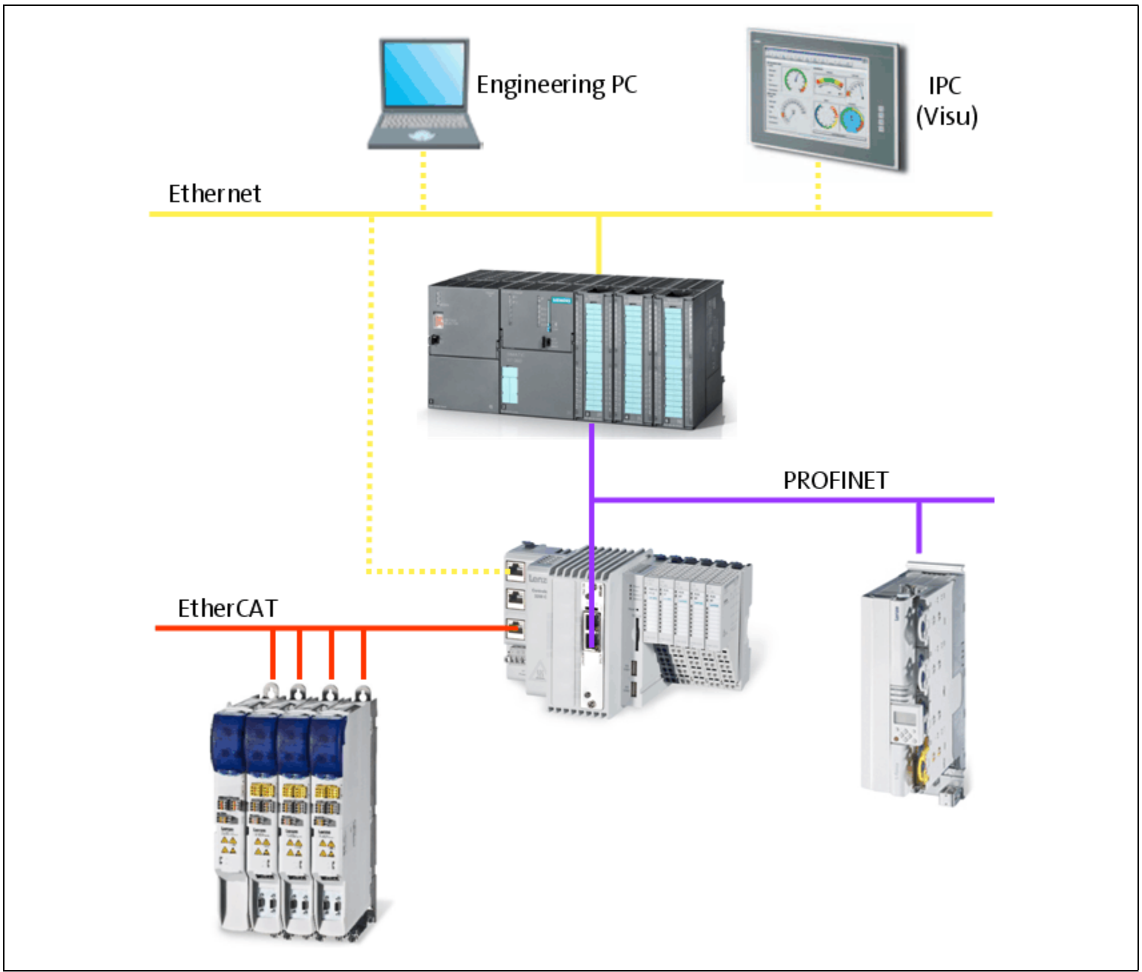
Communication manual for Controller-based Automation with PROFIBUS

Here you can find information on how to commission PROFIBUS components.

9.3 EtherCAT and PROFINET

Note!

- In the Lenze automation system, no PROFINET master functionality is supported. In a PROFINET network, a Lenze Controller can only be driven as I/O device (slave), e.g. by a Siemens SIMATIC S7 PLC.
- In the Lenze automation system, Logic field devices can be exclusively operated via PROFINET. Thus, as an I/O device, the Lenze Controller is a Logic field device.
- The Motion functionality is not supported when PROFINET is used. Always use EtherCAT to connect inverters to be controlled via the central motion functionality.



[9-3] **Example:** Mixed operation of PROFINET with EtherCAT on the Lenze Controller 3221 C

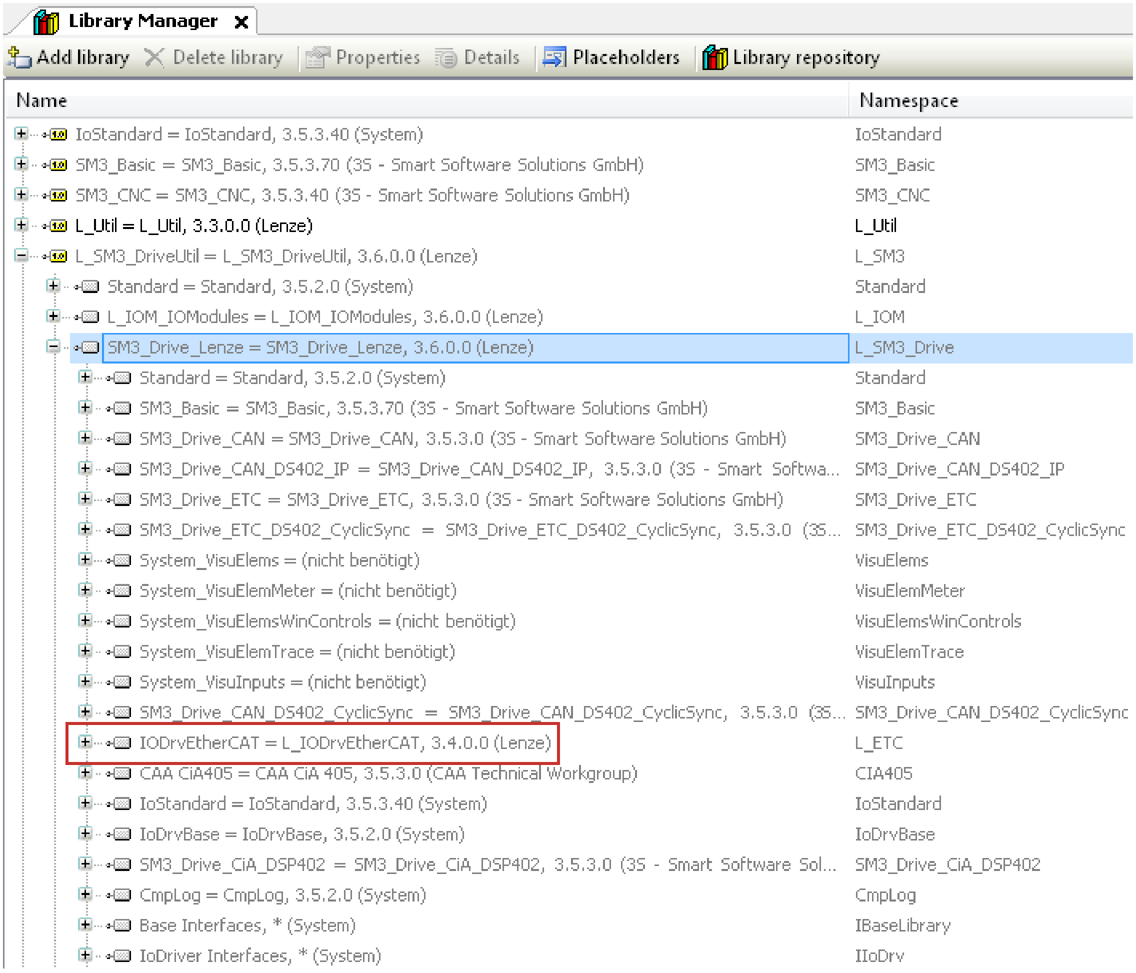
Communication manual Controller-based Automation PROFINET
Here you can find information on how to commission PROFINET components.

10 L_IODrvEtherCAT function library

The **L_IODrvEtherCAT** function library contains all the functions and function blocks for controlling the master and slave status, for diagnostics and for sending and receiving service data.

The interface and its behaviour is in compliance with the "CoDeSys Automation Alliance" (CAA).

The **L_IODrvEtherCAT** function library is part of the **SM3_Drive_Lenze** function library.



Tip!

A sample test project in which parameters of an EtherCAT node is read and written via SDOs and how diagnostic functions of the **L_IODrvEtherCAT** function library are used, can be found in the download area at www.Lenze.com:

Application Knowledge Base: All articles → Application Ideas Pool → Controller 3200 C

The function blocks of the function library **L_IODrvEtherCAT** have inputs and outputs for ...

- activation of the POU;
- display of the current POU state;
- output of error messages.

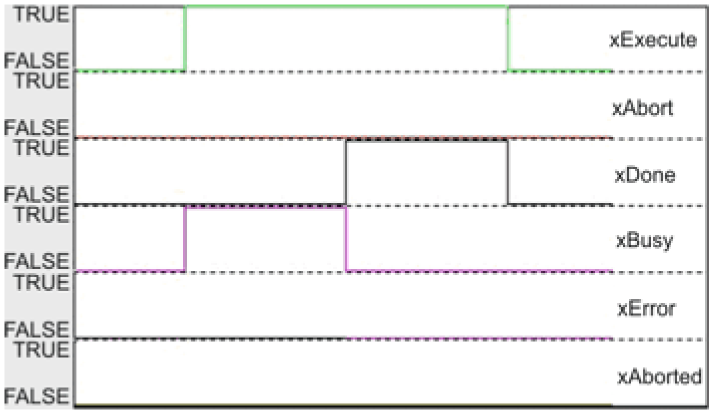
Input/output	Data type	Action
xExecute	BOOL	In the case of a positive edge (TRUE), the function block is executed.
xAbort	BOOL	With *xAbort* = TRUE, the requested service can be cancelled. **Note:** An abort can only be executed in the EtherCAT state "Init" (before, set *xStopBus* = TRUE, see L_IODrvEtherCAT (FB) (📖 151)).
xDone	BOOL	If a function block has been executed, *xDone* is set to TRUE and *xBusy* is set to FALSE. • If *xExecute* has been reset, *xDone* is only active during the function block call. • If *xExecute* = TRUE, then *xDone* = TRUE as long as *xExecute* is reset.
xBusy	BOOL	As long as a function block is executed, *xBusy* is TRUE and *xDone* is FALSE.
xError	BOOL	If an error has occurred, *bError* is set to TRUE. • The *eErrorCode* output displays the error code. • The error code is an enumeration of the L_ETC_ERRORCODE (📖 173) type. • An error in online mode is shown as string. A detailed description of the EtherCAT error codes is given in the chapter "System error messages" (📖 204).
xAborted	BOOL	When *xAborted* = TRUE, a requested service is aborted. **Note:** An abort can only be executed in the EtherCAT state "Init" (before, set *xStopBus* = TRUE, see L_IODrvEtherCAT (FB) (📖 151)).

Behaviour of the function blocks

The function blocks of the **L_IODrvEtherCAT** function library behave in accordance with PLCopen (IEC 61131-3).

The following diagrams show the behaviour when a function block is executed free of errors and when it is executed with errors:

- Error-free execution:



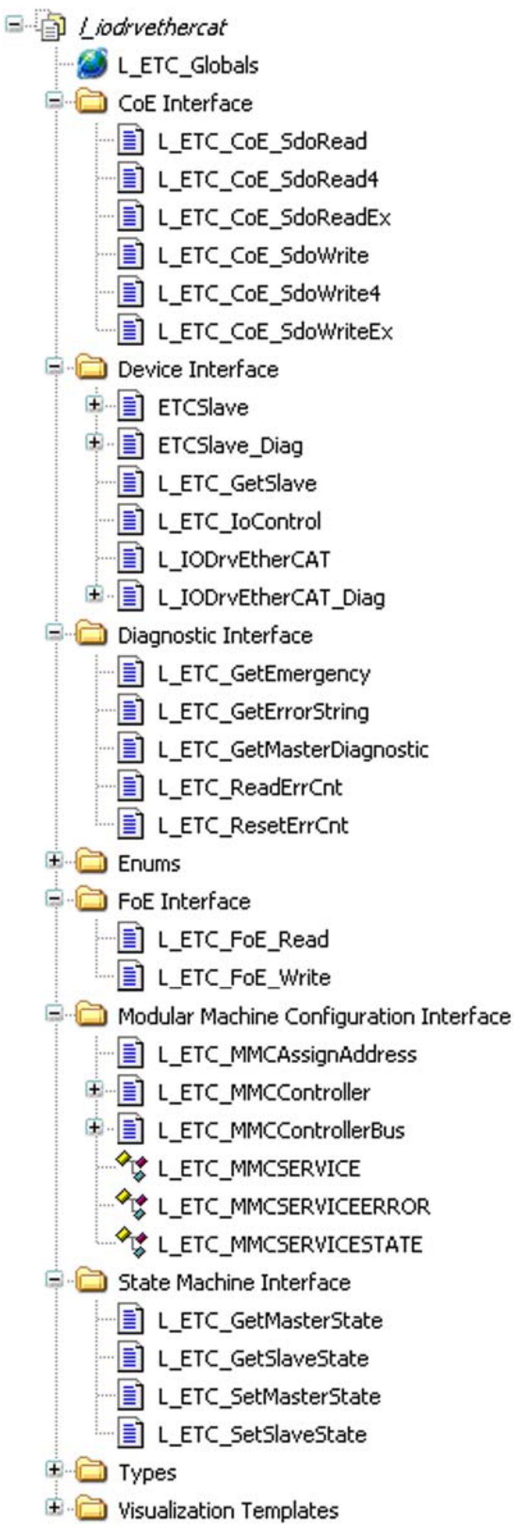
- Error-free execution with falling edge at *xExecute* while *xBusy* = TRUE:

- Error case:

- Error case with falling edge at *xExecute* while *xBusy* = TRUE:

10.1 Overview of the functions and function blocks

The functions and function blocks of the EtherCAT interface are divided into different groups.



CoE Interface (🕮 124)
- L_ETC_CoE_SdoRead (FB) (🕮 134)
- L_ETC_CoE_SdoRead4 (FB) (🕮 136)
- L_ETC_CoE_SdoReadEx (FB) (🕮 138)
- L_ETC_CoE_SdoWrite (FB) (🕮 140)
- L_ETC_CoE_SdoWrite4 (FB) (🕮 142)
- L_ETC_CoE_SdoWriteEx (FB) (🕮 144)

Device Interface (🕮 146)
- ETCSlave (FB) (🕮 146)
- ETCSlave_Diag (FB) (🕮 148)
- L_ETC_GetSlave (FUN) (🕮 149)
- L_ETC_IoControl (FUN) (🕮 150)
- L_IODrvEtherCAT (FB) (🕮 151)
- L_IODrvEtherCAT_Diag (FB) (🕮 152)

Diagnostic Interface (🕮 153)
- L_ETC_GetEmergency (FB) (🕮 153)
- L_ETC_GetErrorString (FUN) (🕮 155)
- L_ETC_GetMasterDiagnostic (FB) (🕮 156)
- L_ETC_ReadErrCnt (FB) (🕮 157)
- L_ETC_ResetErrCnt (FB) (🕮 158)

FoE interface (🕮 159)
- L_ETC_FoE_Read (FB) (🕮 159)
- L_ETC_FoE_Write (FB) (🕮 161)

Modular Machine Configuration Interface (🕮 163)
- L_ETC_MMCAssignAddress (FB) (🕮 163)
- L_ETC_MMCController (🕮 164)
- L_ETC_MMCControllerBus (🕮 166)

State Machine Interface (🕮 168)
- L_ETC_GetMasterState (🕮 168)
- L_ETC_GetSlaveState (🕮 169)
- L_ETC_SetMasterState (🕮 170)
- L_ETC_SetSlaveState (🕮 171)

10.2 CoE Interface

The function blocks of the "CoE interface" (CAN over EtherCAT) allow objects on the EtherCAT master and the EtherCAT slaves to be read and written.

The SDO read and write services are performed serially in the case of EtherCAT. In the Lenze R3.x control technology (Controller-based Automation), a maximum of 100 services can be temporarily stored for processing. If no more services can be accepted because the temporary storage buffer is full, the value '7' is sent back as the error code.

10.2.1 Reading and writing parameters

Parameters ...

- for instance are set for one-time system settings or if materials are changed within a machine;
- are transmitted with a low priority.

In the case of Lenze inverters, the parameters to be changed are contained in codes or in the case of the CANopen device profile "CiA402" as device profile objects.

Indexing of the Lenze codes

When they are accessed, the codes of the Lenze Controllers are addressed by the index.

The index for Lenze code numbers is in the manufacturer-specific area of the object directory between 8192 (0x2000) and 24575 (0x5FFF).

Conversion formula	
Index [dec]	**Index [hex]**
24575 - Lenze code	0x5FFF - Lenze code [hex]

Example for C00002 (device commands)	
Index [dec]	**Index [hex]**
24575 - 2 = 24573	0x5FFF - 2 = 0x5FFD

Structure of a mailbox datagram

In a datagram, mailbox data are transferred within an EtherCAT frame. The data area of the mailbox datagram has the following structure:

Mailbox header	CoE header	SDO control byte	Index	Subindex	Data	Data
6 bytes	2 bytes	1 byte	2 bytes	1 byte	4 bytes	1 ... n bytes

10.2.1.1 Reading parameters (SDO upload)

1. The master sends "Initiate Domain Upload Request".
2. The slave acknowledges the request with a positive response ("Initiate Domain Upload Response").
   In the event of an error the slave responds with "Abort Domain Transfer".

Note!

In the case of jobs for the inverter, please make sure that you convert the code into an index.

▸ Indexing of the Lenze codes (📖 124)

SDO Upload Request

Detailed breakdown of the data for an "SDO Upload Request":

SDO frame area	Data field	Data type / length		Value / description
Mailbox header	Length	WORD	2 bytes	0x0A: Length of the mailbox service data
	Address	WORD	2 bytes	Station address of the source if an EtherCAT master is the instructing party. Station address of the target if an EtherCAT slave is the instructing party.
	Channel	WORD	6 bits (0 ... 5)	0x00: Reserved
	Priority		2 bits (6, 7)	0x00: Lowest priority ... 0x03: Highest priority
	Type		4 bits (8 ... 11)	0x03: CANopen over EtherCAT (CoE)
	Reserved		4 bits (12 ... 15)	0x00
CANopen header	Number	WORD	9 bits (0 ... 8)	0x00
	Reserved		3 bits (9 ... 11)	0x00
	Service		4 bits (12 ... 15)	0x02: SDO request
SDO	Reserved	BYTE	4 bits (0 ... 3)	0x00
	Complete access		1 bit (4)	0x00: The entry addressed with index and subindex is read. 0x01: The complete object is read. (Currently not supported.)
	Command specifier		3 bits (5 ... 7)	0x02: Upload request
	Index	WORD	2 bytes	Index of the object
	Subindex	BYTE	1 byte	Subindex of the object 0x00 or 0x01 if "Complete access" = 0x01.
	Reserved	DWORD	4 bytes	0x00

SDO Upload Expedited Response

An "SDO Upload Expedited Response" is effected if the data length of the parameter data to be read is up to 4 bytes.

Detailed breakdown of the data for an "SDO Upload Expedited Response":

SDO frame area	Data field	Data type / length		Value / description
Mailbox header	Length	WORD	2 bytes	0x0A: Length of the mailbox service data
	Address	WORD	2 bytes	Station address of the source if an EtherCAT master is the instructing party. Station address of the target if an EtherCAT slave is the instructing party.
	Channel	WORD	6 bits (0 ... 5)	0x00: Reserved
	Priority		2 bits (6, 7)	0x00: Lowest priority ... 0x03: Highest priority
	Type		4 bits (8 ... 11)	0x03: CANopen over EtherCAT (CoE)
	Reserved		4 bits (12 ... 15)	0x00
CANopen header	Number	WORD	9 bits (0 ... 8)	0x00
	Reserved		3 bits (9 ... 11)	0x00
	Service		4 bits (12 ... 15)	0x03: SDO response
SDO	Size indicator	BYTE	1 bit (0)	0x01: Size of the data in "Data set size"
	Transfer type		1 bit (1)	0x01: Expedited transfer
	Data set size		2 bits (2, 3)	0x00: 4 bytes of data 0x01: 3 bytes of data 0x02: 2 bytes of data 0x03: 1 byte of data
	Complete access		1 bit (4)	0x00: The entry addressed with index and subindex is read. 0x01: The complete object is read. (Currently not supported.)
	Command specifier		3 bits (5 ... 7)	0x02: Upload response
	Index	WORD	2 bytes	Index of the object
	Subindex	BYTE	1 byte	Subindex of the object 0x00 or 0x01 if "Complete access" = 0x01.
	Data	DWORD	4 bytes	Data of the object

SDO Upload Expedited Response

An "SDO Upload Normal Response" is effected if the data length of the parameter data to be read is ≥ 4 bytes.

Detailed breakdown of the data for an "SDO Upload Normal Response":

SDO frame area	Data field	Data type / length		Value / description
Mailbox header	Length	WORD	2 bytes	n ≥ 0x0A: Length of the mailbox service data
	Address	WORD	2 bytes	Station address of the source if an EtherCAT master is the instructing party. Station address of the target if an EtherCAT slave is the instructing party.
	Channel	WORD	6 bits (0 ... 5)	0x00: Reserved
	Priority		2 bits (6, 7)	0x00: Lowest priority ... 0x03: Highest priority
	Type		4 bits (8 ... 11)	0x03: CANopen over EtherCAT (CoE)
	Reserved		4 bits (12 ... 15)	0x00
CANopen header	Number	WORD	9 bits 0 ... 8)	0x00
	Reserved		3 bits (9 ... 11)	0x00
	Service		4 bits (12 ... 15)	0x03: SDO response
SDO	Size indicator	BYTE	1 bit (0)	0x01
	Transfer type		1 bit (1)	0x00: Normal transfer
	Data set size		2 bits (2, 3)	0x00
	Complete access		1 bit (4)	0x00: The entry addressed with index and subindex is read. 0x01: The complete object is read. (Currently not supported.)
	Command specifier		3 bits (5 ... 7)	0x02: Upload response
	Index	WORD	2 bytes	Index of the object
	Subindex	BYTE	1 byte	Subindex of the object 0x00 or 0x01 if "Complete access" = 0x01.
	Complete size	DWORD	4 bytes	Total data length of the object
	Data	BYTE	n - 10 bytes	Data of the object

Example

In the case of an **upload** to index 0x5FC2 (standard setting of C00061/0 (heatsink temperature) = 0x0000002B (43 °C)), the transmitted response structure contains the following data:

SDO frame area	Data field	Data type / length		Value / description
Mailbox header	Length	WORD	2 bytes	0x0A: Length of the mailbox service data
	Address	WORD	2 bytes	0x00
	Channel	WORD	6 bits (0 ... 5)	0x00: Reserved
	Priority		2 bits (6, 7)	0x00: Lowest priority
	Type		4 bits (8 ... 11)	0x03: CANopen over EtherCAT (CoE)
	Reserved		4 bits (12 ... 15)	0x00
CANopen header	Number	WORD	9 bits (0 ... 8)	0x00
	Reserved		3 bits (9 ... 11)	0x00
	Service		4 bits (12 ... 15)	0x03: SDO response
SDO	Size indicator	BYTE	1 bit (0)	0x01: Length of the data in "Data set size"
	Transfer type		1 bit (1)	0x01: Expedited transfer
	Data set size		2 bits (2, 3)	0x00: 4 bytes of data
	Complete access		1 bit (4)	0x00: The entry addressed with index and subindex is read.
	Command specifier		3 bits (5 ... 7)	0x02: Upload response
	Index	WORD	2 bytes	0xC2: Index low byte of the object 0x5F: Index high byte of the object
	Subindex	BYTE	1 byte	0x00
	Data	DWORD	4 bytes	0x0000002B

10.2.1.2 Writing parameters (SDO download)

1. The master sends "Initiate Domain Download Request".
2. The slave acknowledges the request with a positive response ("Initiate Domain Download Response").
   In the event of an error the slave responds with "Abort Domain Transfer".

Note!

In the case of jobs for the inverter, please make sure that you convert the code into an index.

▸ Indexing of the Lenze codes (📖 124)

SDO Download Expedited Request

An "SDO Download Expedited Request" is effected if the data length of the parameter data to be written is up to 4 bytes.

Detailed breakdown of the data for an "SDO Download Expedited Request":

SDO frame area	Data field	Data type / length		Value / description
Mailbox header	Length	WORD	2 bytes	0x0A: Length of the mailbox service data
	Address	WORD	2 bytes	Station address of the source if an EtherCAT master is the instructing party. Station address of the target if an EtherCAT slave is the instructing party.
	Channel	WORD	6 bits (0 ... 5)	0x00: Reserved
	Priority		2 bits (6, 7)	0x00: Lowest priority ... 0x03: Highest priority
	Type		4 bits (8 ... 11)	0x03: CANopen over EtherCAT (CoE)
	Reserved		4 bits (12 ... 15)	0x00
CANopen header	Number	WORD	9 bits (0 ... 8)	0x00
	Reserved		3 bits (9 ... 11)	0x00
	Service		4 bits (12 ... 15)	0x02: SDO request
SDO	Size indicator	BYTE	1 bit (0)	0x01: Size of the data in "Data set size"
	Transfer type		1 bit (1)	0x01: Expedited transfer
	Data set size		2 bits (2, 3)	0x00: 4 bytes of data 0x01: 3 bytes of data 0x02: 2 bytes of data 0x03: 1 byte of data
	Complete access		1 bit (4)	0x00: The entry addressed with index and subindex is written. 0x01: The complete object is written. (Currently not supported.)
	Command specifier		3 bits (5 ... 7)	0x01: Download request
	Index	WORD	2 bytes	Index of the object
	Subindex	BYTE	1 byte	Subindex of the object 0x00 or 0x01 if "Complete access" = 0x01.
	Data	DWORD	4 bytes	Data of the object

SDO Download Expedited Request

An "SDO Download Normal Request" is effected if the data length of the parameter data to be written is ≥ 4 bytes.

Detailed breakdown of the data for an "SDO Download Normal Request":

SDO frame area	Data field	Data type / length		Value / description
Mailbox header	Length	WORD	2 bytes	n ≥ 0x0A: Length of the mailbox service data
	Address	WORD	2 bytes	Station address of the source if an EtherCAT master is the instructing party. Station address of the target if an EtherCAT slave is the instructing party.
	Channel	WORD	6 bits (0 ... 5)	0x00: Reserved
	Priority		2 bits (6, 7)	0x00: Lowest priority ... 0x03: Highest priority
	Type		4 bits (8 ... 11)	0x03: CANopen over EtherCAT (CoE)
	Reserved		4 bits (12 ... 15)	0x00
CANopen header	Number	WORD	9 bits (0 ... 8)	0x00
	Reserved		3 bits (9 ... 11)	0x00
	Service		4 bits (12 ... 15)	0x02: SDO request
SDO	Size indicator	BYTE	1 bit (0)	0x01
	Transfer type		1 bit (1)	0x00: Normal transfer
	Data set size		2 bits (2, 3)	0x00
	Complete access		1 bit (4)	0x00: The entry addressed with index and subindex is written. 0x01: The complete object is written. (Currently not supported.)
	Command specifier		3 bits (5 ... 7)	0x01: Download request
	Index	WORD	2 bytes	Index of the object
	Subindex	BYTE	1 byte	Subindex of the object 0x00 or 0x01 if "Complete access" = 0x01.
	Complete size	DWORD	4 bytes	Total data length of the object
	Data	BYTE	n - 10 bytes	Data of the object

SDO Download Response

Detailed breakdown of the data for an "SDO Download Response":

SDO frame area	Data field	Data type / length		Value / description
Mailbox header	Length	WORD	2 bytes	0x0A: Length of the mailbox service data
	Address	WORD	2 bytes	Station address of the source if an EtherCAT master is the instructing party. Station address of the target if an EtherCAT slave is the instructing party.
	Channel	WORD	6 bits (0 ... 5)	0x00: Reserved
	Priority		2 bits (6, 7)	0x00: Lowest priority ... 0x03: Highest priority
	Type		4 bits (8 ... 11)	0x03: CANopen over EtherCAT (CoE)
	Reserved		4 bits (12 ... 15)	0x00
CANopen header	Number	WORD	9 bits (0 ... 8)	0x00
	Reserved		3 bits (9 ... 11)	0x00
	Service		4 bits (12 ... 15)	0x03: SDO response
SDO	Size indicator	BYTE	1 bit (0)	0x0
	Transfer type		1 bit (1)	0x0
	Data set size		2 bits (2, 3)	0x0
	Complete access		1 bit (4)	0x00: The entry addressed with index and subindex is written. 0x01: The complete object is written. (Currently not supported.)
	Command specifier		3 bits (5 ... 7)	0x3: Download response
	Index	WORD	2 bytes	Index of the object
	Subindex	BYTE	1 byte	Subindex of the object 0x00 or 0x01 if "Complete access" = 0x01.
	Reserved	DWORD	4 bytes	0x00

Example

In the case of a **download** to index 0x5FA7 (C00088/0, rated motor current I = 10.2 A), the transmitted request structure contains the following data:

SDO frame area	Data field	Data type / length		Value / description
Mailbox header	Length	WORD	2 bytes	0x0A: Length of the mailbox service data
	Address	WORD	2 bytes	0x00
	Channel	WORD	6 bits (0 ... 5)	0x00: Reserved
	Priority		2 bits (6, 7)	0x00: Lowest priority
	Type		4 bits (8 ... 11)	0x03: CANopen over EtherCAT (CoE)
	Reserved		4 bits (12 ... 15)	0x00
CANopen header	Number	WORD	9 bits (0 ... 8)	0x00
	Reserved		3 bits (9 ... 11)	0x00
	Service		4 bits (12 ... 15)	0x02: SDO request
SDO	Size indicator	BYTE	1 bit (0)	0x01: Size of the data in "Data set size"
	Transfer type		1 bit (1)	0x01: Expedited transfer
	Data set size		2 bits (2, 3)	0x00: 4 bytes of data
	Complete access		1 bit (4)	0x00: The entry addressed with index and subindex is written.
	Command specifier		3 bits (5 ... 7)	0x01: Download request
	Index	WORD	2 bytes	0xA7: Index low byte of the object 0x5F: Index high byte of the object
	Subindex	BYTE	1 byte	0x00: Subindex of the object
	Data	DWORD	4 bytes	0x00000066 (10.2 x 10 = 102_{dec})

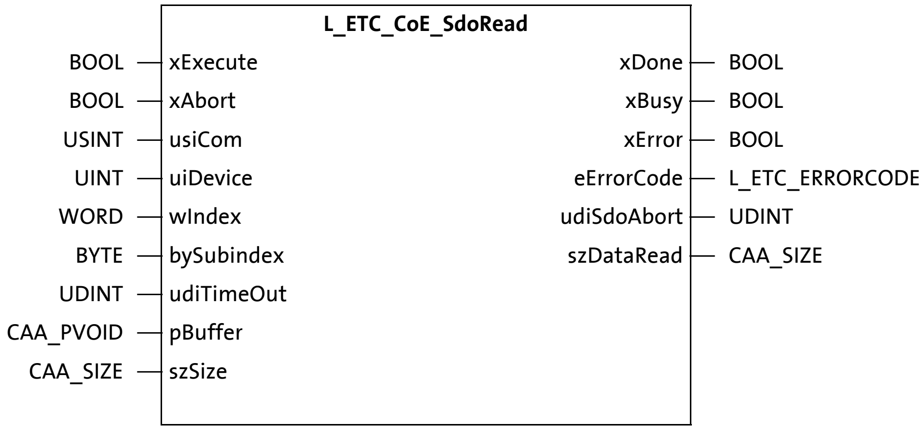
10.2.2 L_ETC_CoE_SdoRead (FB)

This function block triggers uploading of a CoE object (SDO) from the slave or from the master.

Visualisation: VISU_L_ETC_CoE_SdoRead

Note!

For executing the function block, the EtherCAT master and the slave must be at least in the "Pre-Operational" state.



Inputs (VAR_INPUT)

Identifier/data type	Meaning/possible settings
xExecute BOOL	A positive edge (TRUE) triggers a read request (uploading) of a CoE object.
xAbort BOOL	A positive edge (TRUE) aborts a running read request (upload). **Note:** An abort can only be executed in the EtherCAT state "Init" (before, set *xStopBus* = TRUE, see L_IODrvEtherCAT (FB) (📖 151)).
usiCom USINT	EtherCAT master instance number • In the Lenze R3.x control technology, only the instance number ’1’ is currently supported. *usiCom* has the number ’1’ pre-assigned to it so that the input can be left open. • A value that does not equal '1' causes the error ETC_E_INVALIDPARM (0x9811000B).
uiDevice UINT	EtherCAT address (station address) of the master/slave • The station addresses are assigned by the master during the start-up phase (fixed-address assignment, see also Addressing of the slaves (📖 23)). • The address ’0’ directly accesses the object directory of the EtherCAT master. • The first EtherCAT slave is given the address ’1001’, the second the address ’1002’ and so on. • The EtherCAT addresses cannot be altered.
wIndex WORD	CANopen index to be read in the object directory of the master or slave. • Formula for converting a Lenze code number into a CANopen index: **Index = 0x5FFF - code number**
bySubindex BYTE	Subindex in the object directory of the master or slave.
udiTimeout UDINT	Timeout in milliseconds (ms) • The value '0' is not permissible and causes the error ETC_E_INVALIDPARM (0x9811000B).

Identifier/data type	Meaning/possible settings
pBuffer CAA_PVOID	Reference to memory buffer to which the values to be read are to be copied.
szSize CAA_SIZE	Size of the memory buffer transmitted to *pBuffer*. • The memory buffer must be big enough to accept the read object.

Outputs (VAR_OUTPUT)

Identifier/data type	Meaning/possible settings
xDone BOOL	• TRUE: An action has been executed successfully. • FALSE: No action active / action is still being executed.
xBusy BOOL	• TRUE: An action is currently being executed. • FALSE: No action active
xError BOOL	• TRUE: An error has occurred. • FALSE: No error
eErrorCode L_ETC_ERRORCODE	Error code of the incorrectly executed action (*xError* = TRUE). A detailed description of the EtherCAT error codes is given in the chapter "System error messages" (🕮 204).
udiSdoAbort UDINT	If a read request (upload) from the slave or master is rejected with an error, here the CANopen abort code is returned.
szDataRead CAA_SIZE	Number of bytes for the read request (upload) that is actually read

10.2.3 L_ETC_CoE_SdoRead4 (FB)

This function block triggers uploading of a CoE object (SDO) from the slave or from the master.

Visualisation: VISU_L_ETC_CoE_SdoRead4

Note!

- The function of this function block is identical with the function of L_ETC_CoE_SdoRead (FB) (📖 134), except that only up to 4 bytes can be read with **L_ETC_CoE_SdoRead4**.
- For executing the function block, the EtherCAT master and the slave must be at least in the "Pre-Operational" state.

L_ETC_CoE_SdoRead4

BOOL	xExecute	xDone	BOOL
BOOL	xAbort	xBusy	BOOL
USINT	usiCom	xError	BOOL
UINT	uiDevice	eErrorCode	L_ETC_ERRORCODE
WORD	wIndex	udiSdoAbort	UDINT
BYTE	bySubindex	abyData	ARRAY [0..3] OF BYTE
UDINT	udiTimeOut	szDataRead	CAA_SIZE

Inputs (VAR_INPUT)

Identifier/data type	Meaning/possible settings
xExecute BOOL	A positive edge (TRUE) triggers a read request (uploading) of a CoE object.
xAbort BOOL	A positive edge (TRUE) aborts a running read request (upload). **Note:** An abort can only be executed in the EtherCAT state "Init" (before, set *xStopBus* = TRUE, see L_IODrvEtherCAT (FB) (📖 151)).
usiCom USINT	EtherCAT master instance number • In the Lenze R3.x control technology, only the instance number '1' is currently supported. *usiCom* has the number '1' pre-assigned to it so that the input can be left open. • A value that does not equal '1' causes the error ETC_E_INVALIDPARM (0x9811000B).
uiDevice UINT	EtherCAT address of the master/slave. • The address 0 directly accesses the object directory of the EtherCAT master.
wIndex WORD	CANopen index to be read in the object directory of the master or slave. • Formula for converting a Lenze code number into a CANopen index: **Index = 0x5FFF - code number**
bySubindex BYTE	Subindex in the object directory of the master or slave.
udiTimeout UDINT	Timeout in milliseconds (ms) • The value '0' is not permissible and causes the error ETC_E_INVALIDPARM (0x9811000B).

Outputs (VAR_OUTPUT)

Identifier/data type	Meaning/possible settings
xDone BOOL	• TRUE: An action has been executed successfully. • FALSE: No action active / action is still being executed.
xBusy BOOL	• TRUE: An action is currently being executed. • FALSE: No action active
xError BOOL	• TRUE: An error has occurred. • FALSE: No error
eErrorCode L_ETC_ERRORCODE	Error code of the incorrectly executed action (*xError* = TRUE). A detailed description of the EtherCAT error codes is given in the chapter "System error messages" (204).
udiSdoAbort UDINT	If a read request (upload) from the slave or master is rejected with an error, here the CANopen abort code is returned.
abyData ARRAY [0..3] OF BYTE	Memory buffer with the value to be written.
szDataRead CAA_SIZE	Number of bytes for the read request (upload) that is actually read

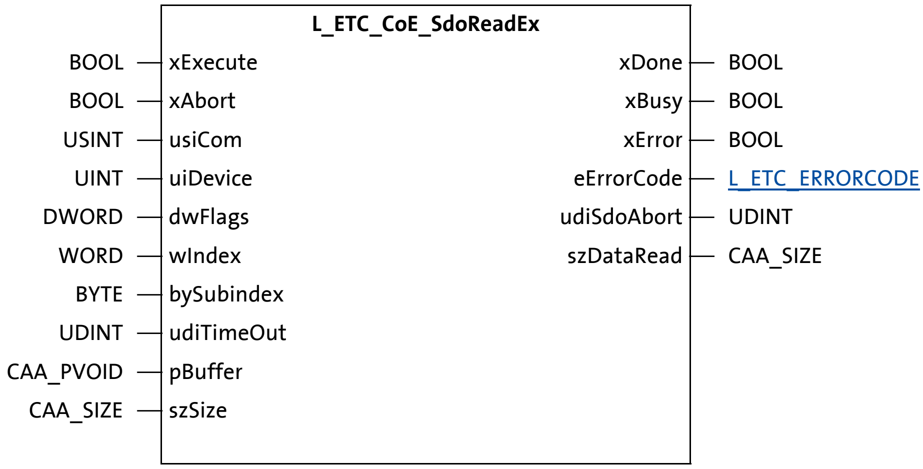
10.2.4 L_ETC_CoE_SdoReadEx (FB)

This function block triggers uploading of a CoE object (SDO) from the slave or from the master.

Visualisation: VISU_L_ETC_CoE_SdoReadEx

Note!

For executing the function block, the EtherCAT master and the slave must be at least in the "Pre-Operational" state.



Complete access

By setting bit 0 at *dwFlags*, all subindices of an object are read by the slave using a single SDO service. The following rules apply for transmission via "complete access":

- Subindex 1 always begins at an even byte address.
- Boolean/bit variables are packed in bytes. Following non-bit objects begin at the next byte address.
- Non-existent subindices do not require any memory space.
- "Complete access" can start with subindex 0 or subindex 1. Other subindices are not permitted.

Inputs (VAR_INPUT)

Identifier/data type	Meaning/possible settings
xExecute BOOL	A positive edge (TRUE) triggers a read request (uploading) of a CoE object.
xAbort BOOL	A positive edge (TRUE) aborts a running read request (upload). **Note:** An abort can only be executed in the EtherCAT state "Init" (before, set *xStopBus* = TRUE, see L_IODrvEtherCAT (FB) (🕮 151)).
usiCom USINT	EtherCAT master instance number • In the Lenze R3.x control technology, only the instance number '1' is currently supported. *usiCom* has the number '1' pre-assigned to it so that the input can be left open. • A value that does not equal '1' causes the error ETC_E_INVALIDPARM (0x9811000B).
uiDevice UINT	EtherCAT address of the master/slave. • The address 0 directly accesses the object directory of the EtherCAT master.
dwFlags DWORD	*dwFlags* controls the function block behaviour: • 16#00000000: Function block behaviour L_ETC_CoE_SdoRead (FB) (🕮 134) • 16#00000001: "Complete access" Type definition see L_ETC_COE_FLAGS (🕮 172).
wIndex WORD	CANopen index to be read in the object directory of the master or slave. • Formula for converting a Lenze code number into a CANopen index: **Index = 0x5FFF - code number**
bySubindex BYTE	Subindex in the object directory of the master or slave.
udiTimeout UDINT	Timeout in milliseconds (ms) • The value '0' is not permissible and causes the error ETC_E_INVALIDPARM (0x9811000B).
pBuffer CAA_PVOID	Reference to memory buffer to which the values to be read are to be copied.
szSize CAA_SIZE	Size of the memory buffer transmitted to *pBuffer*. • The memory buffer must be big enough to accept the read object.

Outputs (VAR_OUTPUT)

Identifier/data type	Meaning/possible settings
xDone BOOL	• TRUE: An action has been executed successfully. • FALSE: No action active / action is still being executed.
xBusy BOOL	• TRUE: An action is currently being executed. • FALSE: No action active
xError BOOL	• TRUE: An error has occurred. • FALSE: No error
eErrorCode L_ETC_ERRORCODE	Error code of the incorrectly executed action (*xError* = TRUE). A detailed description of the EtherCAT error codes is given in the chapter "System error messages" (🕮 204).
udiSdoAbort UDINT	If a read request (upload) from the slave or master is rejected with an error, here the CANopen abort code is returned.
szDataRead CAA_SIZE	Number of bytes for the read request (upload) that is actually read

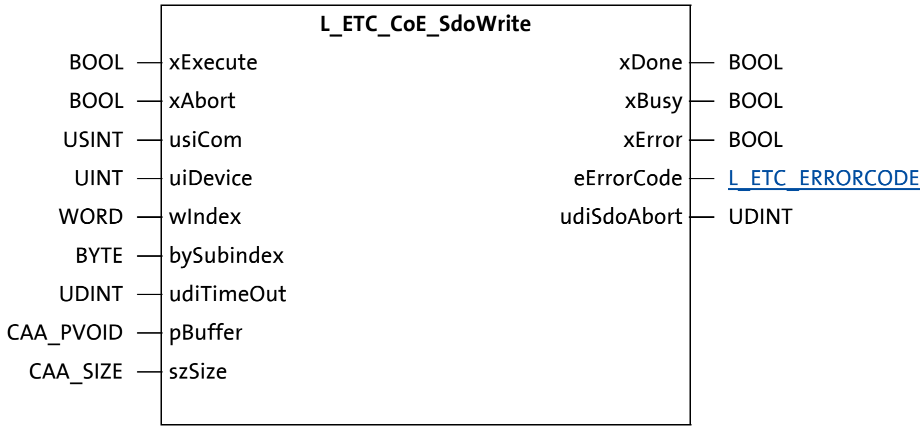
10.2.5 L_ETC_CoE_SdoWrite (FB)

This function block triggers downloading of a CoE object (SDO) to the slave or to the master.

Visualisation: VISU_L_ETC_CoE_SdoWrite

Note!

For executing the function block, the EtherCAT master and the slave must be at least in the "Pre-Operational" state.



Inputs (VAR_INPUT)

Identifier/data type	Meaning/possible settings
xExecute BOOL	A positive edge (TRUE) triggers a write request (downloading) of a CoE object.
xAbort BOOL	A positive edge (TRUE) aborts a running write request (download). **Note:** An abort can only be executed in the EtherCAT state "Init" (before, set *xStopBus* = TRUE, see L_IODrvEtherCAT (FB) (📖 151)).
usiCom USINT	EtherCAT master instance number • In the Lenze R3.x control technology, only the instance number '1' is currently supported. *usiCom* has the number '1' pre-assigned to it so that the input can be left open. • A value that does not equal '1' causes the error ETC_E_INVALIDPARM (0x9811000B).
uiDevice UINT	EtherCAT address of the master/slave. • The address 0 directly accesses the object directory of the EtherCAT master.
wIndex WORD	CANopen index to be read in the object directory of the master or slave. • Formula for converting a Lenze code number into a CANopen index: **Index = 0x5FFF - code number**
bySubindex BYTE	Subindex in the object directory of the master or slave.
udiTimeout UDINT	Timeout in milliseconds (ms) • The value '0' is not permissible and causes the error ETC_E_INVALIDPARM (0x9811000B).
pBuffer CAA_PVOID	Reference to the memory buffer from which the values to be written are to be taken.
szSize CAA_SIZE	Number of the bytes to be written

Outputs (VAR_OUTPUT)

Identifier/data type	Meaning/possible settings
xDone BOOL	• TRUE: An action has been executed successfully. • FALSE: No action active / action is still being executed.
xBusy BOOL	• TRUE: An action is currently being executed. • FALSE: No action active
xError BOOL	• TRUE: An error has occurred. • FALSE: No error
eErrorCode L_ETC_ERRORCODE	Error code of the incorrectly executed action (*xError* = TRUE). A detailed description of the EtherCAT error codes is given in the chapter "System error messages" (204).
udiSdoAbort UDINT	If a write request (download) to the slave or master is rejected with an error, here the CANopen abort code is returned.

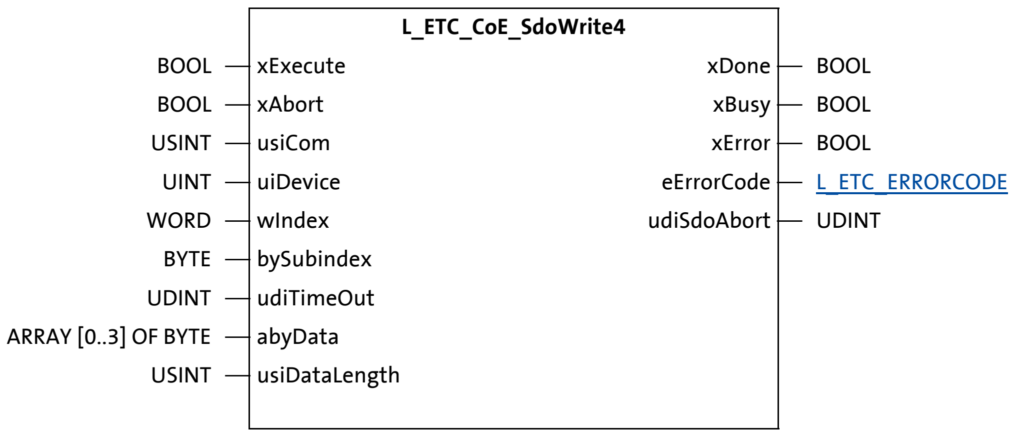
10.2.6 L_ETC_CoE_SdoWrite4 (FB)

This function block triggers downloading of a CoE object (SDO) to the slave or to the master.

Visualisation: VISU_L_ETC_CoE_SdoWrite4

Note!

- The function of this function block is identical with the function of L_ETC_CoE_SdoWrite (FB) (📖 140), except that only up to 4 bytes can be written with **L_ETC_CoE_SdoWrite4**.
- For executing the function block, the EtherCAT master and the slave must be at least in the "Pre-Operational" state.



Inputs (VAR_INPUT)

Identifier/data type	Meaning/possible settings
xExecute BOOL	A positive edge (TRUE) triggers a write request (downloading) of a CoE object.
xAbort BOOL	A positive edge (TRUE) aborts a running write request (download). **Note:** An abort can only be executed in the EtherCAT state "Init" (before, set *xStopBus* = TRUE, see L_IODrvEtherCAT (FB) (📖 151)).
usiCom USINT	EtherCAT master instance number • In the Lenze R3.x control technology, only the instance number ’1’ is currently supported. *usiCom* has the number ’1’ pre-assigned to it so that the input can be left open. • A value that does not equal '1' causes the error ETC_E_INVALIDPARM (0x9811000B).
uiDevice UINT	EtherCAT address of the master/slave. • The address 0 directly accesses the object directory of the EtherCAT master.
wIndex WORD	CANopen index to be read in the object directory of the master or slave. • Formula for converting a Lenze code number into a CANopen index: **Index = 0x5FFF - code number**
bySubindex BYTE	Subindex in the object directory of the master or slave.
udiTimeout UDINT	Timeout in milliseconds (ms) • The value '0' is not permissible and causes the error ETC_E_INVALIDPARM (0x9811000B).

Identifier/data type	Meaning/possible settings
abyData ARRAY [0..3] OF BYTE	Memory buffer with the value to be written.
usiDataLength USINT	Number of the bytes to be written

Outputs (VAR_OUTPUT)

Identifier/data type	Meaning/possible settings
xDone BOOL	• TRUE: An action has been executed successfully. • FALSE: No action active / action is still being executed.
xBusy BOOL	• TRUE: An action is currently being executed. • FALSE: No action active
xError BOOL	• TRUE: An error has occurred. • FALSE: No error
eErrorCode L_ETC_ERRORCODE	Error code of the incorrectly executed action (*xError* = TRUE). A detailed description of the EtherCAT error codes is given in the chapter "System error messages" (📖 204).
udiSdoAbort UDINT	If a write request (download) to the slave or master is rejected with an error, here the CANopen abort code is returned.

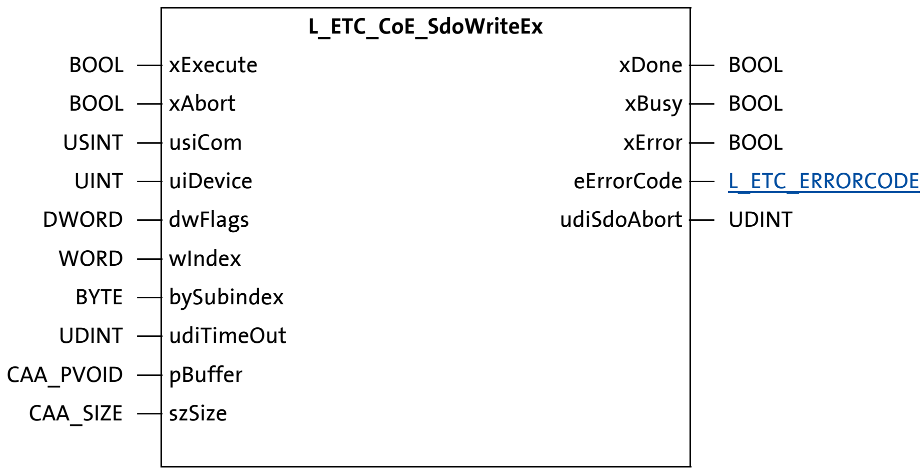
10.2.7 L_ETC_CoE_SdoWriteEx (FB)

This function block triggers downloading of a CoE object (SDO) to the slave or to the master.

Visualisation: VISU_L_ETC_CoE_SdoWriteEx

Note!

For executing the function block, the EtherCAT master and the slave must be at least in the "Pre-Operational" state.



Complete access

By setting bit 0 at *dwFlags*, all subindices of an object are read by the slave using a single SDO service. The following rules apply for transmission via "complete access":

- Subindex 1 always begins at an even byte address.
- Boolean/bit variables are packed in bytes. Following non-bit objects begin at the next byte address.
- Non-existent subindices do not require any memory space.
- "Complete access" can start with subindex 0 or subindex 1. Other subindices are not permitted.

Inputs (VAR_INPUT)

Identifier/data type	Meaning/possible settings
xExecute BOOL	A positive edge (TRUE) triggers a write request (downloading) of a CoE object.
xAbort BOOL	A positive edge (TRUE) aborts a running write request (download). **Note:** An abort can only be executed in the EtherCAT state "Init" (before, set *xStopBus* = TRUE, see L_IODrvEtherCAT (FB) (📖 151)).
usiCom USINT	EtherCAT master instance number • In the Lenze R3.x control technology, only the instance number ’1’ is currently supported. *usiCom* has the number ’1’ pre-assigned to it so that the input can be left open. • A value that does not equal '1' causes the error ETC_E_INVALIDPARM (0x9811000B).
uiDevice UINT	EtherCAT address of the master/slave. • The address 0 directly accesses the object directory of the EtherCAT master.
dwFlags DWORD	*dwFlags* controls the function block behaviour: • 16#00000000: Function block behaviour L_ETC_CoE_SdoWrite (FB) (📖 140) • 16#00000001: "Complete access" Type definition see L_ETC_COE_FLAGS (📖 172).
wIndex WORD	CANopen index to be read in the object directory of the master or slave. • Formula for converting a Lenze code number into a CANopen index: **Index = 0x5FFF - code number**
bySubindex BYTE	Subindex in the object directory of the master or slave.
udiTimeout UDINT	Timeout in milliseconds (ms) • The value '0' is not permissible and causes the error ETC_E_INVALIDPARM (0x9811000B).
pBuffer CAA_PVOID	Reference to the memory buffer from which the values to be written are to be taken.
szSize CAA_SIZE	Number of the bytes to be written

Outputs (VAR_OUTPUT)

Identifier/data type	Meaning/possible settings
xDone BOOL	• TRUE: An action has been executed successfully. • FALSE: No action active / action is still being executed.
xBusy BOOL	• TRUE: An action is currently being executed. • FALSE: No action active
xError BOOL	• TRUE: An error has occurred. • FALSE: No error
eErrorCode L_ETC_ERRORCODE	Error code of the incorrectly executed action (*xError* = TRUE). A detailed description of the EtherCAT error codes is given in the chapter "System error messages" (📖 204).
udiSdoAbort UDINT	If a write request (download) to the slave or master is rejected with an error, here the CANopen abort code is returned.

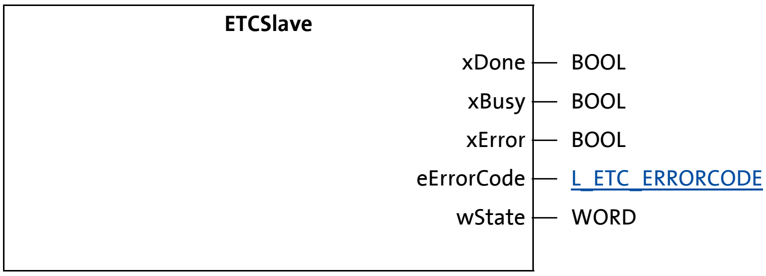
10.3 Device Interface

In addition to the EtherCAT master and slave types for access to slaves, this group also provides a generic function for accessing the EtherCAT master.

10.3.1 ETCSlave (FB)

For every slave in the control configuration, the system creates an object of the type **ETCSlave**. The name of the slave and the name of the EtherCAT slave instance are identical. In the case of a SoftMotion slave, the instance is called cyclically in the context of the SoftMotion application. If the slave is not a SoftMotion slave, the instance must be called when the *wState* output is used.

Visualisation: VISU_ETCSlave



Outputs (VAR_OUTPUT)

Identifier/data type	Meaning/possible settings
xDone BOOL	• TRUE: An action has been executed successfully. • FALSE: No action active / action is still being executed.
xBusy BOOL	• TRUE: An action is currently being executed. • FALSE: No action active
xError BOOL	• TRUE: An error has occurred. • FALSE: No error
eErrorCode L_ETC_ERRORCODE	Error code of the incorrectly executed action (*xError* = TRUE). A detailed description of the EtherCAT error codes is given in the chapter "System error messages" (📖 204).
wState WORD	Current slave state of the L_ETC_STATE (📖 178) type

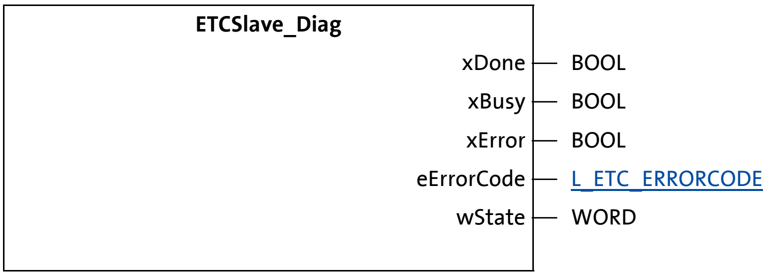
Product features

Identifier/data type	Meaning/possible settings
DCEnable BOOL	• TRUE: The "Distributed clocks" (DC) function is activated for the slave. • FALSE: The "Distributed clocks" (DC) function is not activated. ▶ Synchronisation with "Distributed clocks" (DC) (🕮 37)
IODrvEtherCAT POINTER TO L_IODrvEtherCAT	Reference to the EtherCAT master function block L_IODrvEtherCAT (FB) (🕮 151).
Name STRING	Name of the EtherCAT slave
PhysSlaveAddr UINT	EtherCAT address of the EtherCAT slave
IsMandatory (GET) BOOL	Depending on the currently selected modular machine configuration, the feature is set or not set for the EtherCAT slave. If the modular machine configuration is not used, the value 'TRUE' is automatically returned. • TRUE: Slave is "mandatory". • FALSE: Slave is "optional". ▶ Modular Machine Configuration Interface (🕮 163)
IsPresent (GET) BOOL	• TRUE: The slave is currently available at the bus. • TRUE: The slave is currently not available at the bus.

10.3.2 ETCSlave_Diag (FB)

The **ETCSlave_Diag** function block is an extension for the diagnostics of the EtherCAT slave and contains the same inputs and outputs as ETCSlave (FB) (📖 146).

Visualisation: VISU_ETCSlave



Outputs (VAR_OUTPUT)

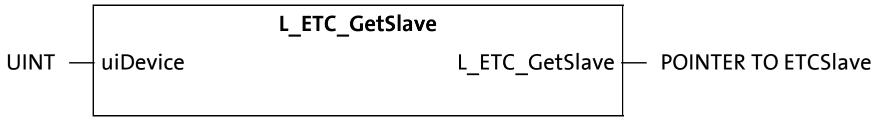
Identifier/data type	Meaning/possible settings
xDone BOOL	• TRUE: An action has been executed successfully. • FALSE: No action active / action is still being executed.
xBusy BOOL	• TRUE: An action is currently being executed. • FALSE: No action active
xError BOOL	• TRUE: An error has occurred. • FALSE: No error
eErrorCode L_ETC_ERRORCODE	Error code of the incorrectly executed action (*xError* = TRUE). A detailed description of the EtherCAT error codes is given in the chapter "System error messages" (📖 204).
wState WORD	Current slave state of the L_ETC_STATE (📖 178) type

Product features

Identifier/data type	Meaning/possible settings
DCEnable BOOL	• TRUE: The "Distributed clocks" (DC) function is activated for the slave. • FALSE: The "Distributed clocks" (DC) function is not activated. ▸ Synchronisation with "Distributed clocks" (DC) (📖 37)
IODrvEtherCAT POINTER TO L_IODrvEtherCAT	Reference to the EtherCAT master function block L_IODrvEtherCAT (FB) (📖 151).
Name STRING	Name of the EtherCAT slave
PhysSlaveAddr UINT	EtherCAT address of the EtherCAT slave
IsMandatory (GET) BOOL	Depending on the currently selected modular machine configuration, the feature is set or not set for the EtherCAT slave. If the modular machine configuration is not used, the value 'TRUE' is automatically returned. • TRUE: Slave is "mandatory". • FALSE: Slave is "optional". ▸ Modular Machine Configuration Interface (📖 163)
IsPresent (GET) BOOL	• TRUE: The slave is currently available at the bus. • TRUE: The slave is currently not available at the bus.

10.3.3 L_ETC_GetSlave (FUN)

On the basis of the EtherCAT address, this function provides the reference to the slave instance of the ETCSlave (FB) (📖 146) type.



Inputs (VAR_INPUT)

Identifier/data type	Meaning/possible settings
uiDevice UINT	The EtherCAT address of the slave for which the reference is to be returned.

Return value

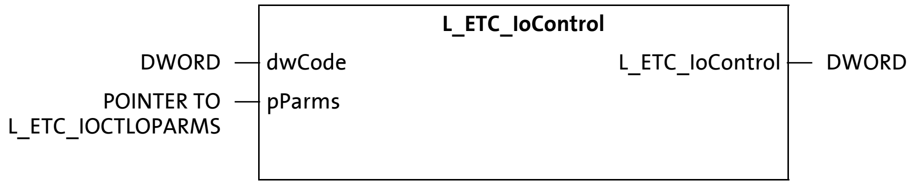
Identifier/data type	Meaning/possible settings
L_ETC_GetSlave POINTER TO ETCSlave	Reference to the EtherCAT slave object by the EtherCAT address provided by *uiDevice*.

10.3.4 L_ETC_IoControl (FUN)

This function transmits a "IoControl" to the EtherCAT master stack.

Note!

This function is only to be used Lenze-internally!



Inputs (VAR_INPUT)

Identifier/data type	Meaning/possible settings
dwCode DWORD	The EtherCAT address of the slave for which the reference is to be returned.
pParms POINTER TO L_ETC_IOCTLOPARMS	Reference to input and output buffers of the L_ETC_IOCTLOPARMS (174) type

Return value

Identifier/data type	Meaning/possible settings
L_ETC_IoControl DWORD	The return value of this function corresponds to the error code (L_ETC_ERRORCODE (173)).

10.3.5 L_IODrvEtherCAT (FB)

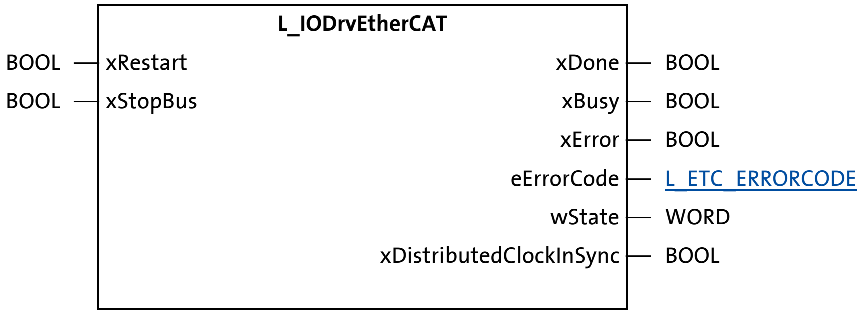
If an EtherCAT master is created in the configuration tree of a »PLC Designer« project, the system automatically creates an object of the **L_IODrvEtherCAT** type.

The name of the function block for the EtherCAT master ist "EtherCAT_Master".

Visualisation: VISU_L_IODrvEtherCAT

Note!

The **L_IODrvEtherCAT** function block is called in the context of the bus cycle task and must not be called again in the PLC program.



Inputs (VAR_INPUT)

Identifier/data type	Meaning/possible settings
xRestart BOOL	A positive edge (TRUE) executes a **bus restart**. • The EtherCAT master is first set to the Init state and then to "Operational". • When *xStopBus* = TRUE, the positive edge is ignored. ▸ Restarting the EtherCAT fieldbus (📖 179)
xStopBus BOOL	A positive edge (TRUE) executes a bus stop. • The fieldbus is set to the "Init" state.

Outputs (VAR_OUTPUT)

Identifier/data type	Meaning/possible settings
xDone BOOL	• TRUE: An action has been executed successfully. • FALSE: No action active / action is still being executed.
xBusy BOOL	• TRUE: An action is currently being executed. • FALSE: No action active
xError BOOL	• TRUE: An error has occurred. • FALSE: No error
eErrorCode L_ETC_ERRORCODE	Error code of the incorrectly executed action (*xError* = TRUE). A detailed description of the EtherCAT error codes is given in the chapter "System error messages" (📖 204).
wState WORD	Current master state of the L_ETC_STATE (📖 178) type
xDistributedClockInSync BOOL	• TRUE: All DC slaves have been synchronised. • FALSE: Not all DC slaves have been synchronised.

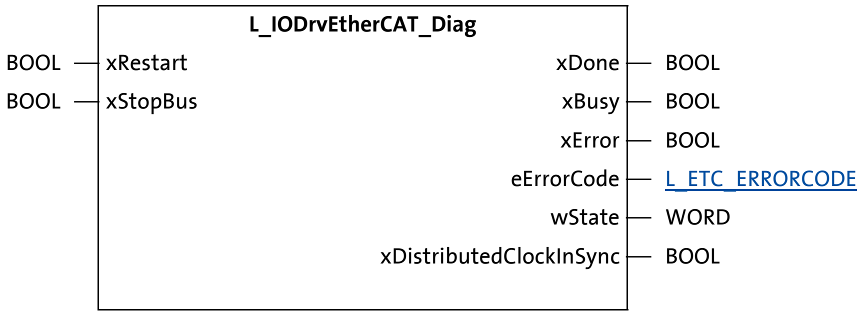
10.3.6 L_IODrvEtherCAT_Diag (FB)

The **L_IODrvEtherCAT_Diag** function block is an extension for the diagnostics of the EtherCAT network and contains the same inputs and outputs as L_IODrvEtherCAT (FB) (📖 151).

Visualisation: VISU_L_IODrvEtherCAT

Note!

The **L_IODrvEtherCAT** function block is called in the context of the bus cycle task and must not be called again in the PLC program.



Inputs (VAR_INPUT)

Identifier/data type	Meaning/possible settings
xRestart BOOL	A positive edge (TRUE) executes a **bus restart**. • The EtherCAT master is first set to the Init state and then to "Operational". • When *xStopBus* = TRUE, the positive edge is ignored. ▸ Restarting the EtherCAT fieldbus (📖 179)
xStopBus BOOL	A positive edge (TRUE) executes a bus stop. • The fieldbus is set to the "Init" state.

Outputs (VAR_OUTPUT)

Identifier/data type	Meaning/possible settings
xDone BOOL	• TRUE: An action has been executed successfully. • FALSE: No action active / action is still being executed.
xBusy BOOL	• TRUE: An action is currently being executed. • FALSE: No action active
xError BOOL	• TRUE: An error has occurred. • FALSE: No error
eErrorCode L_ETC_ERRORCODE	Error code of the incorrectly executed action (*xError* = TRUE). A detailed description of the EtherCAT error codes is given in the chapter "System error messages" (📖 204).
wState WORD	Current master state of the L_ETC_STATE (📖 178) type
xDistributedClockInSync BOOL	• TRUE: All DC slaves have been synchronised. • FALSE: Not all DC slaves have been synchronised.

10.4 Diagnostic Interface

The "Diagnostic Interface" provides diagnostic blocks for the master and the slaves. The **L_ETC_GetErrorString** function converts the internal error code into a readable string.

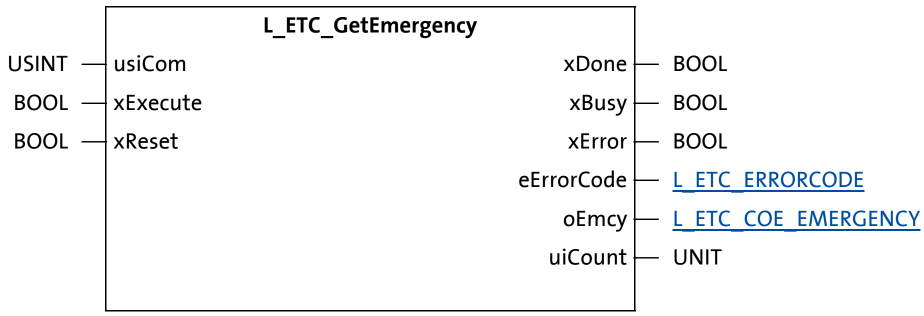
10.4.1 L_ETC_GetEmergency (FB)

This function block outputs CoE emergency frames which have been stored in a buffer by the I/O driver before.

Visualisation: VISU_L_ETC_GetEmergency

Note!

For executing the function block, the EtherCAT master and the slave must be at least in the "Pre-Operational" state.



Inputs (VAR_INPUT)

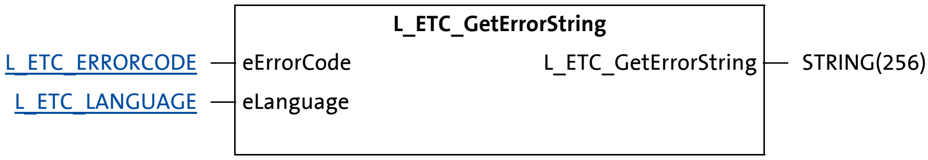
Identifier/data type	Meaning/possible settings
usiCom USINT	EtherCAT master instance number • In the Lenze R3.x control technology, only the instance number '1' is currently supported. *usiCom* has the number '1' pre-assigned to it so that the input can be left open. • A value that does not equal '1' causes the error ETC_E_INVALIDPARM (0x9811000B).
xExecute BOOL	A positive edge (TRUE) activates a read request of a CoE emergency frame from the buffer.
xReset BOOL	A positive edge (TRUE) resets the buffer and deletes all CoE emergency frames saved until then.

Outputs (VAR_OUTPUT)

Identifier/data type	Meaning/possible settings
xDone BOOL	• TRUE: An action has been executed successfully. • FALSE: No action active / action is still being executed.
xBusy BOOL	• TRUE: An action is currently being executed. • FALSE: No action active
xError BOOL	• TRUE: An error has occurred. • FALSE: No error
eErrorCode L_ETC_ERRORCODE	Error code of the incorrectly executed action (*xError* = TRUE). If no emergency telegram was received, the error message "0x9811000C: ETC_E_NOTFOUND" is output. A detailed description of the EtherCAT error codes is given in the chapter "System error messages" (204).
oEmcy L_ETC_COE_EMERGENCY	If *xExecute* is TRUE, a CoE emergency frame read from the buffer is pending at the *oEmcy* output. Information in the CoE emergency frame: • *uiDevice* : UNIT (slave address) • *dwTimestamp* : DATE_AND_TIME (timestamp UTC, see FUNC SysTimeRtcGet) • *wErrorCode* : WORD (error code, device-specific) • *byErrorRegister* : BYTE (error register, device-specific) • *abyData* : ARRAY[0..4] OF BYTE (error data, device-specific)
uiCount UINT	If *xExecute* is TRUE, the number of CoE emergency frames remaining in the buffer is specified at the *uiCount* output.

10.4.2 L_ETC_GetErrorString (FUN)

This function returns a language-specific string for an **L_ETC_ErrorCode**.



Inputs (VAR_INPUT)

Identifier/data type	Meaning/possible settings
eErrorCode L_ETC_ERRORCODE	Error code of the incorrectly executed action (*xError* = TRUE). A detailed description of the EtherCAT error codes is given in the chapter "System error messages" (📖 204).
eLanguage L_ETC_LANGUAGE	Language selection • In the Lenze R3.x control technology, only English strings are supported at the moment.

Return value

Identifier/data type	Meaning/possible settings
L_ETC_GetErrorString STRING(256)	Language-specific error string for *eErrorCode*

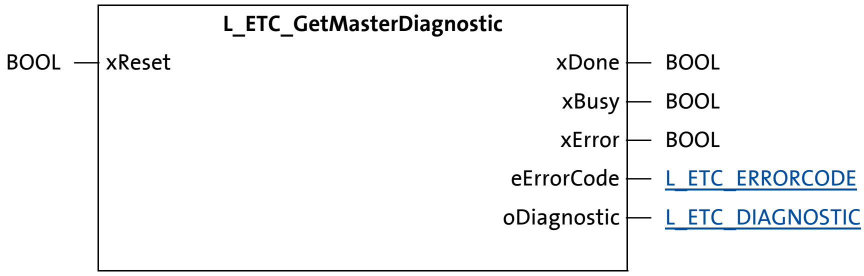
10.4.3 L_ETC_GetMasterDiagnostic (FB)

When called, this function block returns a structure with diagnostic information at the *oDiagnostic* output (type L_ETC_DIAGNOSTIC (173)).

The processing of the POU requires some microseconds (µs).

Visualisation: VISU_L_ETC_GetMasterDiagnostic

▸ Visualisation of the function block L_ETC_GetMasterDiagnostic (186)



Inputs (VAR_INPUT)

Identifier/data type	Meaning/possible settings
xReset BOOL	A positive edge (TRUE) resets the error counter of the output structure *oDiagnostic*. (Exception "Frame Lost Counter")

Outputs (VAR_OUTPUT)

Identifier/data type	Meaning/possible settings
xDone BOOL	• TRUE: An action has been executed successfully. • FALSE: No action active / action is still being executed.
xBusy BOOL	• TRUE: An action is currently being executed. • FALSE: No action active
xError BOOL	• TRUE: An error has occurred. • FALSE: No error
eErrorCode L_ETC_ERRORCODE	Error code of the incorrectly executed action (*xError* = TRUE). A detailed description of the EtherCAT error codes is given in the chapter "System error messages" (204).
oDiagnostic L_ETC_DIAGNOSTIC	Diagnostic information that is returned when the function block is called. • The output *oDiagnostic.xDC_InSync* is set (TRUE) if the DC master and all DC slaves have been synchronised.

10.4.4 L_ETC_ReadErrCnt (FB)

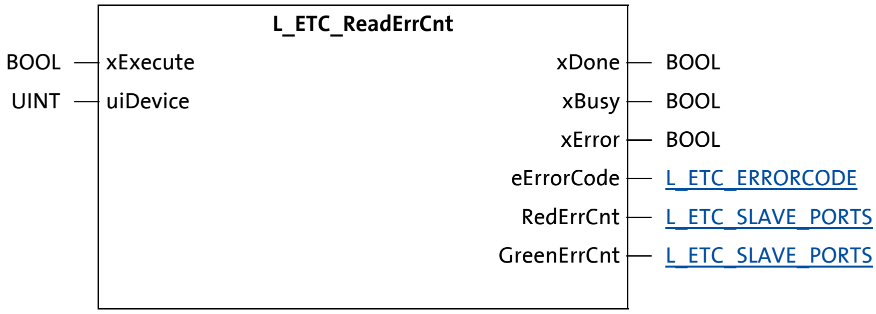
This function block reads the frame error counters of the connected slaves.

On the basis of the values in the *RedErrCnt* array, the wiring quality (EMC sensitivity) of the EtherCAT bus can be evaluated.

Visualisation: VISU_L_ETC_ReadErrCnt

Note!

- We recommend executing the **L_ETC_ReadErrCnt** function block cyclically in greater distances (e.g. every 10 minutes) in order to evaluate the bus state.
- For executing the function block, the EtherCAT master and the slave must be at least in the "Pre-Operational" state.



Inputs (VAR_INPUT)

Identifier/data type	Meaning/possible settings
xExecute BOOL	A positive edge (TRUE) reads the error counter of the slave given by *uiDevice*.
uiDevice UINT	The EtherCAT address of the slave

Outputs (VAR_OUTPUT)

Identifier/data type	Meaning/possible settings
xDone BOOL	• TRUE: An action has been executed successfully. • FALSE: No action active / action is still being executed.
xBusy BOOL	• TRUE: An action is currently being executed. • FALSE: No action active
xError BOOL	• TRUE: An error has occurred. • FALSE: No error
eErrorCode L_ETC_ERRORCODE	Error code of the incorrectly executed action (*xError* = TRUE). A detailed description of the EtherCAT error codes is given in the chapter "System error messages" (📖 204).
RedErrCnt L_ETC_SLAVE_PORTS	Error counter for errors detected in the slave for the first time ▸ Error counters of the EtherCAT slaves (📖 192)
GreenErrCnt L_ETC_SLAVE_PORTS	Error counter for forwarded errors, i.e. errors that have been detected in a previous slave

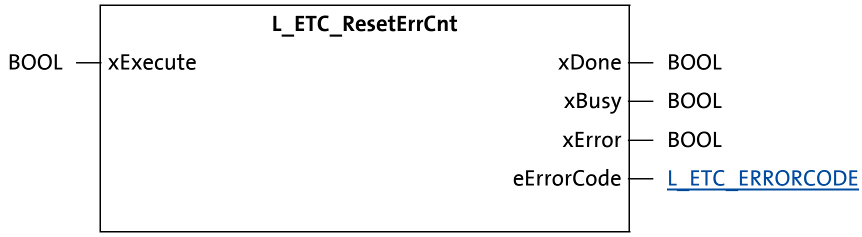
10.4.5 L_ETC_ResetErrCnt (FB)

This function block resets the frame error counters of all connected slaves.

Visualisation: VISU_L_ETC_ResetErrCnt

Note!

We recommend executing the **L_ETC_ResetErrCnt** function block before one of the frame error counters of the slaves has reached the maximum value of '255'.



Inputs (VAR_INPUT)

Identifier/data type	Meaning/possible settings
xExecute BOOL	A positive edge (TRUE) resets the frame error counters of all connected slaves.

Outputs (VAR_OUTPUT)

Identifier/data type	Meaning/possible settings
xDone BOOL	• TRUE: An action has been executed successfully. • FALSE: No action active / action is still being executed.
xBusy BOOL	• TRUE: An action is currently being executed. • FALSE: No action active
xError BOOL	• TRUE: An error has occurred. • FALSE: No error
eErrorCode L_ETC_ERRORCODE	Error code of the incorrectly executed action (*xError* = TRUE). A detailed description of the EtherCAT error codes is given in the chapter "System error messages" (📖 204).

10.5 FoE interface

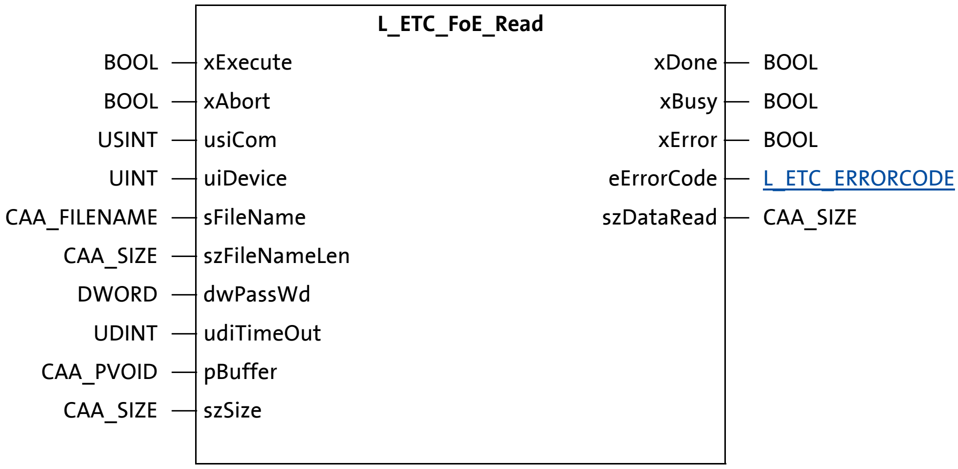
The function blocks of the "FoE interface" (File over EtherCAT) allow you to transmit files between the EtherCAT master and the EtherCAT slaves.

10.5.1 L_ETC_FoE_Read (FB)

This function block activates a file upload from the slave or from the master.

Visualisation: VISU_L_ETC_FoE_Read

> **Note!**
>
> For executing the function block, the EtherCAT master and the slave must be at least in the "Pre-Operational" state.



Inputs (VAR_INPUT)

Identifier/data type	Meaning/possible settings
xExecute BOOL	A positive edge (TRUE) activates a read request (upload) of a file.
xAbort BOOL	A positive edge (TRUE) aborts a running read request (upload). **Note:** An abort can only be executed in the EtherCAT state "Init" (before, set *xStopBus* = TRUE, see L_IODrvEtherCAT (FB) (📖 151)).
usiCom USINT	EtherCAT master instance number • In the Lenze R3.x control technology, only the instance number '1' is currently supported. *usiCom* has the number '1' pre-assigned to it so that the input can be left open. • A value that does not equal '1' causes the error ETC_E_INVALIDPARM (0x9811000B).
uiDevice UINT	EtherCAT address of the master/slave. • The address '0' directly accesses the object directory of the EtherCAT master.
sFileName CAA_FILENAME	File name of file to be read • Max. 32 characters (see *szFileNameLen*) • Directory structures are not supported.

Identifier/data type	Meaning/possible settings
szFileNameLen CAA_SIZE	File name length • The file name must not be longer than 32 characters. • In the case of more than 32 characters, the ETC_E_INVALIDPARM (0x9811000B) error is returned. Example: "firmware.efw" corresponds to 12 characters.
dwPassWd DWORD	Password
udiTimeout UDINT	Timeout in milliseconds (ms) • The value '0' is not permissible and causes the error ETC_E_INVALIDPARM (0x9811000B).
pBuffer CAA_PVOID	Reference to memory buffer to which the values to be read are to be copied.
szSize CAA_SIZE	Size of the memory buffer transmitted to *pBuffer*. • The memory buffer must be big enough to accept the read object.

Outputs (VAR_OUTPUT)

Identifier/data type	Meaning/possible settings
xDone BOOL	• TRUE: An action has been executed successfully. • FALSE: No action active / action is still being executed.
xBusy BOOL	• TRUE: An action is currently being executed. • FALSE: No action active
xError BOOL	• TRUE: An error has occurred. • FALSE: No error
eErrorCode L_ETC_ERRORCODE	Error code of the incorrectly executed action (*xError* = TRUE). A detailed description of the EtherCAT error codes is given in the chapter "System error messages" (204).
szDataRead CAA_SIZE	Number of bytes for the read request (upload) that is actually read

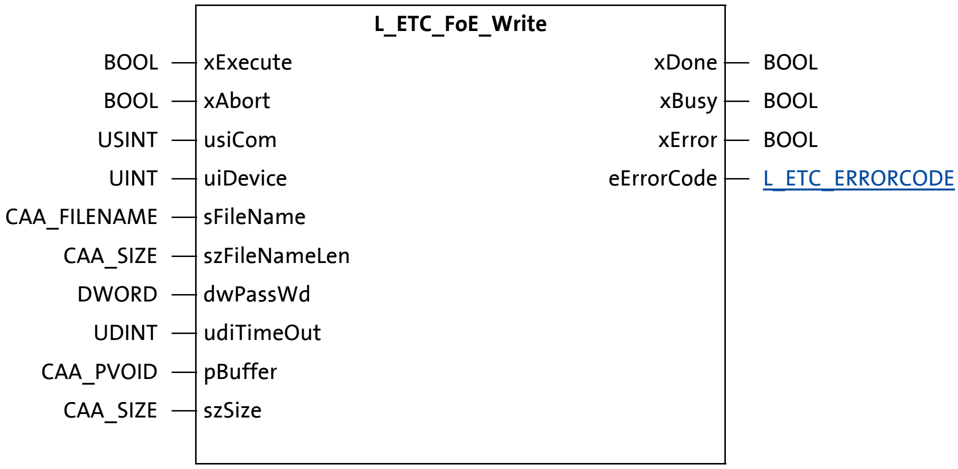
10.5.2 L_ETC_FoE_Write (FB)

This function block activates a file download to the slave or to the master.

Visualisation: VISU_L_ETC_FoE_Write

Note!

For executing the function block, the EtherCAT master and the slave must be at least in the "Pre-Operational" state.



Inputs (VAR_INPUT)

Identifier/data type	Meaning/possible settings
xExecute BOOL	A positive edge (TRUE) activates a write request (download) of a file.
xAbort BOOL	A positive edge (TRUE) aborts a running write request (download). **Note:** An abort can only be executed in the EtherCAT state "Init" (before, set *xStopBus* = TRUE, see L_IODrvEtherCAT (FB) (📖 151)).
usiCom USINT	EtherCAT master instance number • In the Lenze R3.x control technology, only the instance number '1' is currently supported. *usiCom* has the number '1' pre-assigned to it so that the input can be left open. • A value that does not equal '1' causes the error ETC_E_INVALIDPARM (0x9811000B).
uiDevice UINT	EtherCAT address of the master/slave. • The address 0 directly accesses the object directory of the EtherCAT master.
sFileName CAA_FILENAME	File name of file to be written • Max. 32 characters (see *szFileNameLen*) • Directory structures are not supported.
szFileNameLen CAA_SIZE	File name length • The file name must not be longer than 32 characters. • In the case of more than 32 characters, the ETC_E_INVALIDPARM (0x9811000B) error is returned. Example: "firmware.efw" corresponds to 12 characters.
dwPassWd DWORD	Password
udiTimeout UDINT	Timeout in milliseconds (ms) • The value '0' is not permissible and causes the error ETC_E_INVALIDPARM (0x9811000B).

Identifier/data type	Meaning/possible settings
pBuffer CAA_PVOID	Reference to the memory buffer from which the values to be written are to be taken.
szSize CAA_SIZE	Number of the bytes to be written

Outputs (VAR_OUTPUT)

Identifier/data type	Meaning/possible settings
xDone BOOL	• TRUE: An action has been executed successfully. • FALSE: No action active / action is still being executed.
xBusy BOOL	• TRUE: An action is currently being executed. • FALSE: No action active
xError BOOL	• TRUE: An error has occurred. • FALSE: No error
eErrorCode L_ETC_ERRORCODE	Error code of the incorrectly executed action (*xError* = TRUE). A detailed description of the EtherCAT error codes is given in the chapter "System error messages" (🕮 204).

10.6 Modular Machine Configuration Interface

The function blocks of the "Modular Machine Configuration Interface" can be used from release 3.10 and provide the following options:

- Assigning "Second Station Addresses" (alias addresses) to the EtherCAT slaves
- Controlling the EtherCAT master and the behaviour
- Controlling the EtherCAT bus

▶ Modular machine configuration (📖 108)

10.6.1 L_ETC_MMCAssignAddress (FB)

This function block assigns a "Second Station Address" (alias address) to an EtherCAT slave.

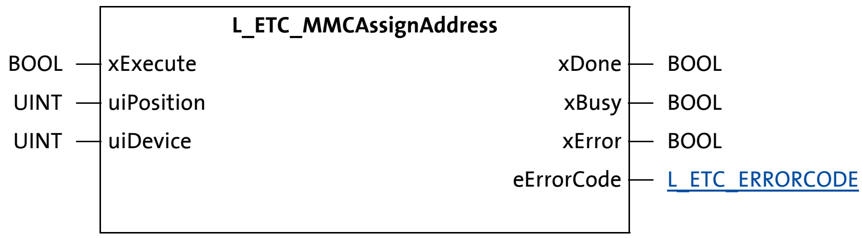
This function block only works if the function block L_ETC_MMCController (📖 164) or L_ETC_MMCControllerBus (📖 166) is operated with the ADDR_ASSIGNMENT_EXTERNALLY service. At the *szConfiguration* input, the current machine configuration that is really available at the EtherCAT bus needs to be selected.

Note!

Transfer of the "Second Station Address" (alias address)

- For transferring the alias address, the slave must be restarted by mains switching.
- In case of the **I/O module EPM-S130** from firmware version >1.28 and from release 3.12, the transfer of the alias address is executed by setting the index ’0x2359’ (subindex ’0’) = 1. An access is only possible in the "Pre-Operational" state.

Visualisation: VISU_L_ETC_MMCAssignAddress



Inputs (VAR_INPUT)

Identifier/data type	Meaning/possible settings
xExecute BOOL	A positive edge (TRUE) assigns the slave at the (*uiPosition*) position in the logical ring of the EtherCAT network to the *uiDevice* address.
uiPosition UINT	Position in the logical ring of the EtherCAT network The first slave behind the EtherCAT master has the position ’1’.
uiDevice UINT	"Second Station Address" (alias address) to be assigned The value is written into the EEPROM of the slave.

Outputs (VAR_OUTPUT)

Identifier/data type	Meaning/possible settings
xDone BOOL	• TRUE: An action has been executed successfully. • FALSE: No action active / action is still being executed.
xBusy BOOL	• TRUE: An action is currently being executed. • FALSE: No active action.
xError BOOL	• TRUE: An error has occurred. • FALSE: No error
eErrorCode L_ETC_ERRORCODE	Error code of the incorrectly executed action (*xError* = TRUE). A detailed description of the EtherCAT error codes is given in the chapter "System error messages" (📖 204).

If additional parameters have to be written or initialised for a slave, this is the responsibility of the user.

10.6.2 L_ETC_MMCController

> **Note!**
>
> This function block ...
> - may only be instanced once in the PLC program.
> - must be called in the context of the EtherCAT bus cycle task.

This function block control the EtherCAT master and its behaviour.

If an instance of the **L_ETC_MMCController** function block is created in the PLC program, the EtherCAT master waits before starting the EtherCAT bus. Only after a service/mode has been defined (*eService* input) and the configuration (*szConfiguration* input), the EtherCAT master can be started.

The EtherCAT master is started with *xRestart* = TRUE (see L_IODrvEtherCAT (FB) (📖 151)).

Visualisation: VISU_L_ETC_MMCController

	L_ETC_MMCController		
BOOL	xExecute	xDone	BOOL
BOOL	xAbort	xBusy	BOOL
L_ETC_MMCSERVICE	eService	xError	BOOL
STRING	szConfiguration	xAborted	BOOL
BOOL	xResetTopologyChangedCounter	eErrorCode	L_ETC_MMCSERVICEERROR
		eServiceState	L_ETC_MMCSERVICESTATE
		udiTopologyChangedCounter	UDINT

Inputs (VAR_INPUT)

Identifier/data type	Meaning/possible settings
xExecute BOOL	In case of a positive edge (TRUE), the data at the inputs *szConfiguration* and *eService* is accepted and the EtherCAT master is set to the mode given under *eService*.
xAbort BOOL	A positive edge (TRUE) stops the running service. **Note:** An abort can only be executed in the EtherCAT state "Init" (before, set *xStopBus* = TRUE, see L_IODrvEtherCAT (FB) (📖 151)).
eService L_ETC_MMCSERVICE	Service to be executed
szConfiguration STRING	Name/designator of the current machine configuration (file mmc-0-conf.csv). The designator may only contain these characters: [a..z], [A..Z], [0..9], und [.-].
xResetTopologyChanged Counter BOOL	A positive edge (TRUE) resets the counter for topology changes to 'zero' (*udiTopologyChangedCounter* output).

Outputs (VAR_OUTPUT)

Identifier/data type	Meaning/possible settings
xDone BOOL	• TRUE: An action has been executed successfully. • FALSE: No action active / action is still being executed.
xBusy BOOL	• TRUE: An action is currently being executed. • FALSE: No action active
xError BOOL	• TRUE: An error has occurred. • FALSE: No error
xAborted BOOL	When *xAborted* = TRUE, the requested service is aborted. **Note:** An abort can only be executed in the EtherCAT state "Init" (before, set *xStopBus* = TRUE, see L_IODrvEtherCAT (FB) (📖 151)).
eErrorCode L_ETC_MMCSERVICEERROR	Error code of the incorrectly executed service (xError = TRUE)
eServiceState L_ETC_MMCSERVICESTATE	Current status of the active service: • STOPPED: No active service. • RUNNING: Service is started and executed.
udiTopologyChanged Counter UDINT	Counter for topology changes • If the physical EtherCAT bus is changed and with an active service, the counter is incremented. • Between the change and the incrementation of the counter, a certain time elapses because if new slaves are added, they can be detected and initialised. • The counter is reset by the start of a service and by the xResetTopologyChangedCounter input.

Product features

Identifier/data type	Meaning/possible settings
GetActualConfiguration STRING	Current configuration under *szConfiguration* input If no configuration is active, an empty string is displayed.
GetRunningService L_ETC_MMCSERVICE	Currently running service

10.6.3 L_ETC_MMCControllerBus

Note!

This function block ...
- may only be instanced once in the PLC program.
- must be called in the context of the EtherCAT bus cycle task.

This function block is an extension for controlling the EtherCAT bus and contains the same inputs, outputs and features as L_ETC_MMCController (📖 164).

The following table shows the bus control as a function of the service:

Service	Final state when xExecute = TRUE	Final state when xAbort = TRUE
RUN_WITHOUT_CHECK RUN_OPTIONAL_SLAVES_ALLOWED RUN_OPTIONAL_SLAVES_PROHIBITED	Operational	Init
ADDR_ASSIGNMENT_EXTERNALLY	Pre-Operational 1)	Init
ADDR_ASSIGNMENT_CONFIG_SLAVEORDER ADDR_ASSIGNMENT_CONFIG_PARAMETER	Pre-Operational 1)	Init 2)

1) Due to a temporarily generated configuration, the master sets the slaves to the "Pre-Operational" state.

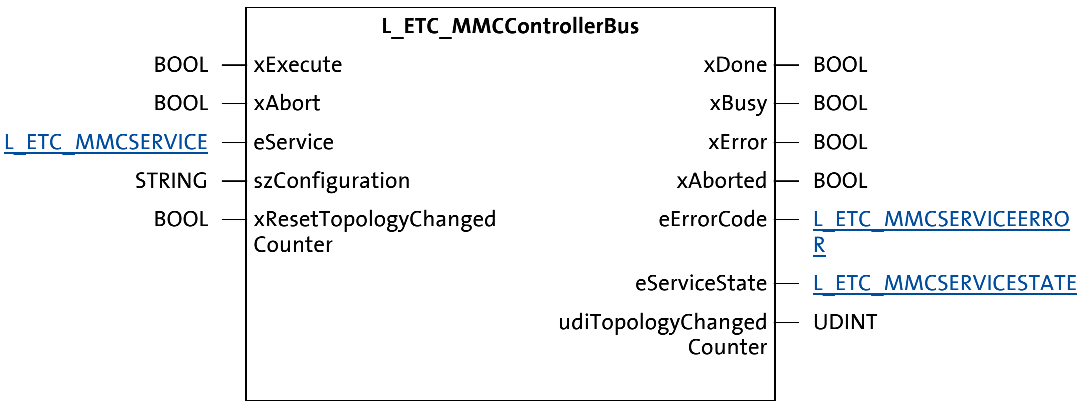
2) After the addresses have been automatically assigned, the master is set to the "Init" state again.

As soon as a RUN service is started, the EtherCAT bus is automatically set internally to the "Operational" state.

When a RUN service is started, the EtherCAT master is set to the "Operational" state.

When a RUN service is terminated, the EtherCAT master is set to the "Init" state.

Visualisation: VISU_L_ETC_MMCControllerBus



Inputs (VAR_INPUT)

Identifier/data type	Meaning/possible settings
xExecute BOOL	In case of a positive edge (TRUE), the data at the inputs *szConfiguration* and *eService* is accepted and the EtherCAT master is set to the mode given under *eService*.
xAbort BOOL	A positive edge (TRUE) stops the running service. **Note:** An abort can only be executed in the EtherCAT state "Init" (before, set *xStopBus* = TRUE, see L_IODrvEtherCAT (FB) (📖 151)).
eService L_ETC_MMCSERVICE	Service to be executed
szConfiguration STRING	Name/designator of the current machine configuration (file mmc-0-conf.csv). The designator may only contain these characters: [a..z], [A..Z], [0..9], und [.-].
xResetTopologyChanged Counter BOOL	A positive edge (TRUE) resets the counter for topology changes to 'zero' (*udiTopologyChangedCounter* output).

Outputs (VAR_OUTPUT)

Identifier/data type	Meaning/possible settings
xDone BOOL	• TRUE: An action has been executed successfully. • FALSE: No action active / action is still being executed.
xBusy BOOL	• TRUE: An action is currently being executed. • FALSE: No action active
xError BOOL	• TRUE: An error has occurred. • FALSE: No error
xAborted BOOL	When *xAborted* = TRUE, the requested service is aborted. **Note:** An abort can only be executed in the EtherCAT state "Init" (before, set *xStopBus* = TRUE, see L_IODrvEtherCAT (FB) (📖 151)).
eErrorCode L_ETC_MMCSERVICEERROR	Error code of the incorrectly executed service (xError = TRUE)
eServiceState L_ETC_MMCSERVICESTATE	Current status of the active service: • STOPPED: No active service. • RUNNING: Service is started and executed.
udiTopologyChanged Counter UDINT	Counter for topology changes • If the physical EtherCAT bus is changed and with an active service, the counter is incremented. • Between the change and the incrementation of the counter, a certain time elapses because if new slaves are added, they can be detected and initialised. • The counter is reset by the start of a service and by the xResetTopologyChangedCounter input.

Product features

Identifier/data type	Meaning/possible settings
GetActualConfiguration STRING	Current configuration under *szConfiguration* input If no configuration is active, an empty string is displayed.
GetRunningService L_ETC_MMCSERVICE	Currently running service

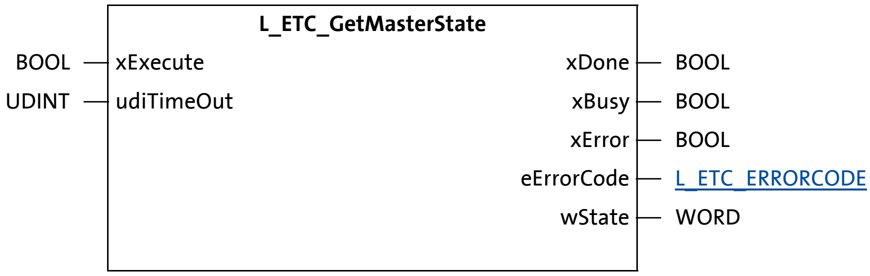
10.7 State Machine Interface

The "State Machine Interface" provides function blocks for setting and maintaining the state of master and slave.

10.7.1 L_ETC_GetMasterState

This function block detects the current state of the EtherCAT master.

Visualisation: VISU_L_ETC_GetMasterState



Inputs (VAR_INPUT)

Identifier/data type	Meaning/possible settings
xExecute BOOL	A positive edge (TRUE) detects the current EtherCAT master state.
udiTimeout UDINT	Timeout in milliseconds (ms) • The value '0' is not permissible and causes the error ETC_E_INVALIDPARM (0x9811000B).

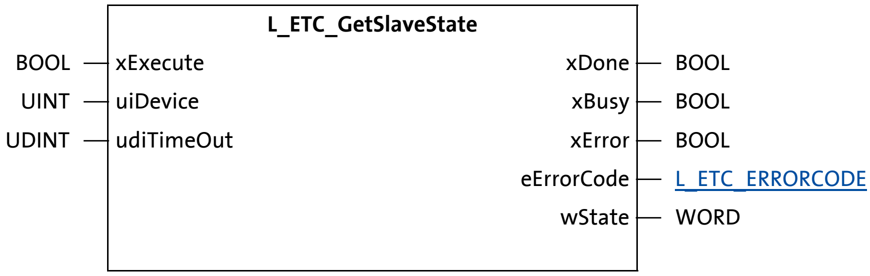
Outputs (VAR_OUTPUT)

Identifier/data type	Meaning/possible settings
xDone BOOL	• TRUE: An action has been executed successfully. • FALSE: No action active / action is still being executed.
xBusy BOOL	• TRUE: An action is currently being executed. • FALSE: No action active
xError BOOL	• TRUE: An error has occurred. • FALSE: No error
eErrorCode L_ETC_ERRORCODE	Error code of the incorrectly executed action (*xError* = TRUE). A detailed description of the EtherCAT error codes is given in the chapter "System error messages" (204).
wState WORD	Current master state of the L_ETC_STATE (178) type

10.7.2 L_ETC_GetSlaveState

This function block detects the current state of the EtherCAT slave.

Visualisation: VISU_L_ETC_GetSlaveState



Inputs (VAR_INPUT)

Identifier/data type	Meaning/possible settings
xExecute BOOL	A positive edge (TRUE) detects the current EtherCAT slave status of the slave given by *uiDevice*.
uiDevice UINT	EtherCAT address of the slave.
udiTimeout UDINT	Timeout in milliseconds (ms) • The value '0' is not permissible and causes the error ETC_E_INVALIDPARM (0x9811000B).

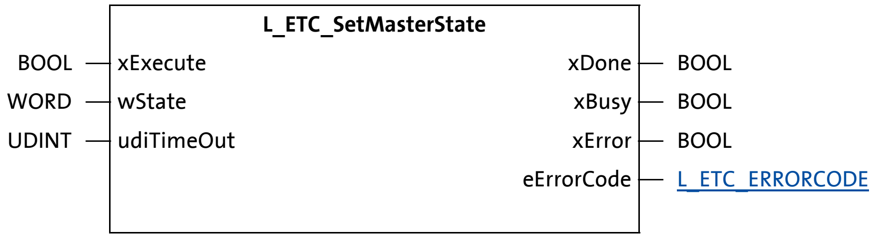
Outputs (VAR_OUTPUT)

Identifier/data type	Meaning/possible settings
xDone BOOL	• TRUE: An action has been executed successfully. • FALSE: No action active / action is still being executed.
xBusy BOOL	• TRUE: An action is currently being executed. • FALSE: No action active
xError BOOL	• TRUE: An error has occurred. • FALSE: No error
eErrorCode L_ETC_ERRORCODE	Error code of the incorrectly executed action (*xError* = TRUE). A detailed description of the EtherCAT error codes is given in the chapter "System error messages" (🕮 204).
wState WORD	Current slave state of the L_ETC_STATE (🕮 178) type

10.7.3 L_ETC_SetMasterState

This function block sets the state of the EtherCAT master.

Visualisation: VISU_L_ETC_SetMasterState



Inputs (VAR_INPUT)

Identifier/data type	Meaning/possible settings
xExecute BOOL	A positive edge (TRUE) sets the EtherCAT master state given by *wState*.
wState WORD	EtherCAT master state to be set of the L_ETC_STATE (📖 178) type
udiTimeout UDINT	Timeout in milliseconds (ms) • The value '0' is not permissible and causes the error ETC_E_INVALIDPARM (0x9811000B).

Outputs (VAR_OUTPUT)

Identifier/data type	Meaning/possible settings
xDone BOOL	• TRUE: An action has been executed successfully. • FALSE: No action active / action is still being executed.
xBusy BOOL	• TRUE: An action is currently being executed. • FALSE: No action active
xError BOOL	• TRUE: An error has occurred. • FALSE: No error
eErrorCode L_ETC_ERRORCODE	Error code of the incorrectly executed action (*xError* = TRUE). A detailed description of the EtherCAT error codes is given in the chapter "System error messages" (📖 204).

10.7.4 L_ETC_SetSlaveState

This function block sets the state of the EtherCAT slave.

Visualisation: VISU_L_ETC_SetSlaveState

Note!

The status of the slaves cannot be higher than the status of the EtherCAT master.

Example: If the master is in the Pre-Operational state, the state of a slave cannot be "Safe-Operational" or "Operational".

	L_ETC_SetSlaveState		
BOOL	xExecute	xDone	BOOL
UINT	uiDevice	xBusy	BOOL
WORD	wState	xError	BOOL
UDINT	udiTimeout	eErrorCode	L_ETC_ERRORCODE

Inputs (VAR_INPUT)

Identifier/data type	Meaning/possible settings
xExecute BOOL	A positive edge (TRUE) sets the EtherCAT slave state by the slave given by *uiDevice*.
uiDevice UINT	EtherCAT address of the slave.
wState WORD	EtherCAT slave state to be set of the L_ETC_STATE (178) type
udiTimeout UDINT	Timeout in milliseconds (ms) • The value '0' is not permissible and causes the error ETC_E_INVALIDPARM (0x9811000B).

Outputs (VAR_OUTPUT)

Identifier/data type	Meaning/possible settings
xDone BOOL	• TRUE: An action has been executed successfully. • FALSE: No action active / action is still being executed.
xBusy BOOL	• TRUE: An action is currently being executed. • FALSE: No action active
xError BOOL	• TRUE: An error has occurred. • FALSE: No error
eErrorCode L_ETC_ERRORCODE	Error code of the incorrectly executed action (*xError* = TRUE). A detailed description of the EtherCAT error codes is given in the chapter "System error messages" (204).

10.8 Data types

The Lenze-specific data types described in the following are used in some functions and function blocks of the L_IODrvEtherCAT function library (📖 119).

10.8.1 L_ETC_COE_EMERGENCY

The type **L_ETC_COE_EMERGENCY** describes errors that occur at a particular EtherCAT slave.

```
TYPE L_ETC_COE_EMERGENCY :
STRUCT
   uiDevice : UNIT;                // Slave address
   dwTimestamp : DATE_AND_TIME;    // Timestamp UTC (see FUNC SysTimeRtcGet())
   wErrorCode : WORD;                // Error code, see specification for details
   byErrorRegister : BYTE;           // Error register
   abyData : ARRAY[0..4] OF BYTE;    // Error data

END_STRUCT
END_TYPE
```

10.8.2 L_ETC_COE_EMERGENCY_BUFFER_DATA

The **L_ETC_COE_EMERGENCY_BUFFER_DATA** type describes a flag for displaying an emergency buffer overflow.

```
TYPE L_ETC_COE_EMERGENCY_BUFFER_DATA :
STRUCT
   structEmergency : L_ETC_COE_EMERGENCY; // Struct Emergency
   bBufferOverflow : BOOL;                // Flag for emergency buffer overflow

END_STRUCT
END_TYPE
```

10.8.3 L_ETC_COE_FLAGS

The **L_ETC_COE_FLAGS** type controls the behaviour of the function blocks L_ETC_CoE_SdoReadEx (FB) (📖 138) and L_ETC_CoE_SdoWriteEx (FB) (📖 144).

```
TYPE L_ETC_COE_FLAGS :
(
   ETC_E_COMPLETEACCESS := 16#00000001
)  DWORD;
END_TYPE
```

10.8.4 L_ETC_DIAGNOSTIC

The **L_ETC_DIAGNOSTIC** type describes diagnostic information which is returned e.g. via the L_ETC_GetMasterDiagnostic (FB) (📖 156) function block.

A detailed description of the EtherCAT error codes is given in the chapter "System error messages" (📖 204).

```
TYPE L_ETC_DIAGNOSTIC :
STRUCT
   wState : L_ETC_STATE;                       // Master state
   stState : STRING;                           // Master state as text

   xEthernetLinkup : BOOL;                     // Link up
   xNotAllSlavesOperational : BOOL;           // Indicates that not all slaves set in
                                                  operational

   udiFrameCounter : UDINT;                    // Ethernet frame counter
   udiFrameResponseErrorCounter : UDINT;       // Frame loss counter
   udiCycWorkingCounterErrorCounter : UDINT;   // WKC error counter

   xBusMismatch : BOOL;                       // Configuration and topology mismatch
   uiNumberOfSlavesFound : UINT;               // Number of found slaves
   uiNumberOfSlavesConfigured : UINT;          // Number of slaves configured

   dwLastErrorCode : DWORD;                    // Last error
   stLastError : STRING;                       // Last error as string

   xDC_Enabled : BOOL;                         // DC in use
   xDC_InSync : BOOL;                          // DC all slaves are in-sync
   xDC_Busy : BOOL;                            // DC re-sync running
   diDC_CurrentDeviation : DINT;               // DC current deviation

   udiEmergencyNo : UDINT;                     // COE Emergency Counter

END_STRUCT
END_TYPE
```

10.8.5 L_ETC_ERRORCODE

The type **L_ETC_ERRORCODE** describes all possible EtherCAT error codes that can be output at the *eErrorCode* output of the EtherCAT function blocks.

A detailed description of the EtherCAT error codes is given in the chapter "System error messages" (📖 204).

10.8.6 L_ETC_EVTPARAM_PARAMETERTRANSFER

The type **L_ETC_EVTPARAM_PARAMETERTRANSFER** describes parameter transfer to a particular EtherCAT node or to all EtherCAT nodes.

(See also L_ETC_PARAMETERTRANSFERSERVICE_CODE (📖 177))

```
TYPE L_ETC_EVTPARAM_PARAMETERTRANSFER :
   (wService,   // Parameter transfer service
                // (start: 16#0001, state: 16#0002, done: 16#0004, abort: 16#8000)
    uiDevice    // EtherCAT address (All devices: 16#FFFF)
   ) WORD;

   (dwResult    // Local parameter transfer in progress (wired OR)
   )  DWORD;
END_TYPE
```

10.8.7 L_ETC_IOCTLOPARMS

The **L_ETC_IOCTLOPARMS** type describes the reference to the I/O buffer e.g. for the L_ETC_IoControl (FUN) (📖 150) function.

```
TYPE L_ETC_IOCTLOPARMS :
STRUCT
   pbyInBuf : POINTER TO BYTE;        // Input data buffer
   dwInBufSize : DWORD;               // Size of input data buffer in bytes
   pbyOutBuf : POINTER TO BYTE;       // Output data buffer
   dwOutBufSize : DWORD;              // Size of output data buffer in bytes
   pdwNumOutData : POINTER TO DWORD;  // Number of output data bytes stored in
                                      // output data buffer
END_STRUCT
END_TYPE
```

10.8.8 L_ETC_LANGUAGE

The **L_ETC_LANGUAGE** type describes the languages in which e.g. an error text can be output via the L_ETC_GetErrorString (FUN) (📖 155) function.

```
TYPE L_ETC_LANGUAGE :
(
   eng,    // english
   de      // german
);
END_TYPE
```

10.8.9 L_ETC_MMCSERVICE

The **L_ETC_MMCSERVICE** type contains the potential services for the function blocks L_ETC_MMCController (164) and L_ETC_MMCControllerBus (166).

Principally, there are two types of services:

- RUN services for the operating mode
- ADDR_ASSIGNMENT services for the automatic and manual address allocation

Please note: Dependencies between configuration files and services (113)

No. [hex]	Identifier/data type	Meaning/possible settings
0x0000	NONE	No operation (NOP)
0x0001	RUN_WITHOUT_CHECK	When the EtherCAT bus is started, no check for mandatory or optional slaves and the additional slave identification is carried out. The prerequisite for this is the address assignment to the slaves. Once started successfully, this service runs until a positive edge is applied to the xAbort input.
0x0002	RUN_OPTIONAL_SLAVES_ALLOWED	When the bus is started, a check for mandatory or optional slaves is carried out. The EtherCAT bus changes to the "Pre-Operational" state and more if ... • all mandatory slaves are available; • no or several optional slaves are available; • the additional identification for all available slaves is correct. The prerequisite for this is the address assignment to the slaves. Once started successfully, this service runs until a positive edge is applied to the xAbort input.
0x0003	RUN_OPTIONAL_SLAVES_PROHIBITED	When the bus is started, a check for mandatory slaves is carried out. The EtherCAT bus changes to the "Pre-Operational" state and more if ... • all mandatory slaves are available; • no optional slave is available; • the additional identification for the mandatory slaves is correct. The prerequisite for this is the address assignment to the slaves. Once started successfully, this service runs until a positive edge is applied to the xAbort input.
0x0100	ADDR_ASSIGNMENT_EXTERNALLY	Address allocation via an external tool or from the PLC application: • The EtherCAT master is set to the "Pre-Operational" state based on the slave EEPROM information. • An access via an external tool or from the PLC application can be carried out. • The addresses can be assigned via the L_ETC_MMCAssignAddress (FB) (163) function block. • CoE objects can be accessed in read-only mode. Once started successfully, this service runs until a positive edge is applied to the xAbort input.

No. [hex]	Identifier/data type	Meaning/possible settings
0x0200	ADDR_ASSIGNMENT_CONFIG_SLAVEORDER	Automatic address allocation via the configuration: • The EtherCAT master is set to the "Pre-Operational" state based on the slave EEPROM information. • If the order of the slaves at the EtherCAT complies with the selected configuration (vendor-ID and product code), the persistent addresses are written to the EEPROM. Once started successfully, this service runs until the service has been completed successfully (xDone = TRUE) or an error has occurred (xError = TRUE).
0x0300	ADDR_ASSIGNMENT_CONFIG_PARAMETER	Is currently not supported.

10.8.10 L_ETC_MMCSERVICEERROR

The **L_ETC_MMCSERVICEERROR** type describes the error that may be provided at the *eErrorCode* output of the function blocks L_ETC_MMCController (📖 164) and L_ETC_MMCControllerBus (📖 166).

More error descriptions regarding modular machine configuration can be found in the "Diagnostics" chapter:

▸ Error messages for modular machine configuration (📖 211)

No. [hex]	Identifier/data type	Meaning/possible settings
0x00000000	NO_ERROR	No error
0x00000001	CONFIG_INVALID	The selected configuration is not valid.
0x00000002	CONFIG_FILE_ERROR	• The **mmc-0-conf.csv** configuration file is not available. • An error has occurred during data import (Parsing Error).
0x00000003	IDENT_FILE_ERROR	No configuration file (mmc-0-conf-csv) is available or an error has occurred during data import (Parsing Error)
0x00000100	SERVICE_INVALID	The selected service (L_ETC_MMCSERVICE (📖 175)) is not valid.
0x00000101	SERVICE_INVALID_STATE	In the current status of the EtherCAT master, the selected service cannot be executed. The EtherCAT master has to be set to the "Init" state.
0x00010000	ADDR_INTERNAL_ERROR	When the service has been started, an internal error has occurred. Please contact Lenze!
0x00010001	ADDR_NO_MEMORY	When the service has been started, an internal memory error has occurred. Please contact Lenze!
0x00010002	ADDR_LESS_CONNECTED	In the address assignment, less slaves are connected to the physical EtherCAT bus than given in the active configuration.
0x00010003	ADDR_MORE_CONNECTED	In the address assignment, more slaves are connected to the physical etherCAT bus than given in the active configuration.
0x00010004	ADDR_UNEXPECTED_DEVICE	In the address assignment, the physical EtherCAT bus and the configuration provide the same number of slaves but at least one slave comes with a different device type. (For more information, see the logbook).
0x00010005	ADDR_ASSIGN_ERROR	An error has occurred during the address assignment. (For more information, see the logbook).

10.8.11 L_ETC_MMCSERVICESTATE

The **L_ETC_MMCSERVICESTATE** type displays the status of a service (L_ETC_MMCSERVICE (📖 175)) during processing.

```
TYPE L_ETC_MMCSERVICESTATE :
(
    STOPPED,   // Service was stopped
    RUNNING,   // Service was started and is running
    NOT_READY  // Function block is not ready
) WORD;
END_TYPE
```

No. [hex]	Identifier/data type	Meaning/possible settings
0x0000	STOPPED	Currently, no service is started.
0x0001	Running	Current service has been started and running.
0x0002	NOT_READY	The function block is not ready at the moment. Especially during the initialisation phase when the PLC program has been started but the MMC files have not been imported yet. If the NOT_READY state has been set, the inputs are ignored and no actions can be executed.

10.8.12 L_ETC_PARAMETERTRANSFERSERVICE_CODE

The type **L_ETC_PARAMETERTRANSFERSERVICE_CODE** describes all possible services or actions for parameter data transfer.

(See also L_ETC_EVTPARAM_PARAMETERTRANSFER (📖 174))

```
TYPE L_ETC_PARAMETERTRANSFERSERVICE_CODE :
   (START := 16#0001,   // Start of service
    STATE := 16#0002,   // State polling
    DONE  := 16#0004,   // End of service
    ABORT := 16#8000,   // Abort of service
   ) WORD;
END_TYPE
```

10.8.13 L_ETC_SLAVE_PORTS

The type **L_ETC_SLAVE_PORTS** describes ports 0 ... 3 of the EtherCAT slaves.

```
TYPE L_ETC_SLAVE_PORTS :
STRUCT
   Port0 : BYTE;    // Slave port 0
   Port1 : BYTE;    // Slave port 1
   Port2 : BYTE;    // Slave port 2
   Port3 : BYTE;    // Slave port 3

END_STRUCT
END_TYPE
```

10.8.14 L_ETC_STATE

The **L_ETC_STATE** describes all possible states of an EtherCAT master and the EtherCAT slaves.

```
TYPE L_ETC_STATE :                              // EtherCAT states
(
    ETC_STATE_NONE             := 16#0000,   // Unknown state
    ETC_STATE_INIT             := 16#0001,   // INIT state
    ETC_STATE_PREOPERATIONAL   := 16#0002,   // Pre-Operational state
    ETC_STATE_BOOT             := 16#0003,   // Bootstrap state
    ETC_STATE_SAVEOPERATIONAL := 16#0004,   // Safe-Operational state
    ETC_STATE_OPERATIONAL      := 16#0008,   // Operational state
    ETC_STATE_UNKNOWN          := 16#FFEF    // Unknown state
) WORD;
END_TYPE
```

ETC_STATE_NONE is available for the Modular machine configuration (🕮 108).

ETC_STATE_UNKNOWN is currently not available for Lenze devices.

11 Restarting the EtherCAT fieldbus

During operation it may be required to restart the EtherCAT fieldbus. This can for instance be necessary after fatal faults like a cable break.

How to restart the EtherCAT fieldbus:

1. Activate the controller inhibit for the inverters.
2. Request restart of the EtherCAT master.

   In the L_IODrvEtherCAT (FB) (📖 151)/L_IODrvEtherCAT_Diag (FB) (📖 152) function block from the L_IODrvEtherCAT function library (📖 119), set input **xRestart = TRUE**:

   ```
   Ethercat_Master.xRestart := TRUE;
   ```

   When "FAST Motion" function blocks are used:

   - Execute the **L_MC1P_ReinitNode** function block from the **L_MC1P_MotionControlBasic** function library:
     `input bExecute := TRUE;`

   When "L-force Motion" function blocks are used:

   - Execute the **SMC3_ReInitDrive** function block from the **SM3_Basic** function library:
     `input bExecute := TRUE;`
   - Execute the **MC_Reset** function block from the **SM3_Basic** function library:
     `input bExecute := TRUE;`

Note!

Use of "FAST Motion" function blocks

When the EtherCAT bus is restarted, the control mode of the Motion drives is set to the "PosCtrlDrive" mode.

If a different control mode was active before the restart, change the control mode by means of the **L_MC1P_SetControlMode** function block.

Use of "L-force Motion" function blocks

When the EtherCAT bus is restarted, the control mode of the Motion drive is set to the "SMC_position" mode or in case of the Servo Inverter i700 to "CSP: Cyclic Synchronous Position".

If a different control mode was active before the restart, change the control mode by means of the **SMC_SetControllerMode** function block.

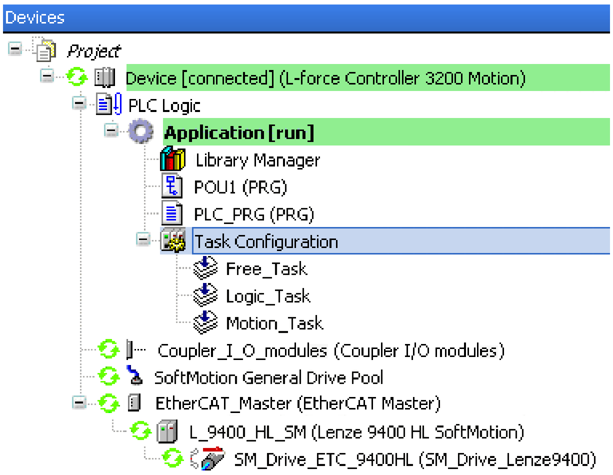
12 Defining the cycle time of the PLC project

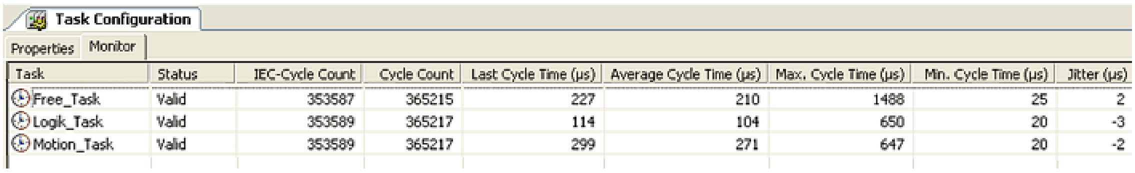
In this chapter you'll learn how to ...

- die Determine the task utilisation of the application (📖 180);
- das Optimising the system (📖 182).

12.1 Determine the task utilisation of the application

In the online mode, the **Monitor** tab of the **Task Configuration** shows current status details and measurements of the cycles, cycle times, and jitters of the tasks contained.





Task	Status	IEC-Cycle Count	Cycle Count	Last Cycle Time (µs)	Average Cycle Time (µs)	Max. Cycle Time (µs)	Min. Cycle Time (µs)	Jitter (µs)
Free_Task	Valid	353587	365215	227	210	1488	25	2
Logic_Task	Valid	353589	365217	114	104	650	20	-3
Motion_Task	Valid	353589	365217	299	271	647	20	-2

The values are updated in the same time interval as that used for monitoring the values from the controller.

If the cursor is on a task name field, the values displayed can be reset to 0 by the **Reset** context menu command (right-click the task name field).

How to determine the task utilisation:

Initial situation: A complete project, e.g. with a EtherCAT task and 2 lower priority tasks has been created.

1. For a first measurement of the task utilisation, set the cycle times of all cyclic tasks available in the PLC system "high" (e.g. EtherCAT task = 10 ms, all other cyclic tasks = 20 ms).
2. Use the menu command **Online → Login**, or log in on the Lenze Controller with **<Alt>+<F8>**.
   - For this, the PLC program must be error-free.
   - With the log-in, the fieldbus configuration and the PLC program are loaded to the controller.
3. Reset the values displayed on the **Monitor** tab of the **Task Configuration** to 0 after the complete run-up of the system.

   Execute the **Reset** command from the context menu of the task name field.
4. Read the displayed maximum computing time of the task with the highest priority.

   In the example above, the max. cycle time of the EtherCAT task is 647 µs.

The minimum cycle time (T_{min}) for a system can be calculated by means of the formula:

$$T_{min} = \text{Task utilisation x safety factor}$$

Note!

A safety factor of 1.5 should be included in the calculation.

12.2 Optimising the system

How to optimise the system:

1. Use the menu command **Online → Login**, or log in on the Lenze Controller with **<Alt>+<F8>**.
   - For this, the PLC program must be error-free.
   - With the log-in, the fieldbus configuration and the PLC program are loaded to the controller.
2. Check the task processing times.
3. Optimising the cycle times:
   - If technologically required, the cycle times of the remaining tasks with lower priorities can be decreased.
   - Condition: No task with a low priority must assign more than 60 percent of the corresponding cycle time in its task utilisation.

13 Diagnostics

This chapter provides information on diagnostics using the »PLC Designer« and the »WebConfig«. Moreover, error scenarios for the most frequent user errors are shown and system error message are described.

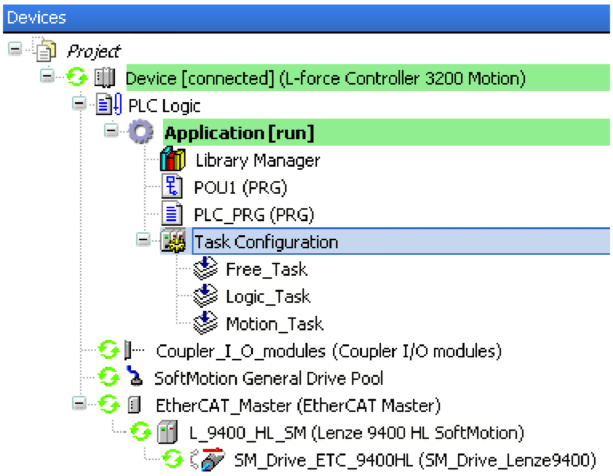
13.1 Diagnostics in the »PLC Designer«

A test project in which the diagnostic functions of the L_IODrvEtherCAT function library (📖 119) are used can be found in the Download area at www.Lenze.com:

Application Knowledge Base: All articles → Application Ideas Pool → Controller 3200 C

13.1.1 Representation in the online mode

When an online connection has been established to the Lenze Controller, the icon in front of the individual entries in the configuration tree provides information on the status of the respective EtherCAT node (in the example: two green arrows in each case):

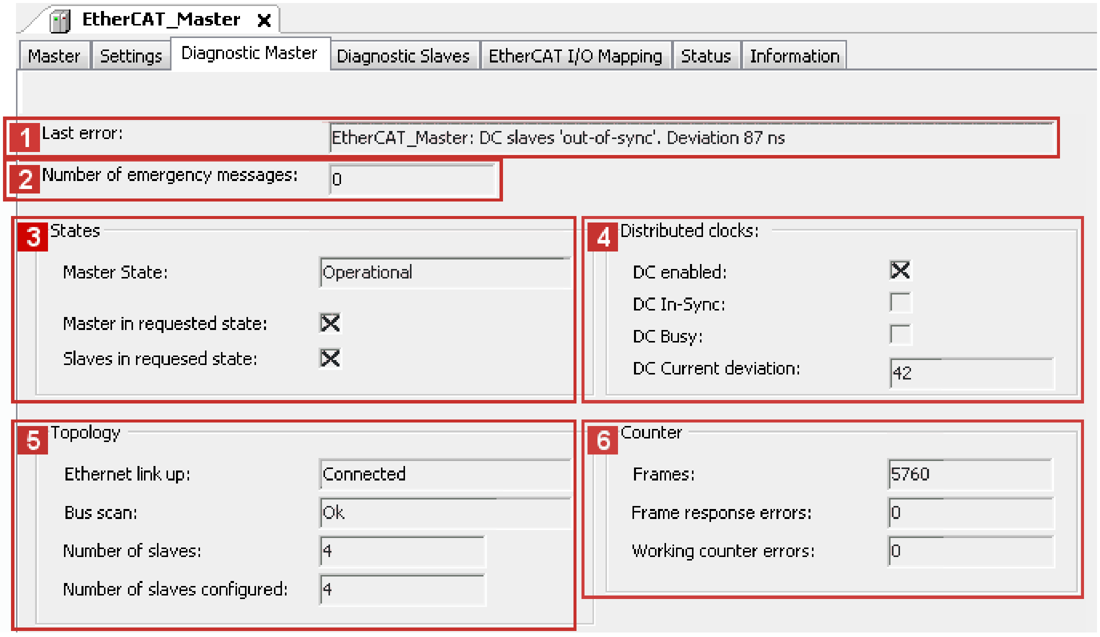


Symbol	Meaning
	Device is online • Successful online connection to the Lenze Controller • Successful online connection to the EtherCAT node. • Status of the node: Operational (OP)
	Device is online Possible states of the EtherCAT node: • INIT (initialisation) • PREOP (Pre-operational) • SAFEOP (Safe-operational) • No information on the node
no icon	Device is offline • No connection to the Lenze Controller

13.1.2 Diagnostic tabs of the EtherCAT master

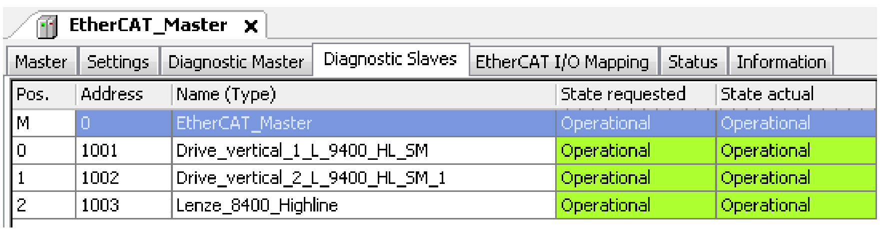
Only if an online connection to the Lenze Controller has been established, information is displayed in the diagnostic tabs.

The following information is displayed on the **Diagnostic Master** tab:



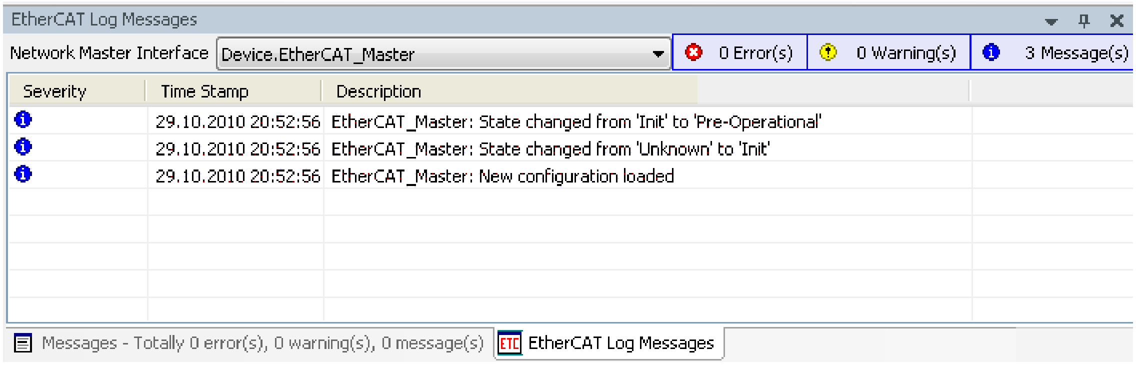
- 1 Error occurred last
- 2 Number of emergency frames
- 3 Status information
- 4 Information on "Distributed clocks"
- 5 Information on network topology
- 6 Frame and error counter

In addition to the EtherCAT states, the **Diagnostic Slaves** tab also displays the positions, addresses, and names of the EtherCAT nodes:



13.1.3 Display window for EtherCAT logbook messages

If you execute the menu command **View → EtherCAT log messages**, a window is opened in which Lenze Controller logbook messages (🕮 213) are displayed.



Column	Description
Severity	Message type: error, warning, information
Time stamp	Date and time of occurrence of a message
Description	For message text, see Lenze Controller logbook messages (🕮 213)

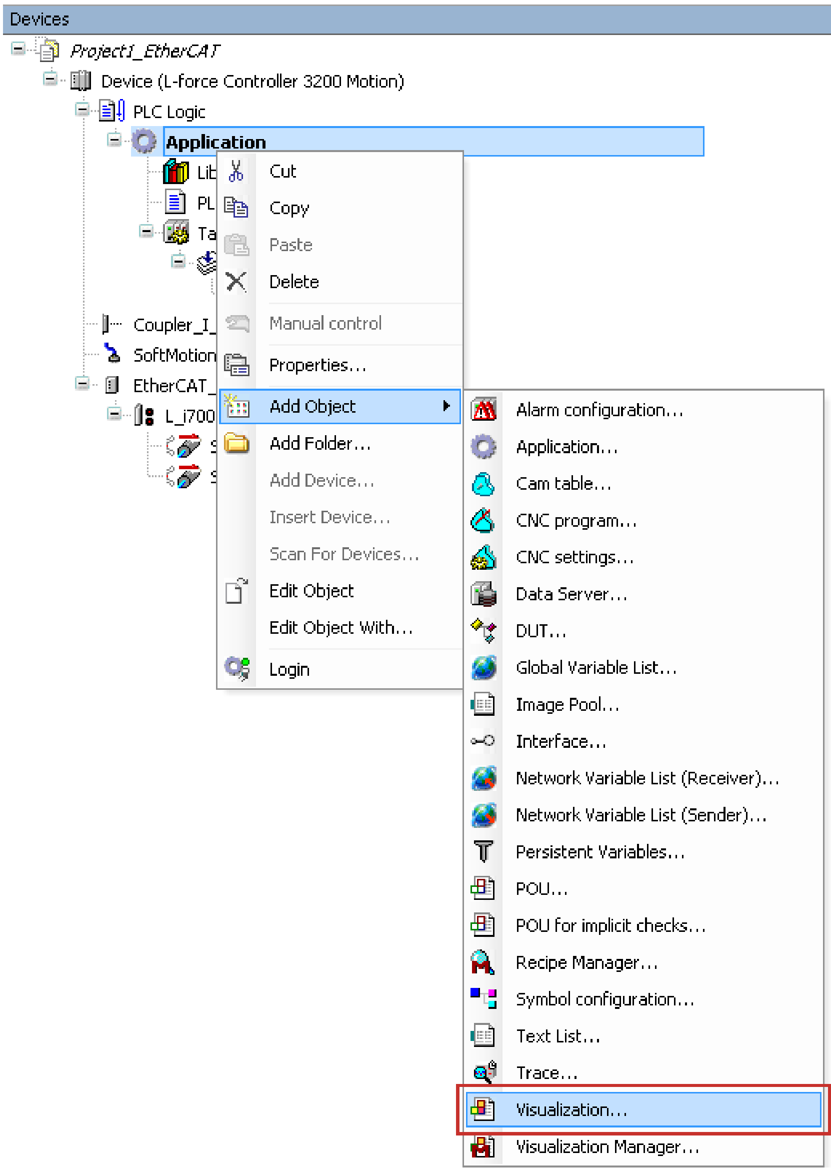
13.1.4 Visualisation of the function block L_ETC_GetMasterDiagnostic

In the visualisation of the L_ETC_GetMasterDiagnostic (FB) (📖 156) function block, variables for EtherCAT and DC states, (error) counters, error numbers etc. are displayed for diagnostic purposes.

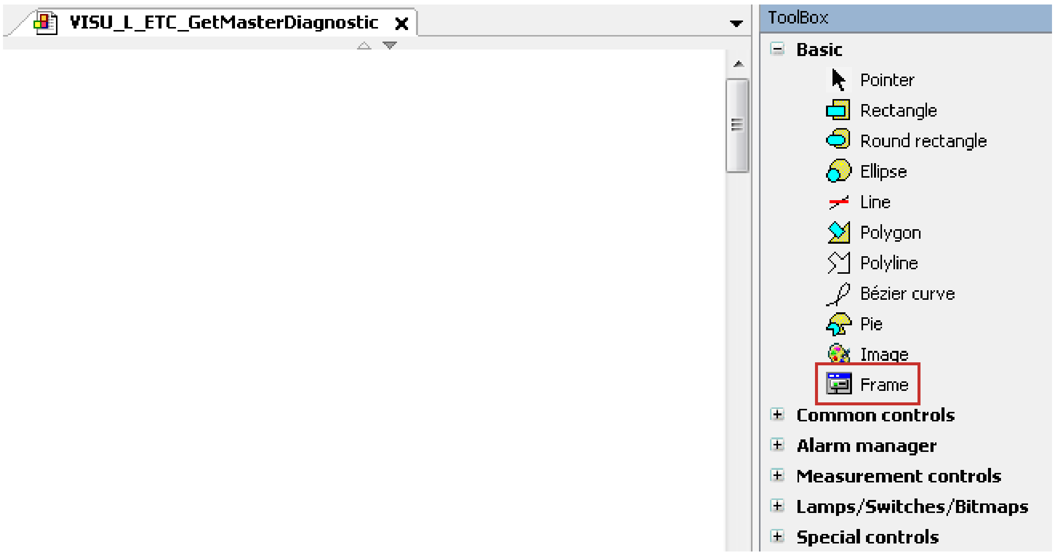
How to create the visualisation of the L_ETC_GetMasterDiagnostic function block

1. Open the context menu for **Application**, select the command **Add object → Visualisation...** and insert the visualisation of the function block **L_ETC_GetMasterDiagnostic**.

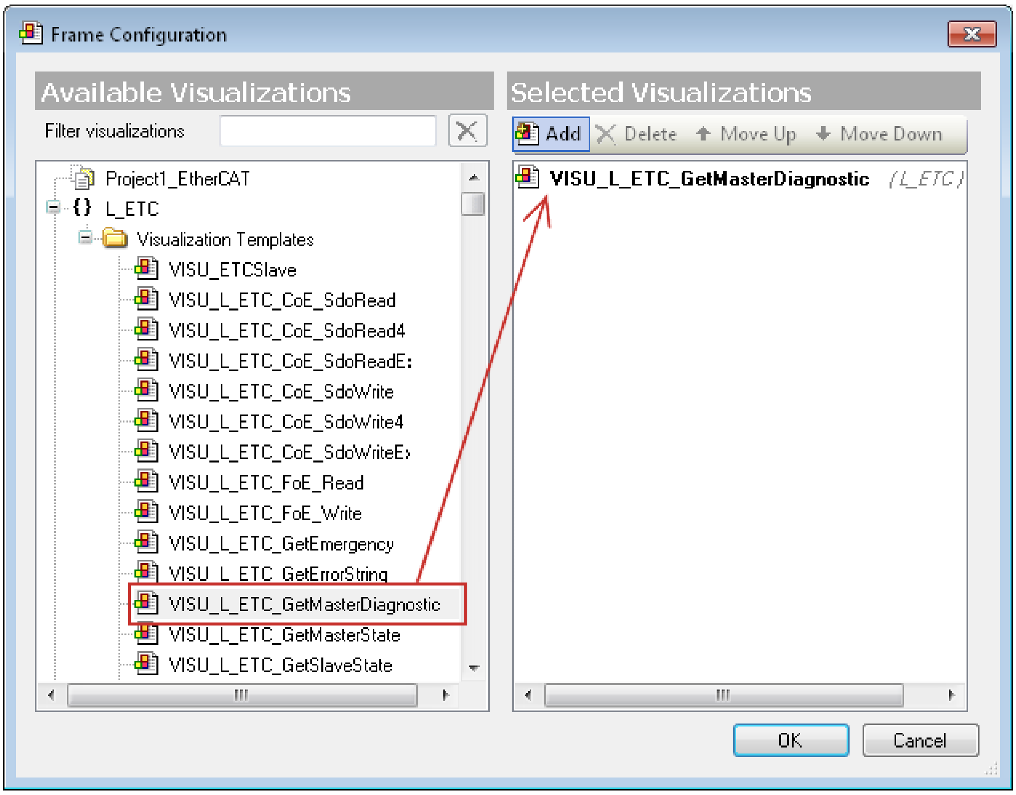
   Enter an appropriate name (e.g. "VISU_L_ETC_GetMasterDiagnostic").



2. Insert a frame in the visualisation with the help of the **frame tool**.



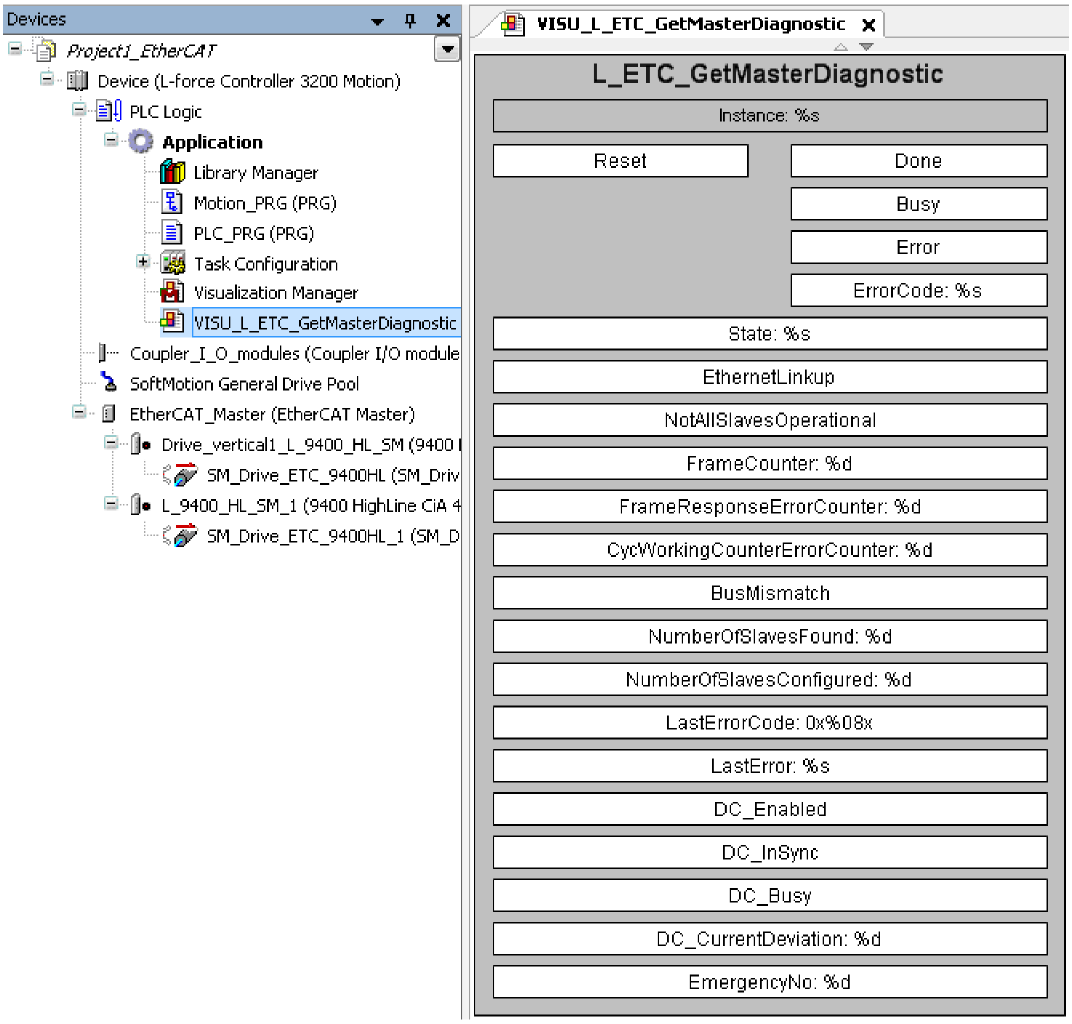
3. Add the frame visualisation of the function block **L_ETC_GetMasterDiagnostic** and close the dialog box by clicking the **OK** button.



The visualisation is added to the configuration tree of the »PLC Designer« project.

In general, all fields in the visualisation are initially white. If a status variable or a state is set or active (TRUE), the corresponding field is shown in green or red:

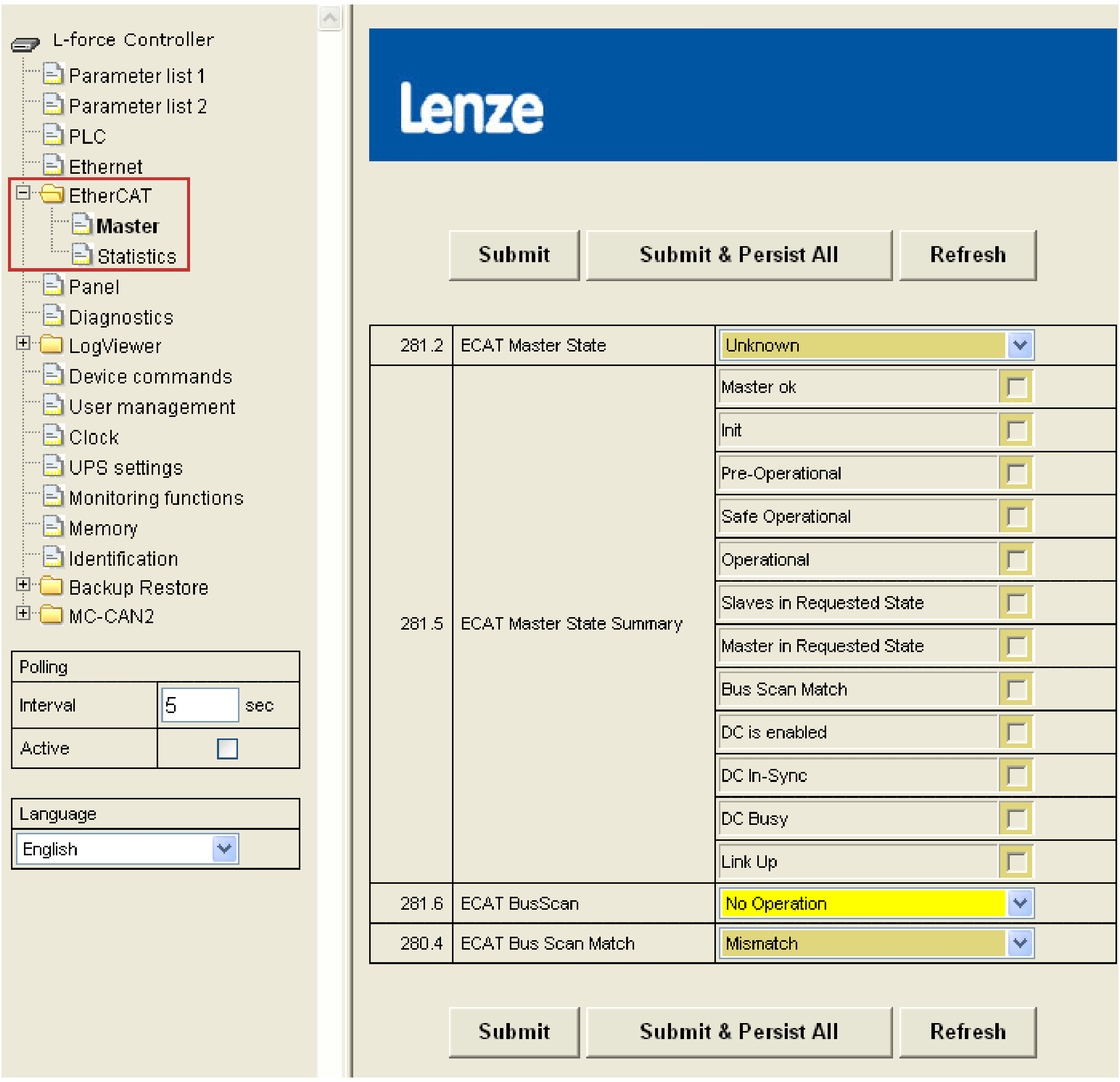
- Red fields represents an "error".
- Green fields display "information".



13.2 Diagnostic codes in the »WebConfig«

In »WebConfig«, you can view the EtherCAT diagnostic parameters under **EtherCAT → Master** and **EtherCAT → Statistics**.

▸ Parameter reference (📖 219)

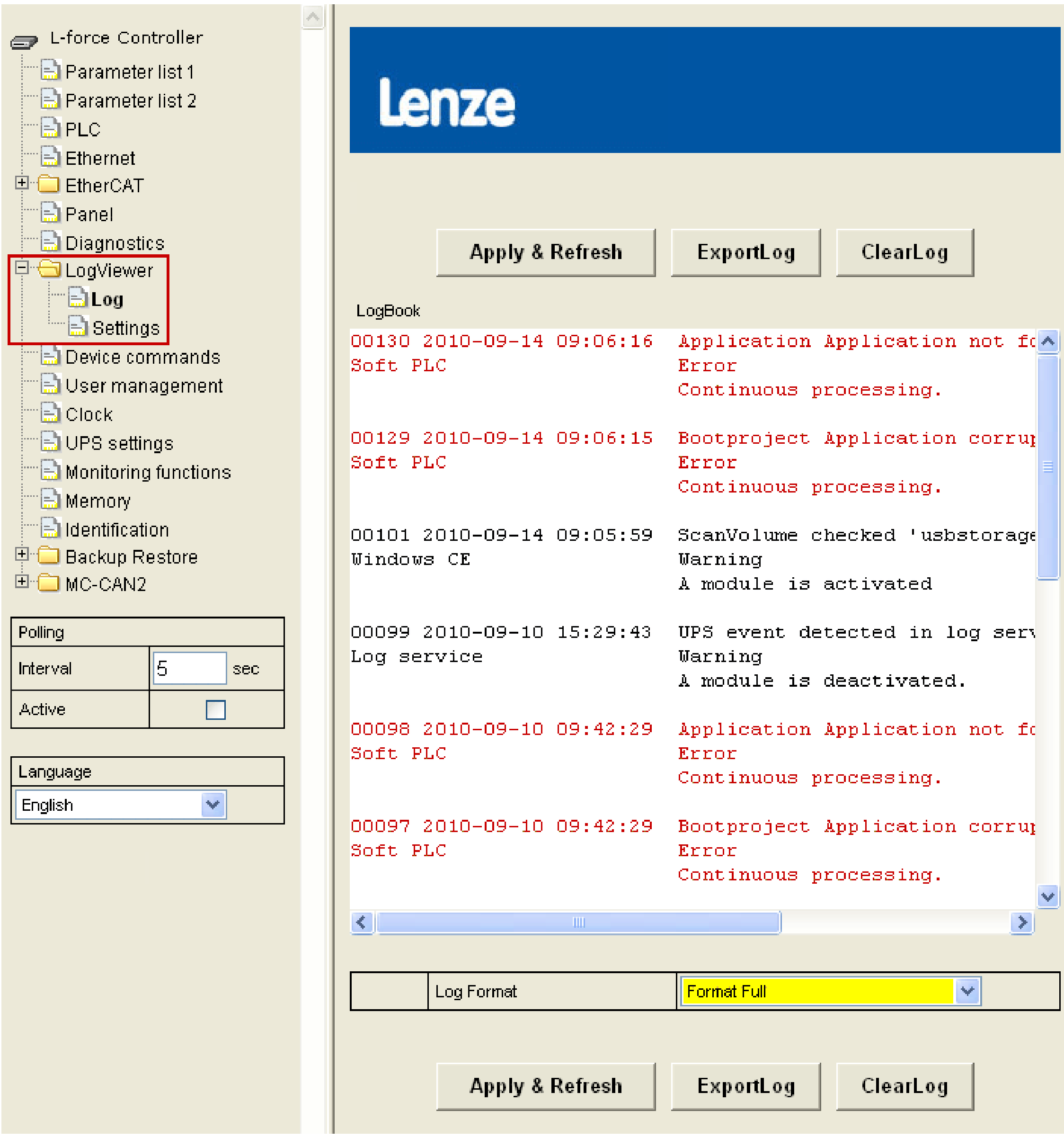


13.3 Logbook of the Lenze Controller in the »WebConfig«

The **logbook** of the »WebConfig« displays errors (highlighted in red), warnings, or information.

Read the messages in the logbook from bottom to top. The most recent message always appears at the top of the logbook.

▸ Lenze Controller logbook messages (📖 213)



Structure of the messages in the logbook:

• Running number • Date/time of occurrence • Location of occurrence	• Error description • Message type: error / warning / information • Action which caused the message

Settings for a compact representation of the messages:

1. Under **Logbook → Settings** in the **Application** area, set a checkmark in the checkbox for **EtherCAT Master Stack**.
2. Under **Logbook → Log** for **Log Format**, select "Format severity".

   Compact representation of the messages:

```
00130 F  Application Application not found to start
00129 F  Bootproject Application corrupt. CRC File is not valid
00101 W  ScanVolume überprüfte die 'usbstorage'-Partition und behob alle Fehler.
00099 W  Log-Service hat USV-Ereignis erkannt
00098 F  Application Application not found to start
00097 F  Bootproject Application corrupt. CRC File is not valid
00069 W  ScanVolume überprüfte die 'usbstorage'-Partition und behob alle Fehler.
00067 W  Log-Service hat USV-Ereignis erkannt
00066 F  Application Application not found to start
00065 F  Bootproject Application corrupt. CRC File is not valid
00035 W  Korrupte Restoredaten im Log-Service
00034 W  Log-Service hat USV-Ereignis erkannt
00033 F  Application Application not found to start
00032 F  Bootproject Application corrupt. CRC File is not valid
```

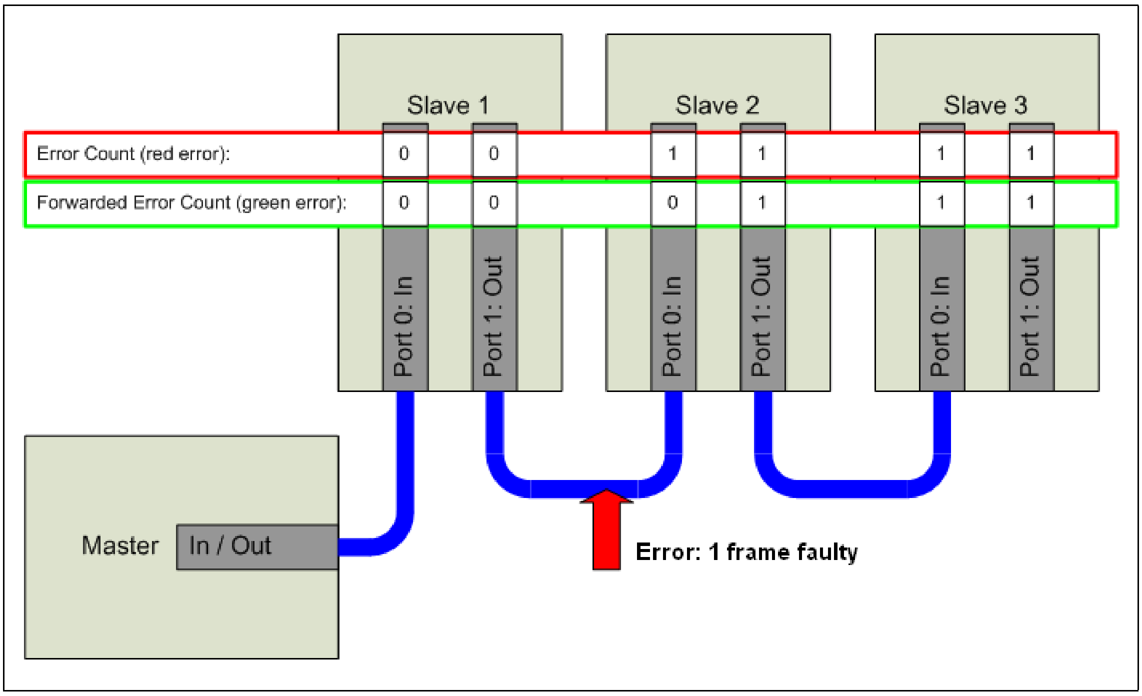
13.4 Error counters of the EtherCAT slaves

The EtherCAT slaves have numerical error counters for detecting and analysing error states. All error counters have a limited counting range of 0 ... 255. After the maximum value of 255 is reached, no "wrap-around" takes place. If the PLC application is to make a sensible evaluation of the error counters, these must be deleted by the application after the evaluation via a write access.

13.4.1 Error types "Errors" and "Forwarded Errors"

The EtherCAT slaves differ between errors detected in the slave for the first time (red error) and forwarded errors, i.e. errors that have already been detected in a previous slave (green error).

When the corresponding error counters are evaluated, an error in the EtherCAT network can be clearly assigned to a bus segment or a slave.



[13-1] Error types "Errors" and "Forwarded Errors"

13.4.2 Error counter reset from the application

The L_ETC_ReadErrCnt (FB) (157) function block enables the PLC application to access the error counters by reading.

The L_ETC_ResetErrCnt (FB) (158) function block resets the error counters to the value '0'.

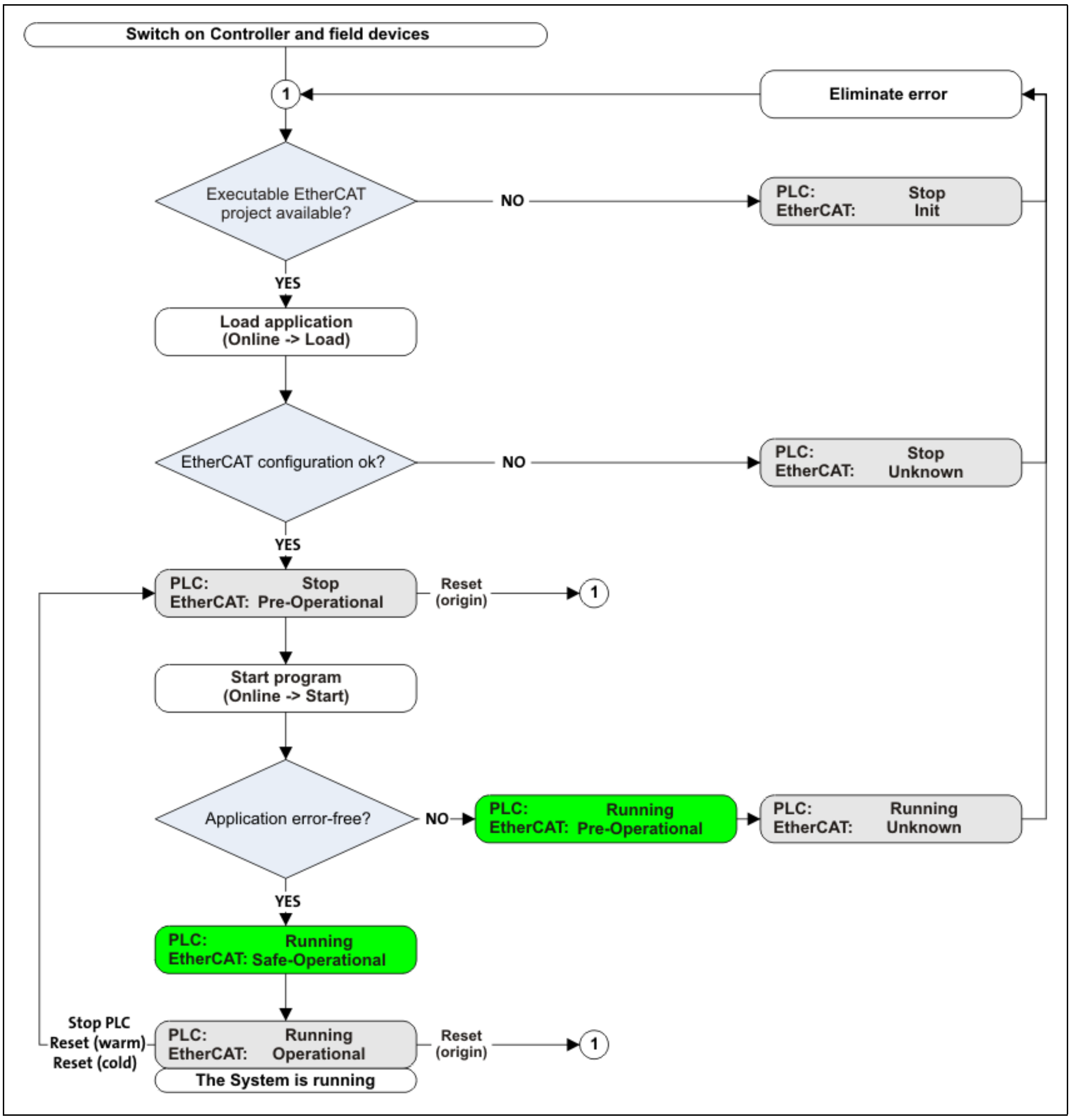
Example

Once per minute the PLC reads the error counters and evaluates the contents. If it recognises an error counter value of '250', the application will reset the error counters. Depending on the EMC load of the network environment, approx. 1 to 2 frame errors per day in the fieldbus are normal. The error counters should not reach the value of 255.

13.5 Error scenarios

In the following sections, the causes and remedies for the most frequent user errors are described.

The state diagram and descriptions on the next pages serve to localise and remove an error.



[13-2] Status diagram for system behaviour

13.5.1 The EtherCAT bus does not assume the "Pre-Operational" state.

During the start-up of the EtherCAT bus, a check is carried out at the transition from "Init" to "Pre-Operational" to determine whether the physical bus configuration corresponds to the bus configuration configured. If theses configurations are different, the master does not enter the "Pre-Operational" state.

Furthermore, the slaves are initialised during the transition from "Init" to "Pre-Operational". If this fails because, for instance, a slave denies the configuration, the master does not enter the "Pre-Operational" state.

Cause	**Faulty bus configuration and bus structure** • Slaves are missing. • Slaves have been inverted or their configuration is faulty. • A wrong slave type has been configured or is at the fieldbus. • The inputs and outputs of the EtherCAT communication module are reversed (IN/OUT connections). • Faulty wiring

Error message	`EtherCAT_Master: Start master failed. Bus mismatch`
	`...`
	`[DeviceName] (1002): Configuration mismatch. Check VendorID failed (0x3B / missing )`

Remedies	Correct the bus configuration or the physical bus structure. Afterwards, reload the »PLC Designer« application into the automation system.

Cause	**Wiring error: EtherCAT cable is not connected to the master.**
Error message	`EtherCAT_Master: Start master failed. EtherCAT cable disconnected`
Remedies	Correct wiring. Afterwards, reload the »PLC Designer« application into the automation system.

13.5.2 The EtherCAT bus does not assume the "Operational" state

The Lenze Controller causes the EtherCAT bus to assume the "Operational" state when the controller is set in RUN mode. The EtherCAT bus can only reach the "Operational" state if the fieldbus has previously allowed itself to be set to the "Pre-Operational" state.

Cause	**Start parameter could not be written.**

Error message	`EtherCAT_Master: Set master 'Operational' failed (0x4000005)`
	
	`[DeviceName] (1001): CoE 0x1234:5 - SDO Abort 'Object does not exist in the object dictionary (0x06020000)'`

Remedies	Correct the start-up parameters in »PLC Designer« on the **Configuration** tab of the EtherCAT master.

Cause	**Wiring error: The EtherCAT terminals (IN/OUT) of the slave were inverted.** A fieldbus scan does not indicate this error!
Error message	`EtherCAT_Master: Set master 'Operational' failed. DCM not in-sync`
Remedies	Correct wiring. Afterwards, reload the »PLC Designer« application into the automation system.

Cause	**The real structure at an EPM-S130 head end does not correspond to the control configuration configured.** This error can occur if I/O panels have been added manually to the control configuration.

Error message	`EtherCAT_Master: Set master 'Operational' failed. Timeout`
	
	`I_O_System_1000_EPM_S130 (1005): CoE emergency request. id=0x0, len=8, ErrCode=0xa000, ErrReg=0x2, data: 0xe 0x9 0x0 0x9 0x0`
	
	`I_O_System_1000_EPM_S130 (1005): Slaves signals Error. AL Status: 'PRE OPERATIONAL' (0x12), AL Status Code: 'Invalid Input Configuration'`

Remedies	Correct control configuration in the »PLC Designer«. **Note:** In "`data: ...`", a coded information says which error has occurred in which slave device/module. Detailed information on the coding of error messages can be found in the documentation of the corresponding slave device/module.

Cause	**Synchronisation settings in the standard device (slave) and the »PLC Designer« configuration are inconsistent.** Example : • »Engineer«: Sync source C01120 = MXI1 • »PLC Designer«: Distributed clocks deactivated ("DC = unused")

Error message	`EtherCAT_Master: Set master 'Operational' failed (0x4000005)`
	
	`[DeviceName] (1001): Slaves signals Error. AL Status: 'SAFE OPERATIONAL' (0x14), AL Status Code: 'Synchronization error' (0x1A)`

Remedies	Adapt the synchronisation settings in the slave.

Cause	**The first slave after the master has not been defined as DC master.**
Error message	`EtherCAT_Master: DC slaves 'out-of-sync'. Deviation xxxxxxxx ns`
Remedies	Declare the first slave after the master as DC master ("DC for Synchronization").

13.5.3 Messages: WKC Error / Not all slaves "Operational" / SyncManager Watchdog

In the "Operational" state, the process data are exchanged cyclically.

If a slave does not accept the cyclical frame (WKC is not increased), this error is caused.

Cause	• The bus cable between two EtherCAT nodes has been unplugged. • The node at position X is deenergised. • A slave no longer receives cyclical frames so that the watchdog determined by the device description is triggered. This message will only be transmitted when the connection to the master has been re-established.

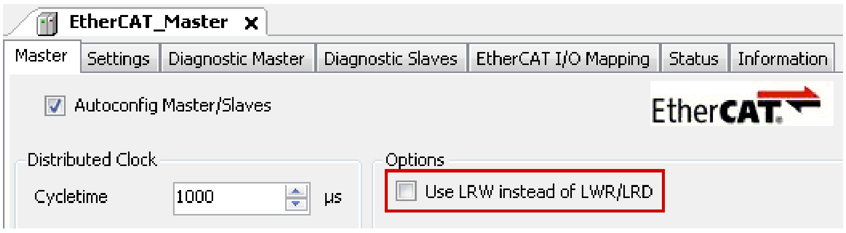
Error message	`EtherCAT_Master: Not all slaves 'Operational' (repeated 100 times)`
	`EtherCAT_Master: Cyclical command WKC error (repeated 100 times)`
	`EtherCAT_Master: Not all slaves 'Operational' (repeated 10 times)`
	`[DeviceName] (1002): Communication to device interrupted`
	`EtherCAT_Master: Cyclical command WKC error (repeated 10 times)`
	`EtherCAT_Master: Not all slaves 'Operational' (repeated 1 times)`
	`EtherCAT_Master: Cyclical command WKC error (repeated 1 times)`

Remedies	Correct bus topology and Restarting the EtherCAT fieldbus (📖 179).

13.5.4 Message: Invalid SyncManager Configuration

When the state is changed from "Pre-operational" to "Safe-operational", a slave reports "Invalid SyncManager Configuration".

<table>
<tr><td>Cause</td><td>• One of the slaves does not support an LRW command (Logical Read/Write).
• A slave is not written to correctly.</td></tr>
<tr><td>Error message</td><td><code>[DeviceName] (1001): Invalid SyncManager Configuration</code></td></tr>
<tr><td>Remedies</td><td>In the EtherCAT master tab, do <u>not</u> select the "Use LRW instead of LWR/LRD" checkbox.



</td></tr>
</table>

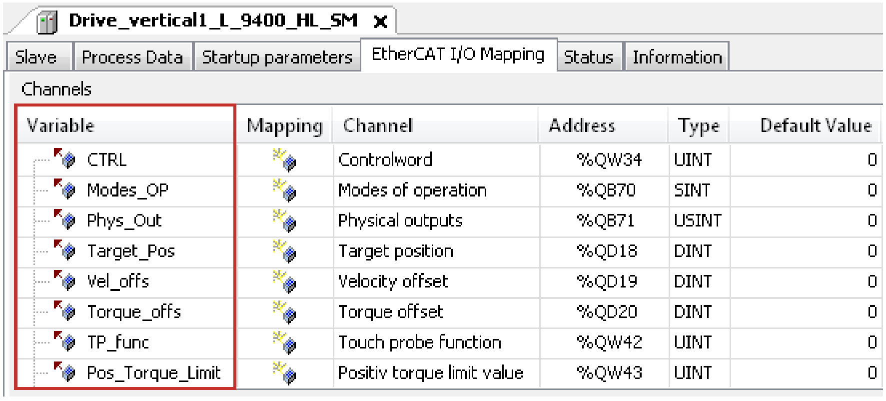
13.5.5 Messages: Invalid Input Configuration / Invalid Output Configuration

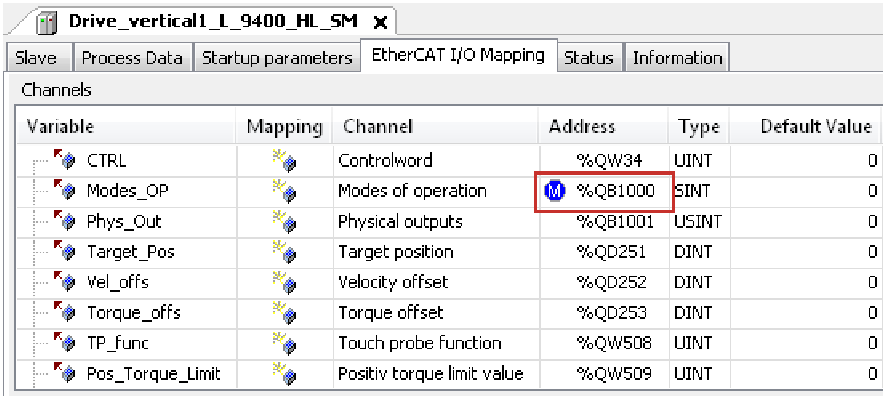
When the status changes from "Pre-Operational" to "Safe-Operational", a slave reports "Invalid Input Configuration" or "Invalid Output Configuration".

Cause	The process data configuration of a slave is not correct. • In case of a modular device such as the I/O system 1000 (EPM-Sxxx), the configuration in the project does not comply with the real assembly. • More process data than permissible are mapped for the device.
Error message	• `[DeviceName] (1001): Slave signals Error. AL state: 'PRE OPERATIONAL' (0x12), AL state code: 'Invalid Input Configuration' (0x1E)` • `[DeviceName] (1001):  Slave signals Error. AL state: 'PRE OPERATIONAL' (0x12), AL state code: 'Invalid Output Configuration' (0x1D)`
Remedies	• In case of modular devices such as the I/O system 1000 (EPM-Sxxx): Correct the control configuration in the »PLC Designer« (adjustment with the real setup). • Reduction of the process data: The maximum process data length must not be exceeded (see also the documentation of the device).

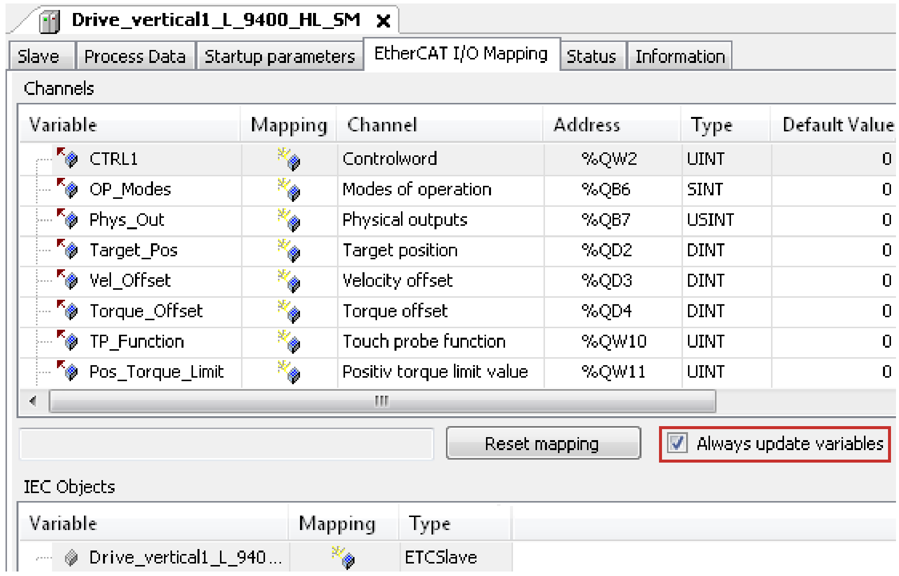
13.5.6 Error during process data transfer

A faulty EtherCAT I/O mapping causes errors during the process data transfer.

<table>
<tr><td>Cause</td><td>Use of logic addresses
In the »PLC Designer« application, access does not take place symbolically but directly via the I/O addresses (%Ixx, %Qxx) of the EtherCAT input and output objects <u>and</u> the bus structure, the PDO selection, etc, have changed.</td></tr>
<tr><td>Error message</td><td>–</td></tr>
<tr><td>Remedies</td><td>In the »PLC Designer« application, the input and output objects must be accessed via individual, non-ambiguous variables. The variable names must comply with the IEC 61131 syntax (no space characters and leading digits in the variable names).
Example :

Drive_vertical1_L_9400_HL_SM
Slave | Process Data | Startup parameters | EtherCAT I/O Mapping | Status | Information
Channels
<table>
<tr><th>Variable</th><th>Mapping</th><th>Channel</th><th>Address</th><th>Type</th><th>Default Value</th></tr>
<tr><td>CTRL</td><td></td><td>Controlword</td><td>%QW34</td><td>UINT</td><td>0</td></tr>
<tr><td>Modes_OP</td><td></td><td>Modes of operation</td><td>%QB70</td><td>SINT</td><td>0</td></tr>
<tr><td>Phys_Out</td><td></td><td>Physical outputs</td><td>%QB71</td><td>USINT</td><td>0</td></tr>
<tr><td>Target_Pos</td><td></td><td>Target position</td><td>%QD18</td><td>DINT</td><td>0</td></tr>
<tr><td>Vel_offs</td><td></td><td>Velocity offset</td><td>%QD19</td><td>DINT</td><td>0</td></tr>
<tr><td>Torque_offs</td><td></td><td>Torque offset</td><td>%QD20</td><td>DINT</td><td>0</td></tr>
<tr><td>TP_func</td><td></td><td>Touch probe function</td><td>%QW42</td><td>UINT</td><td>0</td></tr>
<tr><td>Pos_Torque_Limit</td><td></td><td>Positiv torque limit value</td><td>%QW43</td><td>UINT</td><td>0</td></tr>
</table>
Processing EtherCAT I/O mapping (96)</td></tr>
</table>

<table>
<tr><td>Cause</td><td>Manual definition of the logic address in the EtherCAT I/O mapping
Example: The address %QB70 has been changed to %QB1000.

Drive_vertical1_L_9400_HL_SM
Slave | Process Data | Startup parameters | EtherCAT I/O Mapping | Status | Information
Channels
<table>
<tr><th>Variable</th><th>Mapping</th><th>Channel</th><th>Address</th><th>Type</th><th>Default Value</th></tr>
<tr><td>CTRL</td><td></td><td>Controlword</td><td>%QW34</td><td>UINT</td><td>0</td></tr>
<tr><td>Modes_OP</td><td></td><td>Modes of operation</td><td>M %QB1000</td><td>SINT</td><td>0</td></tr>
<tr><td>Phys_Out</td><td></td><td>Physical outputs</td><td>%QB1001</td><td>USINT</td><td>0</td></tr>
<tr><td>Target_Pos</td><td></td><td>Target position</td><td>%QD251</td><td>DINT</td><td>0</td></tr>
<tr><td>Vel_offs</td><td></td><td>Velocity offset</td><td>%QD252</td><td>DINT</td><td>0</td></tr>
<tr><td>Torque_offs</td><td></td><td>Torque offset</td><td>%QD253</td><td>DINT</td><td>0</td></tr>
<tr><td>TP_func</td><td></td><td>Touch probe function</td><td>%QW508</td><td>UINT</td><td>0</td></tr>
<tr><td>Pos_Torque_Limit</td><td></td><td>Positiv torque limit value</td><td>%QW509</td><td>UINT</td><td>0</td></tr>
</table></td></tr>
<tr><td>Error message</td><td>–</td></tr>
<tr><td>Remedies</td><td>It is not permissible to manually manipulate the I/O addresses for the EtherCAT bus!</td></tr>
</table>

Cause	**Missing or incorrect I/O mapping** In the case of Servo Drives 9400 and Inverter Drives 8400, the ports in the »Engineer« are displayed incorrectly or not at all.
Error message	–
Remedies	Check and correct the mapping settings in the control configuration and in the inverter. When the Lenze Controller is started, the complete configuration/PDO mapping is written into the EtherCAT slaves. When this is done, mapping entries, e.g. from the »Engineer« are overwritten.

<table>
<tr><td>Cause</td><td>Variables are not used in the »PLC Designer« application.
I/O variables that are not used in the »PLC Designer« application are not copied into the process image (and vice versa) so that they are not updated by the peripherals.</td></tr>
<tr><td>Error message</td><td>–</td></tr>
<tr><td>Remedies</td><td>With the slave, go to the EtherCAT I/O Mapping tab and set a checkmark at Always update variables:

</td></tr>
</table>

13.5.7 Messages: EtherCAT cable not connected / EtherCAT cable connected

Cause	**The bus cable between the Lenze Controller and the first node has been unplugged.** If a previously removed bus cable has been plugged into the first EtherCAT node, the message "`EtherCAT_Master: EtherCAT cable connected`" is entered in the logbook of the controller. The EtherCAT connection is re-established. Since the EtherCAT slave sync managers do not receive any messages, a time-out expires and the slaves change to the "Safe-Operational" state.

Error message	`EtherCAT_Master: EtherCAT cable not connected`
	`...`
	`EtherCAT_Master: EtherCAT cable connected`

Remedies	After the bus cable has been plugged in again, Restarting the EtherCAT fieldbus (🕮 179).

13.5.8 Message: Frame Response Error

Cause	**A frame sent by the master does not return to the master until the next cycle.** • The task utilisation is too high so that a sent frame takes longer than the time to the next start of the bus cycle task. • The EtherCAT bus cycle task does not have the highest IEC task priority or another task has the same IEC task priority so that the EtherCAT bus cycle task is suppressed. • Due to an error, the slave does not forward any frames. • Only a switch or an ET2000 is connected to the Lenze Controller, but no further slave.

Error message	`EtherCAT_Master: Frame response error (repeated 1 time)`
	`EtherCAT_Master: Frame response error (repeated 10 times)`
	`EtherCAT_Master: Frame response error (repeated 100 times)`

Remedies	• Reduce the program code or increase the bus task cycle time. • Assign the sole and highest IEC task priority to the EtherCAT bus cycle task. • Correct the slave error. • Correct the bus structure.

13.5.9 Shafts make clicking noises

If the shafts make clicking noises, this is often caused by faulty synchronisation or a shift of data in the process image.

Cause	**The task and DC cycle times set in the logic/motion system differ.**
Error message	–
Remedies	Adjust the task cycle time and DC cycle time. ▸ Adjusting the task cycle time and DC cycle time (📖 89)

Cause	**Wiring error: The EtherCAT terminals (IN/OUT) of the slave were inverted.** A fieldbus scan does not indicate this error!
Error message	`EtherCAT_Master: Set master 'Operational' failed. DCM not in-sync`
Remedies	Correct wiring. Afterwards, reload the »PLC Designer« application into the automation system.

Cause	**Clicking noise of the shafts after "out-of-sync"** If due to a fault, the preset DC deviation limit is exceeded, a re-synchronisation of the DC slaves is carried out until the slaves are synchronised again ("In-Sync") and the DC deviation is under the preset limit value again. Currently, the Lenze Controller is not re-synchronised to the distributed clocks so that the sync pulses of the master and the ones of the slaves are different.
Error message	`EtherCAT_Master: DC slaves 'out-of-sync'. Deviation xxxxxxxx ns`
Remedies	Restarting the EtherCAT fieldbus (📖 179) so that the DC slaves and the DC master synchronise again.

Cause	**Wrong selection of the device sync source (9400 HighLine CiA402, C01120)** After the sync source has been changed by **C01120**, the subsequent download and the setting of the slave to the "Operational" state may fail
Error message	–
Remedies	• Manual setting of code C01120 to MXI1 or MXI2 • Repeated download with PLC start • Restarting the EtherCAT fieldbus (📖 179) with reset of the SoftMotion drive

13.5.10 Shafts do not rotate

Cause	**The EtherCAT bus could not be set to the "Operational" state** ▶ The EtherCAT bus does not assume the "Operational" state (📖 196)
Error message	–
Remedies	▶ The EtherCAT bus does not assume the "Operational" state (📖 196)

Cause	**Clicking noise of the shafts after "out-of-sync"** ▶ Error during process data transfer (📖 199)
Error message	–
Remedies	▶ Error during process data transfer (📖 199)

Cause	**Faulty SoftMotion scaling/mapping** With SoftMotion scaling/mapping, the increments per revolution are not set.
Error message	–
Remedies	Check the following settings and correct them if required: • Gearbox ratio in the »PLC Designer« application • Mapping settings in the master configuration When the Lenze Controller is started, the complete configuration/PDO mapping is written into the EtherCAT slaves. When this is done, mapping entries, e.g. from the »Engineer«, are overwritten. **Tip:** In the case of the Servo Drive 9400 HighLine Cia402, 65536 increments per revolution are correct.

13.6 System error messages

In the case of system error messages, the following types of error are distinguished:

- General error codes (L_ETC_ERRORCODE) (204)
- Error messages for modular machine configuration (211)
- Lenze Controller logbook messages (213)
- SDO abort codes (218)

13.6.1 General error codes (L_ETC_ERRORCODE)

General error codes are provided at the *eErrorCode* output of the EtherCAT function blocks (see L_IODrvEtherCAT function library (119)).

If required, these error messages are also output as additional error information in the Logbook of the Lenze Controller in the »WebConfig« (190).

Error no. [hex]	Name	Description
0x00000000	ETC_E_NOERROR	No error / function completed successfully.
0x02000000	ETC_TEXTBASE	Unknown (basis) text **Internal error. Please contact Lenze.**
0x03000000	ETC_ALSTATEBASE	AL status: No error **Internal error. Please contact Lenze.**
0x98110000	ETC_E_ERROR	Unspecified error
0x98110180	ETC_EMRAS_E_ERROR	Unspecified RAS error **Internal error. Please contact Lenze.**
0x981201C0	ETC_DCM_E_ERROR	Unspecified DCM error **Internal error. Please contact Lenze.**
0x98110001	ETC_E_NOTSUPPORTED	Function or feature not available.
0x98110002	ETC_E_INVALIDINDEX	CoE: invalid SDO index
0x98110003	ETC_E_INVALIDOFFSET	Invalid offset value during access to process data image **Internal error. Please contact Lenze.**
0x98110005	ETC_E_INVALIDSIZE	Invalid offset value ... during access to process data image during storage of data in a data area
0x98110006	ETC_E_INVALIDDATA	Invalid data
0x98110007	ETC_E_NOTREADY	Internal software error (numerous possible causes) **Internal error. Please contact Lenze.**
0x98110008	ETC_E_BUSY	The master is busy at the moment and cannot process the API function. The function should be repeated at a later time.
0x98110009	ETC_E_ACYC_FRM_FREEQ_EMPTY	The queue/the memory for acyclic commands is full. **Internal error. Please contact Lenze.**
0x9811000A	ETC_E_NOMEMORY	Not enough application memory available. **Internal error. Please contact Lenze.**
0x9811000B	ETC_E_INVALIDPARM	An API function has been called with incorrect parameters.
0x9811000C	ETC_E_NOTFOUND	An API function has been called with an invalid slave ID.
0x9811000E	ETC_E_INVALIDSTATE	Invalid state
0x9811000F	ETC_E_TIMER_LIST_FULL	Not enough application memory available. **Internal error. Please contact Lenze.**
0x98110010	ETC_E_TIMEOUT	A time-out is active.
0x98110011	ETC_E_OPENFAILED	Internal software error (numerous possible causes) **Internal error. Please contact Lenze.**
0x98110012	ETC_E_SENDFAILED	The transmission of the frame has failed. **Internal error. Please contact Lenze.**
0x98110013	ETC_E_INSERTMAILBOX	The mailbox command cannot be enqueued on the internal queue. **Internal error. Please contact Lenze.**
0x98110014	ETC_E_INVALIDCMD	Unknown mailbox command code **Internal error. Please contact Lenze.**

Error no. [hex]	Name	Description
0x98110015	ETC_E_UNKNOWN_MBX_PROTOCOL	Unknown mailbox protocol Mailbox command ID with unknown protocol assignment **Internal error. Please contact Lenze.**
0x98110016	ETC_E_ACCESSDENIED	Access denied (internal software error on the master) **Internal error. Please contact Lenze.**
0x9811001A	ETC_E_PRODKEY_INVALID	The evaluation version of the master is used. The master enters the "Stop" mode after 30 minutes. **Internal error. Please contact Lenze.**
0x9811001B	ETC_E_WRONG_FORMAT	The XML file contains no or faulty content. **Internal error. Please contact Lenze.**
0x9811001C	ETC_E_FEATURE_DISABLED	Attempt to execute a non-existing or deactivated function. **Internal error. Please contact Lenze.**
0x9811001D	ETC_E_SHADOW_MEMORY	The shadow memory has been requested in wrong mode. **Internal error. Please contact Lenze.**
0x9811001E	ETC_E_BUSCONFIG_MISMATCH	The EtherCAT configuration of the master and the connected slaves does not comply with the physical bus structure.
0x9811001F	ETC_E_CONFIGDATAREAD	The XML file cannot be read. **Internal error. Please contact Lenze.**
0x98110021	ETC_E_XML_CYCCMDS_MISSING	The XML file of the master does not contain any cyclical commands. **Internal error. Please contact Lenze.**
0x98110022	ETC_E_XML_ALSTATUS_READ_MISSING	The XML file of the master does not contain the command for reading the AL Status Register. **Internal error. Please contact Lenze.**
0x98110023	ETC_E_MCSM_FATAL_ERROR	The master state machine is in an invalid state. **Internal error. Please contact Lenze.**
0x98110024	ETC_E_SLAVE_ERROR	The slave cannot be addressed.
0x98110025	ETC_E_FRAME_LOST	An EtherCAT frame was lost on the fieldbus, i.e. it has not been received. If this error occurred frequently, this indicates the wiring may be faulty.
0x98110026	ETC_E_CMD_MISSING	The received EtherCAT frame is not complete. **Internal error. Please contact Lenze.**
0x98110028	ETC_E_INVALID_DCL_MODE	This function cannot be used when DC latching is in the "Auto Read" operating mode. **Internal error. Please contact Lenze.**
0x98110029	ETC_E_AI_ADDRESS	The connected slaves do not comply with the control configuration. This error only occurs if a previously existing slave disappears. **Internal error. Please contact Lenze.**
0x9811002A	ETC_E_INVALID_SLAVE_STATE	The mailbox commands are not permissible in the current slave state.
0x9811002B	ETC_E_SLAVE_NOT_ADDRESSABLE	The slave has been switched on/off.
0x9811002C	ETC_E_CYC_CMDS_OVERFLOW	Error during XML file creation by the configurator **Internal error. Please contact Lenze.**
0x9811002D	ETC_E_LINK_DISCONNECTED	The EtherCAT cable has not be connected/plugged into the Lenze controller.
0x9811002E	ETC_E_MASTERCORE_INACCESSIBLE	The connection to the master (server) is interrupted or the master has been stopped. **Internal error. Please contact Lenze.**
0x9811002F	ETC_E_COE_MBXSND_WKC_ERROR	It is not possible to write to the CoE mailbox in the slave. The slave has not yet read the mailbox. **Internal error. Please contact Lenze.**
0x98110030	ETC_E_COE_MBXRCV_WKC_ERROR	It is not possible to read the CoE mailbox in the slave. **Internal error. Please contact Lenze.**
0x98110031	ETC_E_NO_MBX_SUPPORT	The slave does not support mailbox transfer.
0x98110032	ETC_E_NO_COE_SUPPORT	Configurator error or the slave description file does not correspond with the slave firmware.
0x98110033	ETC_E_NO_EOE_SUPPORT	Configurator error or the slave description file does not correspond with the slave firmware.
0x98110034	ETC_E_NO_FOE_SUPPORT	Configurator error or the slave description file does not correspond with the slave firmware.
0x98110035	ETC_E_NO_SOE_SUPPORT	Configurator error or the slave description file does not correspond with the slave firmware. **Is not supported.**
0x98110036	ETC_E_NO_VOE_SUPPORT	Configurator error or the slave description file does not correspond with the slave firmware. Is not supported.
0x98110037	ETC_E_EVAL_VIOLATION	The number of slaves indicated in the XML file is too large for the evaluation version of the master. **Internal error. Please contact Lenze.**

Error no. [hex]	Name	Description
0x98110038	ETC_E_EVAL_EXPIRED	The evaluation time has expired. The fieldbus is stopped. **Internal error. Please contact Lenze.**
0x98110040	ETC_E_SDO_ABORTCODE_TOGGLE	The status of the toggle bit has not changed. Abort code 0x05030000
0x98110041	ETC_E_SDO_ABORTCODE_TIMEOUT	SDO protocol time-out Abort code 0x05040000
0x98110042	ETC_E_SDO_ABORTCODE_CCS_SCS	Invalid or unknown specification symbol for the client/server command Abort code 0x05040001
0x98110043	ETC_E_SDO_ABORTCODE_BLK_SIZE	Invalid block size (only in "Block mode") Abort code 0x05040002
0x98110044	ETC_E_SDO_ABORTCODE_SEQNO	Invalid sequence number (only in "Block mode") Abort code 0x05040003
0x98110045	ETC_E_SDO_ABORTCODE_CRC	CRC error (only in "Block mode") Abort code 0x05040004
0x98110046	ETC_E_SDO_ABORTCODE_MEMORY	The space in the main memory is not sufficient. Abort code 0x05040005
0x98110047	ETC_E_SDO_ABORTCODE_ACCESS	Access to object not supported Abort code 0x06010000
0x98110048	ETC_E_SDO_ABORTCODE_WRITEONLY	Read access to a write-protected object Abort code 0x06010001
0x98110049	ETC_E_SDO_ABORTCODE_READONLY	Write access to a write-protected object Abort code 0x06010002
0x9811004A	ETC_E_SDO_ABORTCODE_INDEX	An object does not exist in the object directory Abort code 0x06020000
0x9811004B	ETC_E_SDO_ABORTCODE_PDO_MAP	An object cannot be mapped into the PDO Abort code 0x06040041
0x9811004C	ETC_E_SDO_ABORTCODE_PDO_LEN	The number and/or length of the objects mapped would exceed the PDO length Abort code 0x06040042
0x9811004D	ETC_E_SDO_ABORTCODE_P_INCOMP	General parameter incompatibility Abort code 0x06040043
0x9811004E	ETC_E_SDO_ABORTCODE_I_INCOMP	General internal device incompatibility Abort code 0x06040047
0x9811004F	ETC_E_SDO_ABORTCODE_HARDWARE	Access has failed due to a fault in the hardware Abort code 0x06060000
0x98110050	ETC_E_SDO_ABORTCODE_DATA_SIZE	The data type or the parameter length does not correspond Abort code 0x06070010
0x98110051	ETC_E_SDO_ABORTCODE_DATA_SIZE1	Incorrect data type (The parameter length is too large) Abort code 0x06070012
0x98110052	ETC_E_SDO_ABORTCODE_DATA_SIZE2	Wrong data type (parameter length is too small). Abort code 0x06070013
0x98110053	ETC_E_SDO_ABORTCODE_OFFSET	A subindex is not available Abort code 0x06090011
0x98110054	ETC_E_SDO_ABORTCODE_DATA_RANGE	The value range for parameters is too great (only for write access) Abort code 0x06090030
0x98110055	ETC_E_SDO_ABORTCODE_DATA_RANGE1	The parameter value is too high Abort code 0x06090031
0x98110056	ETC_E_SDO_ABORTCODE_DATA_RANGE2	The parameter value is too low Abort code 0x06090032
0x98110057	ETC_E_SDO_ABORTCODE_MINMAX	The maximum value is lower than the minimum value Abort code 0x06090036
0x98110058	ETC_E_SDO_ABORTCODE_GENERAL	General error Abort code 0x08000000
0x98110059	ETC_E_SDO_ABORTCODE_TRANSFER	Data cannot be transferred/saved to the application. Abort code 0x08000020
0x9811005A	ETC_E_SDO_ABORTCODE_TRANSFER1	Data cannot be transferred/saved to the application because of local control. Abort code 0x08000021
0x9811005B	ETC_E_SDO_ABORTCODE_TRANSFER2	Due to the current device state, data cannot be transferred to the application or stored in the application Abort code 0x08000022
0x9811005C	ETC_E_SDO_ABORTCODE_DICTIONARY	The dynamic generation of an object directory has failed, or no object directory is available. Abort code 0x08000023
0x9811005D	ETC_E_SDO_ABORTCODE_UNKNOWN	Unknown internal slave error

Error no. [hex]	Name	Description
0x98110060	ETC_E_FOE_ERRCODE_NOTDEFINED	Manufacturer-specific FoE error
0x98110061	ETC_E_FOE_ERRCODE_NOTFOUND	Not found
0x98110062	ETC_E_FOE_ERRCODE_ACCESS	Access denied
0x98110063	ETC_E_FOE_ERRCODE_DISKFULL	Memory (floppy disk/hard disk) is full.
0x98110064	ETC_E_FOE_ERRCODE_ILLEGAL	Invalid/impermissible
0x98110065	ETC_E_FOE_ERRCODE_PACKENO	Wrong package number
0x98110066	ETC_E_FOE_ERRCODE_EXISTS	Already available
0x98110067	ETC_E_FOE_ERRCODE_NOUSER	User/consumer is missing.
0x98110068	ETC_E_FOE_ERRCODE_BOOTSTRAPONLY	Only bootstrap state
0x98110069	ETC_E_FOE_ERRCODE_NOTINBOOTSTRAP	No bootstrap state
0x9811006A	ETC_E_FOE_ERRCODE_INVALIDPASSWORD	No required access authorisation
0x9811006B	ETC_E_FOE_ERRCODE_PROGERROR	Program error
0x98110070	ETC_E_CFGFILENOTFOUND	The master configuration has not been found.
0x98110071	ETC_E_EEPROMREADERROR	Command error during EEPROM upload
0x98110072	ETC_E_EEPROMWRITEERROR	Command error during EEPROM download
0x98110073	ETC_E_XML_CYCCMDS_SIZEMISMATCH	The cyclical command has a wrong size or is too long. **Internal error. Please contact Lenze.**
0x98110074	ETC_E_XML_INVALID_INP_OFF	Invalid input offset in cyclical command **Internal error. Please contact Lenze.**
0x98110075	ETC_E_XML_INVALID_OUT_OFF	Invalid output offset in cyclical command **Internal error. Please contact Lenze.**
0x98110076	ETC_E_PORTCLOSE	Closing of the port failed. **Internal error. Please contact Lenze.**
0x98110077	ETC_E_PORTOPEN	Opening of the port failed. **Internal error. Please contact Lenze.**
0x98110078	ETC_E_SOE_ERRORCODE_INVALID_ACCESS	Impermissible access to element 0 **Is not supported.**
0x98110079	ETC_E_SOE_ERRORCODE_NOT_EXIST	Nonexistent Is not supported.
0x9811007a	ETC_E_SOE_ERRORCODE_INVL_ACC_ELEM1	Impermissible access to element 1 Is not supported.
0x9811007b	ETC_E_SOE_ERRORCODE_NAME_NOT_EXIST	The name is not available. Is not supported.
0x9811007c	ETC_E_SOE_ERRORCODE_NAME_UNDERSIZE	The name is too short for transmission. Is not supported.
0x9811007d	ETC_E_SOE_ERRORCODE_NAME_OVERSIZE	The name is too short for transmission. Is not supported.
0x9811007e	ETC_E_SOE_ERRORCODE_NAME_UNCHANGE	The name cannot be changed. Is not supported.
0x9811007f	ETC_E_SOE_ERRORCODE_NAME_WR_PROT	The name is currently write-protected. Is not supported.
0x98110080	ETC_E_SOE_ERRORCODE_UNDERS_TRANS	The attribute is too small for transmission. Is not supported.
0x98110081	ETC_E_SOE_ERRORCODE_OVERS_TRANS	The attribute is too big for transmission. Is not supported.
0x98110082	ETC_E_SOE_ERRORCODE_ATTR_UNCHANGE	The attribute cannot be changed. Is not supported.
0x98110083	ETC_E_SOE_ERRORCODE_ATTR_WR_PROT	The attribute is currently write-protected. Is not supported.
0x98110084	ETC_E_SOE_ERRORCODE_UNIT_NOT_EXIST	The unit is not available. Is not supported.
0x98110085	ETC_E_SOE_ERRORCODE_UNIT_UNDERSIZE	The unit is too small for transmission. Is not supported.
0x98110086	ETC_E_SOE_ERRORCODE_UNIT_OVERSIZE	The unit is too big for transmission. Is not supported.
0x98110087	ETC_E_SOE_ERRORCODE_UNIT_UNCHANGE	The unit cannot be changed. Is not supported.
0x98110088	ETC_E_SOE_ERRORCODE_UNIT_WR_PROT	The unit is currently write-protected. Is not supported.

Error no. [hex]	Name	Description
0x98110089	ETC_E_SOE_ERRORCODE_MIN_NOT_EXIST	The minimum input value is not available. Is not supported.
0x9811008a	ETC_E_SOE_ERRORCODE_MIN_UNDERSIZE	The minimum input value is too small for transmission. Is not supported.
0x9811008b	ETC_E_SOE_ERRORCODE_MIN_OVERSIZE	The minimum input value is too big for transmission. Is not supported.
0x9811008c	ETC_E_SOE_ERRORCODE_MIN_UNCHANGE	The minimum input value cannot be changed. Is not supported.
0x9811008d	ETC_E_SOE_ERRORCODE_MIN_WR_PROT	The minimum input value is currently write-protected. Is not supported.
0x9811008e	ETC_E_SOE_ERRORCODE_MAX_NOT_EXIST	The maximum input value is not available. Is not supported.
0x9811008f	ETC_E_SOE_ERRORCODE_MAX_UNDERSIZE	The maximum input value is too small for transmission. Is not supported.
0x98110090	ETC_E_SOE_ERRORCODE_MAX_OVERSIZE	The maximum input value is too big for transmission. Is not supported.
0x98110091	ETC_E_SOE_ERRORCODE_MAX_UNCHANGE	The maximum input value cannot be changed. Is not supported.
0x98110092	ETC_E_SOE_ERRORCODE_MAX_WR_PROT	The maximum input value is currently write-protected. Is not supported.
0x98110093	ETC_E_SOE_ERRORCODE_DATA_NOT_EXIST	The data element is not available. Is not supported.
0x98110094	ETC_E_SOE_ERRORCODE_DATA_UNDERSIZE	The data element is too small for transmission. Is not supported.
0x98110095	ETC_E_SOE_ERRORCODE_DATA_OVERSIZE	The data element is too big for transmission. Is not supported.
0x98110096	ETC_E_SOE_ERRORCODE_DATA_UNCHANGE	The data element cannot be changed. Is not supported.
0x98110097	ETC_E_SOE_ERRORCODE_DATA_WR_PROT	The data element is currently write-protected. Is not supported.
0x98110098	ETC_E_SOE_ERRORCODE_DATA_MIN_LIMIT	The data element is smaller than the minimum input value limit. Is not supported.
0x98110099	ETC_E_SOE_ERRORCODE_DATA_MAX_LIMIT	The data element exceeds the maximum input value limit. Is not supported.
0x9811009a	ETC_E_SOE_ERRORCODE_DATA_INCOR	The data element is not correct. Is not supported.
0x9811009b	ETC_E_SOE_ERRORCODE_PASWD_PROT	The data element is protected by a password. Is not supported.
0x9811009c	ETC_E_SOE_ERRORCODE_TEMP_UNCHANGE	The data element can currently not be changed (in AT or MDT). Is not supported.
0x9811009d	ETC_E_SOE_ERRORCODE_INVL_INDIRECT	Invalid/indirect Is not supported.
0x9811009e	ETC_E_SOE_ERRORCODE_TEMP_UNCHANGE1	The data element can currently not be changed (parameter or OP mode). Is not supported.
0x9811009f	ETC_E_SOE_ERRORCODE_ALREADY_ACTIVE	The command is already active. Is not supported.
0x98110100	ETC_E_SOE_ERRORCODE_NOT_INTERRUPT	The command can be interrupted. Is not supported.
0x98110101	ETC_E_SOE_ERRORCODE_CMD_NOT_AVAIL	The command is not available in this phase. Is not supported.
0x98110102	ETC_E_SOE_ERRORCODE_CMD_NOT_AVAIL1	The command is not available (invalid parameter) Is not supported.
0x98110103	ETC_E_SOE_ERRORCODE_DRIVE_NO	The received drive number does not comply with the requested drive number. Is not supported.
0x98110104	ETC_E_SOE_ERRORCODE_IDN	The received ID does not comply with the requested ID. Is not supported.
0x98110105	ETC_E_SOE_ERRORCODE_FRAGMENT_LOST	At least on fragment got lost. Is not supported.
0x98110106	ETC_E_SOE_ERRORCODE_BUFFER_FULL	The Rx memory buffer is full (EtherCAT call with too little data buffer). Is not supported.
0x98110107	ETC_E_SOE_ERRORCODE_NO_DATA	No data state Is not supported.

Error no. [hex]	Name	Description
0x98110108	ETC_E_SOE_ERRORCODE_NO_DEFAULT_VALUE	No standard value Is not supported.
0x98110109	ETC_E_SOE_ERRORCODE_DEFAULT_LONG	The standard value transmission is too long. Is not supported.
0x9811010a	ETC_E_SOE_ERRORCODE_DEFAULT_WP	The standard value cannot be changed (read only). Is not supported.
0x9811010b	ETC_E_SOE_ERRORCODE_INVL_DRIVE_NO	Invalid drive number Is not supported.
0x9811010c	ETC_E_SOE_ERRORCODE_GENERAL_ERROR	General error Is not supported.
0x9811010d	ETC_E_SOE_ERRCODE_NO_ELEM_ADR	No element has been addressed. Is not supported.
0x9811010e	ETC_E_SLAVE_NOT_PRESENT	The slave is not available at the fieldbus.
0x9811010f	ETC_E_NO_FOE_SUPPORT_BS	The FoE protocol is not supported in the bootstrap state.
0x98110110	ETC_E_EEPROMRELOADERROR	Command error during EEPROM reload
0x98110111	ETC_E_SLAVECTRLRESETERROR	Command error during slave controller reset
0x98110112	ETC_E_SYSDRIVERMISSING	The system driver ect.sys could not be opened. **Internal error. Please contact Lenze.**
0x9811011E	ETC_E_BUSCONFIG_TOPOCHANGE	The fieldbus configuration could not be detected. The bus topology has been changed.
0x9811011F	ETC_E_EOE_MBX_WKC_ERROR	Error at EoE mailbox reception: Working counter
0x98110120	ETC_E_FOE_MBX_WKC_ERROR	Error at FoE mailbox reception: Working counter
0x98110121	ETC_E_SOE_MBX_WKC_ERROR	Error at SoE mailbox reception: Working counter Is not supported.
0x98110122	ETC_E_AOE_MBX_WKC_ERROR	Error at AoE mailbox reception: Working counter Is not supported.
0x98110123	ETC_E_VOE_MBX_WKC_ERROR	Error at VoE mailbox reception: Working counter Is not supported.
0x98110124	ETC_E_EEPROMASSIGNERROR	The EEPROM assignment has failed. **Internal error. Please contact Lenze.**
0x98110125	ETC_E_MBX_ERROR_TYPE	Error at mailbox reception **Internal error. Please contact Lenze.**
0x981201C1	ETC_DCM_E_NOTINITIALIZED	The initialisation has not been successful. The initialisation function has not been called. **Internal error. Please contact Lenze.**
0x981201C2	ETC_DCM_E_MAX_CTL_ERROR_EXCEED	Controller error: Synchronisation beyond the limits **Internal error. Please contact Lenze.**
0x981201C3	ETC_DCM_E_NOMEMORY	Not enough memory location available. **Internal error. Please contact Lenze.**
0x981201C4	ETC_DCM_E_INVALID_HWLAYER	Hardware error: Invalid (BSP) **Internal error. Please contact Lenze.**
0x981201C5	ETC_DCM_E_TIMER_MODIFY_ERROR	Hardware error: Error at timer change **Internal error. Please contact Lenze.**
0x981201C6	ETC_DCM_E_TIMER_NOT_RUNNING	Hardware error: The timer does not run. **Internal error. Please contact Lenze.**
0x981201C7	ETC_DCM_E_WRONG_CPU	Hardware error: The function has been called on the wrong CPU. **Internal error. Please contact Lenze.**
0x981201C8	ETC_DCM_E_INVALID_SYNC_ PERIOD	Invalid DC-Sync. Period length (invalid DC-master?) **Internal error. Please contact Lenze.**
0x981201C9	ETC_DCM_E_INVALID_SETVAL	DCM controller error: The set value is too low. **Internal error. Please contact Lenze.**
0x981201CA	ETC_DCM_E_DRIFT_TO_HIGH	DCM controller error: The deviation between the local timer and the reference clock is too high. **Internal error. Please contact Lenze.**
0x98110181	ETC_EMRAS_E_INVALIDCOOKIE	Reconnecting using the old cookie has failed. A new reconnection attempt is made automatically. **Internal error. Please contact Lenze.**
0x98110183	ETC_EMRAS_E_MULSRVDISMULCON	Attempt to connect to another remote server has been rejected because the multi-instance API has not been used for establishing an already existing connection. **Internal error. Please contact Lenze.**
0x98110184	ETC_EMRAS_E_LOGONCANCELLED	Server aborts connection during client logon. **Internal error. Please contact Lenze.**

Error no. [hex]	Name	Description
0x98110186	ETC_EMRAS_E_INVALIDVERSION	Server and client version are not identical (different protocol versions). Therefore, connecting has been rejected. **Internal error. Please contact Lenze.**
0x98110187	ETC_EMRAS_E_INVALIDACCESSCONFIG	Access configuration is invalid **Internal error. Please contact Lenze.**
0x98110188	ETC_EMRAS_E_ACCESSLESS	No access to call on access level **Internal error. Please contact Lenze.**
0x98110191	ETC_EMRAS_EVT_SERVERSTOPPED	Detailed description for connection abort/termination if connection to server has been closed due to "API call (local)". **Internal error. Please contact Lenze.**
0x98110192	ETC_EMRAS_EVT_WDEXPIRED	Detailed description for connection abort/termination if connection has been closed due to missing keep-alive messages. **Internal error. Please contact Lenze.**
0x98110193	ETC_EMRAS_EVT_RECONEXPIRED	Client attempts to reopen an old connection (after the connection has been aborted), but the server has already cleared the session. A new connection must be established (register client and mailbox objects must be created again). **Internal error. Please contact Lenze.**
0x98110194	ETC_EMRAS_EVT_CLIENTLOGON	Server message when a new client has connected. **Internal error. Please contact Lenze.**
0x98110195	ETC_EMRAS_EVT_RECONNECT	Server message when a client has successfully reopened an old connection. **Internal error. Please contact Lenze.**
0x98110196	ETC_EMRAS_EVT_SOCKCHANGE	Detailed description (event) which marks the successful socket transfer of a new connection to an already existing session object (reconnect). **Internal error. Please contact Lenze.**
0x98110197	ETC_EMRAS_EVT_CLNTDISC	Client disconnected/switched off **Internal error. Please contact Lenze.**

13.6.2 Error messages for modular machine configuration

The error messages are arranged in alphabetical order (A - Z).

Error message	Description
MMC - address assignment - done	INFO: The address assignment has been completed successfully.
MMC - address assignment - invalid device on position ... (..._.../..._...)	ERROR: During the address assignment by means of the ADDR_ASSIGNMENT_EXTERNALLY or ADDR_ASSIGNMENT_CONFIG_SLAVEORDER service, a slave has been detected unexpectedly at the given position.
MMC - address assignment - less slaves connected (...) as configured (...)	ERROR: In the address assignment, less slaves are connected to the physical EtherCAT bus than given in the active configuration.
MMC - address assignment - more slaves connected as configured	ERROR: In the address assignment, more slaves are connected to the physical etherCAT bus than given in the active configuration.
MMC - address assignment - writing address ... at position ... by CoE ... (error ...)	ERROR: During the address assignment, an error has occurred for the slave at the given position. For further information see General error codes (L_ETC_ERRORCODE).
MMC - address assignment - writing address ... at position ... failed (error ...)	ERROR: During the address assignment, an error has occurred for the slave at the given position. For further information see General error codes (L_ETC_ERRORCODE).
MMC - address assignment - written address ... at position ... successfully	INFO: The address has been successfully assigned to the slave at the given position.
MMC - devices not ascending or device(s) missing at line ... (...)	ERROR: In the **mmc-0-conf.csv** configuration file, the slaves are not given in ascending order (starting with '1001') or are missing. Or the address space is incomplete.
MMC - duplicated Alias Address ... at positions ... and ...	ERROR: While the EtherCAT bus is booted and the slaves are checked, several slaves have been found with identical "Second Station Address" (alias address) at the given position. The first slave behind the EtherCAT master has the position '1'.
MMC - Error in configuration files	ERROR: During the analysis of the **mmc-0-conf.csv** configuration file or the **mmc-0-ident.csv** identification file, errors have been detected. Further information about the error is displayed in the logbook above.
MMC - file does not exist ...	INFO: The **mmc-0-conf.csv** configuration file or **mmc-0-ident.csv** identification file have not been found in the directory `.../USBStorage/IPC/PLC` or `.../SDCard/IPC/PLC`. As both files do not have to be available in the system (depending on the service), this is only an information.
MMC - Internal Error (...)	ERROR: An internal error has occurred. The internal error number is output in the error message. Please contact Lenze!
MMC - invalid Alias Address ... at position ... (.../.../.../...)	ERROR: While the EtherCAT bus is booted and the slaves are checked, a slave with invalid or unexpected "Second Station Address" (alias address) has been detected at the given position. The first slave behind the EtherCAT master has the position '1'. Parameters in bracket: Vendor-ID/Product code/Revision/Serial number.
MMC - invalid configuration	ERROR: An action has been aborted because no valid configuration is active.
MMC - mandatory slave ... is not present	ERROR: While the EtherCAT bus and the slaves are checked, a mandatory slave has not been found at the bus. In the error message, the EtherCAT address or "Second Station Address" (alias address) of the slave is given.
MMC - 'Modular Machine Configuration' is active - EtherCAT Master is controlled by L_ETC_MMCController	INFO: The L_ETC_MMCController function block is used in the PLC program. The behaviour of the EtherCAT master is controlled by the function block.
MMC - no configuration checks	INFO: While the EtherCAT bus is booted, the configuration is not checked because the RUN_WITHOUT_CHECK service is active.
MMC - no valid service active	ERROR: When the L_ETC_MMCController function block was used, it was tried to boot the EtherCAT bus (xRestart = TRUE). But no service is active.
MMC - number of device in device tree differs at line ... (...)	Warning: The number of slaves from the **mmc-0-conf.csv** configuration file is higher than defined in the »PLC Designer« project. This is a warning because the excessive devices from the configuration are simply ignored.
MMC - optional slave ... is present, but not allowed	ERROR: While the EtherCAT bus is booted and the RUN_OPTIONAL_SLAVES_PROHIBITED service is active, a "non-mandatory" slave has been found.

Error message	Description
MMC - parsing error at line ... (...)	ERROR: During the analysis of the **mmc-0-conf.csv** configuration file or **mmc-0-ident.csv** identification file, errors in the given line numbers have been detected. The line number starts with '1'.
MMC - parsing file succeeded ...	INFO: The given **mmc-0-conf.csv** configuration file or **mmc-0-ident.csv** identification file have been analysed without any errors.
MMC - service ... started, configuration '...'	INFO: A service has been started with the given configuration.
MMC - service stopped	INFO: A service has been stopped by the user via the L_ETC_MMCController function block or per reset.
MMC - slave ... ident data failed - CoE ... (set .../ act ...)	ERROR: While the EtherCAT bus is booted and the additional identification parameters from the **mmc-0-ident.csv** configuration file are checked, an error has occurred in the given slave. The expected and current parameter contents do not match.
MMC - slave ... ident failed - CoE ... (error ...)	ERROR: While the EtherCAT bus is booted and the additional identification parameters from the **mmc-0-ident.csv** configuration file are checked, an error has occurred in the given slave. For further information see General error codes (L_ETC_ERRORCODE).
Modular Machine Configuration: Only one instance of L_ETC_MMCController allowed!	ERROR: When the EtherCAT master configuration is generated, this message is sent if more than one instance of the L_ETC_MMCController function block is used in the application. **Caution:** An instance might also have been created in a library!

13.6.3 Lenze Controller logbook messages

Lenze Controller logbook messages are displayed in the »WebConfig« as errors (highlighted in red), warnings, or information.

The same message text is displayed in the »PLC Designer« logbook and the EtherCAT logbook.

▸ Logbook of the Lenze Controller in the »WebConfig« (📖 190)

Note!

Cyclical messages as "WKC Error" or "Frame Lost" are displayed at the 1., 10., 100., 1000., 10000. etc. occurrence.

Error no.	Message text in the Lenze Controller logbook	Description
5063	... (...): CoE 0x... : ... - invalid slave state	CAN over EtherCAT: CoE is only provided from the "Pre-Operational" state onwards. An attempt has been made to access a CoE parameter in the "Bootstrap" or "Init" state.
5064	... (...): CoE 0x... : ... - SDO abort 'toggle bit unchanged (0x05030000)'	The status of the toggle bit has not changed.
5065	... (...): CoE 0x... : ... - SDO abort 'Protocol timeout (0x05040000)'	SDO protocol time-out
5066	... (...): CoE 0x... : ... - SDO abort 'Client/server command specifier invalid or unknown (0x05040001)'	Invalid or unknown specification symbol for the client/server command
5067	... (...): CoE 0x... : ... - SDO abort 'Invalid block size (only block mode) (0x05040002)'	Invalid block size (only in "Block mode")
5068	... (...): CoE 0x... : ... - SDO abort 'Invalid sequence number (only block mode) (0x05040003)'	Invalid sequence number (only in "Block mode")
5069	... (...): CoE 0x... : ... - SDO abort 'CRC error (only block mode) (0x05040004)'	CRC error (only in "Block mode")
5070	... (...): CoE 0x... : ... - SDO abort 'Memory overflow (0x05040005)'	The space in the main memory is not sufficient.
5071	... (...): CoE 0x... : ... - SDO abort 'Not supported access to object (0x06010000)'	Access to object not supported
5072	... (...): CoE 0x... : ... - SDO abort 'Attempt to read a write-only object (0x06010001)'	Read access to a write-only object
5073	... (...): CoE 0x... : ... - SDO abort 'Attempt to write to a read-only object (0x06010002)'	Write access to a read-only object
5074	... (...): CoE 0x... : ... - SDO abort 'Object does not exist in the object directory (0x06020000)'	An object does not exist in the object directory
5075	... (...): CoE 0x... : ... - SDO abort 'Object cannot be mapped to the PDO (0x06040041)'	An object cannot be mapped into the PDO
5076	... (...): CoE 0x... : ... - SDO abort 'Number and length of the objects to be mapped are greater than the PDO length (0x06040042)'	The number and/or length of the objects mapped would exceed the PDO length
5077	... (...): CoE 0x... : ... - SDO abort 'General incompatibility of the parameters (0x06040043)'	General parameter incompatibility
5078	... (...): CoE 0x... : ... - SDO abort 'General internal incompatibility within the device (0x06040047)'	General internal device incompatibility
5079	... (...): CoE 0x... : ... - SDO abort 'Access failed due to a hardware error (0x06060000)'	Access has failed due to a fault in the hardware
5080	... (...): CoE 0x... : ... - SDO abort 'Data format incompatible, length of the service parameter incompatible (0x06070010)'	The data type or the parameter length does not correspond
5081	... (...): CoE 0x... : ... - SDO abort 'Data format incompatible, service parameter too long (0x06070012)'	Incorrect data type (The parameter length is too large)
5082	... (...): CoE 0x... : ... - SDO abort 'Data format incompatible, service parameter too short (0x06070013)'	Incorrect data type (The parameter length is too small)
5083	... (...): CoE 0x... : ... - SDO abort 'Subindex does not exist (0x06090011)'	A subindex is not available
5084	... (...): CoE 0x... : ... - SDO abort 'Write access - parameter value beyond the permissible range (0x06090030)'	The value range for parameters is too great (only for write access)
5085	... (...): CoE 0x... : ... - SDO abort 'Write access - parameter value too high (0x06090031)'	The parameter value is too high
5086	... (...): CoE 0x... : ... - SDO abort 'Write access - parameter value too low (0x06090032)'	The parameter value is too low

Error no.	Message text in the Lenze Controller logbook	Description
5087	... (...): CoE 0x... : ... - SDO abort ’Maximum value is lower than minimum value (0x06090036)’	The maximum value is lower than the minimum value
5088	... (...): CoE 0x... : ... - SDO abort ’General error (0x08000000)’	General error
5089	... (...): CoE 0x... : ... - SDO abort 'Data cannot be transferred or saved in the application (0x08000020)'	Data cannot be transferred/saved to the application.
5090	... (...): CoE 0x... : ... - SDO abort 'Data cannot be transferred or saved in the application due to local control (0x08000021)'	Data cannot be transferred/saved to the application because of local control.
5091	... (...): CoE 0x... : ... - SDO abort 'Data cannot be transferred or saved in the application due to device status (0x08000022)'	Due to the current device state, data cannot be transferred to the application or stored in the application
5092	... (...): CoE 0x... : ... - SDO abort ’Dynamic generation of the object directory failed, or the object directory is missing (0x08000023)’	The dynamic generation of an object directory has failed, or no object directory is available.
5093	... (...): CoE 0x... : ... - SDO abort ’Unknown code’	Unknown code
5094	... (...): CoE 0x... : ... - invalid parameter	CAN over EtherCAT: An invalid parameter was transferred to a CoE function block (e.g: 'timeout = 0' or invalid slave address).
5095	... (...): CoE 0x... : ... - CoE protocol is not supported.	CAN over EtherCAT: An attempt was made to access a slave parameter, but the slave supports no CoE protocol.
5096	SLV: Undefined FoE error	File over EtherCAT: Manufacturer-specific error (see slave documentation)
5097	SLV: FoE error - not found	File over EtherCAT: Internal error
5098	SLV: FoE error - access denied	File over EtherCAT: No access to file
5099	SLV: FoE error - storage medium full	File over EtherCAT: No memory space to store file
5100	SLV: FoE error - illegal	File over EtherCAT: Internal error
5101	SLV: FoE error - wrong package number	File over EtherCAT: Internal error
5102	SLV: FoE error - already existing	File over EtherCAT: Internal error
5103	SLV: FoE error - user missing	File over EtherCAT: Internal error
5104	SLV: FoE error - bootstrap only	File over EtherCAT: Transfer only permitted in bootstrap.
5105	SLV: FoE error - no bootstrap	File over EtherCAT: Transfer only permitted in bootstrap.
5106	SLV: FoE error - no rights	File over EtherCAT: No access authorisation
5107	SLV: FoE - program error	File over EtherCAT: Internal error
5108	SLV: FoE - invalid parameter	File over EtherCAT: An invalid parameter was transferred to a FoE function block (e.g. 'timeout = 0' or invalid slave address).
5513	...: State change from '...' to '...'	Info: The EtherCAT master has successfully executed a state change.
5518	...: CoE - SDO download failed. statVal=..., errCode=0x... (...)	CAN over EtherCAT: Internal error - error during CoE download: parameter, mailbox status, error code
5519	...: CoE - SDO upload failed. statVal=..., errCode=0x... (...)	CAN over EtherCAT: Internal error - error during CoE upload: parameter, mailbox status, error code
5520	...: CoE - OD list upload failed. statVal=..., errCode=0x... (...)	CAN over EtherCAT: Internal error - error uploading the object directory: parameter, mailbox status, error code
5521	...: CoE - Object entry description upload failed. statVal=..., errCode=0x... (...)	CAN over EtherCAT: Internal error - error uploading the object directory/parameter description: parameter, mailbox status, error code
5522	...: CoE - Object entry description upload failed. statVal=..., errCode=0x... (...)	CAN over EtherCAT: Internal error - error uploading the object directory/parameter description: parameter, mailbox status, error code
5523	...: CoE - emergency transfer failed. statVal=..., errCode=0x... (...)	CAN over EtherCAT: Internal error during emergency message transfer
5524	... (...): CoE - emergency request. id=0x..., len=..., errCode=0x..., ErrReg=0x..., data: 0x... 0x... 0x... 0x... 0x...	CAN over EtherCAT: Internal error during emergency message transfer **Note:** "data: ..." indicates by means of codes which error has occurred in which slave device/module. Detailed information on the coding of error messages is provided in the documentation for the corresponding slave device/module.
5525	...:Cyclical command WKC error command: ... - logic/physical address: 0x... - WKC act/set=.../...	Error during cyclical command: One or more slaves have not processed the command.
5526	...: Master init command WKC error - command: ... - logic/ physical address: 0x..., WKC act/set=.../...	Error during the initialisation command: One or more slaves have not processed the command.
5527	... (...): Slave init command WKC error. Command: ..., logic/ physical address: 0x..., WKC act/set=.../...	Error during the initialisation command: One or more slaves have not processed the command.
5528	... (...): EoE receive WKC error. Command: ..., logic/physical address: 0x..., WKC act/set=.../...	Ethernet over EtherCAT: error during initialisation command for 'EoE Receive Request' One or more slaves have not processed the command.

Error no.	Message text in the Lenze Controller logbook	Description
5529	... (...): CoE receive WKC error. Command: ..., logic/physical address: 0x..., WKC act/set=.../...	CAN over EtherCAT: error during initialisation command for 'CoE Receive Request' One or more slaves have not processed the command.
5530	... (...): FoE receive WKC error. Command: ..., logic/physical address: 0x..., WKC act/set=.../...	File over EtherCAT: error during initialisation command for 'FoE Receive Request' One or more slaves have not processed the command.
5531	... (...): SoE receive WKC error. Command: ..., logic/physical address: 0x..., WKC act/set=.../...	Sercos over EtherCAT: error during initialisation command for 'SoE Receive Request' One or more slaves have not processed the command.
5532	... (...): EoE send WKC error. Command: ..., logic/physical address: 0x..., WKC act/set=.../...	Ethernet over EtherCAT: error during initialisation command for 'EoE Send Request' One or more slaves have not processed the command.
5533	... (...): CoE send WKC error. Command: ..., logic/physical address: 0x..., WKC act/set=.../...	CAN over EtherCAT: error during initialisation command for 'CoE Send Request' One or more slaves have not processed the command.
5534	... (...): FoE send WKC error. Command: ..., logic/physical address: 0x..., WKC act/set=.../...	File over EtherCAT: error during initialisation command for 'FoE Send Request' One or more slaves have not processed the command.
5535	... (...): SoE send WKC error. Command: ..., logic/physical address: 0x..., WKC act/set=.../...	Sercos over EtherCAT: error during initialisation command for 'SoE Send Request' One or more slaves have not processed the command.
5541	... (...): Error at response to Init command - no response. State change='...'	Slave does not respond to Init command
5542	... (...): Error at response to Init command - validation error. State change='...'	Slave does not respond correctly to Init command
5543	... (...): Error at response to Init command - failed. State change='...'	Init command cannot be written to slave
5544	...: Error at response to master init command - no response. State change='...'	Slaves do not respond to Init command (broadcast)
5545	...: Error at response to master init command - validation error. State change='...'	Slaves do not respond correctly to Init command (broadcast)
5546	... (...): EtherCAT command ... is missing in the Ethernet frame. Index of the missing command in the Ethernet frame=...	Internal error
5547	... (...): Mailbox init command timeout. Current state change of the slave='...'	Time-out during mailbox initialisation in the case of a state change
5549	...: Ethernet cable is connected	Ethernet cable reconnected (link-up available)
5550	...:Ethernet cable is not connected	Ethernet cable is not connected (no link-up available)
5551	...: Timeout for cyclical commands. Time between transmission processes too long	Internal error
5552	...: Redundant operation. Ethernet cable missing at the 2. EtherCAT interface	Lenze controller does not support redundancy.
5554	... (...): Slaves signals error. AL status: '...' (0x...), AL status code: '...' (0x...)	Slave indicates an error. AL status and AL status code are slave-specific.
5555	... (...): Communication to device interrupted	Connection to slave is interrupted. The slave does not respond or is no longer available.
5557	...: DC slaves 'in-sync'. Deviation ... ns	Information that the DC deviation is within the permissible limits (standard 8 µs).
5558	...: DC slaves 'out-of-sync'. Deviation ... ns	Information that the DC deviation is not within the permissible limits (standard 8 µs).
5562	...: Client registration lost	Internal error
5704	...: Connection was changed: connection is established	Internal information
5705	...: Connection was changed. Cookie: 0x... cause: ... (0x...)	Internal information
5706	...: Client was registered by cookie 0x... instance 0x... Id 0x... result ...	Internal information
5707	...: Client was deleted by cookie 0x... instance 0x... Id 0x... result ...	Internal information
5708	...: Unknown registration: 0x...	Internal error: Master outputs unknown message.
5719	... (...): Slave has incorrect status. Status set/act '...'/'...'	Slave is not in expected status: 'req' is the requested status and 'act' the current status
5729	...: RAP - invalid parameter size for ...	Internal error
5730	...: RAP - Marshaling error. Cookie: 0x..., Command: 0x..., Cause: ... (0x...), Protocol Header: 0x...	Internal error
5731	...: ... could not be included in queue (missing calls of the ProcessNotificationJobs)	Internal error

Error no.	Message text in the Lenze Controller logbook	Description
5733	...: ecatSetTargetState - EtherCAT master could not be set to "target state". The master is busy (timeout)	Time-out during status change 'Request'
5740	...: Error 0x... when reading out the bus scan status	Internal error: Internal bus scan failed.
5743	...: Error 0x... during restart of the bus scan	Internal error: Internal bus scan failed.
6200	... (...): Configuration mismatch. VendorID check failed (0x... / 0x...)	Configured and current bus structure do not match. At the indicated position, a slave with the following vendor ID is expected (expected/current).
6201	... (...): Configuration mismatch. ProductCode check failed (0x... / 0x...)	Configured and current bus structure do not match. At the indicated position, a slave with the following vendor product code is expected (expected/current).
6202	... (...): Configuration mismatch. Revision check failed (0x... / 0x...)	Configured and current bus structure do not match. At the indicated position, a slave with the following revision is expected (expected/current).
6203	... (...): Configuration mismatch. VendorID check failed (0x... / missing...)	Configured and current bus structure do not match. At the indicated position, a slave with the following vendor ID is expected (currently, there is no slave available here).
6204	...: Configuration mismatch. Too many slaves after '... (...)' on the bus. Identification 0x... / 0x... / 0x...)	Configured and current bus structure do not match. More slaves than configured are connected to the bus. For the first odd slave, vendor ID, product code and revision are indicated.
6212	...: All slaves 'Operational' again	Information: Slaves reset from "Operational" state to a lower state were set to "Operational" again (e.g. by means of L_ETC_SetSlaveState())
6213	...: Cyclical command WKC error (repeated ... times)	One or several slaves do not process the commands of the cyclic frames. Possible causes may e.g. be that slaves are no longer available or were reset from the "Operational" status to a lower status. Only the 1st, 10th, 100th, 1000th, 10000th, etc. error is logged.
6214	...: Frame response error (repeated ... times)	Master did not receive transmitted EtherCAT frame with the next bus cycle task call. Causes may be wiring errors, contact problems, and an excessive cycle time load of the EtherCAT task.
6215	...: Not all slaves 'Operational' (repeated ... times)	The master is in the "Operational" state and one or more slaves are falling back to a lower status. Only the 1st, 10th, 100th, 1000th, 10000th, etc. error is logged.
6216	... (...): Emergency message overflow. Further emergency messages are blocked	A slave repeatedly sends the same emergency message. After receipt of five messages, this error message appears and further emergency messages from the slave are no longer logged until the status of the slave changes from "Init" to "Pre-Operational".
6220	...: New configuration loaded	Information: New IEC application with EtherCAT master was loaded.
6221	...: New configuration loaded. No slaves defined	Information: New IEC application with EtherCAT master was loaded. There are no slaves defined.
6222	...: Master start failed. Configuration error	Internal error: Master was not configured correctly.
6230	...: Master start failed	The master cannot be started. General message (no 'Bus mismatch', DC/DCM, cable problem). Observe previous messages in the logbook!
6231	...: Master start failed. Bus mismatch	The master does not start due to a 'Bus mismatch': Configured and current bus structure do not match. Which slave is not correct is logged shortly before this message.
6232	...: Master start failed. EtherCAT cable not connected	The master cannot be started because the Ethernet cable is not connected (no link-up).
6233	...: Master start failed. DC/DCM configuration	Internal error: The master cannot be started because of a faulty DC/DCM configuration.
6234	...: Master start failed. Slaves cannot be set to Pre-Operational.	The master cannot be started because of a slave error. Observe previous messages (slave error) in the logbook!
6240	...: Status change 'Operational' failed (0x...)	The master cannot be set to "Operational". General message, none of the following errors. Observe previous messages in the logbook!
6241	...: Status change 'Operational' failed. Master is not initialised	Internal error: The master cannot be started because of a faulty DC/DCM configuration.
6242	...: Status change 'Operational' failed. EtherCAT cable not connected	The master cannot be set to "Operational" because the Ethernet cable is not connected (no link-up).
6243	...: Status change 'Operational' failed. DCM is not in-sync	The master cannot be set to "Operational" because the Ethernet cable is not connected (no link-up).
6244	...: Status change 'Operational' failed. Times for bus cycle task and DC are not identical	Internal error: The DC cycle time and the cycle time of the EtherCAT bus cycle task are not identical.

Error no.	Message text in the Lenze Controller logbook	Description
6245	...: Status change 'Operational' takes some time ...	Information: The transition from "Safe-operational" -> "Operational" takes longer. This message will be output after 10 s if one or several slaves did not change to the "Operational" state. E.g. for Servo Drives 9400 with long cycle times, because the standard device must synchronise with the communication module (SYNC0).
6246	...: Status change 'Operational' failed. Time-out.	The master cannot be set to "Operational" because of a time-out (standard 55 s).
6247	...: Status change 'Operational' failed. Slave error.	Setting the master to "Operational" failed due to a slave error. Observe previous messages (slave error) in the logbook!
6248	...: Status change 'Operational' cancelled by reset command.	The master cannot be set to "Operational" because the user cancelled the procedure.
6250	...: Master stop failed (0x...).	The master cannot be stopped or cannot be set to the "Init" state.
6251	...: Master stop failed (0x...). Slaves cannot be set to Pre-operational.	The master cannot be stopped and slaves cannot be set to the "Pre-operational" state.
6260	...: Master shutdown failed (0x...).	The master cannot be shut down and cannot be set to the "Init" state.
6270	...: Remote API server start failed	Internal error: The remote API server cannot be started. Communication of CoE parameters from the engineering tool (»EASY Starter«/»Engineer«) is not possible.
6280	...: Start download	Information: Firmware/parameter set download was started.
6281	...: Download complete.	Information: Firmware/parameter set download was completed.

13.6.4 SDO abort codes

The abort codes are relevant for ...

- the *eErrorCode* output in some function blocks of the L_IODrvEtherCAT function library (📖 119);
- SDO read/write error messages, cause by SDO queries from the system (e.g. initialisation code or SDO queries from the engineering tool).

Error number [hex]	Description
0x00000000	No error
0x05030000	The status of the toggle bit has not changed.
0x05040000	SDO protocol time-out
0x05040001	Invalid or unknown specification symbol for the client/server command
0x05040002	Invalid block size (only in "Block mode")
0x05040003	Invalid sequence number (only in "Block mode")
0x05040004	CRC error (only in "Block mode")
0x05040005	The space in the main memory is not sufficient.
0x06010000	Access to object not supported
0x06010001	Read access to a write-protected object
0x06010002	Write access to a write-protected object
0x06020000	An object does not exist in the object directory
0x06040041	An object cannot be mapped into the PDO
0x06040042	The number and/or length of the objects mapped would exceed the PDO length
0x06040043	General parameter incompatibility
0x06040047	General internal device incompatibility
0x06060000	Access has failed due to a fault in the hardware
0x06070010	The data type or the parameter length does not correspond
0x06070012	Incorrect data type (The parameter length is too large)
0x06070013	Incorrect data type (The parameter length is too small)
0x06090011	A subindex is not available
0x06090030	The value range for parameters is too great (only for write access)
0x06090031	The parameter value is too high
0x06090032	The parameter value is too low
0x06090036	The maximum value is lower than the minimum value
0x08000000	General error
0x08000020	Data cannot be transferred/saved to the application.
0x08000021	Data cannot be transferred/saved to the application because of local control.
0x08000022	Due to the current device state, data cannot be transferred to the application or stored in the application
0x08000023	The dynamic generation of an object directory has failed, or no object directory is available.

14 Parameter reference

This chapter complements the parameter list of the online help for the Lenze Controller by the parameters of the EtherCAT communication interface.

These parameters ...

- are for instance shown in the Lenze »WebConfig« (Engineering tool for web-based parameterisation);
- are listed in numerically ascending order.

C280/4

Parameter | Name: **C280/4 | ECAT bus scan compliance**		Data type: UNSIGNED_8 Index: 24295.4 = 0x5EE7.0x04
Brief information on whether the master configuration corresponds to the physical bus structure. The master configuration of the stack is compared to the actual bus structure.		
Selection list(Lenze setting printed in bold)		**Info**
0	**No match**	The master configuration does not correspond to the bus structure.
1	OK	The master configuration corresponds to the bus structure.
☑ Read access ☐ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer		

C281/2

Parameter | Name: **C281/2 | ECAT master: State**		Data type: UNSIGNED_8 Index: 24294.2 = 0x5EE6.0x02
Display of the current state of the master		
Selection list (read only)		
0	Unknown	
1	Init	
2	Pre-Operational	
3	Bootstrap mode	The Bootstrap mode is not supported
4	Safe-operational	
8	Operational	
☑ Read access ☐ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer		

C281/5

Parameter | Name: **C281/5 | ECAT master: State info**		Data type: UNSIGNED_32 Index: 24294.5 = 0x5EE6.0x05
Display of additional information on the current state of the master The bits are set to the value 1 when the respective states are reached.		
Value is bit-coded:		
Bit 0	Master ok	
Bit 1	Reserved 1	
Bit 2	Reserved 2	
Bit 3	Reserved 3	
Bit 4	Init	
Bit 5	Pre-Operational	
Bit 6	Safe-operational	
Bit 7	Operational	
Bit 8	Slaves in requested state	
Bit 9	Master in requested state	
Bit 10	Bus Scan Match	
Bit 11	Reserved 4	
Bit 12	DC: Activated	
Bit 13	DC: Synchronised	
Bit 14	DC: Busy	
Bit 15	Reserved 5	
Bit 16	Link Up	
Bit 17	Reserved 6	
...	...	
Bit 31	Reserved 20	
☑ Read access ☐ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer		

C281/6

Parameter | Name: **C281/6 | ECAT bus scan**		Data type: UNSIGNED_8 Index: 24294.6 = 0x5EE6.0x06
Activation of the fieldbus scan The fieldbus scan updates all EtherCAT codes.		
Selection list(Lenze setting printed in bold)		
0	**No Operation**	
1	The bus is scanned	
☑ Read access ☑ Write access ☐ CINH ☐ PLC-STOP ☑ No transfer		

C282/2

Parameter | Name: **C282/2 | ECAT DC: Slave sync deviation limit**			Data type: UNSIGNED_32 Index: 24293.2 = 0x5EE5.0x02
Permissible deviation of the distributed clocks of all devices in nanoseconds. If the permissible deviation is exceeded, the master will initiate a resynchronisation of the distributed clocks.			
Display range (min. value | unit | max. value)			
0	ns	4294967295	
☑ Read access ☐ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer			

C282/3

Parameter | Name: **C282/3 | ECAT DC: Current deviation**			Data type: INTEGER_32 Index: 24293.3 = 0x5EE5.0x03
Current maximum deviation of the distributed clocks of all devices in nanoseconds.			
Display range (min. value | unit | max. value)			
-2147483647	ns	2147483647	
☑ Read access ☐ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer			

C286/3

Parameter | Name: **C286/3 | ECAT bus: No. of slaves**			Data type: UNSIGNED_32 Index: 24289.3 = 0x5EE1.0x03
Number of slaves connected to the fieldbus			
Display range (min. value | unit | max. value)			
0		4294967295	
☑ Read access ☐ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer			

C286/4

Parameter | Name: **C286/4 | ECATBus: No. of DC slaves**			Data type: UNSIGNED_32 Index: 24289.4 = 0x5EE1.0x04
Number of slaves connected to the fieldbus and supported by distributed clocks			
Display range (min. value | unit | max. value)			
0		4294967295	
☑ Read access ☐ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer			

C286/5

Parameter | Name: **C286/5 | ECAT config.: No. of slaves**			Data type: UNSIGNED_32 Index: 24289.5 = 0x5EE1.0x05
Number of slaves configured in the master configuration file			
Display range (min. value | unit | max. value)			
0		4294967295	
☑ Read access ☐ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer			

C286/6

Parameter | Name: **C286/6 | ECAT config.: No. of mailbox slaves**			Data type: UNSIGNED_32 Index: 24289.6 = 0x5EE1.0x06
Number of mailbox slaves configured in the master configuration file			
Display range (min. value | unit | max. value)			
0		4294967295	
☑ Read access ☐ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer			

C286/7

Parameter | Name: **C286/7 | ECAT counter: Tx frames**			Data type: UNSIGNED_32 Index: 24289.7 = 0x5EE1.0x07
Number of sent frames			
Display range (min. value | unit | max. value)			
0		4294967295	
☑ Read access ☐ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer			

C286/8

Parameter | Name: **C286/8 | ECAT counter: Rx frames**			Data type: UNSIGNED_32 Index: 24289.8 = 0x5EE1.0x08
Number of received frames			
Display range (min. value | unit | max. value)			
0		4294967295	
☑ Read access ☐ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer			

C286/9

Parameter | Name: **C286/9 | ECAT counter: Lost frames**			Data type: UNSIGNED_32 Index: 24289.9 = 0x5EE1.0x09
Number of lost frames			
Display range (min. value | unit | max. value)			
0		4294967295	
☑ Read access ☐ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer			

C286/10

Parameter | Name: **C286/10 | ECAT counter: Cyclic frames**			Data type: UNSIGNED_32 Index: 24289.10 = 0x5EE1.0x0A
Number of cyclic frames			
Display range (min. value | unit | max. value)			
0		4294967295	
☑ Read access ☐ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer			

C286/11

Parameter | Name: **C286/11 | ECAT counter: Cyclic datagrams**			Data type: UNSIGNED_32 Index: 24289.11 = 0x5EE1.0x0B
Number of cyclic datagrams			
Display range (min. value | unit | max. value)			
0		4294967295	
☑ Read access ☐ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer			

C286/12

Parameter | Name: **C286/12 | ECAT counter: Acyclic frames**			Data type: UNSIGNED_32 Index: 24289.12 = 0x5EE1.0x0C
Number of acyclic frames			
Display range (min. value | unit | max. value)			
0		4294967295	
☑ Read access ☐ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer			

C286/13

Parameter | Name: **C286/13 | ECAT counter: Acyclic datagrams**			Data type: UNSIGNED_32 Index: 24289.13 = 0x5EE1.0x0D
Number of acyclic datagrams			
Display range (min. value | unit | max. value)			
0		4294967295	
☑ Read access ☐ Write access ☐ CINH ☐ PLC-STOP ☐ No transfer			

C286/14

Parameter | Name: **C286/14 | ECAT Clear specific counters**		Data type: UNSIGNED_32 Index: 24289.14 = 0x5EE1.0x0E
Reset frame and datagram counters (C1086/7 ... 13)		
Selection list(Lenze setting printed in bold)		
0	**No Operation**	
1	Reset - All counters	
2	Reset - Tx frame counters	
4	Reset - Rx frame counters	
8	Reset - Lost frame counters	
16	Clear Cyclical Frame Counter	
32	Clear Cyclical Datagram Counter	
64	Reset - Acyclic frame counters	
128	Reset - Acyclic datagram counters	
☑ Read access ☑ Write access ☐ CINH ☐ PLC-STOP ☑ No transfer		

Index

Index

FEEDBACK

Your opinion is important to us

These instructions were created to the best of our knowledge and belief to give you the best possible support for handling our product.

Perhaps we have not succeeded in achieving this objective in every respect. If you have suggestions for improvement, please e-mail us to:

feedback-docu@lenze.com

Thank you very much for your support.

Your Lenze documentation team

Lenze Automation GmbH
Postfach 10 13 52, 31763 Hameln
Hans-Lenze-Straße 1, 31855 Aerzen
GERMANY
HR Hannover B 205381
+49 5154 82-0
+49 5154 82-2800
lenze@lenze.com
www.lenze.com

Service

Lenze Service GmbH
Breslauer Straße 3, 32699 Extertal
GERMANY
008000 24 46877 (24 h helpline)
+49 5154 82-1112
service@lenze.com

PLC Designer

PLC
3

Application Sample i700 - Lenze teachware case _______________

Software Manual EN

13479663

Lenze

1 About this documentation

This documentation describes the sample project for the i700 servo inverter for commissioning a Lenze automation system. The used automation system consists of a PLC for "Controller-based Automation" and drive components connected (via a bus system).

Read the mounting instructions accompanying the controller first before you start working!

The mounting instructions include safety instructions which must be observed!

Tip!

Current information on the Lenze products can be found at: http://www.Lenze.com

This documentation is part of the "Controller-based Automation" manual collection. The manual collection consists of the following documents:

Documentation type	Subject
System manuals	System overview/sample topologies • Controller-based Automation • Visualisation
Communication manuals	Bus systems • Controller-based Automation EtherCAT® • Controller-based Automation CANopen® • Controller-based Automation PROFIBUS® • Controller-based Automation PROFINET®
Online helps/ Software manuals	Lenze Engineering tools • »PLC Designer«: Programming • »Engineer«: Configuring drives • »VisiWinNET® Smart«: Visualisation • »Backup & Restore«: Back up/restore data

Technical documentation for Lenze products

Further information on Lenze products which can be used in connection with "Controller-based Automation" can be found in the following documentation:

Mounting & wiring
🕮 **Mounting instructions** • Controllers • Communication cards (MC-xxx) • I/O system 1000 (EPM-Sxxx) • Inverter • Communication modules
Using sample application/application template
☐ **Online help/software manuals** • Application Sample i700 - Lenze teachware case • Application samples Inverter Drives 8400/Servo Drives 9400 • ApplicationTemplate Lenze standard • ApplicationTemplate PackML standard
Parameterisation, configuration, commissioning
☐ **Online help/software manuals** • L-force Controller • i700 servo inverter • Servo Drives 9400 HighLine/PLC/ regenerative power supply module • Inverter Drives 8400 StateLine/HighLine/TopLine • I/O system 1000 (EPM-Sxxx)
☐ **Online help/communication manuals** • Bus systems • Communication modules

Icons
🕮 Printed documentation ☐ Online help in the Lenze Engineering tool/ software manuals and communication manuals are provided as PDF files in the download area of the Lenze website.

Target group

This documentation addresses to all persons who plan, commission, and program a Lenze automation system on the basis of the Lenze "ApplicationTemplate" as part of the "Controller-based Automation".

Screenshots/application examples

All screenshots in this documentation are application examples. Depending on the firmware version of the Lenze device and the software version of the engineering tools installed (here: »PLC Designer«), the representation of the actual screen display may deviate.

Information on validity

The information in this documentation is valid for the following Lenze software:

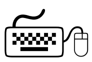
Software	From software version
»PLC Designer«	3.9.x

1.1 Document history

Version			Description
1.0	09/2012	TD11	First edition
1.1	01/2015	TD11	Updated to the »PLC Designer« V3.9.

1.2 Conventions used

This documentation uses the following conventions to distinguish between different types of information:

Type of information	Highlighting	Examples/notes
Spelling of numbers		
Decimal separator	Point	The decimal point is always used. Example: 1234.56
Hexadecimal number	0x	For hexadecimal numbers, the "0x" prefix is used. Example: 0x60F4
Binary number	0b	For binary numbers, the "0b" prefix is used. Example: 0b00010111
Text		
Version information	Blue text colour	All information that applies to from a certain software version of the drive onwards are marked accordingly in this documentation. Example: This function extension is available as from software version V3.0!
Program name	» «	»PLC Designer«
Window	*italics*	The *Message window*... / the *Options* dialog box...
Variable names		By setting *bEnable* to TRUE...
Control element	**bold**	The **OK**... button / The **Copy**... command / The **Properties**... tab / The **Name** ... input field
Sequence of menu commands		If the execution of a function requires several commands in a row, the individual commands are separated by an arrow: Select **File→Open** to...
Shortcut	**<bold>**	Use **<F1>** to open the online help.
		If a key combination is required for a command, a "+" is placed between the key identifiers: With **<Shift>+<ESC>**...
Hyperlink	Underlined	Optically highlighted reference to another topic. It is activated with a mouse-click in this online documentation.
Icons		
Page reference	(🕮 5)	Optically highlighted reference to another page. In this online documentation activated via mouse-click.
Step-by-step instructions		Step-by-step instructions are indicated by a pictograph.

1.3 Notes used

The following signal words and symbols are used in this documentation to indicate dangers and important information:

Safety instructions

Layout of the safety instructions:

Pictograph and signal word!

(characterise the type and severity of danger)

Note

(describes the danger and gives information about how to prevent dangerous situations)

Pictograph	Signal word	Meaning
	**Danger!**	**Danger of personal injury through dangerous electrical voltage** Reference to an imminent danger that may result in death or serious personal injury if the corresponding measures are not taken.
	**Danger!**	**Danger of personal injury through a general source of danger** Reference to an imminent danger that may result in death or serious personal injury if the corresponding measures are not taken.
	**Stop!**	**Danger of property damage** Reference to a possible danger that may result in property damage if the corresponding measures are not taken.

Application notes

Pictograph	Signal word	Meaning
	**Note!**	Important note to ensure trouble-free operation
	**Tip!**	Useful tip for easy handling
		Reference to another document

2 Safety instructions

Please observe the following safety instructions when you want to commission a controller or system using the »Engineer«.

Read the documentation supplied with the controller or the individual components of the system carefully before you start commissioning the devices with the »Engineer«!

The device documentation contains safety instructions which must be observed!

Danger!

According to today's scientific knowledge it is not possible to ensure absolute freedom from defects of a software product.

If necessary, systems with built-in controllers must be provided with additional monitoring and protective equipment complying with the relevant safety regulations (e.g. law on technical equipment, regulations for the prevention of accidents) in each case, so that an impermissible operating status does not endanger persons or facilities.

During commissioning persons must keep a safe distance from the motor or the machine parts driven by the motor. Otherwise there is a risk of injury by the moving machine parts.

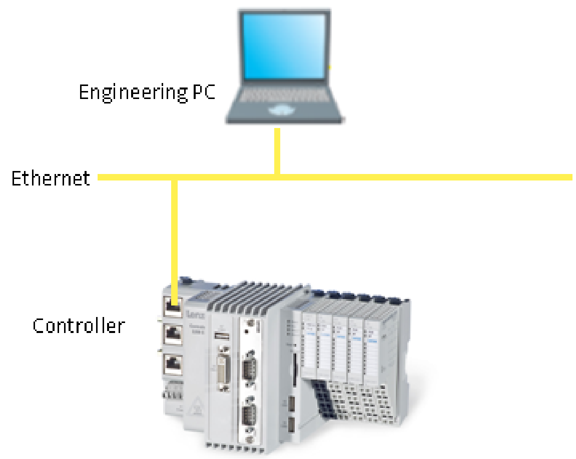
Stop!

If you change parameters in the »PLC Designer« during an online connection to the device is established, the changes are directly accepted in the device!

A wrong parameter setting can cause unpredictable motor movements. By an unintended direction of rotation, a too high speed, or jerky operation, the driven machine parts may be damaged!

3 Conditions

3.1 System requirements



	Engineering PC	Controllers
Hardware	PC/notebook	Controller 3200 C/p500 (Logic) from firmware V3.9 onwards
Operating system	Windows 7	Windows CE
Required Lenze software	»PLC Designer« from V3.9	Runtime software (Logic)
Further requirements	-	Bus system • EtherCAT bus system • EtherCAT node

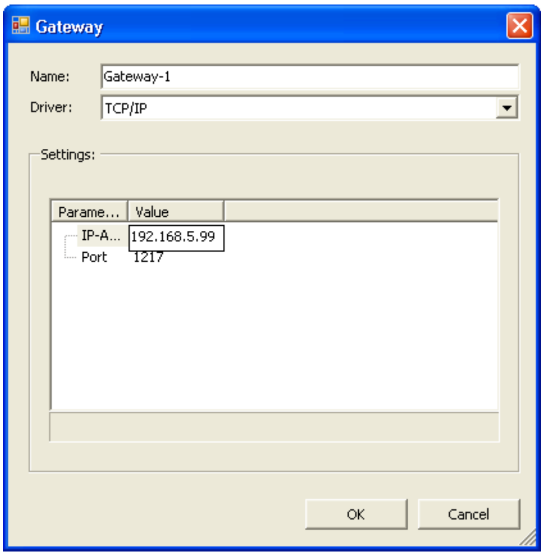
3.2 Setting up communication to the Controller

- Connect the Engineering PC with the controller via a network cable. The »PLC Designer« accesses the controller via Ethernet.
- Make the IP settings with the »PLC Designer« afterwards.

How to check the communication settings:

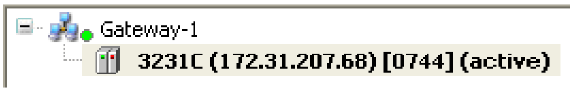
1. Go to the *Device* view and double-click the desired Controller.
2. Make the desired settings on the **Communication settings** tab.
   - Click the **Add gateway** button to insert a gateway.

- Enter the desired IP address of the controller.



[3-1] Example: Enter the IP address of the Controller

3. Click OK to add the controller as gateway.
4. By double-clicking the desired channel (or clicking the **Set active path** button) set the channel selected in the device tree below the gateway as active path for control.
   - Thus, all communication actions directly refer to this channel.
   - The currently active path is represented **in bold** in the list and "(active)" is attached:



5. A device represented in *italics* is set as active path but has not been found during the last network scan.

Note!

- During initial commissioning, observe the following predefined IP addresses of the controllers:: **192.168.5.99**

Further information can be found in the following documentation:

- **Controller** - *Parameter setting & configuration*

4 What is the Application Sample?

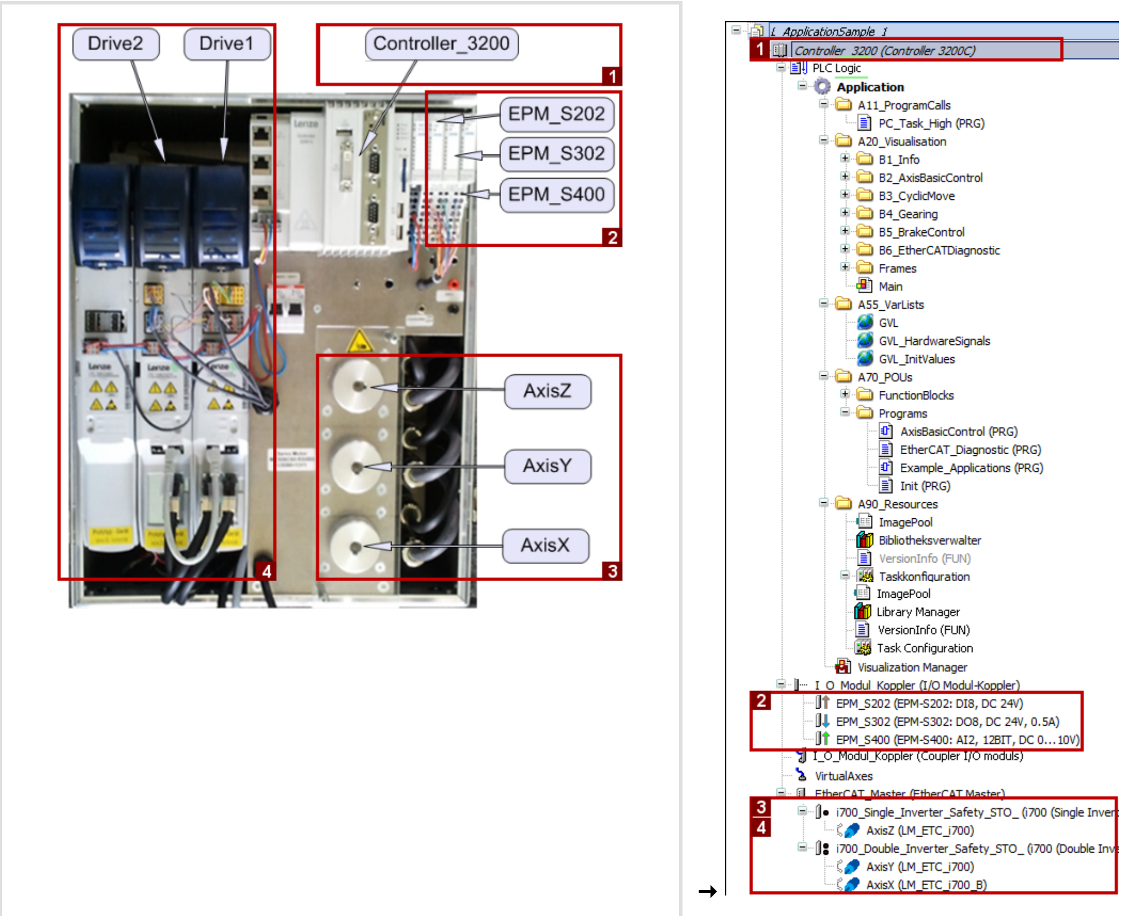
The "Application Sample i700" is a sample project in the »PLC Designer« for commissioning the i700 Lenze teachware case the purpose of which is to facilitate the introduction to the "Controller-based Automation" Lenze system.

- The ready-made sample project enables easy commissioning/diagnostics of the system which consists of a Controller (Controller 3200 C) with a connected I/O system 1000 and i700 servo inverters.

4.1 Target of the Application Sample

The Application Sample i700...

- ... facilitates the introduction to the Lenze "Controller-based Automation" on the basis of the Lenze Teachware case i700.
- ... includes simple examples of commissioning the i700 servo inverters.
- ... can optionally be extended by other hardware configurations.



[4-1] Structure of the Lenze Teachware case i700: It includes the Lenze devices contained in the **Application Sample i700**.

4.2 Overview of the features of the Application Sample

Visualisations

Folder/symbol	Function
AxisBasicControl	Visualisation for the quick commissioning of the controllers of the Lenze application case via the **L_SMC_AxisBasicControl** function.
CyclicMove	Example of application integration in the **L_SMC_AxisBasicControl** FB.
GearingAxisY	Example of integration of PLCopen functionality for coupling/decoupling: Second axis→first axis
BrakeAxisZ	Example of application integration of a holding brake by means of PLCopen functions.
Diagnostic	Example of diagnostics of the EtherCAT master and the status of the controllers.
	"Getting started": Information on the use/structure of the Application Sample

4.3 Elements of the Application Sample

The Application Sample ...

- ... enables easy and quick commissioning of a Lenze Automation system by means of examples.
- ... illustrates the possible applications of the **L_SMC_AxisBasicControl** function block.

4.3.1 The L_SMC_AxisBasicControl block

The **L_SMC_AxisBasicControl** block ...

- ... enables easy traversing of controllers by using PLCopen functionality.
- ... can be controlled via:

1. The visualisation during manual operation
2. The interfaces of the block (via a PLC program)

Controlling the L_SMC_AxisBasicControl block

The Application Sample shows the control possibilities by means of examples:

1. Example of commissioning via visualisation:

   The **AxisBasicControl** visualisation can be used to control the **AxisBasicControl** program.

2. Example of controlling via the interfaces of the **AxisBasicControl** FB:

   The **ExampleApplications** program uses the following blocks which control the interfaces of the corresponding AxisBasicControl block (X axis, Y axis, Z axis):
   - CyclicMove
   - Gearing
   - Brake-Control

   The "programs" are created as function blocks to illustrate by means of an example how software functions can be reused in the »PLC Designer«.

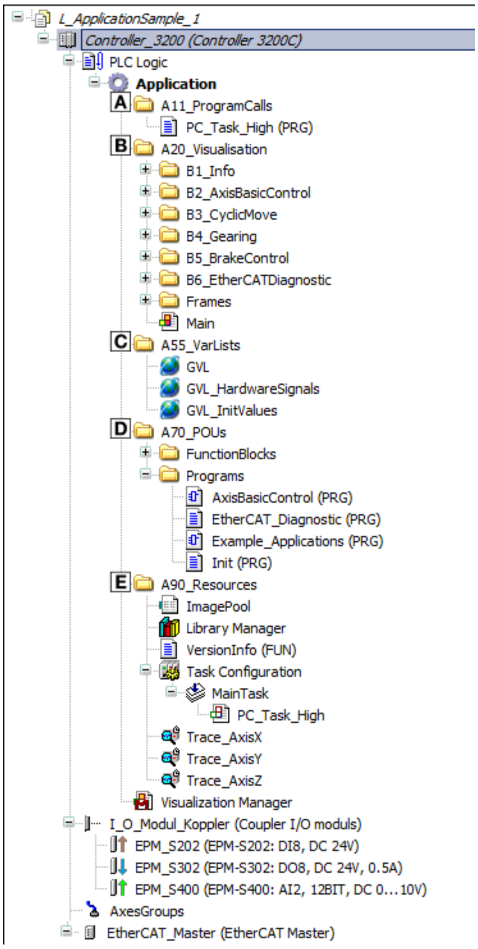
5 Overview - The structure of the Application Sample

The Application Sample i700 is a project template which provides support during commissioning and helps you entering the world of Lenze "Controller-based Automation" with the »PLC Designer«.

- The Application Sample is included in the »PLC Designer« setup as project template:

  L_AppSample_i700.project

- Lenze devices already contained in the project are:
  - I/O modules on the backplane bus of the controller (I/O system 1000),
  - Three axes (*AxisX...Z*) on the EtherCAT fieldbus (single axis of i700 servo inverter each).
- The Application Sample i700 has the following (fixed) structure:



A **A11_ProgramCalls**

- ... contains all program calls.

B **A20_Visualisation**

- ... contains the visualisations for the device-independent functions.

C **A55_VarLists**

- ... contains the declarations of the global variables:
  - Variables for switching over the visualisation: **GVL**
  - I/O variables: **GVL_Hardsignals**
  - Initialisation variables: **GVL_InitValues**

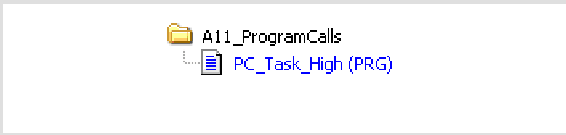
D **A70_POUs**

- ... contains the programs for:
  - BrakeControl/CyclicMove/EtherCAT/Gearing

E **A90_Resources**

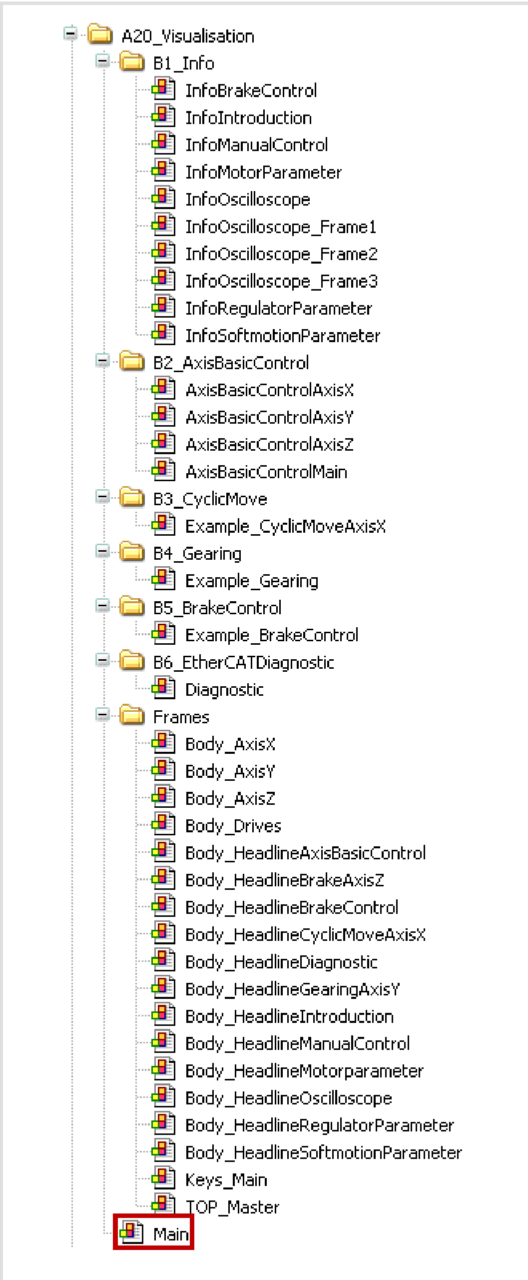
- ... contains all system data such as:
  - task settings,
  - used libraries,
  - predefined traces,
- Visualization manager.

5.1 A11_ProgramCalls



- The **PC_Task_High** program calls all high-priority programs of the application.

5.2 A20_Visualisation



B1_Info: "Getting started"

- ... contains the stored information on the structure and use of the Application Sample.

B2_AxisBasicControl

- ... contains the visualisations for the individual instances of the **L_SMC_AxisBasicControl** block.

B3_CyclicMove

- ... contains the visualisation of the **CyclicMove** block.

B4_Gearing

- ... contains the visualisation of the **Gearing** block.

B5_BrakeControl

- ... contains the visualisation of the **BrakeControl** block.

B6_EtherCATDiagnostic

- ... contains the visualisation of the EtherCAT diagnostic of the application.

Main calls the user interface of the main page of the visualisation which serves to run the entire visualisation.

- ... contains all visualisations of subfolders **B1**-**B6**.

5.3 A70_POUs

BrakeControl

- ... contains the instanceable sample program, **BrakeControl (FB)**, including the brake function, which can be used with the following function blocks:
  - **L_SM3_AxisBasicControl**
  - **SMCBrakeControl**
  - **SMCBrakeState**
- ... contains the corresponding visualisation.

CyclicMove

- ... contains the instanceable sample program, **CyclicMove (FB)**, for using the **L_SMC_AxisBasicControl** function block.
- ... contains the corresponding visualisation.

EtherCAT

- ... contains the following programs:

  MasterHandling: Reinitialise EtherCAT bus.

  ReadStatus: Read the status of the EtherCAT slaves.

  SetState: Set the state of the EtherCAT slaves.

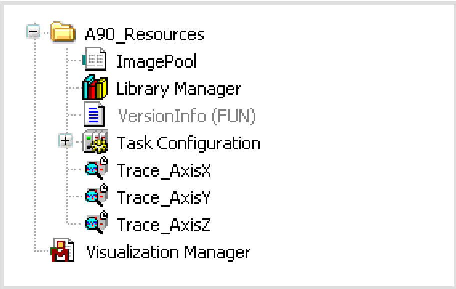
Gearing

- ... contains the instanceable sample program, **Gearing (FB)**, for synchronising the Y axis to the X axis via PLC open functions.

Programs

- ... contains the following programs:
  - **AxisBasicControl:**
    Calls three instances of the **L_SMC_AxisBasicControl** function block.
  - **Example_Applications:**
    Calls instances of the **Gearing**, **BrakeControl**, **CyclicMove** function blocks.
  - **EtherCATDiagnostic:**
    Calls the **MasterHandling**, **ReadStatus**, **SetState** programs.
  - **Init:**
    Writes initial values to those function block inputs which are not interconnected.

5.4 A90_Resources



BrakeControl

- ... contains the graphics referenced in the visualisations.

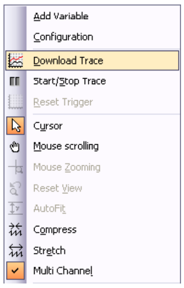
Trace_*AxisX...Z*

- ... pre-configured oscilloscope function of the respective axis.

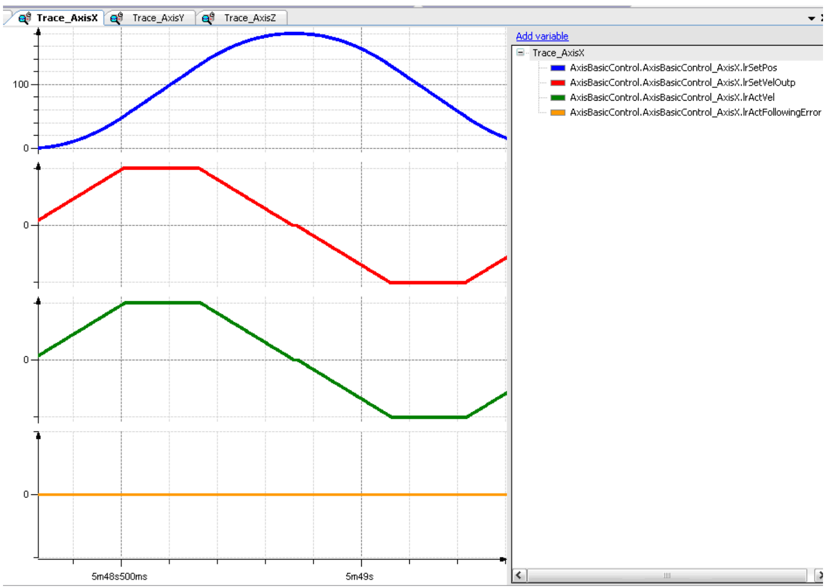
5.4.1 Use of the oscilloscope function: Example of AxisX

How to proceed:

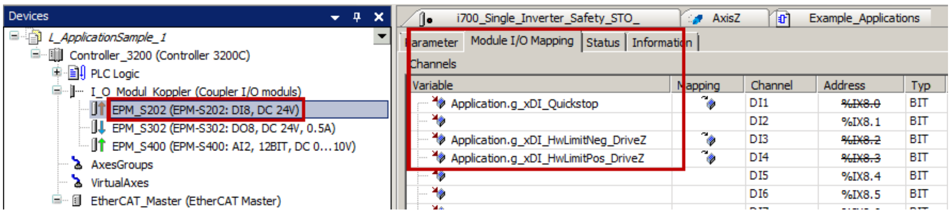
1. Double-click the desired **Trace** in the device view.
2. Select the **Download Trace** command in the context menu of the oscilloscope to load the data to the Controller.
   - Alternatively, execute the command via the menu **Trace→Download Trace**.



   - The **Add Variable** option makes evaluation easier by providing all signals in separate channels.



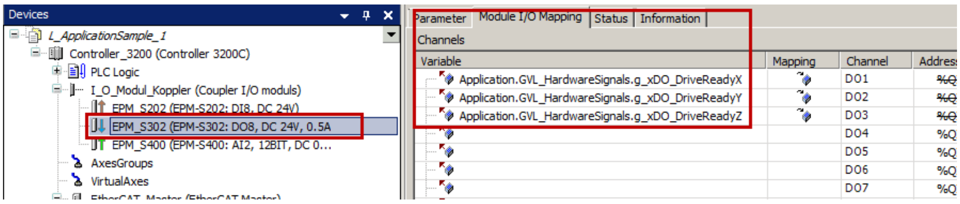
5.4.2 EPM_S202: Digital inputs



The hardware configuration of the sample project contains eight digital inputs. The corresponding Lenze Teachware case i700 contains a control panel for controlling the inputs of the I/O system.

- Digital input 1 "DI1": Mapped to the global variable for quick stop (QSP).
- Digital input 3 "DI3": Mapped to the global variable of the negative hardware limit switch for Axis Z.
- Digital input 4 "DI4": Mapped to the global variable of the positive hardware limit switch for Axis Z.

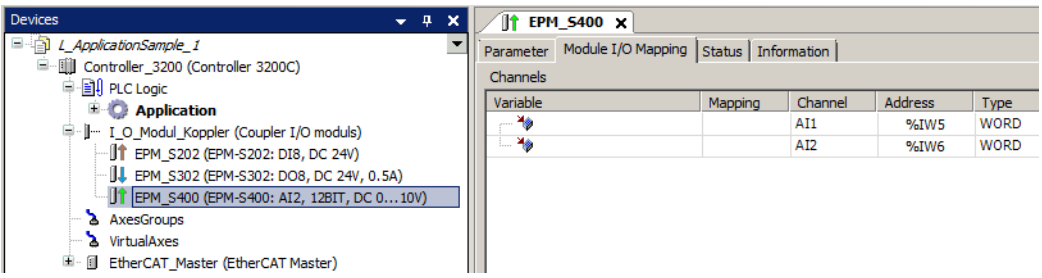
5.4.3 EPM_S303: Digital outputs



The hardware configuration of the sample project contains eight digital outputs. The corresponding Lenze Teachware case i700 contains a control panel for evaluating the outputs of the I/O system.

- Global variable *DriveReadyX* is mapped to digital output 1 "DO1".
- Global variable *DriveReadyY* is mapped to digital output 2 "DO2".
- Global variable *DriveReadyZ* is mapped to digital output 3 "DO3".

5.4.4 EPM_S400: Analog inputs



The hardware configuration of the sample project contains two analog inputs. The corresponding Lenze Teachware case i700 contains a control panel for controlling the inputs of the I/O system.

- The sample project does not contain/require a pre-configured mapping of the analog inputs.

6 Opening the Application Sample

The Application Sample includes a sample program with three axes and a predefined visualisation.

General procedure

The main steps are presented in the table below:

Step	Activity
1.	Creating a new project - opening the Application Sample (📖 18)
2.	Updating the Controller in the project (optional) (📖 18)
3.	Going online (📖 19)
4.	Downloading and starting the PLC program (📖 19)
5.	Getting started - operating the Application Sample (📖 21)

Further information about the parameterisation and configuration of the is provided in the following communication manuals:
- "Controller-based Automation" EtherCAT - *commissioning & configuration*

The commissioning steps in detail

The following section provides a detailed description of every commissioning step.

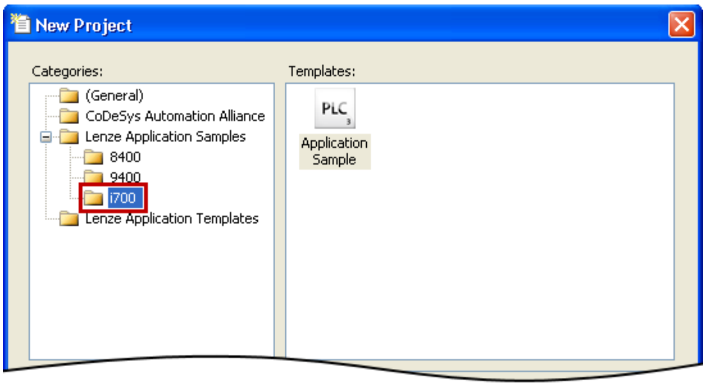
Please follow the instructions below carefully to commission your automation system.

6.1 Creating a new project - opening the Application Sample

The Application Sample is available as a predefined »PLC Designer« project (*.project,). To call the Application Sample, a new project must be created with the Application Sample being the template.

How to proceed:

1. Creating a new project:
   - **File→ New project**



   - Select category *Lenze Application Samples\Lenze conventional\i700*
   - Open template **Application sample**

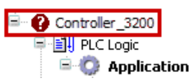
6.2 Updating the Controller in the project (optional)

The Controller must optionally be updated in the »PLC Designer« if the project has older firmware information than the hardware to be used.

- If the Controller is marked with the symbol after opening the project, the device must be updated.
- Use the »WebConfig« to determine the firmware used by the Controller to be able to select the corresponding setting in the »PLC Designer«.

How to proceed:

1. Select the desired controller:



   - Execute the **Update device** command in the context menu.
   - Double-click the desired controller in the **Update device** dialog box to update the controller in the device view.

6.3 Going online

To establish an online connection to the Controller, the communication settings (**Set Active Path**) must be adapted first. ▸ Setting up communication to the Controller (📖 8)

6.3.1 Transferring the project to the control system - "logging in"

The "Login" command transfers the PLC program to the controller.

- Transfer the project to the Controller by "logging in" to the control system:

  Menu command **Online→ Login**

6.4 Downloading and starting the PLC program

- Select the **Online→Load** menu command to load the PLC program to the Controller.
- Select the **Online→Run** menu command to start the PLC program.
  - As an alternative, you can execute the **Debug→Start** menu command or press **<F5>**.

Tip!

If the project is to be loaded automatically after a restart of the device, you can install it as "Boot project".

How to install the project as boot project:

1. Select the **Online→Generate boot project for L-force Controller** menu command.

6.5 Simulation

Note!

"Watchdog" monitoring must be deactivated for all tasks before the simulation mode can be used with the Application Sample.

- If the "Watchdog" monitoring is active, an error message will be issued and the simulation mode cannot be activated.

How to deactivate the "Watchdog" monitoring:

1. Double-click the *A90_Resources* folder in the device view.
   - Double-click **Task Configuration→** **MainTask** .
2. On the **Configuration** tab, deactivate the *Watchdog* by removing the ☑ checkmark:

7 Getting started - operating the Application Sample

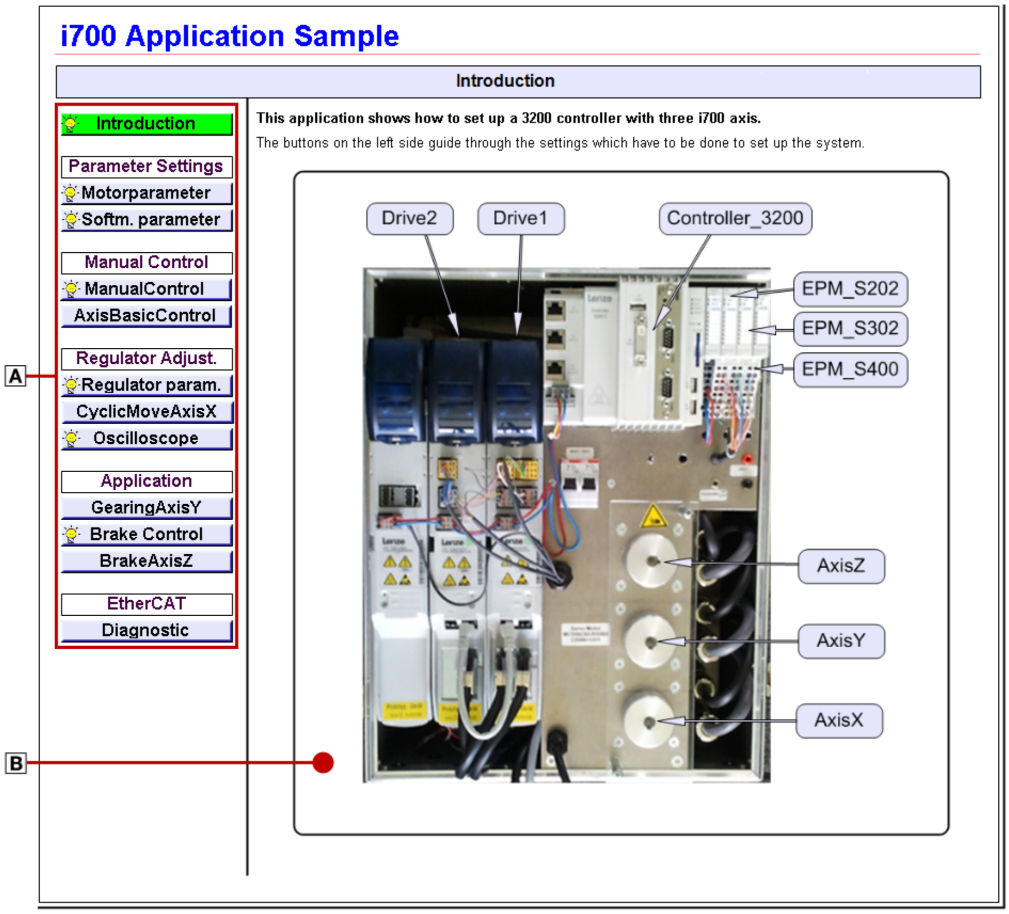
The Application Sample i700 includes visualisations for the operation of the i700 which can be controlled via the **Main** visualisation.

How to proceed:

1. In the *Device View*, select the *A20_Visualisation* folder.
2. Double-click the **Main** visualisation.

Welcome page - Main visualisation

The user interface of the visualisation is divided into the following areas:



A Buttons:
- Select the desired visualisation.

B Operating range of the machine modules
- Displays the visualisation page selected.

7.1 The buttons of the visualisation

Tip!

The information pages of the visualisation are marked with the symbol .

- The pages provide information on what to do to get the system started: "Getting started".

Setting parameters

Motor parameters: information page "Getting started"
- **Step 1:**
  How to parameterise the motor data

Softm. parameters: information page "Getting started"
- **Step 2:**
  How to parameterise the SoftMotion data

Manual control

ManualControl: information page "Getting started"
- **Step 3:**
  How to move the axes (*AxisX/AxisY/AxisZ*) in the manual mode (manual jog)

AxisBasicControl: user interface
- Axis control via the visualisation of the **L_SMC_AxisBasicControl** FB.

Controller settings

Regulator param.: information page "Getting started"
- **Step 4:**
  How to change the controller settings

CyclicMoveAxisX: user interface
- Visualisation of the program **CyclicMoveAxisX** ▶ Axis control in the automatic mode: "CyclicMove"

Application

Oscilloscope: information page "Getting started"
- **Step 5:**
  How to optimise the controller settings

GearingAxisY:
- Visualisation of the **GearingAxisY** program

BrakeControl: information page "Getting started"
- **Step 6:**
  How to use the holding brake

BrakeAxisZ: user interface
- Visualisation of the **BrakeAxisZ** program

EtherCAT

Diagnostic: EtherCAT diagnostic page
- Query/set EtherCAT state (bus/slave)

7.2 Commissioning an axis

Note!

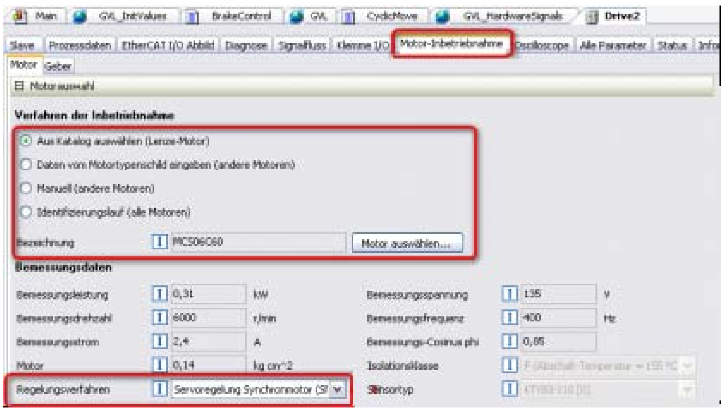
If the Lenze Teachware case i700 is used, steps **1-4** are not required!

- The parameters are predefined and already included in the project.

How to proceed:

1. Motor data parameterisation
2. SoftMotion data parameterisation
3. Axis control during manual operation
4. Controller adjustment by means of oscilloscope function

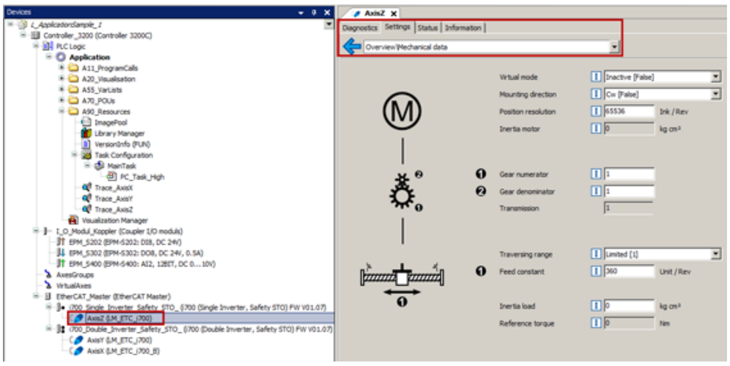
7.3 Motor data parameterisation



How to proceed:

1. Select the desired controller from the device tree.
2. Select the **Motor Commissioning** tab.
3. Select the desired motor.
4. Select the desired control mode.

7.4 Parameterising the axis data



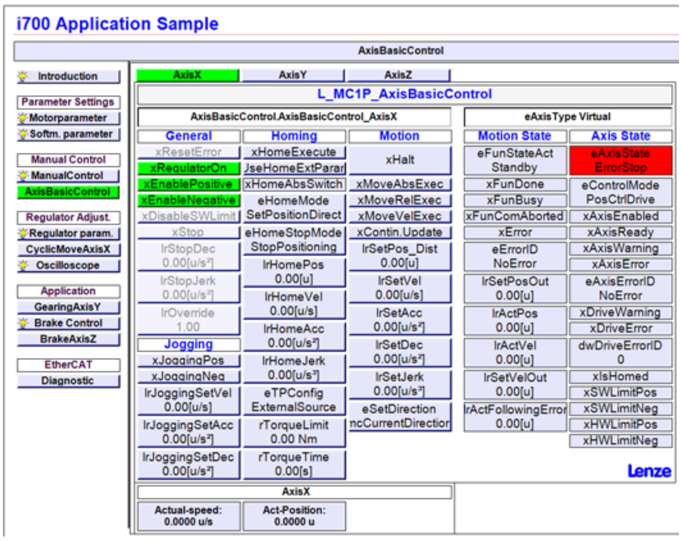
How to proceed:

1. Select the desired axis in the device tree. Example: **AxisZ**
2. Select the **Overview\Mechanical data** tab. Select desired settings such as the traversing range of the axis.
3. Select the **Overview\Basic settings** tab. Select desired settings such as...

   ... the response of the following error monitoring.

   ....the suitable scaling of the units/mounting direction.

7.5 Axis control during manual operation: manual control

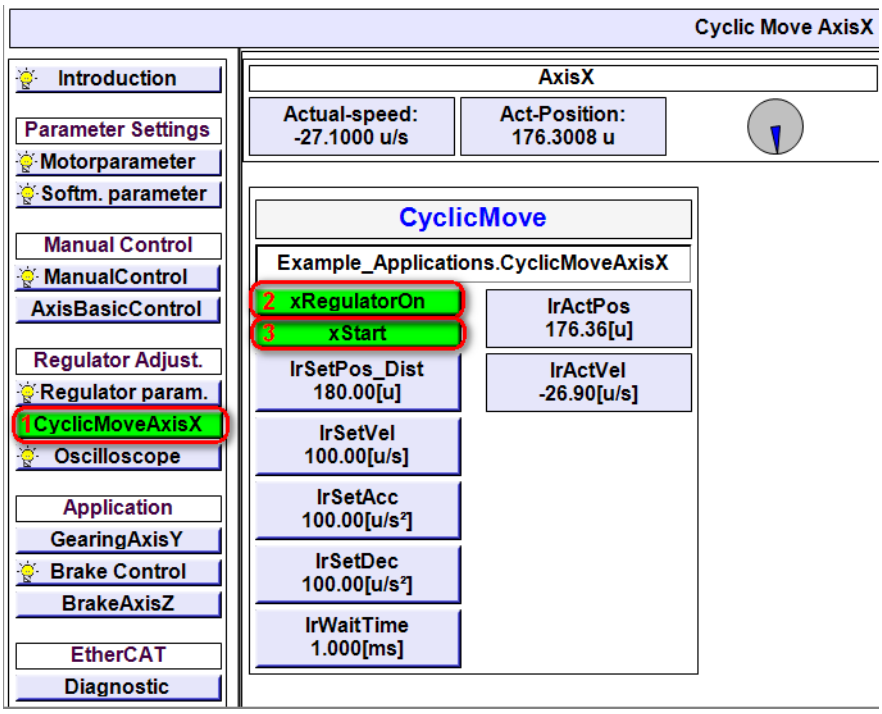


How to proceed:

1. Click the **AxisBasicControl** button.
2. Select the desired axis: **AxisX**/**AxisY**/**AxisZ**.
3. Click the **Internal Control** button.
4. Acknowledge pending errors by clicking the **xResetError** button.
5. Click the **xRegulatorOn** button.
   - Now the controller has been enabled and can be controlled via the buttons.

   Example: Click the **xJoggingPos** button to control the axis in positive direction of rotation.

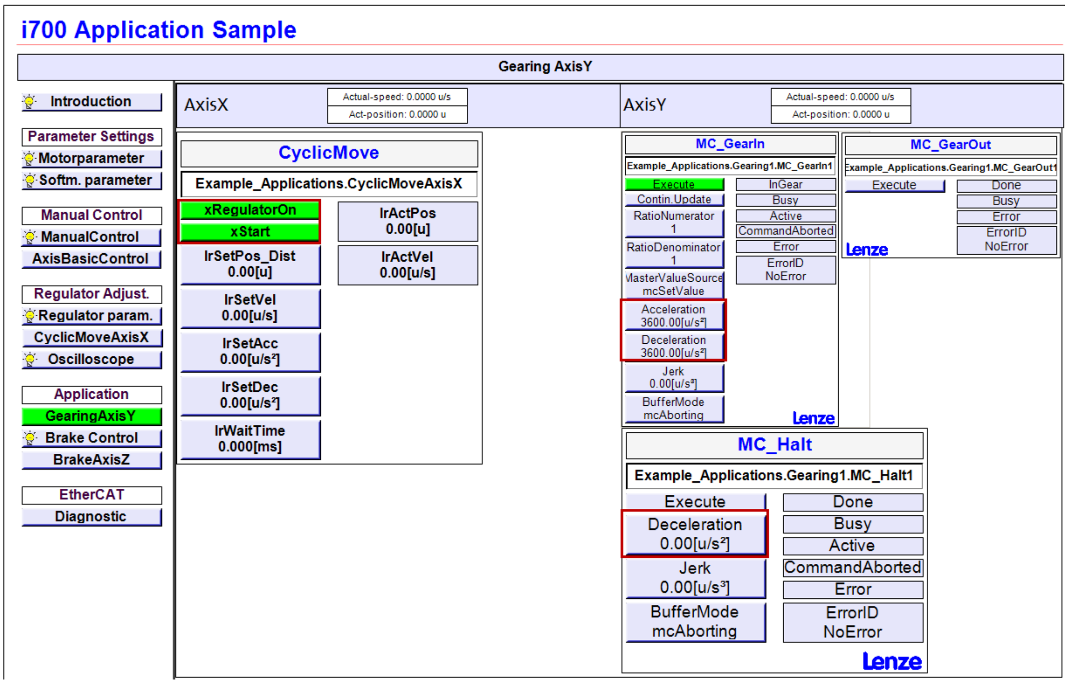
7.6 Axis control in the automatic mode: "CyclicMove"



How to proceed:

1. Click the **CyclicMoveAxisX** button.
2. Enable the controller: Click the **xRegulatorOn** button.
3. Start the positioning process: Click the **xStart** button.
   - The positioning parameters can be edited anytime directly in the visualisation.

7.7 Coupling/decoupling of an axis: "GearingAxisY"

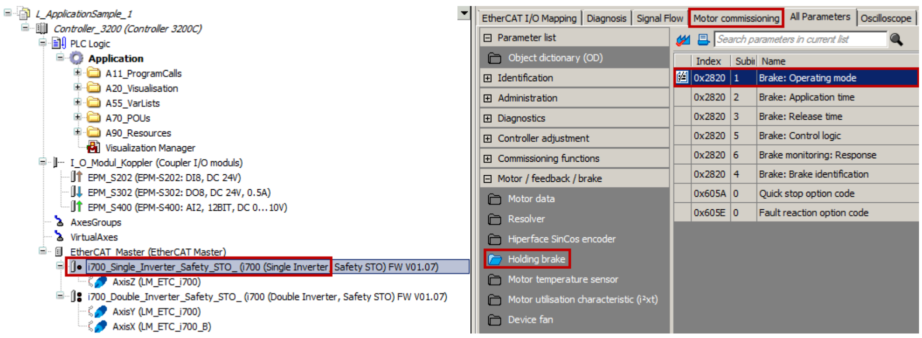


How to proceed:

1. Click the **GearingAxisY** button.
2. Enable the controller: Click the **xRegulatorOn** button.
3. Start the positioning process: Click the **xStart** button.
4. Set the acceleration/deceleration: Enter the desired values in **MC_GearIn**.
5. Set the desired deceleration of **MC_Halt**.
6. Coupling/decoupling
   - Start "Coupling": Click **Execute** in **MC_GearIn**.
   - "Decoupling": Click **Execute** in **MC_GearOut**.
7. To bring **Axis Y** to a standstill after decoupling, click **Execute** in **MC_Halt**.

   The parameters of the function blocks can be edited anytime directly in the visualisation.

7.8 Holding brake: Application "BrakeAxisZ"



For **Axis Z** a holding brake is defined by default, which releases at controller enable.

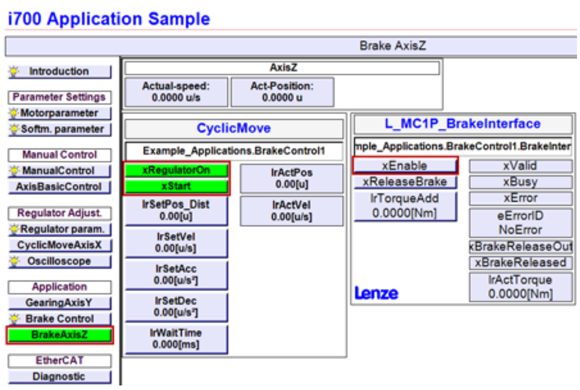
- The holding brake can be additionally switched over to other states (for example for purposes regarding the application) via FB **L_MC1P_BrakeInterface**.

Preparation: First of all, the operating mode of the brake (index **0x2820/1**) must be set to a value of "1", the standard value is "0".

Note: When you quit the example, you must reset the brake setting to the initial value!

Movement of Axis X during automatic operation

After changing the operating mode of the brake, the function of the holding brake can be controlled via the **BrakeAxisZ** visualisation.

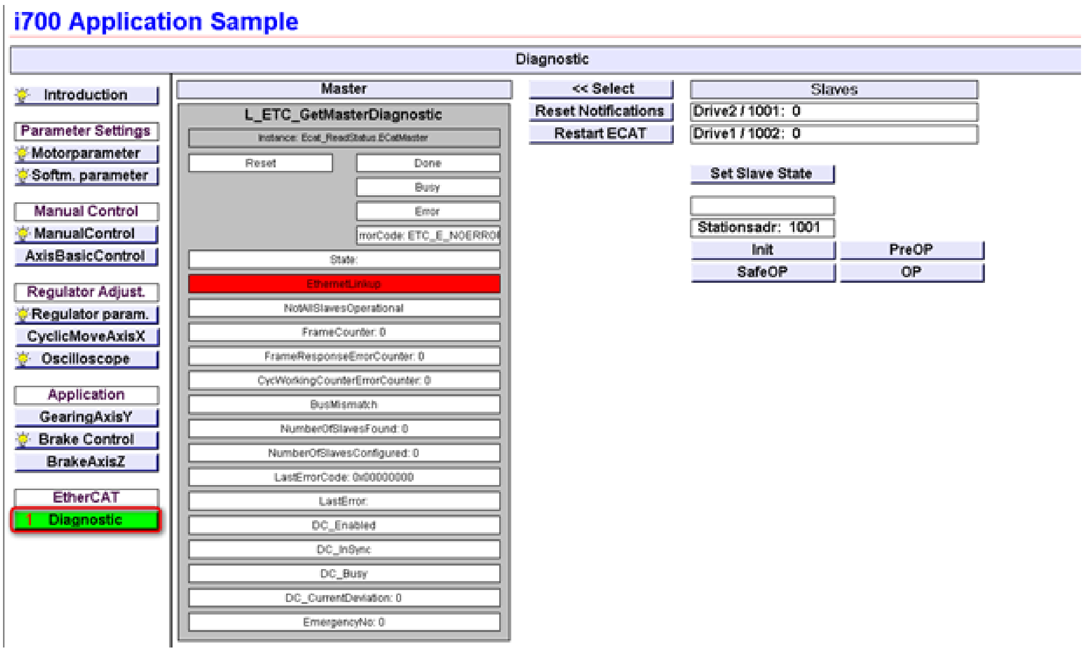


How to proceed:

1. Click the **BrakeAxisZ** button.
2. Select brake mode for **L_MC1P_BrakeControl** :
   - Open brake: **xReleaseBrake = TRUE**
3. Release brake: click **xEnable** button of **L_MC1P_BrakeInterface**.
4. **xBrakeReleased** shows the current brake status.
5. Enable the controller: Click the **xRegulatorOn** button.
6. Start the positioning process: Click the **xStart** button.

   The positioning parameters can be edited anytime directly in the visualisation.

7.9 EtherCAT diagnostics: "Diagnostic"



The **Diagnostic** visualisation displays information on the EtherCAT state:

- State EtherCAT bus

  ... displays the current state of the bus. It shows e.g. whether the bus has been restarted after the communication to the slaves was interrupted (disconnected cable connection between Controller and node).
- EtherCAT slave status
- ... displays the state of the corresponding slave.
- **Set Slave State** serves to set the desired state for the corresponding slave.

Index

FEEDBACK

Your opinion is important to us

These instructions were created to the best of our knowledge and belief to give you the best possible support for handling our product.

If you have suggestions for improvement, please e-mail us to:

feedback-docu@Lenze.de

Thank you for your support.

Your Lenze documentation team

Lenze Automation GmbH
Hans-Lenze-Str. 1
D-31855 Aerzen
Germany
+49 (0)51 54 / 82-0
+49 (0)51 54 / 82-28 00
Lenze@Lenze.de
www.Lenze.com

Service
Lenze Service GmbH
Breslauer Straße 3
D-32699 Extertal
Germany
00 80 00 / 24 4 68 77 (24 h helpline)
+49 (0)51 54 / 82-11 12
Service@Lenze.de