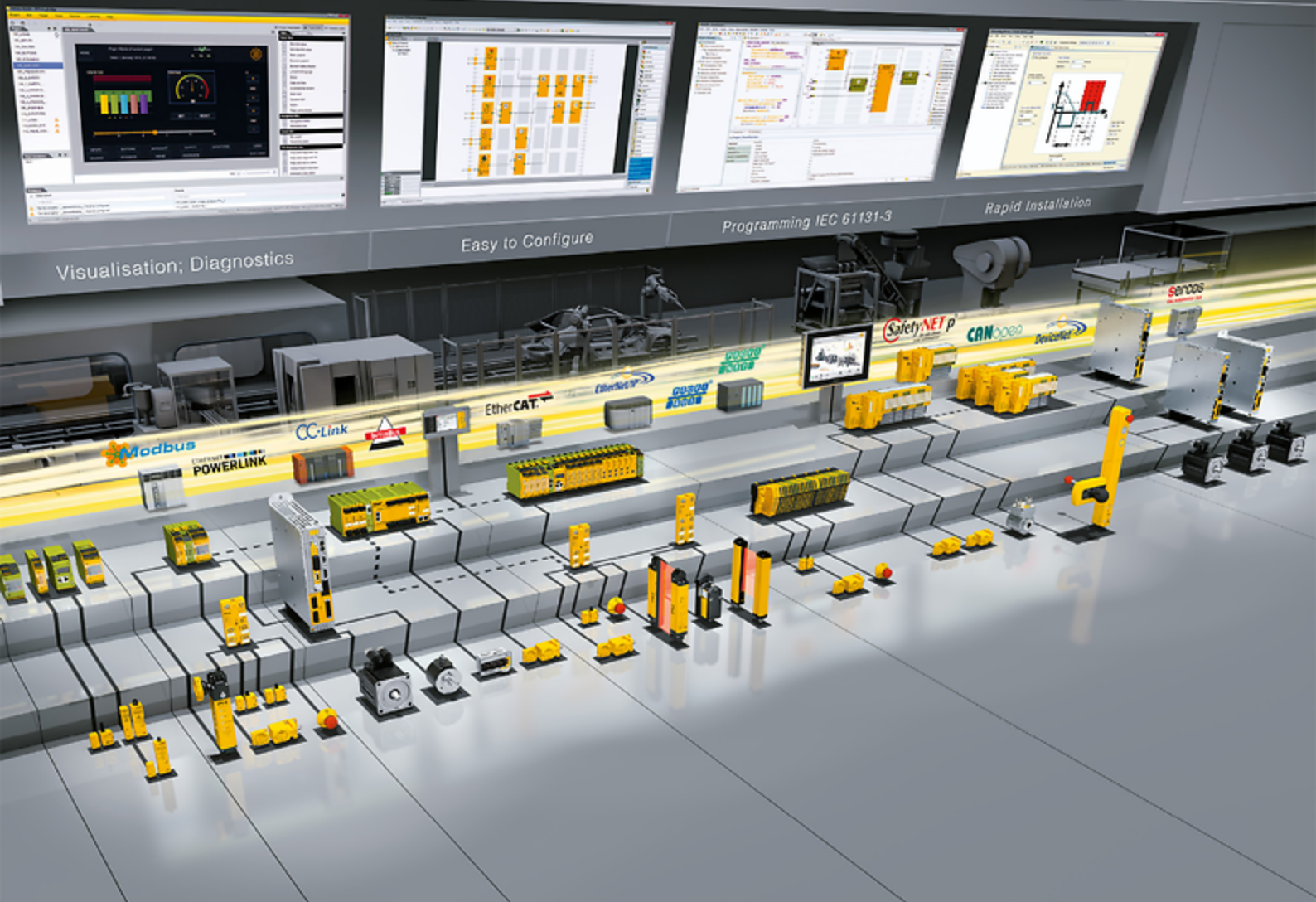


Intern



EtherCAT – PMC SC6 and PMC SI6

Pilz

	Table of contents	2
1	Foreword	6
2	User information	7
2.1	Storage and transfer	7
2.2	Described product	7
2.3	Directives and standards	7
2.4	Timeliness	7
2.5	Original language	7
2.6	Limitation of liability	7
2.7	Formatting conventions	8
2.7.1	Display of warning messages and information	8
2.7.2	Markup of text elements	9
2.7.3	Mathematics and formulas	9
2.8	Trademarks	10
3	Safety notes	11
4	Network structure	12
5	Connection	13
5.1	Selecting suitable cables	13
5.2	X200, X201: Fieldbus connection	13
6	What you should know before commissioning	14
6.1	Program interfaces	14
6.1.1	DS6 program interface	14
6.1.1.1	Configuring the view	16
6.1.1.2	Navigation using sensitive circuit diagrams	17
6.1.2	TwinCAT 3 program interface	18
6.2	Meaning of parameters	19
6.2.1	Parameter groups	19
6.2.2	Parameter types and data types	20
6.2.3	Parameter types	21
6.2.4	Parameter structure	21
6.2.5	Parameter visibility	22
6.3	Signal sources and process data mapping	23
6.4	Non-volatile memory	23
7	Commissioning	24
7.1	DS6: Configuring the drive controller	25
7.1.1	Initiating the project	25
7.1.1.1	Projecting the drive controller and axis	25
7.1.1.2	Configuring safety technology	27
7.1.1.3	Creating other drive controllers and modules	27
7.1.1.4	Projecting the module	28
7.1.1.5	Projecting the project	28
7.1.2	Parameterizing general EtherCAT settings	28
7.1.3	Configuring PDO transmission	29
7.1.3.1	Adapting RxPDO	29
7.1.3.2	Adapting TxPDO	29

Intern

7.1.4	Mapping the mechanical axis model	30
7.1.4.1	Parameterizing the motor	30
7.1.4.2	Parameterizing the axis model	31
7.1.5	Synchronizing EtherCAT nodes	35
7.1.6	Transmitting and saving the configuration	36
7.1.7	Activating the control panel and testing the configuration	38
7.2	TwinCAT 3: Putting the EtherCAT system into operation	39
7.2.1	Creating and exporting an ESI file	39
7.2.2	Activating the EtherCAT master	40
7.2.3	Scanning the hardware environment	41
7.2.4	Expanding the startup list	42
7.2.5	Configuring synchronization using distributed clocks	44
7.2.6	Configuring synchronization via SyncManager event	44
7.2.7	Controller-based axis control	45
7.2.7.1	Parameterizing an axis	45
7.2.7.2	Programming axis control	46
7.2.8	Drive-based axis control	47
7.2.9	Configuring EoE communication	48
7.2.10	Configuring a Station Alias	49
7.2.11	Transmitting the configuration	50
7.2.12	Checking the functionality of the axes	50
7.3	CODESYS V3: Putting the EtherCAT system into operation	51
7.3.1	Creating a standard project	51
7.3.2	Adding a drive controller	51
7.3.3	Configuring synchronization using distributed clocks	52
7.3.4	Controller-based axis control	53
7.3.4.1	Parameterizing a SoftMotion axis	53
7.3.4.2	Programming axis control	54
7.3.5	Drive-based axis control	55
7.3.6	Configuring EoE communication	55
7.3.7	Transmitting the configuration	55
7.3.8	Checking the functionality of the axes	56
7.3.9	Special case: Adding to the PDO transmission	56
8	Monitoring and diagnostics	57
8.1	Connection monitoring	57
8.2	LED display	57
8.2.1	EtherCAT state	58
8.2.2	EtherCAT network connection	59
8.3	Events	60
8.3.1	Event 52: Communication	61
8.4	Parameters	62
8.4.1	A254 EtherCAT Station Alias G6 V0	62
8.4.2	A255 EtherCAT Device State G6 V2	62
8.4.3	A256 EtherCAT address G6 V1	62
8.4.4	A257 EtherCAT diagnosis G6 V1	62
8.4.5	A259 EtherCAT SM-Watchdog G6 V1	64
8.4.6	A261 Sync-Diagnostics G6 V1	64
8.4.7	A287 DC-Sync optimization G6 V3	65
8.4.7.1	A287[2] Result G6 V2	65

9	Looking for more information about EtherCAT?	66
9.1	EtherCAT	66
9.2	Communication protocols.....	67
9.2.1	CoE: CANopen over EtherCAT	67
9.2.2	EoE: Ethernet over EtherCAT	68
9.2.3	EoE: Application cases with Pilz devices	68
9.2.3.1	Topology 1: EtherCAT master and DS6 on one PC.....	69
9.2.3.2	Topology 2: EtherCAT master and DS6 on different PCs	70
9.3	Communication objects.....	72
9.3.1	Process data objects – PDO	72
9.3.1.1	PDO mapping.....	72
9.3.2	Service data objects – SDO	73
9.3.2.1	Addressing axis-dependent parameters	73
9.3.2.2	Expedited transfer	73
9.3.2.3	Segmented transfer.....	75
9.3.3	Emergency objects – EMCY	80
9.4	EtherCAT state machine	82
9.5	Synchronization.....	84
9.5.1	SM-Sync: Synchronization using SyncManager event	85
9.5.2	DC-Sync: Synchronization using distributed clocks	86
9.5.2.1	TwinCAT 3: Synchronization using DC-Sync.....	87
9.5.2.2	CODESYS V3: Synchronization using DC-Sync.....	93
9.6	Modular ESI files	100
9.6.1	Adding to a modular ESI file.....	100
9.6.2	Deleting a module from the ESI file.....	101
9.7	Cycle times.....	101
9.8	Activating and executing actions.....	102
9.9	Fieldbus scaling	104
9.10	SDO Info service	105
9.10.1	Setting SDO Info service in TwinCAT 3	105
9.10.2	Access to objects	106
9.10.3	Check for conformity	106
9.11	Diagnosis History	107
9.11.1	Reading out the Diagnosis History in TwinCAT 3	107
9.11.2	Determination of the system time.....	108
10	Appendix.....	109
10.1	Supported communication objects.....	109
10.1.1	ETG.1000.6 EtherCAT specification: 1000 hex – 1FFF hex	109
10.1.2	ETG.1020 EtherCAT protocol enhancements.....	112
10.1.3	ETG.5000.1 Modular Device Profile: F000 hex – FFFF hex	113
10.1.4	Manufacturer-specific parameters: 2000 hex – 53FF hex.....	114
10.1.5	Manufacturer-specific parameters: A000 hex – D3FF hex.....	116
10.2	SDO transmission: Error codes.....	118
10.3	EMCY message: Incorrect state transition error codes.....	119
10.4	EMCY message: Device fault error codes	120
10.5	EMCY message: EoE-error error codes	121
10.6	Simple Network Time Protocol (SNTP).....	122
10.6.1	Setting up time service on the computer.....	122
10.7	Detailed information	124
10.8	Abbreviations	125

Intern

Glossary	126
List of figures	128
List of tables	129

1 Foreword

Drive controllers of the PMC SC6 and PMC SI6 series are available with the Ethernet-based EtherCAT fieldbus system as standard.

This documentation describes a combination of the drive controllers listed with a controller as an EtherCAT master and the associated automation software.

Drive controllers of the PMC SC6 and PMC SI6 series successfully passed the EtherCAT as well as Fail Safe over EtherCAT (FSoE) Conformance Test. There, the communication interface was tested to ensure the reliability and function of the lower-level communication regardless of vendor.

2 User information

This documentation assists you with commissioning the Pilz PMC SC6 or PMC SI6 series drive controllers in combination with higher-level controller systems over an EtherCAT network.

Technical knowledge

Operating an EtherCAT network requires having familiarity with the basics of the EtherCAT network technology.

Technical requirements

Before you begin operating your EtherCAT network, you need to wire the drive controllers and initially check that they are functioning correctly. To do this, follow the instructions in the manual for the relevant drive controller.

2.1 Storage and transfer

As this documentation contains important information for handling the product safely and efficiently, it must be stored in the immediate vicinity of the product until product disposal and be accessible to qualified personnel at all times.

Also pass on this documentation if the product is transferred or sold to a third party.

2.2 Described product

This documentation is binding for:

PMC SC6 or PMC SI6 series drive controllers in conjunction with the DriveControlSuite software (DS6) in V 6.5-K or later and associated firmware in V 6.5-K-EC or later.

2.3 Directives and standards

Refer to the drive controller documentation for the European directives and standards relevant to the drive controller and accessories.

2.4 Timeliness

Check whether this document is the most up-to-date version of the documentation. We make the latest document versions for our products available for download on our website:

<https://www.pilz.com/en-INT>.

2.5 Original language

The original language of this documentation is German; all other language versions are derived from the original language.

2.6 Limitation of liability

This documentation was created taking into account the applicable standards and regulations as well as the current state of technology.

No warranty or liability claims for damage shall result from failure to comply with the documentation or from use that deviates from the intended use of the product. This is especially true for damage caused by individual technical modifications to the product or the project configuration and operation of the product by unqualified personnel.

2.7 Formatting conventions

Orientation guides in the form of signal words, symbols and special text markups are used to emphasize specific information so that you are able identify it in this documentation quickly.

2.7.1 Display of warning messages and information

Warning messages are identified with symbols. They indicate special risks when handling the product and are accompanied by relevant signal words that express the extent of the risk. Furthermore, useful tips and recommendations for efficient, error-free operation are specially highlighted.



ATTENTION!

Attention indicates that damage to property may occur

- if the stated precautionary measures are not taken.



CAUTION!

Caution with a warning triangle indicates that minor personal injury may occur

- if the stated precautionary measures are not taken.



WARNING!

Warning with a warning triangle means there may be a considerable risk of fatal injury

- if the stated precautionary measures are not taken.



DANGER!

Danger with a warning triangle indicates that there is a considerable risk of fatal injury

- if the stated precautionary measures are not taken.



Information

Information indicates important information about the product or serves to emphasize a section in the documentation that deserves special attention from the reader.

2.7.2 Markup of text elements

Certain elements of the continuous text are distinguished as follows.

Important information	Words or expressions with a special meaning
Interpolated position mode	Optional: File or product name or other name
<u>Detailed information</u>	Internal cross-reference
http://www.samplelink.com	External cross-reference

Software and other displays

The following formatting is used to identify the various information content of elements referenced by the software interface or a drive controller display, as well as any user entries.

Main menu Settings	Window names, dialog box names, page names or buttons, combined proper nouns, functions referenced by the interface
Select Referencing method A	Predefined entry
Save your <own IP address>	User-defined entry
EVENT 52: COMMUNICATION	Displays (status, messages, warnings, faults)

Keyboard shortcuts and command sequences or paths are represented as follows.

[CTRL], [CTRL] + [S]	Key, key combination
Table > Insert table	Navigation to menus/submenus (path specification)

2.7.3 Mathematics and formulas

The following signs are used to represent mathematical relationships and formulas.

–	Subtraction
+	Addition
×	Multiplication
÷	Division
	Absolute value

2.8 Trademarks

The following names used in connection with the device, its optional equipment and its accessories are trademarks or registered trademarks of other companies:

CANopen®,
CiA®

CANopen® and CiA® are registered European Union trademarks of CAN in AUTOMATION e.V., Nuremberg, Germany.

CODESYS®

CODESYS® is a registered trademark of CODESYS GmbH, Kempten, Germany.

EtherCAT®,
Safety over EtherCAT®,
TwinCAT®

EtherCAT®, Safety over EtherCAT® and TwinCAT® are registered trademarks of patented technologies licensed by Beckhoff Automation GmbH, Verl, Germany.

Windows®,
Windows® 7,
Windows® 10,
Windows® 11

Windows®, the Windows® logo, Windows® XP, Windows® 7, Windows® 10, and Windows® 11 are registered trademarks of Microsoft Corporation in the United States and/or other countries.

All other trademarks not listed here are the property of their respective owners.

Products that are registered as trademarks are not specially indicated in this documentation. Existing property rights (patents, trademarks, protection of utility models) are to be observed.

3 Safety notes



WARNING!

Risk of fatal injury if safety notes and residual risks are not observed!

Failure to observe the safety notes and residual risks in the drive controller documentation may result in accidents causing serious injury or death.

- Observe the safety notes in the drive controller documentation.
- Consider the residual risks in the risk assessment for the machine or system.



WARNING!

Malfunction of the machine due to incorrect or modified parameterization!

In the event of incorrect or modified parameterization, malfunctions can occur on machines or systems which can lead to serious injuries or death.

- Observe the security notes in the drive controller documentation.
- Protect the parameterization, e.g. from unauthorized access.
- Take appropriate measures for possible malfunctions (e.g. emergency off or emergency stop).

4 Network structure

An EtherCAT network normally consists of an EtherCAT master (controller) and EtherCAT slaves, i.e. drive controllers from the PMC SC6 or PMC SI6 series. The PMC SI6 drive controllers also require at least one PMC PS6 supply module as an energy supply.

The EtherCAT network structure is generally optimized for a line topology. Each EtherCAT slave has an incoming and continuing bus connection.

Overall network expansion is virtually unlimited because a maximum of 65535 EtherCAT nodes can be connected together.

You can configure and parameterize the drive controllers using the DriveControlSuite commissioning software and the entire EtherCAT network using the automation software of the controller.

The following graphic provides a generalized depiction of an EtherCAT network with an EtherCAT master and EtherCAT slaves, using the PMC SI6 series as an example.

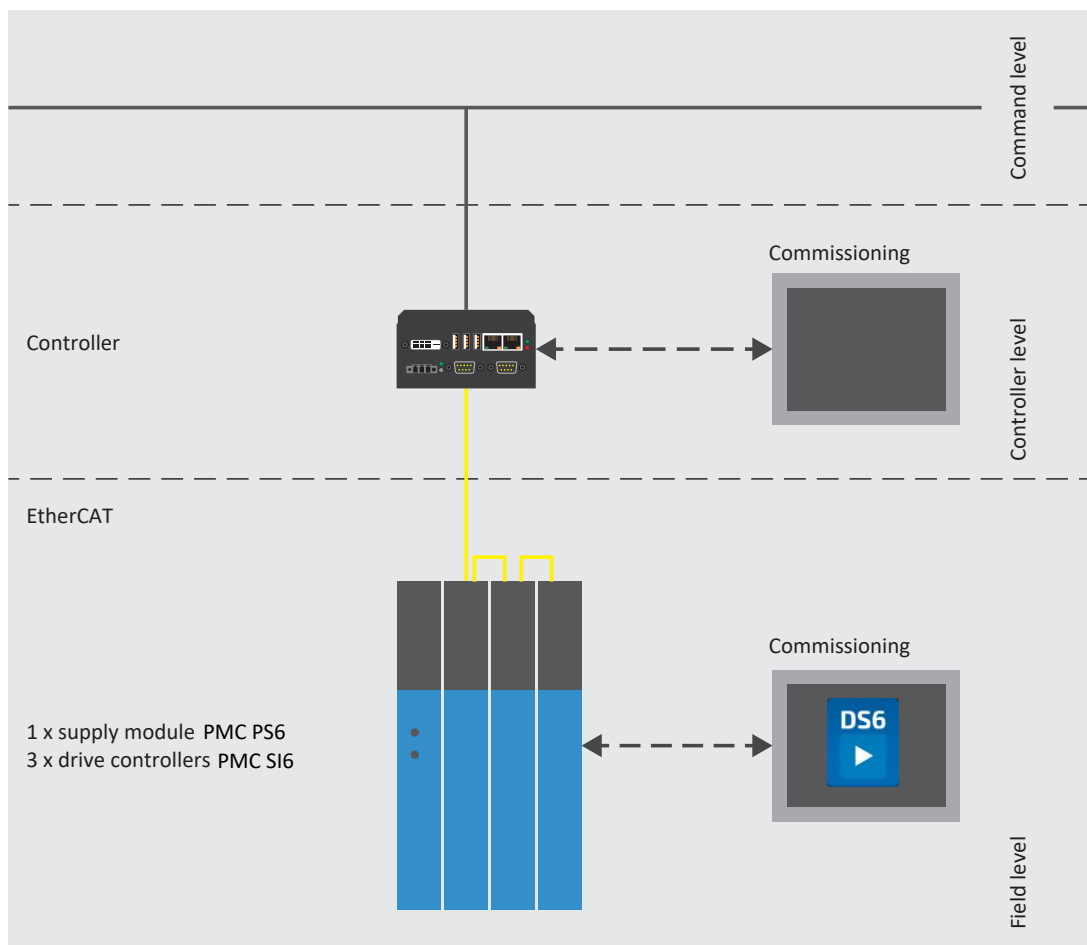


Fig. 1: EtherCAT: Network structure, using the PMC SI6 series as an example

5 Connection

In order to connect drive controllers of the PMC SC6 and PMC SI6 series to other EtherCAT nodes, the top of each device features two RJ-45 sockets.

5.1 Selecting suitable cables

EtherCAT is an Ethernet-based communications technology optimized for automation technology.

Ethernet patch cables or crossover cables meeting the CAT 5e quality level are the ideal cables. The Fast Ethernet technology allows a maximum cable length of 100 m between two nodes.



Information

Ensure that you only use shielded cables with an SF/FTP, S/FTP or SF/UTP design.

5.2 X200, X201: Fieldbus connection

The drive controllers have both RJ-45 sockets X200 and X201. The sockets are located on top of the device. The associated pin assignment and color coding correspond to the EIA/TIA-T568B standard.

Socket	Pin	Designation	Function
	1	Tx+	Communication
	2	Tx-	
	3	Rx+	
	4	—	—
	5	—	—
	6	Rx-	Communication
	7	—	—
	8	—	—

Tab. 1: X200 and X201 connection description

6 What you should know before commissioning

The following chapters provide a quick introduction to the structure of the program interface and accompanying window designations as well as relevant information about parameters and generally saving your project configuration.

6.1 Program interfaces

The following chapters include an overview of the program interfaces for the described software components.

6.1.1 DS6 program interface

Using the graphical interface of the DriveControlSuite commissioning software (DS6), you can project, parameterize and commission your drive project quickly and efficiently. In case of service, you can evaluate diagnostic information such as operating states, fault memories and fault counters of your drive project using DriveControlSuite.



Information

The program interface of DriveControlSuite is available in German, English and French. To change the language of the program interface, select **Settings > Language**.



Information

The DriveControlSuite help in the menu bar can be reached via **Help > Help for DS6** or via the [F1] key on your keyboard. When you press [F1] in an area of the program, the corresponding help topic opens.

Intern

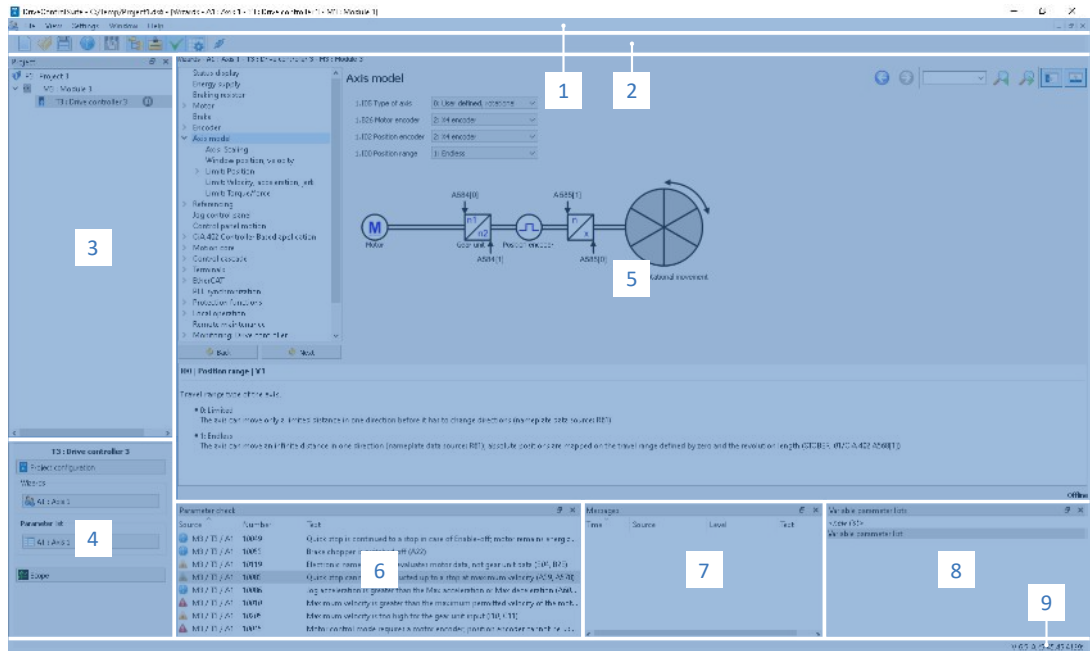


Fig. 2: DS6: Program interface





No.	Area	Description
1	Menu bar	Using the File, View, Settings and Window menus, you can open and save projects, display and hide program windows, select the interface language and access level and change between different windows in the workspace.
2	Toolbar	The toolbar enables quick access to frequently needed functions, like opening and saving projects and hiding and displaying windows in the program interface.
3	Project tree	The project tree forms the structure of your drive project in the form of modules and drive controllers. Select an element using the project tree first in order to edit it using the project menu.
4	Project menu	The project menu offers you various functions for editing the project, module and drive controller. The project menu adapts to the element that you selected in the project tree.
5	Workspace	The different windows which can be used to edit your drive project, such as the configuration dialog, wizards, the parameter list or the scope analysis tool, open in the workspace.
6	Parameter check	The parameter check points out irregularities and inconsistencies that were detected in the plausibility check of calculable parameters.
7	Messages	The entries in the messages log the connection and communication status of the drive controllers, incorrect inputs caught by the system, errors when opening a project or rule violations in the graphical programming.
8	Variable parameter lists	You can use variable parameter lists to compile any parameters in individual parameter lists for a quick overview.
9	Status bar	In the status bar, you can find the specifications of the software version and get additional information about the project file, the devices and the progress of the process during processes such as loading projects.

6.1.1.1 Configuring the view

In DriveControlSuite, you can change the visibility and arrangement of areas and windows, such as to optimize the available space in the workspace when working with smaller screens.

Showing/hiding areas

Use the icons in the toolbar or the items in the View menu to show or hide specific areas in DriveControlSuite as needed.

Icon	Item	Description
–	Reset	Resets the view to factory settings.
	Project	Shows/hides the Project window (project tree, project menu).
	Messages	Shows/hides the Messages window.
	Parameter check	Shows/hides the Parameter check window.
	Variable parameter lists	Shows/hides the Variable parameter lists window.

Arrange and group areas

You can undock and rearrange the individual areas via drag and drop. If you drag an undocked window to the edge of DriveControlSuite, you can release it there in a color-highlighted area either next to or on top of another window to redock it.

When you release the window onto another window, the two areas are merged into one window where you can use tabs to switch between the areas.

6.1.1.2 Navigation using sensitive circuit diagrams

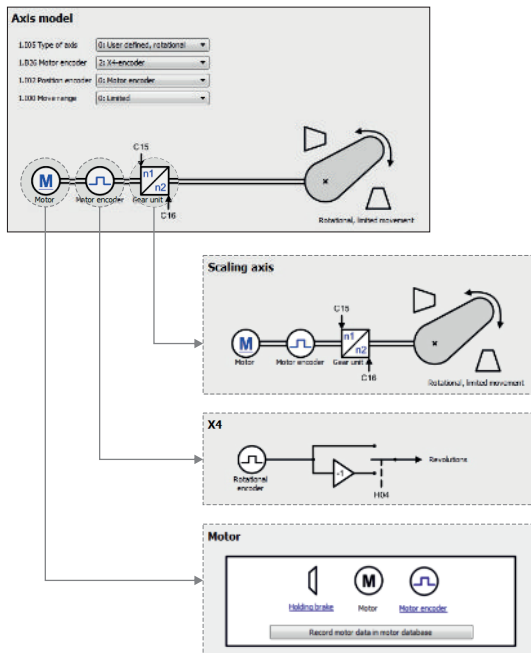


Fig. 3: DriveControlSuite: Navigation using text links and symbols

In order to illustrate graphically the processing sequence of actual and set values, the use of signals or certain drive component arrangements and to make configuring the accompanying parameters easier, they are displayed on the respective wizard pages of the workspace in the form of circuit diagrams.

Blue text links or clickable icons indicate links within the program. These refer to the corresponding wizard pages and, as a result, allow you to reach additional helpful detail pages with just a click.

Intern

6.1.2 TwinCAT 3 program interface

In TwinCAT 3, you operate your EtherCAT system using TwinCAT XAE. The following graphic shows the interface elements relevant to this documentation.

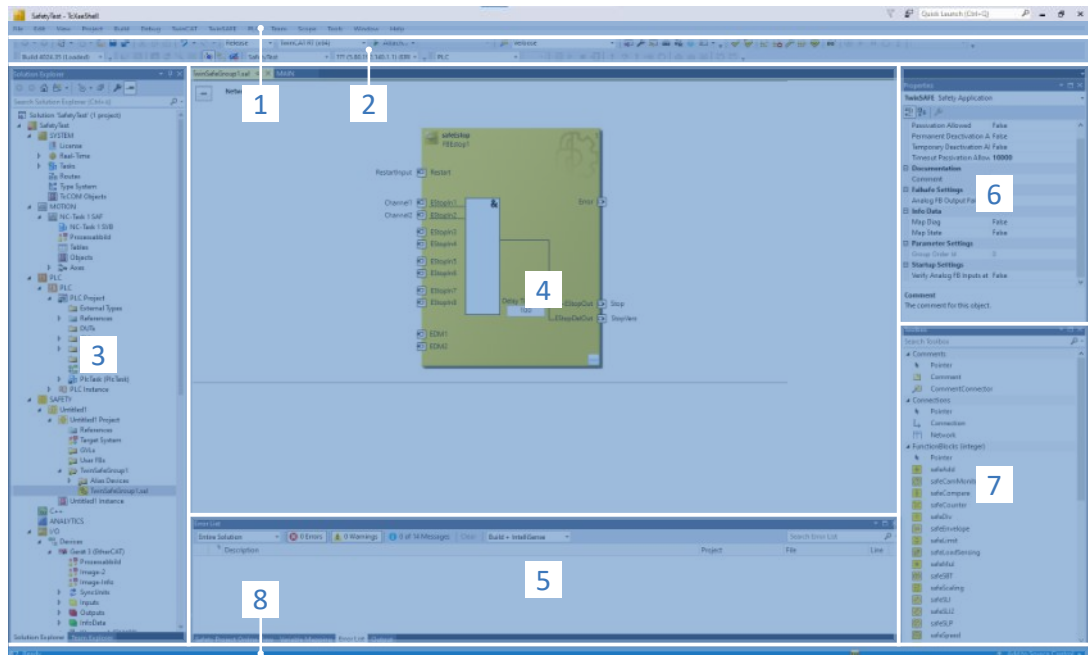


Fig. 4: TwinCAT 3 (TwinCAT XAE): Program interface

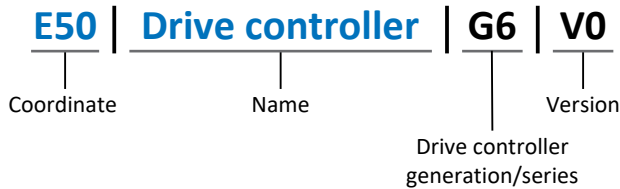
No.	Area	Description
1	Menu bar	The menu bar shows the menus set by default. Editor-specific menus appear only when the corresponding editor is open. Using the Tools menu, you can configure the user interface and add to existing menus or define new ones, for example.
2	Toolbar	The toolbar enables quick access to frequently used functions, such as opening and saving projects.
3	Solution explorer	The solution explorer maps the structure of your project with the project elements it contains. First select an element using the solution explorer to edit it in the main window.
4	Main window (editor)	In the main window, you define and edit objects, e.g. graphical programming elements.
5	Message window	The message window informs you of any errors or warnings currently present. You also receive messages about the syntax check, compilation process, etc.
6	Properties window	The properties window shows the properties of the element selected in the solution explorer.
7	Toolbox	Displays the "tools" available for the active editor, such as graphical programming elements.
8	Information and status bar	The information and status bar informs you about the state of the system (Config, Run, Stop or Exception mode). In online operation, you can see the current status of the program. If an editor window is active, the current position of the cursor and the set editing mode are also displayed.

6.2 Meaning of parameters

You can use parameters to adapt the function of the drive controller to your individual application. In addition, parameters visualize the current actual values (actual velocity, actual torque, etc.) and trigger actions such as Save values, Test phase, etc.

Interpretation of parameter identification

Parameter identification consists of the following elements, where short forms are also possible, i.e. only specifying a coordinate or the combination of coordinate and name.



6.2.1 Parameter groups

Parameters are assigned to individual groups by topic. The 6th generation of drive controllers differentiates between the following parameter groups.

Group	Topic
A	Drive controllers, communication, cycle times
B	Motor
C	Machine, velocity, torque/force, comparators
D	Set value
E	Display
F	Terminals, analog and digital inputs and outputs, brake
G	Technology – Part 1 (application-dependent)
H	Encoder
I	Motion (all motion settings)
J	Motion blocks
K	Control panel
L	Technology – Part 2 (application-dependent)
M	Profiles (application-dependent)
N	Additional functions (application-dependent; e.g. extended cam control unit)
P	Customer-specific parameters (programming)
Q	Customer-specific parameters, instance-dependent (programming)
R	Production data for the drive controller, motor, brakes, motor adapter, gear unit and geared motor
S	Safety (safety technology)
T	Scope
U	Protection functions
Z	Fault counter

Tab. 2: Parameter groups

6.2.2 Parameter types and data types

In addition to topic-based sorting in individual groups, all parameters belong to a certain data type and parameter type. The data type of a parameter is displayed in the parameter list, properties table. The connections between parameter types, data types and their value range can be found in the following table.

Data type	Parameter type	Length	Value range (decimal)
INT8	Integer or selection	1 byte (signed)	-128 – 127
INT16	Integer	2 bytes (1 word, signed)	-32768 – 32767
INT32	Integer or position	4 bytes (1 double word, signed)	-2147483648 – 2147483647
BOOL	Binary number	1 bit (internal: LSB in 1 byte)	0, 1
BYTE	Binary number	1 byte (unsigned)	0 – 255
WORD	Binary number	2 bytes (1 word, unsigned)	0 – 65535
DWORD	Binary number or parameter address	4 bytes (1 double word, unsigned)	0 – 4294967295
REAL32 (single type according to IEE754)	Floating-point number	4 bytes (1 double word, signed)	-3.40282×10^{38} – 3.40282×10^{38}
STR8	Text	8 characters	—
STR16	Text	16 characters	—
STR80	Text	80 characters	—

Tab. 3: Parameters: data types, parameter types, possible values

Parameter types: Use

- ▶ Integer, floating-point number
For general computing processes
Example: Set and actual values
- ▶ Selection
Numeric value to which a direct meaning is assigned
Example: Sources for signals or set values
- ▶ Binary number
Bit-oriented parameter information that is collected in binary
Example: Control and status words
- ▶ Position
Integer combined with associated units and decimal places
Example: Actual and set values of positions
- ▶ Velocity, acceleration, deceleration, jerk
Floating-point number combined with associated units
Example: Actual and set values for velocity, acceleration, deceleration, jerk
- ▶ Parameter address
Referencing of a parameter
Example: In F40 AO1 source, for example, E08 n-motor filtered can be parameterized
- ▶ Text
Outputs or messages

6.2.3 Parameter types

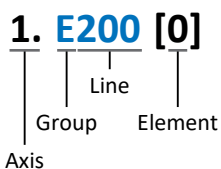
The following types of parameters are differentiated.

Parameter type	Description	Example
Simple parameters	Consist of one group and one line with a defined value.	A21 Brake resistor R: Value = 100 ohms
Array parameters	Consist of a group, a line and multiple sequential (listed) elements, which have the same properties but different values.	A10 Access level ▶ A10[0] access level: Value = Access level via operating unit ▶ A10[2] access level: Value = Access level via CANopen and EtherCAT ▶ A10[4] access level: Value = Access level via PROFINET
Record parameters	Consist of a group, a line and multiple sequential (listed) elements, which can have different properties and different values.	A00 Save values ▶ A00[0] Start: Value = Start action ▶ A00[1] Progress: Value = Display action progress ▶ A00[2] Result: Value = Display action result

Tab. 4: Parameter types

6.2.4 Parameter structure

Every parameter has specific coordinates with the following structure.



- ▶ Axis (optional)
Axis to which an axis-specific parameter is assigned; not applicable for global parameters (value range: 1 - 4).
- ▶ Group
The thematic group to which a parameter belongs (value range: A – Z).
- ▶ Line
Distinguishes the parameters within a parameter group (value range: 0 – 999).
- ▶ Element (optional)
Elements of an array or record parameter (value range: 0 – 16000).

6.2.5 Parameter visibility

The visibility of a parameter is primarily controlled by the access level you set in DriveControlSuite and by the properties you project for the respective drive controller (e.g. hardware, firmware and application). A parameter can also be shown or hidden depending on other parameters or settings. For example, the parameters of an additional function are only shown as soon as you activate the relevant additional function.

Access level

The access options for the individual software parameters are ranked hierarchically and divided into individual levels. This means that parameters can be hidden for a specific purpose and, relatedly, their configuration options can be locked starting from a specific level.

Each parameter has one access level for read access (visibility) and one access level for write access (editability). The following levels are present:

- ▶ Level 0
Elementary parameters
- ▶ Level 1
Important parameters of an application
- ▶ Level 2
Important parameters for service with extensive diagnostic options
- ▶ Level 3
All parameters needed for commissioning and optimizing an application

The parameter A10 Access level controls general access to parameters:

- ▶ Over CANopen or EtherCAT (A10[2])
- ▶ Over PROFINET (A10[3])



Information

It is not possible to write to or read the parameter hidden in DriveControlSuite during communication via fieldbus.

Hardware

Which parameters are available to you in DriveControlSuite is determined by which series you select in the configuration dialog for the drive controller, for example, or whether you project an option module. Basically, the only parameters that are displayed are the ones you need to parameterize the configured hardware.

Firmware

Due to the further development and updating of functions for the 6th generation of drive controllers, new parameters and also new versions of existing parameters are continuously being implemented in DriveControlSuite and in the firmware. The parameters are displayed in the software according to the DriveControlSuite version used and the configured firmware version of the respective drive controller.

Applications

Applications generally differ in terms of functions and their control. For this reason, different parameters are available with each application.

6.3 Signal sources and process data mapping

The transmission of control signals and set values in DriveControlSuite meets the following principles.

Signal sources

Drive controllers are controlled either over a fieldbus, using mixed operation consisting of a fieldbus system and terminals or exclusively using terminals.

You can use the corresponding selection parameters, referred to as signal sources, to configure whether the control signals and set values of the application are obtained over a fieldbus or using terminals.

In case of activation over a fieldbus, parameters that are selected as data sources for control signals or set values must be part of the subsequent process data mapping. In the case of activation using terminals, the respective analog or digital inputs are specified directly.

Process data mapping

If you are working with a fieldbus system and have selected the source parameters for control signals and set values, configure the fieldbus-specific settings, e.g. the assignment of the process data channels for transmitting receive and transmit process data, as the last step.

6.4 Non-volatile memory

All project configurations, parameterizations and related changes to parameter values are in effect after transmission to the drive controller, but are only stored in volatile memory.

Saving to a drive controller

To save the configuration in non-volatile memory on a drive controller, you have the following options:

- ▶ Saving the configuration using the *Save values* wizard:
Project menu > Wizards area > Projected axis > *Save values* wizard: Select the *Save values* action
- ▶ Saving the configuration using the parameter list:
Project menu > Parameter list area > Projected axis > Group A: Drive controller > A00 *Save values*: Set the parameter A00[0] to the value 1: Active
- ▶ Saving the configuration using the S1 operating button:
Drive controller with S1 operating button: Press the operating button for 3 seconds

Saving to all drive controllers within a project

To save the configuration in non-volatile memory on several drive controllers, you have the following options:

- ▶ Saving the configuration using the toolbar:
Toolbar > *Save values* icon: Click the *Save values* icon
- ▶ Saving the configuration using the *Online functions* window:
Project menu > *Online connection* button > *Online functions* window: Click on *Save values (A00)*



Information

Do not shut off the drive controller while saving. If the supply voltage to the control unit is interrupted while saving, the drive controller will start without an executable configuration the next time it is switched on. In this case, the configuration must be transferred to the drive controller again and stored in non-volatile memory.

7 Commissioning

Are you looking to operate drive controllers with a controller over an EtherCAT network?

The following chapters describe commissioning using DriveControlSuite in combination with the TwinCAT 3 or CODESYS V3 automation software.

We assume the following system environment as an example so that you can follow the individual commissioning steps exactly:

- ▶ Drive controllers from the PMC SC6 or PMC SI6 series with firmware version 6.5-K-EC or later
- ▶ DriveControlSuite commissioning software version 6.5-K or later

either in combination with

- ▶ Beckhoff CX2030 embedded PC
- ▶ Beckhoff TwinCAT 3 automation software

or in combination with

- ▶ CODESYS SoftMotion controller
- ▶ CODESYS V3 automation software

Commissioning is divided into the following steps:

1. DriveControlSuite
Project all of the drive controllers, i.e. application type, device control systems, process data for fieldbus communication and mechanical axis model in DriveControlSuite.
Depending on the selected application (CiA 402 or CiA 402 HiRes Motion), scale your axis models on the drive controller side or the controller side.
In both cases, transfer your configuration to the drive controllers of the system network.
2. TwinCAT 3 or CODESYS V3
Scale your axis model if necessary and then map your entire hardware environment in the respective software.
Synchronize the operation of the distributed clocks in all EtherCAT nodes and configure the communication of individual nodes over the EoE protocol.
Finally, transfer the entire configuration to the controller and then start up your EtherCAT system.

7.1 DS6: Configuring the drive controller

Project and configure all drive controllers for your drive system in DriveControlSuite (see the chapter Structure of the program interface).



Information

Since you are working with a controller, the following steps are described based on the CiA 402 application in combination with the CiA 402 device control. Operation with drive-based applications is also possible.



Information

Always perform the steps described below in the specified order!

Some parameters are interdependent and do not become accessible to you until you have first configured certain settings. Follow the steps in the specified sequence so that you can finish the parameterization completely.

7.1.1 Initiating the project

In order to be able to configure all drive controllers and axes of your drive system using DriveControlSuite, you must record them as part of a project.


7.1.1.1 Projecting the drive controller and axis

Create a new project and project the first drive controller along with the accompanying axis.

Creating a new project

1. Start DriveControlSuite.
2. On the start screen, click **Create new project**.
 - ⇒ The new project is created and the configuration dialog for the first drive controller opens.
 - ⇒ The **Drive controller** button is active.

Projecting the drive controller

1. Properties tab:
Establish the relationship between your circuit diagram and the drive controller to be projected in DriveControlSuite.
Reference: Specify the reference code (equipment code) of the drive controller.
Designation: Give the drive controller a unique name.
Version: Version your project configuration.
Description: If necessary, specify additional supporting information, such as the change history of the project configuration.
2. Drive controller tab:
Select the series and device type of the drive controller.
Firmware: Select the EtherCAT version V 6.x -EC.
3. Option modules tab:
Safety module: If the drive controller is part of a safety circuit, select the corresponding safety module.
4. Device control tab:
Device control: Select CiA 402.
Rx process data, Tx process data: Select the receive and transmit process data.
 - 4.1. If you are working with hardware and software products from Beckhoff and are using the SDO Info service, select EtherCAT Rx SDO Info and EtherCAT Tx for the transmission of the EtherCAT process data. You set up the SDO Info service in TwinCAT 3 (see [SDO Info service](#) [ 105]).
 - 4.2. If you are working with a CODESYS SoftMotion controller and the CODESYS V3 automation software, select EtherCAT Rx and EtherCAT Tx for the transmission of the EtherCAT process data.



ATTENTION!

Change of addressing when changing the template

If you change the template from EtherCAT Rx to EtherCAT Rx SDO Info, the addressing of the elements of array and record parameters also changes. Note this in particular for existing configurations. For the templates, various ESI files are created. When changing the template, you must create a new ESI file using the wizards in DriveControlSuite and provide it to TwinCAT 3. A template change also causes a change to the revision number of the drive controller. Therefore, restart the drive controller after changing the template.



Information

Make sure that you project the correct series in the Drive controller tab. The projected series cannot be changed afterwards.

Projecting the axis

1. Click on Axis 1.
2. Properties tab:
Establish the connection between your circuit diagram and the axis to be projected in DriveControlSuite.
Reference: Specify the reference code (equipment code) of the axis.
Designation: Give the axis a unique name.
Version: Version your project configuration.
Description: If necessary, specify additional supporting information, such as the change history of the project configuration.
3. Application tab:
Select the desired application.
If you are working with a CODESYS SoftMotion controller and the CODESYS V3 automation software, we recommend CiA 402 HiRes Motion (version with user-defined units of measure).
If you are working with hardware and software products from Beckhoff, we recommend CiA 402 (incremental version).
4. Motor tab:
Select the type of motor operated using this axis. If you are working with motors from third-party suppliers, enter the accompanying motor data at a later time.
5. Repeat steps 2 – 4 for the 2nd axis (only for double-axis controllers).
6. Confirm with OK.

7.1.1.2

Configuring safety technology

If the drive controller is part of a safety circuit, you must configure the safety technology in accordance with the commissioning steps outlined in the corresponding manual in the next step (see [Detailed information \[124\]](#)).

7.1.1.3

Creating other drive controllers and modules

In DriveControlSuite, all drive controllers within a project are grouped using modules. If you add a new drive controller to your project, be sure to always assign it to an existing module. Group drive controllers in a module if, for example, they are located in the same control cabinet or work together to operate the same machine part.

Creating a drive controller

1. In the project tree, select your project P1 > module M1 > context menu Create new drive controller.
⇒ The drive controller is created in the project tree and the configuration dialog opens.
2. Project the drive controller as described in Projecting the drive controller and axis.
3. Repeat the steps for all other drive controllers that you want to project.

Creating a module

1. In the project tree, select your project P1 > context menu Create new module.
⇒ The module is created in the project tree.
2. Project the module as described in [Projecting the module \[28\]](#).
3. Repeat the steps for all other modules that you want to project.

7.1.1.4 Projecting the module

Give your module a unique name, enter the reference code and, as an option, store additional information like the version and change history of the module.

1. Select the module in the project tree and click on **Project configuration** in the project menu.
⇒ The configuration dialog for the module opens.
2. Establish the relationship between your circuit diagram and the module in DriveControlSuite.
Reference: Specify the reference code (equipment code) of the module.
Designation: Give the module a unique name.
Version: Version the module.
Description: If necessary, specify additional supporting information, such as the change history of the module.
3. Confirm with **OK**.

7.1.1.5 Projecting the project

Give your project a unique name, enter the reference code and, as an option, store additional information like the version and change history of the project.

1. Mark the project in the project tree and click on **Project configuration** in the project menu.
⇒ The configuration dialog for the project opens.
2. Establish the relationship between your circuit diagram and the project in DriveControlSuite.
Reference: Specify the reference code (equipment code) of the project.
Designation: Give the project a unique name.
Version: Version the project.
Description: If necessary, specify additional supporting information, such as the change history of the project.
3. Confirm with **OK**.

7.1.2 Parameterizing general EtherCAT settings

- ✓ You have projected a device control with the process data as part of drive controller and axis projecting.
1. Select the relevant drive controller in the project tree and click on the desired projected axis in the **Project menu > Wizard** area.
 2. Select the **EtherCAT wizard**.
 3. **A213 Fieldbus scaling:**
Leave the default setting at 1: Native (values are passed unchanged).
 4. **A258 EtherCAT PDO-Timeout:**
In order to be able to detect a communication failure, monitor the arrival of cyclical process data by defining a PDO timeout.
Permitted value range: 0 – 65535 ms.
Please note:
0 and 65535 = Monitoring is inactive
1 to 65531 = Monitoring is active
65532 = Monitoring is active but the loss of an individual data packet is ignored
65533 = Monitoring is active but the loss of 3 data packets in a row is ignored
 5. **Optional:** If you would like to use the SDO Info service, define which objects the controller can read out via SDO Info using A268.

7.1.3 Configuring PDO transmission

PDO channels are able to transmit control and status information in real time as well as actual and set values from an EtherCAT master to EtherCAT slaves and vice versa.

PDO communication allows for several PDO channels to be operated simultaneously per transmission and sending direction. The channels for axes A and B each include a PDO with a defined sequence of up to 24 parameters to be transmitted. These are free to be configured in any way. One channel is reserved for FSoE communication and is parameterized automatically.

In order to guarantee error-free communication between the controller and drive controller, Pilz offers an application-dependent pre-assignment of the channels which can be changed at any time.

7.1.3.1 Adapting RxPDO

- ✓ You have configured the global EtherCAT settings.
- 1. Select the relevant drive controller in the project tree and click on the desired projected axis in the Project menu > Wizard area.
- 2. Select the EtherCAT wizard > Received process data RxPDO.
- 3. Check the presets and/or configure the process data according to your requirements.
A225[0] – A225[23], A226[0] – A226[23]:
Parameters whose values are received by the respective drive controller from the controller. The position of the parameters provides information about the associated receiving sequence.

7.1.3.2 Adapting TxPDO

- ✓ You have configured the global EtherCAT settings.
- 1. Select the relevant drive controller in the project tree and click on the desired projected axis in the Project menu > Wizard area.
- 2. Select the EtherCAT wizard > Transmitted process data TxPDO.
- 3. Check the presets and/or configure the process data according to your requirements.
A233[0] – A233[23], A234[0] – A234[23]:
Parameters whose values the respective drive controller sends to the controller. The position of the parameters provides information about the associated transmission sequence.

7.1.4 Mapping the mechanical axis model

To be able to put your real drive train with one or more drive controllers into operation, you must map your complete mechanical environment in DriveControlSuite.



Information

Note that the scaling of the axis depends on the CiA 402 application you have projected.

If you have selected the CiA 402 HiRes Motion application, scale the axis in the drive controller, i.e. parameterize it in DriveControlSuite.

If you have selected the incremental version of the CiA 402 application, scale the axis in the controller.

When scaling the axis, follow the instructions for the application you are projecting.

7.1.4.1 Parameterizing the motor

You have projected one of the following motors:

Synchronous servo motor with EnDat 2.2 digital encoder or EnDat 3 (with optional brake)

By projecting the corresponding motor, limiting values for currents and torques as well as associated temperature data are automatically transferred to the respective parameters of the individual wizards. All additional data on the brake and encoder is transferred at the same time.

Lean motor without encoder (with optional brake)

By projecting the corresponding motor, limiting values for currents and torques as well as associated temperature data are automatically transferred to the respective parameters of the individual wizards. You only have to parameterize the cable length in use. Even the brake purging and engaging times are already stored. You just have to activate the brake.

1. Select the relevant drive controller in the project tree and click on the desired projected axis in the Project menu > Wizard area.
2. Select the Motor wizard.
3. B101 Cable length:
Select the cable length of the power cable in use.
4. Repeat the steps for the 2nd axis (only for double-axis controllers).

Then activate the brake.

1. Select the relevant drive controller in the project tree and click on the first projected axis in the Project menu > Wizard area.
2. Select the Brake wizard.
3. F00 Brake:
Select 1: Active.
4. Repeat the steps for the 2nd axis (only for double-axis controllers).

Motor protection

All models of the 6th drive controller generation feature an i²t model — a computational model for thermal monitoring of the motor. To activate it and set up the protective function, configure the following settings (deviating from the presets): U10 = 2: Warning and U11 = 1.00 s. This model can be used instead of or in addition to temperature-monitored motor protection.

7.1.4.2 Parameterizing the axis model

Parameterize the setup of your drive in this order:

- ▶ Define the axis model
- ▶ Scale the axis
- ▶ Parameterize the position and velocity window
- ▶ Limit the axis (optional)
 - Limit the position
 - Limit the velocity, acceleration and jerk
 - Limit the torque and force



Information

If you are using a double-axis controller with two projected axes, you must parameterize the axis model for each axis individually.

7.1.4.2.1 Defining the axis model

1. Select the relevant drive controller in the project tree and click on the desired projected axis in the Project menu > Wizard area.
2. Select the Axis model wizard.
3. I05 Type of axis:
In order to individually configure the units of measure and the number of decimal places for specifying and displaying positions, velocity values and acceleration values, select 0: Free setting, rotational or 1: Free setting, translational.
4. B26 Motor encoder:
Define the interface to which the motor encoder is connected.
5. I02 Position encoder:
Define the interface to which the position encoder is connected.
6. I00 Position range:
Define the travel range. Note that 1: Endless is possible only in combination with the CiA 402 HiRes Motion application.

7.1.4.2.2 CiA 402: Scaling an axis

- ✓ You have projected the incremental version of the CiA 402 application. Scale the axis in the controller software and, as described below, specify only the increments per motor revolution in DriveControlSuite.
- 1. Select the relevant drive controller in the project tree and click on the desired projected axis in the Project menu > Wizard area.
- 2. Select the Axis model wizard > Axis: Scaling.
- 3. A585[1] Feed constant.Shaft revolutions¹ and A585[0] Feed constant. Feed²
Leave the presets of A585[1] at 1 U and A585[0] at 1048576 inc (= 20 bit = 2²⁰) and adjust the corresponding value in the controller software.
- 4. I06 Decimal places position:
Since you are working with the incremental version of the CiA 402 application, leave the default value at 0.
- 5. A571 Polarity:
Use the polarity to specify the direction of interpretation between the axis movement and motor movement.



Information

Parameter I297 Maximum speed position encoder must be parameterized according to your application case. If I297 is set too low, the permitted maximum speed is exceeded even at normal operating speeds. On the other hand, if I297 is set too high, measuring errors of the encoder can be overlooked.

I297 depends on the following parameters: I05 Type of axis, I06 Decimal places position, I09 Measure unit as well as I07 Distance factor numerator position and I08 Distance factor denominator position or A585 Feed constant for CiA 402. If you have made changes to one of the parameters listed, select I297 accordingly as well.

¹ Corresponds to CiA 402 Feed constant 6092 hex, 2 hex for axis A and 6892 hex, 2 hex for axis B

² Corresponds to CiA 402 Feed constant 6092 hex, 1 hex for axis A and 6892 hex, 1 hex for axis B

7.1.4.2.3 CiA 402 HiRes Motion: Scaling an axis

- ✓ You have projected the HiRes version of the CiA 402 application. Scale the axis as described below and specify only the number of decimal places in the controller software, i.e. the value parameterized in I06.
- 1. Select the relevant drive controller in the project tree and click on the desired projected axis in the Project menu > Wizard area.
- 2. Select the Axis model wizard > Axis: Scaling.
- 3. A584[0] Gear ratio.Motor revolutions and A584[1] Gear ratio.Shaft revolutions:
Specify the gear ratio.
- 4. A585[1] Feed constant.Shaft revolutions and A585[0] Feed constant. Feed:
Specify the feed rate per revolution of the gear unit output.
- 5. I06 Decimal places position:
Specify the number of decimal places for specifying and displaying positions, velocity values and acceleration values. Note that changing this value means the decimal place is moved.
- 6. I09 Measure unit:
Specify the desired unit of measure.
- 7. A571 Polarity:
Use the polarity to specify the direction of interpretation between the axis movement and motor movement.
- 8. A568 Position range limit (only for endless travel range I00 = 1):
Specify the revolution length of the axis.



Information

Parameter I297 Maximum speed position encoder must be parameterized according to your application case. If I297 is set too low, the permitted maximum speed is exceeded even at normal operating speeds. On the other hand, if I297 is set too high, measuring errors of the encoder can be overlooked.

I297 depends on the following parameters: I05 Type of axis, I06 Decimal places position, I09 Measure unit as well as I07 Distance factor numerator position and I08 Distance factor denominator position or A585 Feed constant for CiA 402. If you have made changes to one of the parameters listed, select I297 accordingly as well.

7.1.4.2.4 Parameterizing the position and velocity window

Enter position limits and velocity zones for set values. To do so, parameterize boundary values for reaching a position or velocity.

1. Select the **Axis model wizard** > **Window position, velocity**.
2. **C40 Velocity window:**
Parameterize a tolerance window for velocity tests.
3. **I22 Target window:**
Parameterize a tolerance window for position tests.
4. **I87 Actual position in window time:**
Parameterize how long a drive must stay in the specified position window before a corresponding status message is output.
5. **A546 Following error window:**
Parameterize a tolerance window for lag tests.

7.1.4.2.5 Limiting the axis

If necessary, limit the movement variables for position, velocity, acceleration, jerk as well as torque/force according to the applicable conditions for your axis model.

Limiting the position (optional)

1. Select the relevant drive controller in the project tree and click on the desired projected axis in the **Project menu** > **Wizard** area.
2. Select the **Axis model wizard** > **Limit: Position**.
3. If necessary, limit the position of your axis using a software or hardware limit switch to secure the travel range.

Limiting velocity, acceleration, jerk (optional)

The default values are designed for slow velocities without gear units. For this reason, adapt the saved values.

For example, verify the maximum velocity of the motor (B83) against the velocity of the output (I10).

1. Select the **Motor wizard**.
2. Determine the maximum possible motor velocity in parameter B83 v-max motor.
3. Select the **Axis model wizard** > **Axis: Scaling** > **Conversion of positions, velocities, accelerations, torque/force area**.
4. **Velocity line:**
Enter the maximum motor velocity from B83 in the **Velocity line** of the **Motor** column and confirm with ENTER.
⇒ The maximum velocity of the motor is converted to the output.
5. Repeat the procedure for other limits, such as for the gear unit input speed (C11).
6. Select the **Axis model wizard** > **Limit: Velocity, acceleration, jerk**.
7. **I10 Maximal speed:**
Limit the maximum velocity of the output, taking into account the determined system limits.
8. Determine the limiting values for acceleration and jerk if necessary and enter them into the associated parameters.

Limiting torque/force (optional)

The default values take into account the rated operation together with the overload reserves.


1. Select the **Axis model wizard** > **Limit: Torque/force**.
2. If the motor force must be limited, adapt the saved values as necessary.

7.1.5 Synchronizing EtherCAT nodes

Precise synchronization of the EtherCAT nodes is absolutely required for spatially distributed processes that require simultaneous actions (path interpolation). EtherCAT provides the distributed clocks (DC-Sync) method for this. Synchronization using distributed clocks is more precise when compared to the SyncManager event synchronization (SM-Sync) because it is subject to fewer fluctuations. For this reason, DC-Sync is pre-configured in EtherCAT masters and slaves.

PLL synchronization wizard

Leave the presets in the first step and optimize them if necessary once you have commissioned the EtherCAT network and can assess and evaluate the quality of communication.

For more information on synchronization, and how to adjust it after the fact, see [Synchronization](#) [ 84].

7.1.6 Transmitting and saving the configuration

In order to transmit and save the configuration to one or more drive controllers, you must connect your PC and the drive controllers over the network.



WARNING!

Injury to persons and material damage due to axis movement!

If there is an online connection between DriveControlSuite and the drive controller, changes to the configuration can lead to unexpected axis movements.

- Only change the configuration if you have visual contact with the axis.
- Make sure that no people or objects are within the travel range.
- For access via remote maintenance, there must be a communication link between you and a person on site with eye contact to the axis.



Information

During the search, all drive controllers within the broadcast domain are found via IPv4 limited broadcast.

Requirements for finding a drive controller in the network:

- Network supports IPv4 limited broadcast
- All drive controllers and the PC are in the same subnet (broadcast domain)

Transmitting the configuration

- ✓ The drive controllers are switched on.
- 1. In the project tree, select the module under which you have recorded your drive controller and click **Online connection** in the project menu.
 - ⇒ The **Add connection** dialog box opens. All drive controllers found via IPv4 limited broadcast are displayed.
- 2. **Direct connection tab > IP address column:**
Activate the IP addresses in question and confirm your selection with **OK**.
 - ⇒ The **Online functions** window opens. All drive controllers connected through the selected IP addresses are displayed.
- 3. Select the drive controller to which you would like to transfer the configuration. Change the selection of transmission type from **Read** to **Send**.
- 4. Change the selection **Create new drive controller:**
Select the configuration that you would like to transfer to the drive controller.
- 5. Repeat steps 3 and 4 for all other drive controllers to which you would like to transfer your configuration.
- 6. **Online tab:**
Click **Establish online connections**.
 - ⇒ The configuration is transferred to the drive controllers.

Saving a configuration

- ✓ You have successfully transmitted the configuration.
- 1. Online functions window:
Click Save values (A00).
 - ⇒ The Save values (A00) window opens.
- 2. Click Start action.
 - ⇒ The configuration is stored on the drive controllers in non-volatile memory.
- 3. Close the Save values (A00) window.



Information

For the configuration to take effect on the drive controller, a restart is required when the configuration is saved on the drive controller for the first time or when changes are made to the firmware or process data mapping.

Restarting a drive controller

- ✓ You have stored the configuration on the drive controller in non-volatile memory.
- 1. Online functions window:
Click Restart (A09).
 - ⇒ The Restart (A09) window opens.
- 2. Select which of the connected drive controllers you want to restart.
- 3. Click Start action.
- 4. Confirm the safety note with OK.
 - ⇒ The Restart (A09) window closes.
- ⇒ The fieldbus communication and connection between DriveControlSuite and drive controllers are interrupted.
- ⇒ The selected drive controllers restart.

7.1.7 Activating the control panel and testing the configuration



WARNING!

Injury to persons and material damage due to axis movement!

When you activate the control panel, DriveControlSuite gives you sole control of the motions of the axis. If you are using a controller, it no longer monitors the axis movements after the control panel is activated. The controller cannot intervene to prevent collisions. The controller takes over control again when the control panel is deactivated, which can cause unexpected axis movements.

- Do not switch to other windows when the control panel is active.
- Only use the control panel if you have visual contact with the axis.
- Make sure that no people or objects are within the travel range.
- For access via remote maintenance, there must be a communication link between you and a person on site with eye contact to the axis.

- ✓ You have successfully saved the configuration.
 - ✓ There must not be any active safety function.
 - ✓ The drive controller is switched on and connected to the network.
 - ✓ There is an online connection between DriveControlSuite and the drive controller.
1. Select the relevant drive controller in the project tree and click on the desired projected axis in the Project menu > Wizard area.
 2. Select the Jog control panel wizard.
 3. Click Control panel on and then Enable.
⇒ The drive is controlled using the activated control panel.
 4. Move the axis step-by-step and test the direction of motion, velocity, distances, etc. using the Jog+, Jog-, Jog step+ and Jog step- buttons.
 5. Optimize your project configuration based on your test results as necessary.
 6. To deactivate the control panel, click on Control panel off.



Information

Jog+ and Jog- cause a continual manual movement in the positive or negative direction. Jog step+ and Jog step- move the axis relative to the current actual position by the increment specified in I14.

Jog+ and Jog- have a higher priority than Jog step+ and Jog step-.

7.2 TwinCAT 3: Putting the EtherCAT system into operation

TwinCAT 3 automation software gives you the option to map the hardware environment of your EtherCAT system and to configure and parameterize all necessary bus parameters including data exchange via master and slaves (also see [TwinCAT 3 program interface \[18\]](#)).

Note that all system nodes have to be networked physically before commissioning. In addition, you have projected the drive controllers in question in advance, i.e. EtherCAT slaves in DriveControlSuite, and transmitted the configuration to those drive controllers.



Information

For the following description, we require that you have projected the CiA 402 application.



Information

Always perform the steps described below in the specified order!

Some parameters are interdependent and do not become accessible to you until you have first configured certain settings. Follow the steps in the specified sequence so that you can finish the parameterization completely.

7.2.1 Creating and exporting an ESI file

The functions and properties of the Pilz drive controllers are described in the form of various objects and collected in an ESI file.

Because you are working with TwinCAT 3, generating an ESI file is mandatory. The file must be made available to TwinCAT 3 in the directory specified below. Be aware that TwinCAT 3 can only read in one ESI file per drive controller series. If you use different applications or PDO transmission configurations, you must expand your ESI accordingly (see [Modular ESI files \[100\]](#)).

In case of any change to the PDO transmission or configuration template, a new ESI file must be exported and imported in TwinCAT 3.

- ✓ You are in DriveControlSuite and have completed the configuration of the PDO transmission.
- 1. Select the relevant drive controller in the project tree and click on the desired projected axis in the Project menu > Wizard area.
- 2. Select the EtherCAT wizard.
- 3. Click on Create ESI.
 - ⇒ The Write ESI file dialog box opens.
- 4. Save the XML file in the directory where the controller will read it in (TwinCAT 3 default installation: C:\TwinCAT\3.1\Config\IO\EtherCAT).
 - ⇒ The ESI file is imported the next time TwinCAT 3 is started.

7.2.2 Activating the EtherCAT master

- ✓ You have already projected all drive controllers of your system using DriveControlSuite and transmitted the configuration to the individual drive controllers. The EtherCAT master is connected to the network, all system components are energized and the infrastructure is ready for operation. You have saved the generated ESI file in the specified directory.
- 1. Start TwinCAT XAE.
 - ⇒ The stored ESI file is read in upon program start and the main window opens. Start page tab is active.
- 2. Select File > New > Project...
- ⇒ The New Project window opens.
- 3. Select Installed > Templates > TwinCAT Projects > TwinCAT XAE Project (XML format).
- 4. Name, Location, Solution name:
Label the project and enter a save location and an internal project name.
- 5. Close the window.
- 6. Continue based on the type of installation:
 - 6.1. If the run-time package (EtherCAT master) and TwinCAT XAE have been installed on the same PC, they are connected to each other automatically.
Continue to step 16.
 - 6.2. If the run-time package (EtherCAT master) and TwinCAT System Manager have been installed on different PCs, you must connect them to each other.
If routing to the controller has already been created, continue with step 15.
If a new device is to be connected, perform all of the following steps.
- 7. Click on the <Local> list field in the TwinCAT XAE toolbar and select Choose Target System...
- ⇒ The Choose Target System window opens.
- 8. Click on Search (Ethernet)...
- ⇒ The Add Route Dialog window opens.
- 9. Click on Broadcast Search.
 - ⇒ The Select Adapter(s) window opens.
- 10. Highlight the adapter that is connected with your controller and confirm with OK.
 - ⇒ All available control systems are listed.
- 11. Highlight the desired controller and confirm with Add Route.
 - ⇒ The Add Remote Route window opens.
- 12. Under Remote User Credentials, enter the following data:
User name: Administrator
Password: 1
- 13. Confirm with OK.
- 14. Close the Add Route Dialog and Choose Target System windows.
- 15. Click on the <Local> list field in the TwinCAT XAE toolbar and select the added controller from the picklist.
 - ⇒ The EtherCAT master is saved as the target system.
- 16. In order to be able to configure the EtherCAT system online, you must activate Config mode for the TwinCAT XAE software.
Select the menu TWINCAT > Restart TwinCAT (Config mode).
 - ⇒ The Restart TwinCAT System in Config Mode dialog box opens.
- 17. Confirm with OK.
 - ⇒ The EtherCAT master is saved as the target system, TwinCAT XAE is in Config mode.

7.2.3 Scanning the hardware environment

If all system components are connected to the EtherCAT network and the network is energized, it is possible to scan for connected devices automatically. In this scenario, TwinCAT XAE searches for connected devices and terminals and integrates them into the existing project in accordance with their configuration entries in the accompanying ESI files.

If the actual EtherCAT infrastructure is not available, i.e. you are configuring in offline mode, you must map and project all connected devices manually in TwinCAT XAE. You can get more detailed information on this in the online help tool of the TwinCAT XAE software.

✓ You have activated Config mode.

1. In the solution explorer, navigate to I/O > Devices > Scan context menu.
2. Confirm the HINT: Not all types of devices can be found automatically dialog box with OK.
 - ⇒ TwinCAT XAE scans the EtherCAT system for the EtherCAT master.
 - ⇒ The ... new I/O devices found dialog box opens.
3. Activate the EtherCAT master in question and confirm with OK.
 - ⇒ The EtherCAT master is created in the solution explorer under I/O > Devices as a device (EtherCAT).
 - ⇒ The Scan for boxes? dialog box opens.
4. Confirm with Yes.
 - ⇒ TwinCAT XAE scans the EtherCAT system for the EtherCAT slaves.
 - ⇒ The EtherCAT drive(s) added dialog box opens.
5. Append linked axis to:

If you are using controller-based operation for the drive controller, select the desired option and confirm with OK in order to activate the NC or CNC function. For drive-based control, you prevent the creation of an axis using Cancel.

 - ⇒ The EtherCAT slaves are created in the solution explorer.
The Activate Free Run dialog box opens.
6. In order to shift the system components during configuration into free run mode and thereby enable verification of the signal exchange, confirm with Yes.
 - ⇒ EtherCAT master and slaves are created in TwinCAT XAE.

7.2.4 Expanding the startup list

With the help of the startup list, you can change the values of objects during startup of the EtherCAT state machine via the CoE protocol. You can add objects from the object directory of the ESI file to the startup list. If you are using the SDO Info service, additional objects are also available. The objects are accessed in the order in which they appear in the startup list.



Selecting and adding an object from the list

Objects that are either contained in the object directory of the ESI file or have previously been read from the drive controller via SDO Info are available for selection from a list in TwinCAT XAE.

- ✓ EtherCAT master and slaves are created in TwinCAT XAE.
- 1. In the solution explorer, navigate to the EtherCAT slave for which you want to add an object to the startup list.
- 2. Double-click the EtherCAT slave.
 - ⇒ The settings open in the main window.
- 3. In the main window, switch to the Startup tab.
 - ⇒ The current startup list is displayed.
- 4. Click New...
 - ⇒ The Edit CANopen Startup Entry window opens.
 - ⇒ All available objects are listed.
- 5. In the list, double-click on the object that you want to add to the startup list.
 - ⇒ The Set Value Dialog window opens.
- 6. Dec, Hex, Enum:
Enter the value to be written to the object in the field corresponding to the desired data format (decimal specification, hexadecimal specification or selection from list).
- 7. Confirm the value with OK.
- 8. Transition:
In the Edit CANopen Startup Entry window, activate the state change upon which the object is to be written:
 - 8.1. I -> P: State change from Init to Pre-Operational
 - 8.2. P -> S: State change from Pre-Operational to Safe-Operational
 - 8.3. S -> O: State change from Safe-Operational to Operational
 - 8.4. O -> S: State change from Operational to Safe-Operational
 - 8.5. S -> P: State change from Safe-Operational to Pre-Operational
- 9. Comment:
If required, enter a comment that will be displayed in the startup list for the object.
- 10. Confirm with OK.
 - ⇒ The object is added to the startup list.

Adding an object via index and subindex

Alternatively, you can add objects to the startup list by their index and subindex.

For manufacturer-specific parameters, calculate the index and subindex of the object from the parameter coordinate in advance (see [Manufacturer-specific parameters: 2000 hex – 53FF hex](#) [ 114] for axis A and [Manufacturer-specific parameters: A000 hex – D3FF hex](#) [ 116] for axis B).

- ✓ EtherCAT master and slaves are created in TwinCAT XAE.
- 1. In the solution explorer, navigate to the EtherCAT slave for which you want to add an object to the startup list.
- 2. Double-click the EtherCAT slave.
 - ⇒ The settings open in the main window.
- 3. In the main window, switch to the Startup tab.
 - ⇒ The current startup list is displayed.
- 4. Click New....
 - ⇒ The Edit CANopen Startup Entry window opens.
- 5. Transition:
Activate the state change upon which the object is to be written:
 - 5.1. I -> P: State change from Init to Pre-Operational
 - 5.2. P -> S: State change from Pre-Operational to Safe-Operational
 - 5.3. S -> O: State change from Safe-Operational to Operational
 - 5.4. O -> S: State change from Operational to Safe-Operational
 - 5.5. S -> P: State change from Safe-Operational to Pre-Operational
- 6. Index (hex):
Enter the index of the object (hexadecimal specification).
- 7. Sub-index (dec):
Enter the subindex of the object (decimal specification).
- 8. Data (hexbin):
Enter the value to be written to the object.
- 9. Comment:
If required, store a comment that will be displayed in the startup list for the object.
- 10. Confirm with OK.
 - ⇒ The object is added to the startup list.

7.2.5 Configuring synchronization using distributed clocks

As the more precise of the two sync methods, synchronization using distributed clocks (DC-Sync) is pre-configured in the EtherCAT slaves. Check the associated settings for the EtherCAT master and slaves.

- ✓ You have fully configured the associated axis model in DriveControlSuite.
- 1. Navigate to the EtherCAT master in the solution explorer.
- 2. Double-click the EtherCAT master.
 - ⇒ The settings open in the main window.
- 3. In the main window, switch to the EtherCAT tab and click Advanced Settings....
 - ⇒ The Advanced Settings window opens.
- 4. In the tree view, select Distributed Clocks.
- 5. Automatic DC Mode Selection:
This option must be activated.
- 6. Close the window.
- 7. Navigate to the first EtherCAT slave in the solution explorer.
- 8. In the main window, switch to the DC tab and click Advanced Settings....
 - ⇒ The Advanced Settings window opens.
- 9. Enable:
This option must be activated.
- 10. DC enabled (multiplier = 1):
This list entry must be selected.
- 11. Sync Unit Cycle (μ s):
Check the default value for the cycle time of the controller and change it if necessary.
- 12. Enable SYNC 0:
This option must be activated.
- 13. Close the window.
- 14. Repeat steps 7 – 13 for each additional slave in your EtherCAT network.
 - ⇒ The EtherCAT master and slaves will now be synchronized with the first EtherCAT slave that has the distributed clocks option enabled.

7.2.6 Configuring synchronization via SyncManager event

Synchronization using distributed clocks (DC-Sync) is pre-configured in the EtherCAT slaves. You have the option of switching the synchronization for one or more EtherCAT slaves manually to synchronization via SyncManager event (SM-Sync).

1. In the solution explorer, navigate to the EtherCAT slave for which you want to change the synchronization to SM-Sync.
2. Double-click the EtherCAT slave.
 - ⇒ The settings open in the main window.
3. In the main window, switch to the DC tab.
4. Operation Mode:
Select the SM Synchronous list entry from the picklist.
 - ⇒ You have changed the synchronization for the EtherCAT slave.
5. Repeat the steps for each additional slave of your EtherCAT network for which you want to switch the synchronization to SM-Sync.

7.2.7 Controller-based axis control

In order to control one or more drive controllers with controller-based operation, start by parameterizing the axes and then program your control.

7.2.7.1 Parameterizing an axis

1. In the solution explorer, navigate to Motion > NC-Task 1 SAF > Axes > Axis 1.
2. Double-click the axis.
⇒ The settings open in the main window.
3. In the main window, switch to the Settings tab.
4. Unit:
Select degrees (°) as the unit.
5. Switch to the Parameters tab.
6. Open the Maximum Dynamics parameter list.
7. Parameterize associated limit values for velocity, acceleration and deceleration.
8. Open the Limit Switches parameter list.
9. Soft Position Limit Minimum Monitoring:
If you want to put a lower negative limit on position values, select the True list entry and enter the associated value in Minimum Position.
10. Soft Position Limit Maximum Monitoring:
If you want to put an upper positive limit on position values, select the True list entry and enter the associated value in Maximum Position.
11. In the solution explorer, navigate to Axis > Enc.
12. In the main window, switch to the Parameters tab.
13. Open the Encoder Evaluation parameter list.
14. Scaling Factor Numerator:
Specify the value 0.000343322 ($360 \div 1048576$) – in accordance with the parameterization of $A585[0] = 1048576$ inc for the feed factor in DriveControlSuite.
15. Repeat the steps for each additional axis.
⇒ The axes are parameterized.

7.2.7.2 Programming axis control

You can program control of the axes in TwinCAT 3 using the MC_Power block.

In order to control the drive controller with controller-based operation, the following operating modes are available in parameter A541 Modes of operation:

- ▶ -1: Jog
- ▶ 6: Homing mode
- ▶ 7: Interpolated position mode or
- ▶ 8: Cyclic synchronous position mode
- ▶ 9: Cyclic synchronous velocity mode
- ▶ 10: Cyclic synchronous torque mode

The axes are controlled using control word A515. The device state machine has to receive certain commands for starting operation and the associated state transitions. These commands are the result of a bit combination in the control word. The order of the commands is specified by the device state machine in accordance with CiA 402.

Gravity-loaded axis with brake



Information

If you are using a gravity-loaded axis and a brake, always use a quick stop to switch the drive (state transition 11 in accordance with the device state machine). This prevents the load from dropping until the brake is fully engaged.

In the MC_Power block, start by deactivating the Enable_Positive and Enable_Negative bits and then the Enable bit on a delay in order to bring the drive to a standstill as defined.

For more detailed information on the operating modes, device control and standard mapping, please refer to the manual for the CiA 402 application.

7.2.8 Drive-based axis control

Drive-based axis control requires manual programming in the automation software. The following operating modes are available to you in the parameter A541 Modes of operation:

- ▶ -1: Jog
- ▶ 1: Profile position mode
- ▶ 2: Velocity mode
- ▶ 3: Profile velocity mode
- ▶ 4: Profile torque mode
- ▶ 6: Homing mode

The axes are controlled using control word A515. The device state machine has to receive certain commands for starting operation and the associated state transitions. These commands are the result of a bit combination in the control word; the order of commands is specified by the device state machine in accordance with CiA 402.

Gravity-loaded axis with brake



Information

If you are using a gravity-loaded axis and a brake, always use a quick stop to switch the drive (state transition 11 in accordance with the device state machine). This prevents the load from dropping until the brake is fully engaged.

For more detailed information on the operating modes, device control and standard mapping, please refer to the manual for the CiA 402 application.

7.2.9 Configuring EoE communication

1. Navigate to the EtherCAT master in the solution explorer.
2. Double-click the EtherCAT master.
⇒ The settings open in the main window.
3. In the main window, switch to the EtherCAT tab and click Advanced Settings...
⇒ The Advanced Settings window opens.
4. Select EoE Support from the tree view on the left.
5. Virtual Ethernet Switch > Enable:
This option must be activated.
6. Close the window.
7. Navigate to the first EtherCAT slave in the solution explorer.
8. In the main window, switch to the EtherCAT tab and click Advanced Settings...
⇒ The Advanced Settings window opens.
9. In the tree view on the left, navigate to Mailbox > EoE.
10. Virtual Ethernet Port:
This option must be activated.
11. IP Port:
Activate this option.
12. Define the type of address assignment:
 - 12.1. Activate the DHCP option if an IP address is to be assigned to the EtherCAT slave automatically via DHCP.
 - 12.2. Activate the IP Address option in order to assign a fixed IP address to the EtherCAT slave in accordance with the subnet of your EoE device group. When assigning a fixed IP address for EoE, note that the first and last host address in a subnet must not be used. If one of these addresses is configured in TwinCAT 3, it is not accepted by the drive controller.
13. Default gateway:
When assigning a fixed IP address, the IP address of the EtherCAT network interface of the EtherCAT master must be specified as the default gateway.
14. Close the window.
15. Repeat steps 7 – 14 for each additional slave in your EtherCAT system.
⇒ The EoE communication is enabled for the EtherCAT master and slaves.




Information

Depending on your EoE network structure, you may have to set the routing on your EtherCAT master PC manually to connect the Ethernet and EtherCAT networks (see [EoE: Application cases with Pilz devices \[68\]](#)).



Information

Address assignment via DHCP is possible via either a DHCP server or DriveControlSuite. This requires the DHCP server or DriveControlSuite to be installed directly on the controller PC (see [Topology 1: EtherCAT master and DS6 on one PC](#) [ 69]). Furthermore, the IP address reference must be correctly defined in the drive controller (A166 = 2: DHCP + DS6, default value).

7.2.10 Configuring a Station Alias

You have the option of assigning an EtherCAT Station Alias to each EtherCAT slave. This address is stored in the EEPROM of the respective drive controller. The drive controller can thus be connected to any open port within the network and identified using the Station Alias.

1. In the solution explorer, navigate to the EtherCAT slave to which you want to assign a Station Alias.
 2. Double-click the EtherCAT slave.
⇒ The settings open in the main window.
 3. In the main window, switch to the EtherCAT tab and click Advanced Settings....
⇒ The Advanced Settings window opens.
 4. In the tree view on the left of the Advanced Settings window, navigate to ESC Access > E2PROM > Configured Station:
New Value: Enter the value you want to write as the Station Alias into the EEPROM.
 5. Click Write to E2PROM to write the value to EEPROM.
 6. Confirm the Function succeeded! dialog box with OK.
 7. Close the Advanced Settings window with OK.
 8. Repeat the steps for each additional slave in your EtherCAT system to which you want to assign a Station Alias.
- ⇒ The configuration of the Station Alias is complete.
- ⇒ The change of addresses will take effect upon the next start of TwinCAT 3.



Information

In DriveControlSuite, the Station Alias can be read out via parameter A254.

7.2.11 Transmitting the configuration

Transfer the configuration to the EtherCAT master.

1. Select the menu **TWINCAT > Activate Configuration**.
2. Confirm the transfer of the project configuration to the EtherCAT master with **OK**.
⇒ The **Restart TwinCAT System in Run Mode** dialog box opens.
3. Confirm with **OK**.
⇒ The configuration was transmitted to the EtherCAT master.

7.2.12 Checking the functionality of the axes

Check the functionality of the axes before operation in production.



Information

Ensure that a suitable safety application that ensures safe shut-off of the axis (emergency off, safety switch, etc.) exists before the start of testing.



Information

In order to be able to check the function of the axes, the **A541 Modes of operation** parameter of the respective axis must be set to the value **8** (default value).

1. In the solution explorer, navigate to **Motion > NC-Task 1 SAF > Axes > Axis 1**.
2. Double-click the axis.
⇒ The settings open in the main window.
3. In the main window, switch to the **Online** tab.
4. In the **Enabling** area, click **Set**.
⇒ The **Set Enabling** window opens.
5. Enable the **Controller**, **Feed Fw** and **Feed Bw** options.
6. **Override**:
Specify a value for the override (e.g. 100).
7. Confirm with **OK**.
⇒ The axis is monitored via the active control panel.
8. **F1 – F4**:
Move the axis step-by-step and test the movement direction, velocity, etc. using the corresponding buttons.
9. To deactivate the enable signal, click on **Set Enabling** and deactivate the options **Controller**, **Feed Fw**, **Feed Bw**.
10. Repeat the steps for each additional axis of your system.

7.3 CODESYS V3: Putting the EtherCAT system into operation

CODESYS V3 automation software gives you the option to map the hardware environment of your EtherCAT system and to configure and parameterize all necessary bus parameters including data exchange via master and slaves.

Note that all system nodes have to be networked physically before commissioning. In addition, you have projected the drive controllers in question in advance, i.e. EtherCAT slaves in DriveControlSuite, and transmitted the configuration to those drive controllers.



Information

For the following description, we require that you have projected the CiA 402 HiRes Motion application.



Information

Always perform the steps described below in the specified order!

Some parameters are interdependent and do not become accessible to you until you have first configured certain settings. Follow the steps in the specified sequence so that you can finish the parameterization completely.

7.3.1 Creating a standard project

1. Start the CODESYS V3 automation software.
2. Select Basic Operations > New Project.
⇒ The New Project window opens.
3. Select a standard project that corresponds to your hardware version. Give it a name and save it wherever you want.

7.3.2 Adding a drive controller

1. In the device tree, navigate to the module EtherCAT_Master (EtherCAT Master) > Context menu Add Device.
⇒ The Add Device window opens.
2. Device area > Vendor:
Select STOBBER Antriebstechnik GmbH + Co. KG – Drives and open the folder with the same name.
⇒ All drive controllers that can be mapped are displayed.
3. Highlight the desired drive controller in the SoftMotion_HiRes version and confirm with Add Device.
4. Repeat step 3 for all other drive controllers in your EtherCAT system.
⇒ The selected drive controllers are added in the device tree under the EtherCAT_Master (EtherCAT Master) controller.

7.3.3 Configuring synchronization using distributed clocks

- ✓ As the more precise of the two sync methods, synchronization using distributed clocks (DC-Sync) is pre-configured in the EtherCAT slaves.
In order to reduce jitter in general, we recommend setting data transfer (I/O) of the controller to task start in the EtherCAT configuration.
- 1. In the device tree, navigate to the module EtherCAT_Master (EtherCAT Master) and double-click to open it.
 - ⇒ EtherCAT_Master tab > General opens in the editor window.
- 2. Distributed Clock area > Cycle Time and Sync Offset:
Check the default values and change them if necessary.
- 3. To set the data transfer to task start, select Tools > Options > Device editor.
- 4. Activate the Show generic configuration editors option and confirm with OK.
- 5. Switch to the EtherCAT parameters vertical tab.
- 6. Navigate to the FrameAtTaskStart parameter and set the Value of the parameter to True.
 - ⇒ From now on, controller data transfer will take place at the start of the task.
- 7. In the device tree, double-click the first of the added drive controllers.
 - ⇒ SI6_SC6_Single(Double)Ax_SoftMotion_HiRes tab > General opens in the editor window.
- 8. Distributed Clocks area > Select DC:
DC enabled (multiplier = 1) and Sync 0 as a sync event are enabled by default.
- 9. If you want to change the presets, enable the Additional > Enable Expert Settings option and change the settings.
- 10. Repeat steps 7 – 9 for each additional drive controller in your EtherCAT network.
 - ⇒ The EtherCAT master and slaves will now be synchronized with the first EtherCAT slave that has the distributed clocks option enabled.

7.3.4 Controller-based axis control

In order to control one or more drive controllers with controller-based operation, start by parameterizing the axes and then program your control.

7.3.4.1 Parameterizing a SoftMotion axis

- ✓ You have selected the CiA 402 HiRes Motion application and fully configured the associated axis model in DriveControlSuite.
- 1. In the device tree, navigate to the first SoftMotion axis
SM_Drive_ETC_STOEBER_SI6_SC6_HiRes of the first added PMC SC6 or PMC SI6 drive controller and double click to open it.
 - ⇒ SM_Drive_ETC_STOEBER_SI6_SC6_HiRes tab > General opens in the editor window.
- 2. Axis type and limits area > Modulo/Finite:
Activate your drive according to the listed options and parameterize the conditions necessary in each case:
 - 2.1. Conditions for Modulo > Modulo settings: Define the modulo range by entering the associated modulo value.
 - 2.2. Conditions for Finite > Software limit switches: If you want to put a lower negative limit or upper positive limit on position values, enable the option and enter the associated values.
- 3. Software error reaction:
Delay: If braking is to be done on a delay, enter the associated value.
Maximum distance: Parameterize a maximum distance within which the drive must have reached a stop after an error has occurred.
Set value monitoring of the drive controller is activated by default in the CiA 402 and CiA 402 HiRes Motion applications. In order to prevent the drive controller from transitioning into the **excessive set value jump** state, parameterize ramp that can realistically be implemented.
- 4. Dynamic limits (optional):
If you are using CNC or robotic functions, parameterize the associated limit values for velocity, acceleration, deceleration and jerk.
- 5. Velocity ramp type (optional):
Using the velocity ramp type, define the velocity profile for movement-generating single axis modules and for master/slave modules. Select the appropriate profile.
- 6. Position lag supervision (optional):
Use the associated picklist to define the response of the controller when a following error is detected.
Lag limit: A following error is detected if the difference between the set position and actual position exceeds the lag limit. If you have enabled position lag supervision by selecting a response, specify the associated value.
- 7. Switch to the Scaling/Mapping vertical tab.
- 8. Scaling area > Precision (decimal places):
Specify the parameterized number of decimal places in DriveControlSuite (106 Decimal places position) for specifying and displaying positions, velocity values and acceleration values.
- 9. Repeat the steps for each additional SoftMotion axis in your EtherCAT system.
 - ⇒ The SoftMotion axes are parameterized.

7.3.4.2 Programming axis control

You program the control of the axes in the automation software. See the CODESYS V3 documentation for the required information.

In order to control the drive controller with controller-based operation, the following operating modes are available in parameter A541 Modes of operation:

- ▶ -1: Jog
- ▶ 6: Homing mode
- ▶ 7: Interpolated position mode or
- ▶ 8: Cyclic synchronous position mode
- ▶ 9: Cyclic synchronous velocity mode
- ▶ 10: Cyclic synchronous torque mode

The axes are controlled using control word A515. The device state machine has to receive certain commands for starting operation and the associated state transitions. These commands are the result of a bit combination in the control word. The order of the commands is specified by the device state machine in accordance with CiA 402.

Gravity-loaded axis with brake



Information

If you are using a gravity-loaded axis and a brake, always use a quick stop to switch the drive (state transition 11 in accordance with the device state machine). This prevents the load from dropping until the brake is fully engaged.

For information on how to switch off the drive via a quick stop, refer to the CODESYS V3 documentation.

For more detailed information on the operating modes, device control and standard mapping, please refer to the manual for the CiA 402 application.

7.3.5 Drive-based axis control

Drive-based axis control requires manual programming in the automation software. The following operating modes are available to you in the parameter A541 Modes of operation:

- ▶ -1: Jog
- ▶ 1: Profile position mode
- ▶ 2: Velocity mode
- ▶ 3: Profile velocity mode
- ▶ 4: Profile torque mode
- ▶ 6: Homing mode

The axes are controlled using control word A515. The device state machine has to receive certain commands for starting operation and the associated state transitions. These commands are the result of a bit combination in the control word; the order of commands is specified by the device state machine in accordance with CiA 402.

Gravity-loaded axis with brake



Information

If you are using a gravity-loaded axis and a brake, always use a quick stop to switch the drive (state transition 11 in accordance with the device state machine). This prevents the load from dropping until the brake is fully engaged.

For more detailed information on the operating modes, device control and standard mapping, please refer to the manual for the CiA 402 application.

7.3.6 Configuring EoE communication

Pilz 6th generation drive controllers support EoE communication. For information on whether your controller also supports EoE and how the packets are transferred from your controller to the service PC, refer to the documentation for your controller.



Information

Depending on your EoE network structure, you may have to set the routing on your EtherCAT master PC manually to connect the Ethernet and EtherCAT networks (see [EoE: Application cases with Pilz devices \[68\]](#)).

7.3.7 Transmitting the configuration

Transmit the project to your CODESYS SoftMotion controller and start CODESYS V3.

7.3.8 Checking the functionality of the axes

Check the functionality of the axes before operation in production.



Information

Ensure that a suitable safety application that ensures safe shut-off of the axis (emergency off, safety switch, etc.) exists before the start of testing.

7.3.9 Special case: Adding to the PDO transmission

- ✓ Are you working with a controller-based operating mode (SoftMotion) and need expanded PDO transmission?
Proceed as described in the following steps. Note that you can transmit a maximum of 24 CiA objects or drive controller parameters per channel.
- 1. In the device tree, navigate to the drive controller whose PDO transmission you would like to expand and double-click to open it.
⇒ SI6_SC6_Single(Double)Ax_SoftMotion_HiRes tab > General opens in the editor window.
- 2. Additional area > Enable Expert Settings:
Activate this option.
- 3. Switch to the Expert Process Data Mode vertical tab.
- 4. PDO list:
The list contains one transmit and one receive channel for each parameterized SoftMotion axis. Highlight the channel whose PDO transmission you would like to expand.
⇒ PDO Content: This area shows all PDOs that are exchanged between the controller and drive controller over the selected channel.
- 5. Click on Insert.
⇒ The Select Item from Object Directory dialog box opens. The directory contains a selection of available CiA objects (along with the coordinates and the name of the corresponding drive controller parameter from Pilz).
- 6. Highlight the CiA object for which you would like to extend PDO transmission and confirm with OK.
If the desired CiA object is not included in the directory, enter its index and subindex in the corresponding fields. For manufacturer-specific parameters, calculate the two indices from the parameter coordinate in advance (see [Manufacturer-specific parameters: 2000 hex – 53FF hex \[114\]](#) and [Manufacturer-specific parameters: A000 hex – D3FF hex \[116\]](#)). Also select the data type that matches the data type of the drive controller parameter and confirm with OK.
⇒ The selected CiA object or the specified drive controller parameter has been added to the PDO content of the selected channel.
- 7. Repeat steps 5 – 6 for all other CiA objects for which you would like to extend PDO transmission for the selected channel.
- 8. Switch to the associated DriveControlSuite project and add to the PDO transmission there in the same way as the additions in CODESYS V3 (see [Configuring PDO transmission \[29\]](#)).
⇒ The expansion of PDO transmission takes effect the next time the EtherCAT master is started.

8 Monitoring and diagnostics

For monitoring purposes and in the event of a fault, the various monitoring and diagnostic options described below are available.

8.1 Connection monitoring

In order to be able to detect a communication failure, activate the watchdog function. This means that you monitor the arrival of cyclical process data by defining a PDO timeout in A258 (see [Parameterizing general EtherCAT settings \[📖 28\]](#)).

In the Operational operating state, an activated watchdog triggers fault 52: Communication with the cause 6: EtherCAT PDO-Timeout – if a new PDO is not received within the specified timeout.

Monitoring is not triggered if the EtherCAT master ends communication as intended by leaving the Operational state.

8.2 LED display

The drive controllers feature diagnostic LEDs that visualize the state of fieldbus communication and the states of the physical connection.

8.2.1 EtherCAT state

There are 2 LEDs on the front of the drive controller that provide information about the connection between EtherCAT master and slave and about the state of the data exchange. This information can also be read out in parameter A255. If the drive controller includes the PMC SY6 safety module, the STO and SS1 safety functions are activated via EtherCAT FSoE. In this case, an additional LED on the front of the device provides information about the FSoE state.



Fig. 5: LEDs for the EtherCAT state

- 1 Red: Error
- 2 Green: Run

Red LED	Conduct	Error	Description
	Off	No Error	No error
	Flashing	Invalid Configuration	Invalid configuration
	Single flash	Unsolicited State Change	The EtherCAT slave changed operating states by itself
	2x flash	Application Watchdog Timeout	The EtherCAT slave did not receive new PDO data during the configured watchdog timeout

Tab. 5: Meaning of the red LED (error)

Green LED	Conduct	Operating state	Description
	Off	Init	No communication between the EtherCAT master and slave; the configuration starts, saved values are loaded
	Flashing	Pre-operational	No PDO communication; the EtherCAT master and slave exchange application-specific parameters via SDOs
	1x flash	Safe-operational	The EtherCAT slave sends the current actual values to the EtherCAT master, ignores its set values and refers to internal default values
	On	Operational	Normal operation: The EtherCAT master and slave exchange set and actual values

Tab. 6: Meaning of the green LED (Run)

8.2.2 EtherCAT network connection

The LEDs LA_{EC}IN and LA_{EC}OUT at X200 and X201 on the top of the device indicate the state of the EtherCAT network connection.

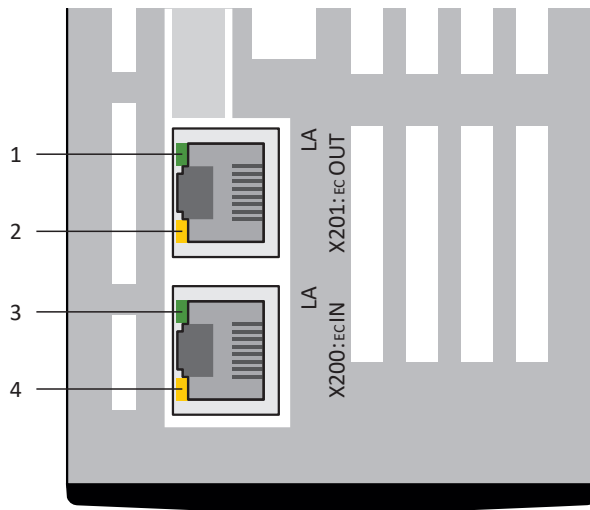


Fig. 6: LEDs for the state of the EtherCAT network connection

- 1 Green: LA_{EC}OUT at X201
- 2 Yellow: No function
- 3 Green: LA_{EC}IN at X200
- 4 Yellow: No function

Green LED	Behavior	Description
	Off	No network connection
	Flashing	Active data exchange with other EtherCAT nodes
	On	Network connection exists

Tab. 7: Meaning of the green LEDs (LA)

8.3 Events

The drive controller has a self-monitoring system that uses test rules to protect the drive system from damage. Violating the test rules triggers a corresponding event. There is no possible way for you as the user to intervene in some events, such as the Short/ground event. In others, you can influence the effects and responses.

Possible effects include:

- ▶ **Message:** Information that can be evaluated by the controller
- ▶ **Warning:** Information that can be evaluated by the controller and becomes a fault after a defined time span has elapsed without the cause being resolved
- ▶ **Fault:** Immediate drive controller response; the power unit is disabled and axis movement is no longer controlled by the drive controller or the axis is brought to a standstill by a quick stop or emergency braking

Depending on the event, there are various measures you can take to rectify the cause. As soon as the cause has been successfully rectified, you can usually acknowledge the event immediately. If the drive controller has to be restarted, a corresponding note can be found in the measures.



ATTENTION!

Damage to property due to interruption of a quick stop or emergency braking!

If, when executing a quick stop or emergency braking, a fault occurs or STO is active, the quick stop or emergency braking is interrupted. In this case, the machine can be damaged by the uncontrolled axis movement.



Information

To make it easier for control system programmers to set up the human-machine interface (HMI), you can contact Pilz Support at support@pilz.com for a list of events and their causes.

8.3.1 Event 52: Communication

The drive controller has a **fault** if:

- ▶ A29 = 0: Inactive for Drive Based device controller
or
- ▶ A540 = 0: Disable drive motor coasting for CiA 402 device controller

Response:

- ▶ The power unit is disabled and axis movement is no longer controlled by the drive controller
- ▶ The brakes engage

The drive controller has a **fault with a quick stop** if:

- ▶ A29 = 1: Active for Drive Based device controller
or
- ▶ A540 = 2: Slow down on quick stop ramp for CiA 402 device controller

Response:

- ▶ The axis is stopped by a quick stop
- ▶ During the quick stop, the brakes remain released
- ▶ At the end of the quick stop, the power unit is disabled and axis movement is no longer controlled by the drive controller
- ▶ The brakes engage



Information

An edge change for the release override signal (source: F06) is expected in the Switch on disabled, Ready to switch on, and Switched on states (E48) so that the brake is released.

Cause		Check and action
6: EtherCAT PDO-Timeout	Missing process data	Check the task cycle time in the EtherCAT master and the timeout time in the drive controller and correct them if necessary (A258)
7: Reserved	Synchronization error	Check the synchronization settings in the EtherCAT master and correct them if necessary
	Connection error	Check the connection and shielding and correct them if necessary
15: Wrong firmware for applicataion	Projected fieldbus identification and that of the drive controller do not match	Check the projected fieldbus identification and the fieldbus identification of the drive controller and change the fieldbus if necessary (E59[2], E52[3])

Tab. 8: Event 52 – Causes and actions

8.4 Parameters

The following diagnostic parameters are available for EtherCAT communication in combination with the drive controller.

8.4.1 A254 | EtherCAT Station Alias | G6 | V0

Alias address of the drive controller (EtherCAT slave) in the EtherCAT network (source: EtherCAT master). This address is stored in the EEPROM of the drive controller and is read out from there when the EtherCAT operating state switches from Init to Pre-Operational.

8.4.2 A255 | EtherCAT Device State | G6 | V2

State of the drive controller in the EtherCAT network (EtherCAT State Machine, ESM).

- ▶ 0: Invalid
- ▶ 1: Init State
No communication between EtherCAT master and EtherCAT slave; the configuration starts, saved values are loaded
- ▶ 2: Pre operational state
No PDO communication; the EtherCAT master and EtherCAT slave exchange application-specific parameters via SDOs
- ▶ 4: Safe operational
The EtherCAT slave sends the current actual values to the EtherCAT master, ignores its set values and refers to internal default values
- ▶ 8: Operational State
Normal operation: The EtherCAT master and EtherCAT slave exchange set and actual values
- ▶ 17: Error - Init State – 21: Error - Operational State (details: A257)

8.4.3 A256 | EtherCAT address | G6 | V1

Address of the drive controller (EtherCAT slave) in the EtherCAT network (source: EtherCAT master).

8.4.4 A257 | EtherCAT diagnosis | G6 | V1

Diagnostic information of the drive controller in the EtherCAT network.

- ▶ [0]: EtherCAT operating state
Format: StX ErX L0X L1X
- ▶ [1]: EtherCAT network connection – error counter
Format: L0 xx L1 xx
- ▶ [2]: Data error – error counter
Format: R0 xxxx R1 xxxx

EtherCAT operating state

- ▶ StX
 - St1 = Init
 - St2 = Pre-Operational
 - St4 = Safe-Operational
 - St8 = Operational

▶ ErX

- Er0 = No Error
- Er1 = Booting Error
EC6 error
- Er2 = General Configuration Error
General configuration error of the data transfer memory
- Er3 = Unsolicited State Change
Drive controller changes state without a request from the master
- Er4 = Watchdog
Timeout A258 expired without receiving process data
- Er6 = Regular process data missing
Change condition from St4 to St8 not met: Stable, regular receipt of PDO data not possible for a duration of more than 200 ms; utilization of the controller is too high (jitter)
- Er7 = Invalid Configuration TxPDO
Data length of the transmit PDO channel does not match the specification
- Er8 = Invalid Configuration RxPDO
Data length of the receive PDO channel does not match the specification
- Er9 = Invalid Configuration Mailbox Tx
Data length of the transmit SDO channel does not match the specification or the EoE configuration is faulty
- Er10 = Invalid Configuration Mailbox Rx
Data length of the receive SDO channel does not match the specification or the EoE configuration is faulty

▶ L0X

- L00 = No Link
No connection to another EtherCAT device via X200 (IN port)
- L01 = Link Detected
Connection to another EtherCAT device via X200 (IN port)

▶ L1X

- L10 = No Link
No connection to another EtherCAT device via X201 (OUT port)
- L11 = Link Detected
Connection to another EtherCAT device via X201 (OUT port)

EtherCAT network connection – Error counter

- ▶ L0 xx = Link Lost Counter
xx = Number of connection failures (hexadecimal) at X200 (IN port)
- ▶ L1 xx = Link Lost Counter
xx = Number of connection failures (hexadecimal) at X201 (OUT port)

Data error – Error counter

- ▶ R0 xxxx = RxPDO Error Counter
xxxx = Number of data errors (hexadecimal) at X200 (IN port)
- ▶ R1 xxxx = RxPDO Link Lost Counter
xxxx = Number of data errors (hexadecimal) at X201 (OUT port)

8.4.5 A259 | EtherCAT SM-Watchdog | G6 | V1

State of the SyncManager watchdog of the drive controller in the EtherCAT network (prerequisite: A258 = 65534).

- ▶ [0]: Tolerated failure time (unit: ms)
Specified by the SyncManager watchdog function of the EtherCAT master
- ▶ [1]: State
0 = not triggered; 1 = triggered = Event 52: Communication, cause 6: EtherCAT PDO-Timeout
- ▶ [2]: Number of times triggered

8.4.6 A261 | Sync-Diagnostics | G6 | V1

Diagnostics of the synchronization of the drive controller in the EtherCAT network.

- ▶ [0]: Error code
 - 0 = No error
 - 1 = SyncManager 2 and 3 have different cycle times
 - 2 = Cycle time < 250 μ s
 - 3 = Odd multiple of 250 μ s
 - 4 = PLL could not be started
 - 6 = Drive controller interrupt not initialized, firmware error
- ▶ [1]: Time difference between the data provision and the Sync 0 signal
- ▶ [2]: Error counter

8.4.7 A287 | DC-Sync optimization | G6 | V3

Evaluate DC-Sync and determine a recommended minimum and maximum value for the phase offset (use: Optimize DC-Sync wizard; prerequisite: EtherCAT network with synchronization via distributed clocks; measurement duration: $1000 \times A150$). Cycle time of the drive controller (A150) and cycle time of the controller (A291) should match, since the accuracy of the measurement result decreases as the difference increases.

8.4.7.1 A287[2] | Result | G6 | V2

Result of the action.

Action successful

- ▶ 7: faultless A292 in the recommended range
Synchronization of the drive controller runs without errors; A292 is in the recommended range
- ▶ 8: Can be optimized by adapting A292
Synchronization of the drive controller runs without errors; RxPDO or TxPDO time cuts the frame or TxPDO is in the next frame, A292 can be optimized

Action not successful

- ▶ 2: Action not executable
State of the drive controller in the EtherCAT network at the start of the measurement is not 4: Safe operational or 8: Operational State (A255), the actual cycle time of the controller and the controller cycle time set in the drive controller are different (A291) or distributed clocks is not used as the synchronization type
- ▶ 3: Cycle time too small, ECMaster frame jitter too large
A150 Cycle time is too small or the jitter of the EtherCAT master frame start is too large
- ▶ 4: Cycle time too small, because jitter of drive controller too large
A150 Cycle time is too small or jitter of the RxPDO or TxPDO time of the drive controller is too large
- ▶ 5: Cycle time too small, runtime of app too great
A150 Cycle time is too small or the duration of the drive controller application is too large
- ▶ 6: Cycle time too small, frame runtime too large
A150 Cycle time is too small or the duration of the frame is too large (too many nodes in the EtherCAT network or PDO data quantity is too large)

9 Looking for more information about EtherCAT?

The following chapters summarize the key terms, services and relationships relating to EtherCAT.

9.1 EtherCAT

EtherCAT (Ethernet for Control Automation Technology) is an industrial Ethernet technology for real-time requirements in automation technology. EtherCAT is focused on short cycle times, low jitter and precise synchronization.

EtherCAT was invented by Beckhoff Automation GmbH & Co. KG and is currently supported by the international EtherCAT Technology Group (ETG) organization. EtherCAT is an open technology standardized in the standard IEC 61158 since 2005.

Master/slave principle and the exchange of data

EtherCAT follows a master/slave principle.

A master sends standard Ethernet frames that pass every slave. The frames are processed as they pass through. More specifically, each EtherCAT slave has an EtherCAT slave controller (ESC) integrated into its hardware which takes the receive data addressed to the respective slave as the frame passes by and attaches the slave's transmit data on the fly. This means any delays are due to the hardware processing time. The last slave in the network sends the frame back to the master.

The EtherCAT master is the only network node that actively sends frames; the EtherCAT slaves simply pass the frame on. This principle avoids potential delays and ensures real-time capability. The order of the data does not depend on the physical order of the slaves in the network.

9.2 Communication protocols

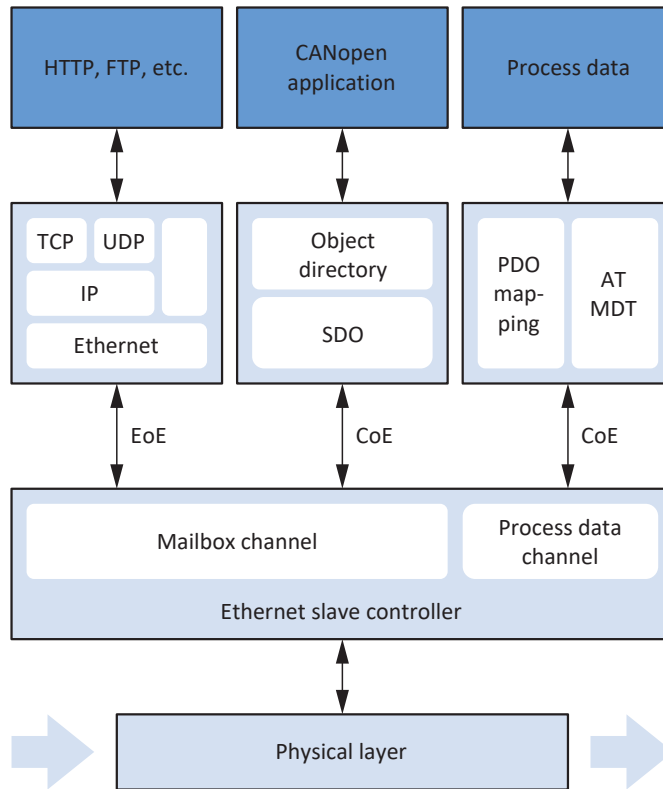


Fig. 7: EtherCAT: Communication protocols


EtherCAT uses standard Ethernet frames containing EtherCAT payloads. Communication normally takes place over a mailbox or process data channel.

Only data that is not time-critical, i.e. service data objects (SDO), are exchanged using the mailbox channel; time-critical process data objects (PDO) are, like in CANopen, transferred using the process data channel.

Pilz drive controllers of the 6th generation support the CoE and EoE EtherCAT protocols.

9.2.1 CoE: CANopen over EtherCAT

EtherCAT, together with the CoE protocol, provides CANopen-compliant communication mechanisms, enabling the use of the entire CANopen profile family over EtherCAT, thereby also allowing full use of the CiA 402 drive profile.

In terms of the respective state machines, CANopen and EtherCAT differ only in that the EtherCAT state machine (see [EtherCAT state machine](#) [ 82]) also has the Safe-Operational state.

9.2.2 EoE: Ethernet over EtherCAT

Using EoE, it is possible to transport any Ethernet data traffic between EoE-capable nodes in an EtherCAT network.

In this process, Ethernet frames are tunneled through the EtherCAT protocol, as is typical for Internet protocols. The EtherCAT master is used as a gateway to the Ethernet network.

EoE is an acyclical protocol, meaning that the EtherCAT real-time properties (process data communication) remain unaffected.

Acyclical frames can be exchanged starting in the Pre-Operational state of the EtherCAT state machine.

The IP address, subnet mask and gateway of the EoE-capable slaves are stored in the EtherCAT master.

9.2.3 EoE: Application cases with Pilz devices

Pilz uses EoE to connect DriveControlSuite to Pilz drive controllers of the 6th generation in combination with an EtherCAT master. A distinction is made between two topologies here:

- ▶ Topology 1
The EtherCAT master and DriveControlSuite are operated on one PC; only the EtherCAT network is used
- ▶ Topology 2
The EtherCAT master and DriveControlSuite are operated on different PCs; transmission takes place between the EtherCAT network and Ethernet

9.2.3.1 Topology 1: EtherCAT master and DS6 on one PC

If the EtherCAT master and DriveControlSuite are installed on one PC, the Ethernet subnet where the drive controllers are operated is automatically known to DriveControlSuite through the master's gateway function.

DriveControlSuite detects the drive controllers; no additional manual configurations are needed.

The following graphic shows the associated network overview together with pre-assigned network addresses on the system end.

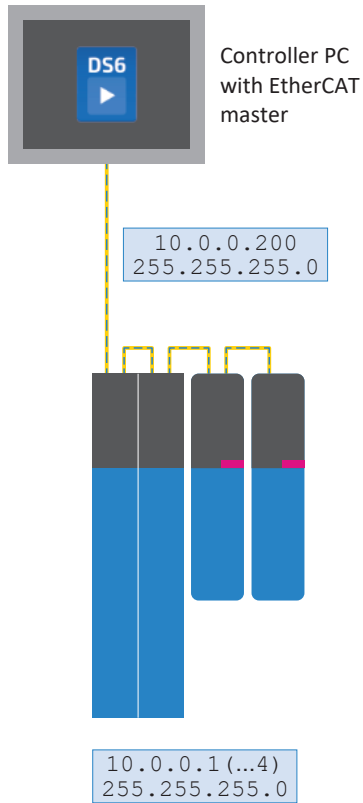


Fig. 8: Network overview: Topology 1

9.2.3.2 Topology 2: EtherCAT master and DS6 on different PCs

If the EtherCAT master and DriveControlSuite are installed on different PCs, the drive controllers are in an Ethernet subnet that is initially unknown to DriveControlSuite. In this case, the address of the master must be manually configured as the gateway for the route, i.e. adding the route to the service PC.



Information

The broadcast-based drive controller search does not work due to the routing, so you must establish the direct connection in DriveControlSuite using the **Direct connection (manual)** tab.

The following graphic shows the associated network overview together with pre-assigned network addresses on the system end.

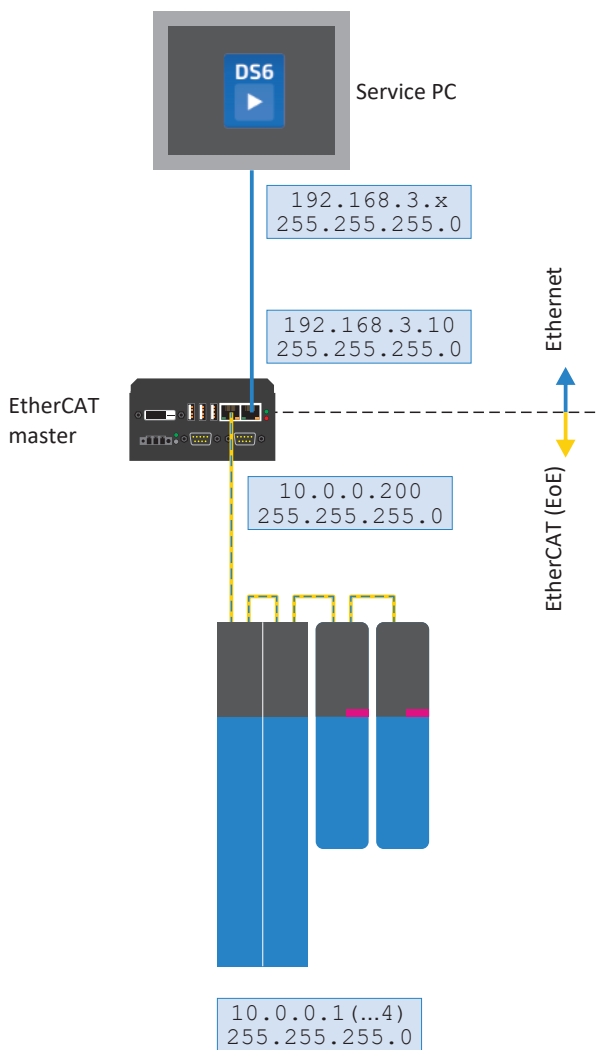


Fig. 9: Network overview: Topology 2

EtherCAT service PC: Setting the route of an Ethernet subnet

In order to make the Ethernet subnet of the drive controllers known to DriveControlSuite, you have to configure a corresponding route on the service PC. The route allows an IP configuration packet to be forwarded to the drive controllers in question via the EtherCat master, which acts as a gateway. Note that the operating system of the EtherCAT master only connects the subnets known to it if IP routing is permitted there.

- ✓ The following information (network of the drive controllers to be triggered, subnet mask, gateway address of the master) is adapted to the Pilz presets and must be replaced by the addresses that correspond to the system environment.
1. To set the Ethernet route using the command line, open the Windows console cmd.exe.
 2. Enter the following command:

```
route add 10.0.0.0 mask 255.255.255.0 192.168.3.10
```
- ⇒ You have now successfully set the route.

9.3 Communication objects

Based on CANopen, the following communication objects are of key importance for data transmission in an EtherCAT network:

- ▶ Process Data Objects (PDO)
 - ... for the transmission of real-time data of the nodes (actual and set values)
- ▶ Service Data Objects (SDOs)
 - ... for access to the object directory of the nodes for device configuration
- ▶ Emergency Objects (EMCY)
 - ... for monitoring the device states of the nodes



Information

It is not possible to write to or read the parameter hidden in DriveControlSuite during communication via fieldbus.


9.3.1 Process data objects – PDO

Process Data Objects are peer-to-peer objects that are used to transmit time-critical real-time data of the nodes, such as set and actual values or control and status information such as set positions, travel velocities, or acceleration specifications.

PDOs allow simultaneous access to several communication parameters defined via the object directory of the respective node. Objects are not addressed with PDO transmission. Instead, the values of the communication parameters are transmitted directly to the respective node.

The process data mapping (PDO mapping) defines which communication parameters are sent and received. With process data mapping, which communication parameters are sent or received in which PDO can be freely selected.

PDOs are generally transmitted via process data channels (PDO channels) with high priority. From the perspective of the respective node, receive PDOs (RxPDO) are differentiated from transmit PDOs (TxPDO).

For information on scaling, see [Fieldbus scaling](#) [ 104].

9.3.1.1 PDO mapping

The process data mapping (PDO mapping) defines which communication parameters are sent and received. The communication parameters from the object directory of a node are mapped to the respective PDO channels for this purpose.

PDO communication enables simultaneous operation of up to 4 independent PDO channels per transmission direction (RxPDO, TxPDO), each of which can transmit 1 PDO with up to 24 communication parameters. Channel 4 is reserved for safety communication, while the remaining PDO channels are freely configurable.

In order to guarantee error-free communication between the controller and drive controller, Pilz offers application-dependent pre-assignment of the process data channels which can be changed at any time.

9.3.2 Service data objects – SDO


Service Data Objects are peer-to-peer objects that are used to transfer non-time-critical data and provide access to entries in a node's object directory in order to configure its device properties.

From the perspective of the drive controller, an SDO transmission always consists of at least one RxSDO message and one TxSDO message. In the RxSDO message, the controller selects an entry from the object directory of the drive controller via index and subindex in order to configure the device properties. The drive controller then acknowledges access to the object directory with a TxSDO message.



SDO messages are transmitted over the mailbox channel acyclically during ongoing cyclical EtherCAT operation, without impairing PDO communication.

Depending on the transmission type, SDOs can generally be used for transmitting data of any length:

- ▶ Expedited transfer
 - ... for transmission of up to 4 bytes in a single message
- ▶ Segmented transfer
 - ... for transmission of more than 4 bytes distributed over several messages

For information on scaling, see [Fieldbus scaling](#) [ 104].

9.3.2.1 Addressing axis-dependent parameters

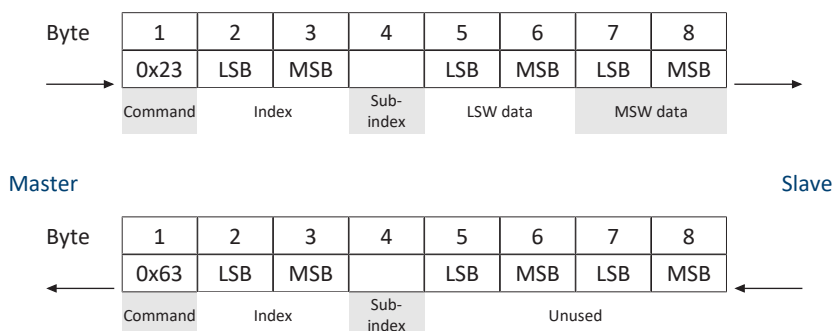
When addressing axis-specific parameters of physical axes using SDOs, the parameters are addressed directly in accordance with the access rules described in the appendix (see [Manufacturer-specific parameters: 2000 hex – 53FF hex](#) [ 114] and [Manufacturer-specific parameters: A000 hex – D3FF hex](#) [ 116]).

9.3.2.2 Expedited transfer

SDO transmission via expedited transfer enables up to 4 bytes of data to be transmitted in a single message. The data is arranged in accordance with the Intel format (little-endian), meaning that the byte with the smallest value is saved at the starting address and transmitted first (compare with big-endian or Motorola format, where the highest-value component is sent first).

Writing parameters (Initiate Domain Download Request)

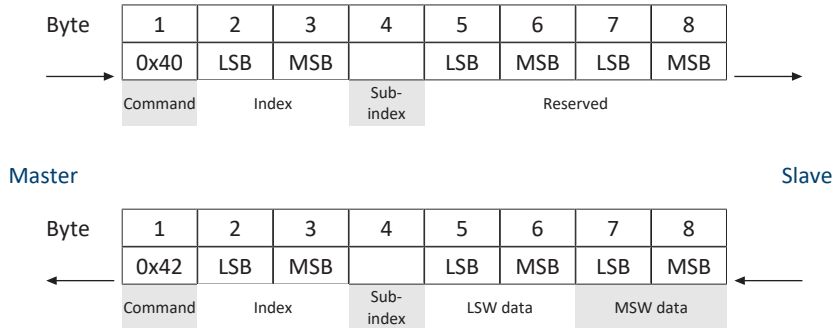
The controller (master) uses an Initiate Domain Download Request to initiate a write process for a communication parameter. The request receives a positive acknowledgement from an Initiate Domain Download Response of the drive controller (slave).



Intern

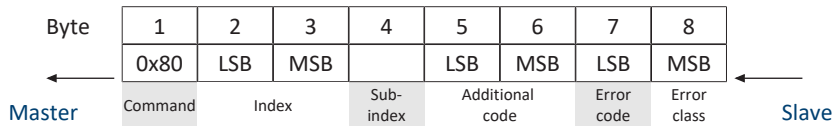
Reading parameters (Initiate Domain Upload Request)

The controller (master) uses an Initiate Domain Upload Request to initiate a read process for a communication parameter. The request receives a positive acknowledgement from an Initiate Domain Upload Response of the drive controller (slave).



Error message (Abort Domain Transfer)

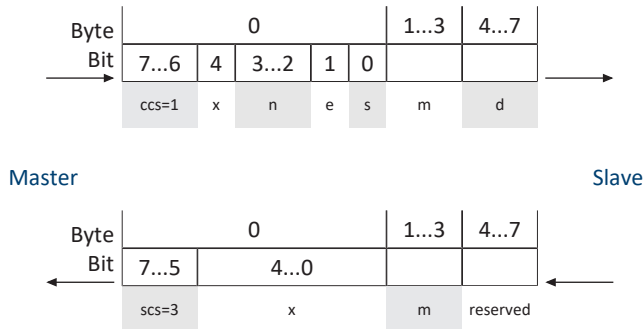
A drive controller (slave) provides a negative response to the write parameter or read parameter requests using an Abort Domain Transfer (see [SDO transmission: Error codes \[118\]](#)).



9.3.2.3 Segmented transfer

With SDO transmission via segmented transfer, more than 4 bytes of data can be transmitted distributed over several messages. The total number of bytes to be transmitted are sent in a first initiate message (Initiate SDO Download); this is followed by the segments (Download SDO Segment), each with 1 byte of control and protocol information and up to 7 bytes of payload.

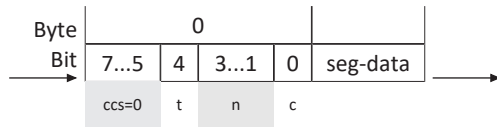
Initiate SDO Download Protocol



ccs	Client command specifier	1 = Initiate download request
scs	Server command specifier	3 = Initiate download response
n	Number of bytes	Number of bytes in "Data" that contain no usable data. If e = 0 , s = 1, then n = valid, otherwise n = 0
e	Transfer type	<ul style="list-style-type: none"> ▶ 0 = Normal transfer ▶ 1 = Expedited transfer
s	Size indicator	<ul style="list-style-type: none"> ▶ 0 = Not displayed ▶ 1 = Displayed
m	Multiplexor	= Index + subindex
d	Data	<ul style="list-style-type: none"> ▶ If e = 0, s = 0, then d = reserved ▶ If e = 0, s = 1, then d = number of bytes to be transmitted ▶ If e = 1, s = 1, then d = 4-n
x	Unused	x = 0

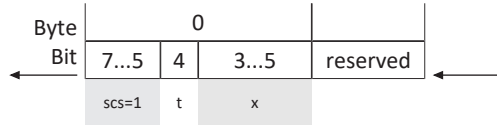
Intern

Download SDO Segment Protocol



Master

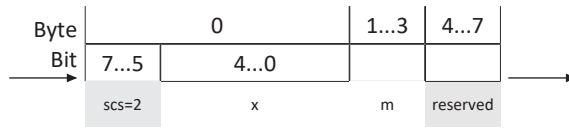
Slave



ccs	Client command specifier	0 = Download segment request
scs	Server command specifier	1 = Download segment response
n	Number of bytes	Number of bytes in "Segment data" that contain no usable data. n = 0: No information about unused data
seg-data	Segment data	7 bytes of usable data
c	Continue	<ul style="list-style-type: none"> ▶ 0 = More segments follow ▶ 1 = Last segment
t	Toggle bit	t = 0 for segment 1; must change for each segment. Identical values for request and response.
x	Unused	x = 0

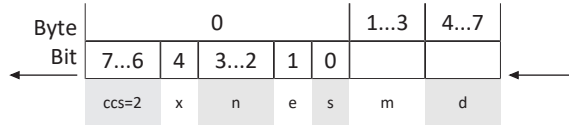
Intern

Initiate SDO Upload Protocol



Master

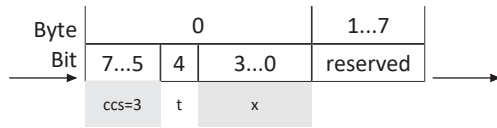
Slave



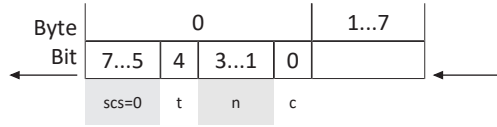
ccs	Client command specifier	2 = Initiate upload request
scs	Server command specifier	2 = Initiate upload response
n	Number of bytes	Number of bytes in "Data" that contain no usable data. If e = 0 , s = 1, then n = valid, otherwise n = 0
e	Transfer type	<ul style="list-style-type: none"> ▶ 0 = Normal transfer ▶ 1 = Expedited transfer
s	Size indicator	<ul style="list-style-type: none"> ▶ 0 = Not displayed ▶ 1 = Displayed
m	Multiplexor	= Index + subindex
d	Data	<ul style="list-style-type: none"> ▶ If e = 0, s = 0, then d = reserved ▶ If e = 0, s = 1, then d = number of bytes to be transmitted ▶ If e = 1, s = 1, then d = 4-n
x	Unused	x = 0

Intern

Upload SDO Segment Protocol



Master Slave



ccs	Client command specifier	3 = Upload segment request
scs	Server command specifier	0 = Upload segment response
n	Number of bytes	Number of bytes in "Segment data" that contain no usable data. n = 0: No information about unused data
seg-data	Segment data	7 bytes of usable data
c	Continue	<ul style="list-style-type: none"> ▶ 0 = More segments follow ▶ 1 = Last segment
t	Toggle bit	t = 0 for segment 1; must change for each segment. Identical values for request and response.
x	Unused	x = 0

Intern

Examples

Segment download with 16 bytes of data; contents: 01, 02, 03 ... 10 hex

Client: IDReq:	21	idx	x	10 00 00 00	(ccs = 1, e = 0 = normal, s = 1 -> data = no. of bytes)
Server: IDRes:	60	idx	x	00 00 00 00	
Client: DSegReq:	00	01 02 03 04 05 06 07			(ccs = 0, t = 0, n = 0, c = 0 -> all data bytes are used)
Server: DSegRes:	20	00 00 00 00 00 00 00			
Client: DSegReq:	10	08 09 0A 0B 0C 0D 0E			(ccs = 0, t = 1, n = 0, c = 0 -> all data bytes are used)
Server: DSegRes:	30	00 00 00 00 00 00 00			
Client: DSegReq:	0b	0F 10 00 00 00 00 00			(ccs = 0, t = 0, n = 5, c = 1 -> 5 data bytes are unused)
Server: DSegRes:	20	00 00 00 00 00 00 00			

Segment upload with 16 bytes of data; contents: 01, 02, 03 ... 10 hex

Client: IDUReq:	40	idx	x	00 00 00 00	(ccs = 2, rest = 0)
Server: IDURes:	41	idx	x	10 00 00 00	(scs = 2, x = 0, e = 0, s = 1 -> data contains no. of bytes to be uploaded)
Client: USegReq:	60	00 00 00 00 00 00 00			(ccs = 3, t = 0)
Server: USegRes:	00	01 02 03 04 05 06 07			(scs = 0, t = 0, n = 0, c = 0 -> all data bytes are used)
Client: USegReq:	70	00 00 00 00 00 00 00			(ccs = 3, t = 1)
Server: USegRes:	10	08 09 0A 0B 0C 0D 0E			(scs = 0, t = 1, n = 0, c = 0 -> all data bytes are used)
Client: USegReq:	60	00 00 00 00 00 00 00			(ccs = 3, t = 0)
Server: USegRes:	0b	0F 10 00 00 00 00 00			(scs = 0, t = 0, n = 5, c = 1 -> 5 data bytes are unused)

9.3.3 Emergency objects – EMCY

Emergency Objects are peer-to-peer objects that are used to monitor the device states of the nodes in the network and are triggered in the event of internal device errors or faults.

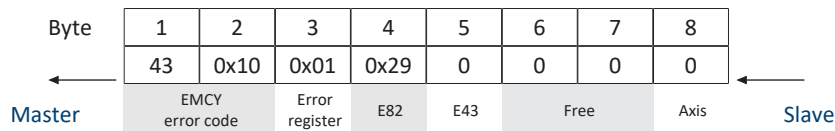
If the EMCY service is active and a drive controller changes to the Fault device state, it sends an EMCY message to the controller. The EMCY message contains an error code that uniquely identifies the fault. As soon as the fault has been corrected and the drive controller leaves the corresponding device state, it sends another EMCY message with error code 0 hex (NO ERROR).

This mechanism automatically notifies the controller of when a drive controller enters and leaves the fault state and of the associated cause for the fault.

Specifically, the drive controller sends EMCY messages in the event of incorrect parameterization of the SyncManager when starting the EtherCAT system, in the event of an incorrect state change within the EtherCAT state machine or in the event of a change in or out of the Fault device state. EMCY messages are transmitted to the EtherCAT master via the mailbox channel.

EMCY message: Switch to the fault state

The following graphic shows an example of the structure of an EMCY message when switching to the Fault device state.



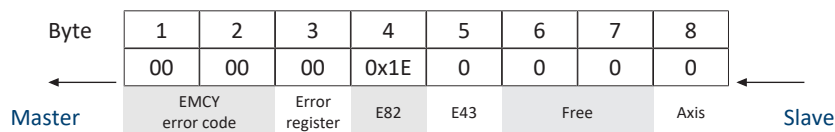
Bytes 1 – 3 contain the error code and error register, bytes 4 – 5 contain the values of the parameters E82 Event type and E43 Event cause.

Byte 8 indicates which axis is affected. If the value is 0, the fault originates from axis A or the global part of the drive controller. If the value is 1, the fault originates from axis B.

A table with the possible error codes of an EMCY message can be found at [EMCY message: Device fault error codes](#) [120].

EMCY message: Exiting the fault state

The following graphic shows an example of the structure of an EMCY message when leaving the Fault device state.



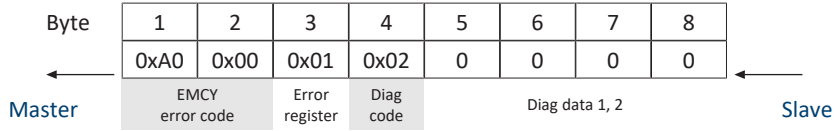
Bytes 1 – 3 contain the error code 0 hex (NO ERROR); byte 4 contains the value 1E hex for parameter E82 = 30: Inactive.

Byte 8 indicates which axis is affected. If the value is 0, the fault originates from axis A or the global part of the drive controller. If the value is 1, the fault originates from axis B.

Intern

EMCY message: Incorrect state transitions

If an error occurs during the state transitions within the EtherCAT state machine, the EtherCAT slave sends a corresponding EMCY message with the associated error code to the EtherCAT master. In accordance with the CANopen standard, an EMCY message is structured as follows in the event of a state change.



Diag data diagnostic data refers to dynamic parameters that are also provided by the firmware. This data is important for diagnostic purposes in the case of support.

You can find a table with the potential encodings for an EMCY message in the appendix (see [EMCY message: Incorrect state transition error codes \[119\]](#)).

9.4 EtherCAT state machine

The EtherCAT state machine (ESM) describes the different states of an EtherCAT slave along with any potential state change. Different functions can be executed in the EtherCAT slaves depending on the individual states.

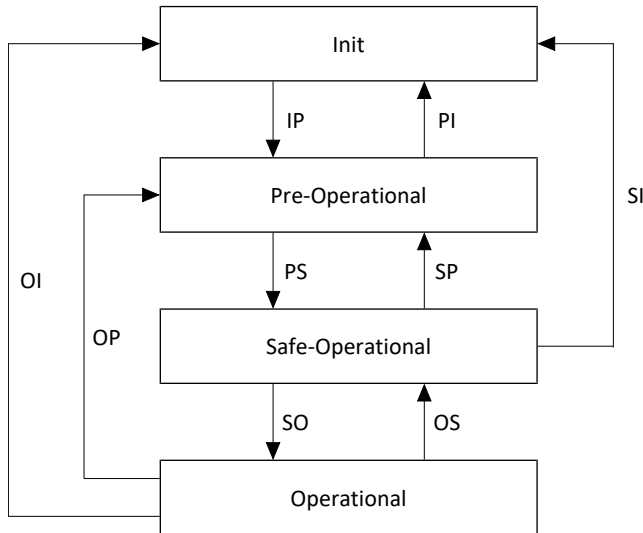


Fig. 10: EtherCAT state machine: States and state changes

States

State	Description
Init	State after an EtherCAT slave is switched on. The configuration starts; saved values are loaded. Neither SDO nor PDO communication is possible using the mailbox and process data channels, i.e. the master and slave do not communicate directly.
Pre-operational	The mailbox channel is active; the master and slaves exchange application-specific parameters using SDO communication.
Safe-operational	The mailbox and process data channels are active. All network nodes are shifted into a safe state. The slaves send current actual values to the master, but they ignore the master's set values and instead refer to internal default values.
Operational	The mailbox and process data channels are active. This state characterizes normal operation, i.e. the master and slaves exchange set and actual values.

State change

State change	Description
IP: Start Mailbox Communication	Start of SDO communication over the mailbox channel.
PI: Stop Mailbox Communication	Stop of SDO communication over the mailbox channel.
PS: Start Input Update Start Input Update	Start of PDO communication over the process data channel.
SP: Stop Input Update	Stop of PDO communication over the process data channel; the slaves do not send any actual values.
SO: Start Output Update	The slaves evaluate the current set value specifications of the master.
OS: Stop Output Update	The slaves ignore the set values of the master and refer to internal default values.
OP: Stop Output Update, Stop Input Update	Stop of PDO communication over the process data channel; neither the master nor the slaves send actual or set values.
SI: Stop Input Update, Stop Mailbox Communication	Stop of PDO and SDO communication over the corresponding channels; neither the master nor the slaves send actual or set values.
OI: Stop Output Update, Stop Input Update, Stop Mailbox Communication	Stop of PDO and SDO communication over the corresponding channels; neither the master nor the slaves send actual or set values.

9.5 Synchronization

For spatially distributed processes that require simultaneous actions, the EtherCAT master and slaves absolutely must work in synchronization with each other in the same cycle. EtherCAT provides two different methods for synchronizing the master and slaves: a SyncManager event (SM-Sync) and distributed clocks (DC-Sync). The FreeRun state exists if the master and slaves are not synchronized.

Each EtherCAT Slave Controller has a SyncManager that manages the memory units of a slave. The slave controller indicates incoming process using an interrupt signal that is used to synchronize individual EtherCAT slaves for SM-Sync; an additional interrupt signal is responsible for synchronization for DC-Sync.

9.5.1 SM-Sync: Synchronization using SyncManager event

In the case of synchronization using a SyncManager event, the EtherCAT slaves synchronize using incoming data as the event.

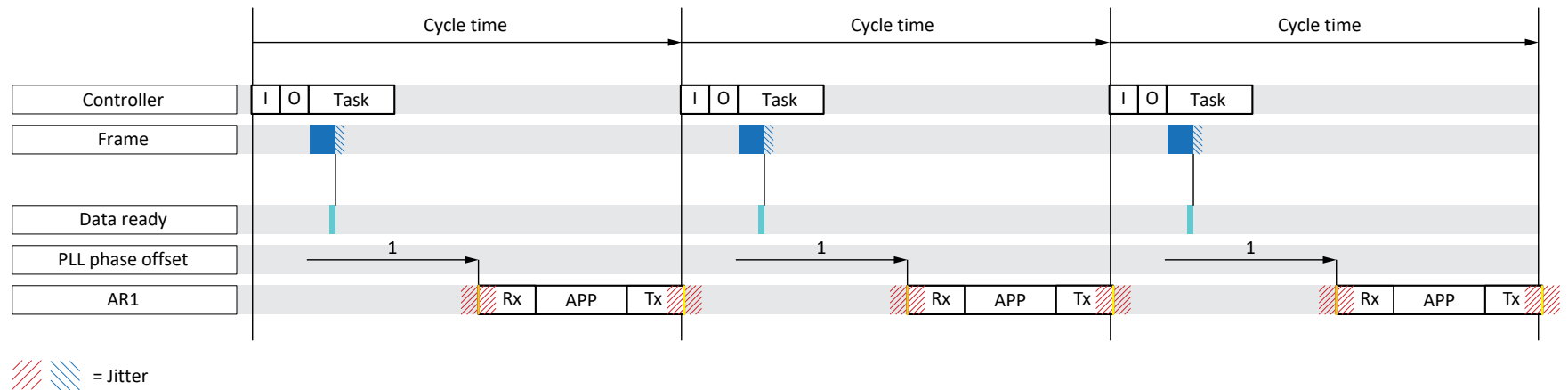


Fig. 11: SM-Sync: Synchronization using SyncManager event

Blue	Frame	Represents the duration of the frame arriving in the drive controller
Light blue	Data ready	The incoming process data for the drive controller was stored in the EtherCAT slave controller (ESC)
Orange	Rx	Beginning of the application in the drive controller; the process data to be calculated is read out from the ESC and calculated in the application
Yellow	Tx	End time of the application in the drive controller; the calculated process data was fully transmitted to the ESC

The following times are significant for SM-Sync.

- ▶ Master cycle time
... Time within which a master task is continuously called up and processed.
- ▶ Slave cycle time
... Time within which a slave task is continuously called up and processed.
- ▶ PLL phase offset
... Time for which the beginning of the individual slave task can be shifted. The task beginning can only be shifted within the scope of the slave cycle time.

The quality of synchronization using SM-Sync suffers in the case of delays in the PDOs from master to slave. Because master jitter has an immediate impact on the slaves, this synchronization method leads to a poorer result than an synchronization using distributed clocks.

9.5.2 DC-Sync: Synchronization using distributed clocks

Synchronization using the distributed clocks method allows the same time to be maintained for all nodes of an EtherCAT network.

Each EtherCAT slave with distributed clocks functionality has a local clock. Normally, the time from the first DC-Sync-capable EtherCAT slave downstream of the master in the network serves as the reference time. Both the master and the slaves synchronize to this reference clock.

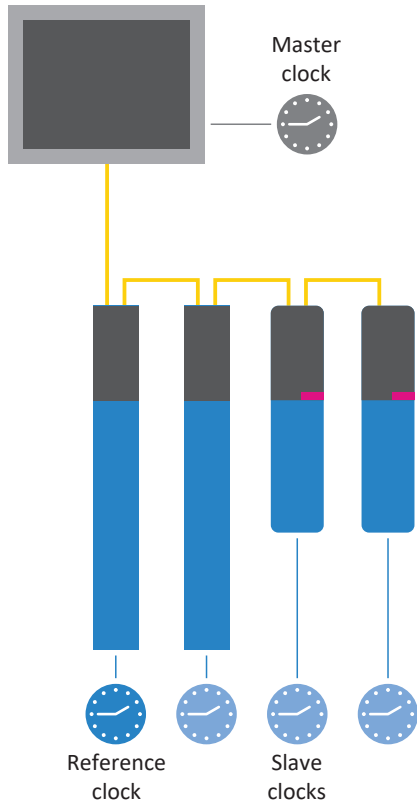


Fig. 12: EtherCAT: Distributed clocks

The EtherCAT master automatically and continuously initiates the time coordination and synchronization.

At specified intervals, the EtherCAT master sends a frame into which the reference slave enters its current time. All other slaves, as well as the master, read out this time from the circulating frame. Because each slave experiences a certain delay when reading in the reference time due to the transmission path, the respective run times between the reference clock and the slave clocks must be taken into account. This is why an individual offset value is measured, calculated and parameterized for each slave.

The synchronous operation of all distributed master and slave clocks in the network allows for highly precise, relative time information.

Moreover, this method has a high level of tolerance for fault-induced delays in the communication system thanks to the clock distribution.

9.5.2.1 TwinCAT 3: Synchronization using DC-Sync

The event for one synchronization is referred to as the SYNC 0 signal in TwinCAT 3. Each slave generates its own SYNC 0 signal cyclically using the respective SyncManager.

9.5.2.1.1 DC settings

The following graphic shows stable synchronization using distributed clocks when using TwinCAT 3. Both the utilization of the control as well as the set times show a stable system, since the frame jitter (controller) and jitter for writing the PDO data in the ESC (drive controller) are separated from each other in time.

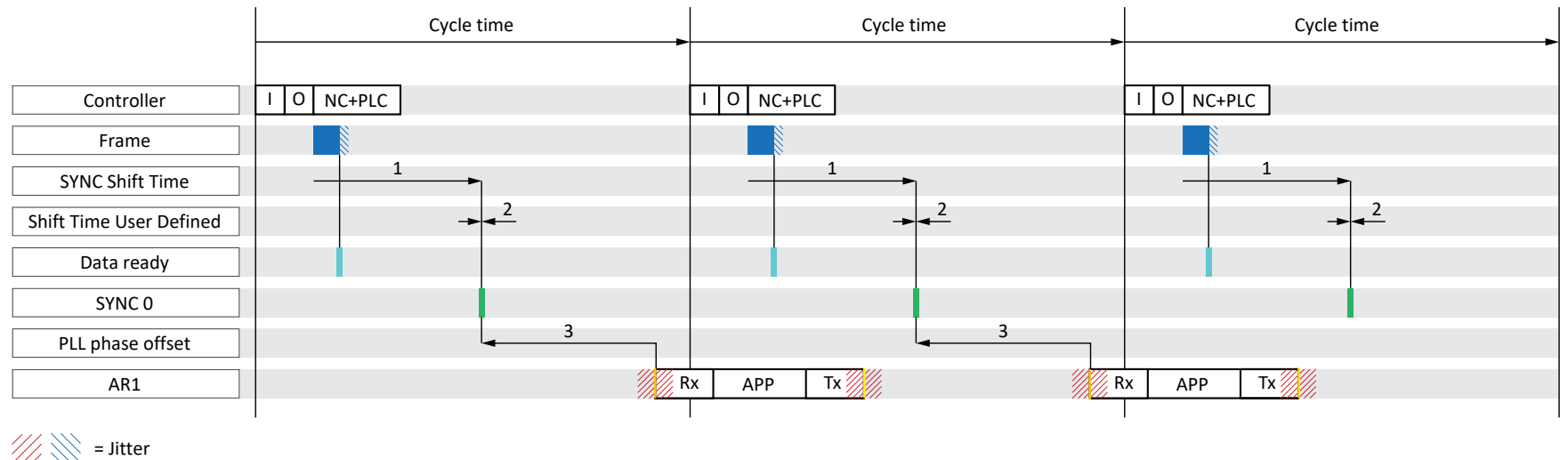


Fig. 13: TwinCAT 3: DC-Sync – Settings

Blue	Frame	Represents the duration of the frame arriving in the drive controller
Light blue	Data ready	The incoming process data for the drive controller was stored in the ESC
Green	SYNC 0	Synchronization signal for the DC synchronization
Orange	Rx	Beginning of the application in the drive controller; the process data to be calculated is read out from the ESC and calculated in the application
Yellow	Tx	End time of the application in the drive controller; the calculated process data was fully transmitted to the ESC

In the example, the data transfer (I/O) of the controller is set to task start in the EtherCAT configuration. For the application cycle sequence, the order is defined as RxPDO, graphical programming, TxPDO (A149 = 0).

Settings on the master's and slave's end

In general, the following settings are particularly significant for DC-Sync.

- ▶ SYNC Shift Time
 - ... Specifies the time span between the release of the process data from the master and the SYNC 0 signal of the slaves for the entire network simultaneously. SYNC Shift Time is parameterized on the master's end.
- ▶ Shift Time User Defined
 - ... Shifts the time of the SYNC 0 event in addition for SYNC Shift Time for each slave individually. Shift Time User Defined is also parameterized on the master's end.
- ▶ PLL phase offset
 - ... Specifies the time span between the SYNC 0 signal and the start of process data processing by the slave. The phase offset is parameterized on the slave's end, i.e. in the drive controller in parameter A292. A negative value pushes the beginning of processing to after the synchronization signal. A292 can only move the beginning of processing within the cycle time of the drive controller.
- ▶ Sync Unit Cycle
 - Permitted cycle times for a SYNC 0 signal must be whole-number multiples of the slave cycle time A150 and must not exceed 8 ms. Non-permitted signal times result in a slave not changing from the pre-operational state to the safe operational state. Sync Unit Cycle is parameterized on the master's end.

Conditions for stable synchronization

If the master cycle time is equal to the slave cycle time, the following conditions must be met for stable synchronization:

- ▶ $\text{SYNC Shift Time (1) + Shift Time User Defined (2) - PLL phase offset (3) + AR1 + Jitter} < \text{Cycle time}$

If the master cycle time is a multiple of the slave cycle time, the following condition also applies:

- ▶ $\text{SYNC Shift Time (1) + Shift Time User Defined (2) - PLL phase offset (3)} < \text{Slave cycle time}$

Checking settings

If you want to check your settings, take the following values into consideration for AR1 and jitter:

- ▶ AR1:
 - The current utilization of the real-time task gives you parameter E191
- ▶ Frame jitter (controller):
 - $\pm 5 \mu\text{s}$
- ▶ Application jitter (drive controller):
 - $\pm 10 \mu\text{s}$

9.5.2.1.1.1 Cycle time < 1 ms

For cycle times < 1 ms, quality defects can occur in the EtherCAT communication if the receipt of the PDO data from the controller and the writing of the process data in the ESC of the drive controller overlap in time. The following graphic shows unstable synchronization using distributed clocks when using TwinCAT 3.

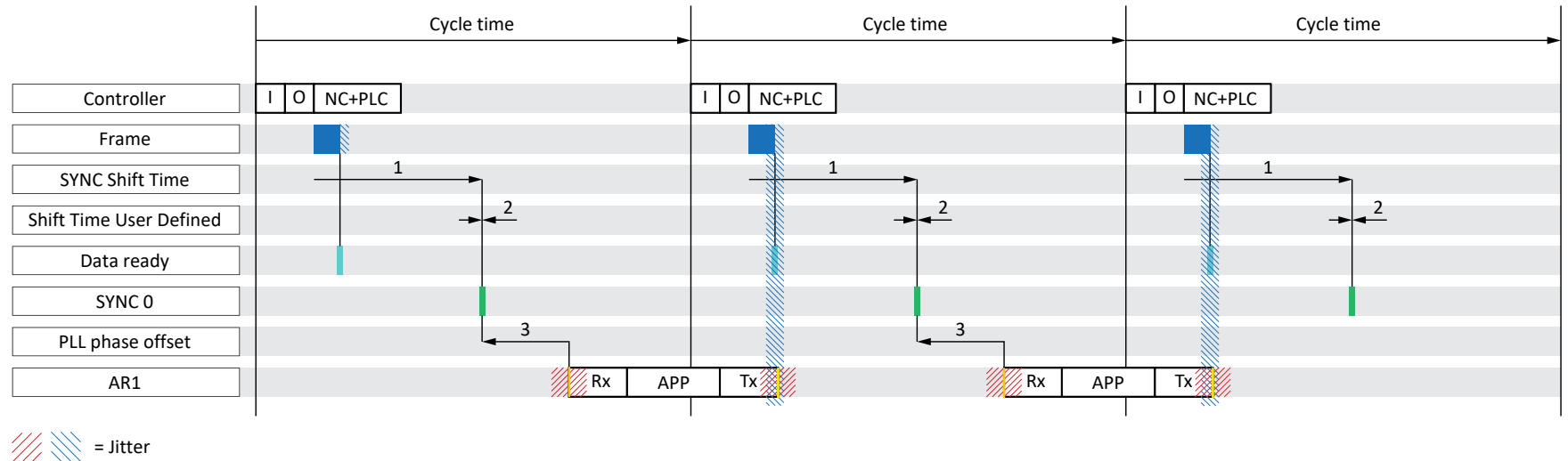


Fig. 14: TwinCAT 3: DC-Sync – Unstable synchronization, cycle time < 1 ms

Blue	Frame	Represents the duration of the frame arriving in the drive controller
Light blue	Data ready	The incoming process data for the drive controller was stored in the ESC
Green	SYNC 0	Synchronization signal for the DC synchronization
Orange	Rx	Beginning of the application in the drive controller; the process data to be calculated is read out from the ESC and calculated in the application
Yellow	Tx	End time of the application in the drive controller; the calculated process data was fully transmitted to the ESC

In the example, the data transfer (I/O) of the controller is set to task start in the EtherCAT configuration. For the application cycle sequence, the order is defined as RxPDO, graphical programming, TxPDO (A149 = 0).

Intern

Changing the cycle sequence

For cycle times < 1 ms, change the cycle sequence of the application to RxPDO, TxPDO, graphical programming (A149 = 1). The following example shows a synchronization with a changed cycle sequence. The frame jitter (controller) and the writing of the PDO data in the ESC of the drive controller (Tx) are separated from each other in time and synchronization is stable.

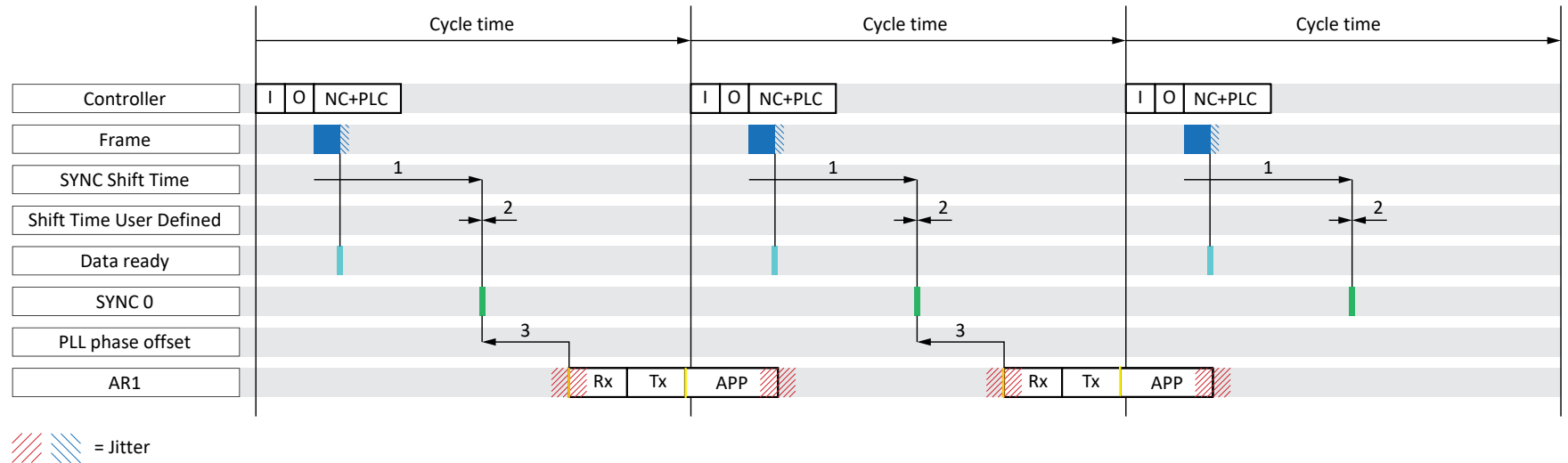


Fig. 15: TwinCAT 3: DC-Sync – Stable synchronization, cycle time < 1 ms

Blue	Frame	Represents the duration of the frame arriving in the drive controller
Light blue	Data ready	The incoming process data for the drive controller was stored in the ESC
Green	SYNC 0	Synchronization signal for the DC synchronization
Orange	Rx	Beginning of the application in the drive controller; the process data to be calculated is read out from the ESC and calculated in the application
Yellow	Tx	End time of the application in the drive controller; the calculated process data was fully transmitted to the ESC

9.5.2.1.1.2 Optimizing DC-Sync using DriveControlSuite

You can check your settings using DriveControlSuite and optimize the DC synchronization. The Optimize DC-Sync wizard suggests a suitable value range for you for the PLL phase offset. Stable DC synchronization is ensured if neither Rx nor Tx (both including jitter) are in the time range of the frame of the controller (including jitter). To achieve this, the beginning the application in the drive controller Rx can be shifted as desired using the PLL phase offset in A292.

- ✓ You are in DriveControlSuite.
- 1. Select the relevant drive controller in the project tree and click on the desired projected axis in the Project menu > Wizard area.
- 2. Select the EtherCAT wizard > Optimize DC-Sync.
 - ⇒ The current settings for cycle sequence, cycle time and PLL phase offset are displayed.
- 3. Click on Start measurement.
 - ⇒ The Start of Frame, End of Frame, Rx, Tx and Sync signals are measured and a suitable PLL phase offset is calculated (measurement duration: approx. 1000 × cycle time).
 - ⇒ A287[1] indicates the progress:
The measured values of End of Frame to Sync, frame duration and Sync to Rx are displayed.
 - ⇒ A287[2] indicates the result:
After the measurement is successfully completed with result 7 of 8, the current situation of the signals is represented in a diagram.
 - ⇒ For the PLL phase offset, a minimum and maximum value are suggested.
- 4. In A292, enter a value that is between the two values.
 - ⇒ The optimization of DC synchronization is concluded.

If a measurement is not successful (result < 7), the value set in A292 is displayed as minimum and maximum recommended PLL phase offset.

Meaning of the measured values

- ▶ End of Frame to Sync: Time difference between frame end and synchronization signal
- ▶ Frame duration: Maximum duration of the frame, including jitter, which contains the process data for these drive controllers.
- ▶ Sync to Rx: Time difference between the synchronization signal and beginning of the application in the drive controller

9.5.2.1.2 Optimize values and correct problems

You have commissioned your EtherCAT network. If you need to optimize synchronization using distributed clocks after the fact due to insufficient EtherCAT communication quality, we recommend the following measures.

9.5.2.1.2.1 EtherCAT master: DC-Sync configured for EtherCAT slaves?

Check whether DC-Sync is configured for all EtherCAT slaves on the master's end; see [Configuring synchronization using distributed clocks \[44\]](#).

9.5.2.1.2.2 EtherCAT slave: Check control

Check the status of control for all EtherCAT slaves and take one of the described measures if necessary.

- ✓ You are in DriveControlSuite.
- 1. Select the relevant drive controller in the project tree and click on the desired projected axis in the Project menu > Wizard area.
- 2. Select the PLL synchronization wizard.
 - ⇒ A298 shows the status of the synchronization between the controller and the drive controller in question.
- 3. Bit 0 – 1: PLL engaged
If one or both of the two associated LEDs lights up, the control range is working at $\geq 50\%$ capacity (frequency too high or too low).
In this case, adjust the cycle time of the Sync 0 signal on the master's end. Note that the cycle time of the Sync 0 signal must be a whole-number multiple of the cycle time A150 and must not exceed 8 ms.
- 4. Bit 2: Cycle time extended
If the associated LED lights up, the PLL has performed an extending control intervention on the task system.
- 5. Bit 3: Maximum control range reached
If the associated LED lights up, check whether the cycle times of the master and drive controller agree. Realign these to each other if necessary.
- 6. Bit 4: Cycle time for the synchronization signals is greater than the specification (A296 > A291)
If the associated LED lights up, check whether the cycle times of the master and drive controller agree. Realign these to each other if necessary.
- 7. Bit 5: Control/synchronization deactivated
If the associated LED lights up, set A290 to 1: Active.

9.5.2.1.2.3 EtherCAT slave: Synchronization – Read out diagnostic parameter

You can get information about the status of the EtherCAT synchronization using the A261 diagnostic parameter. It checks whether a frame arrives at an EtherCAT slave within a certain time period based on the Sync 0 signal.

- ✓ You are in DriveControlSuite.
- 1. Select the relevant drive controller in the project tree and click on the desired projected axis in the Project menu > Wizard area.
- 2. Select the EtherCAT wizard > Diagnostics.
 - ⇒ A261[0] – [3] shows the state of the EtherCAT synchronization.
- 3. A261[0]:
Specifies the number of the error code.
- 4. A261[1]:
Specifies the time difference between the data provision and the Sync 0 signal in μs .
- 5. A261[2]:
If the process data from the master arrived at the slave after the Sync 0 signal in the slave, or if the time difference between process data receipt and Sync 0 signal is greater than half of A150, A261[2] is incremented.

9.5.2.2 CODESYS V3: Synchronization using DC-Sync

The event for one synchronization is referred to as the Sync 0 signal in CODESYS V3. Each slave generates its own Sync 0 signal cyclically using the respective SyncManager.

9.5.2.2.1 DC settings

The following graphic shows stable synchronization using distributed clocks when using CODESYS V3. Both the utilization of the controller as well as the set times show a stable system, since the frame jitter (controller) and the jitter for writing the PDO data in the ESC (drive controller) are separated from each other in time.

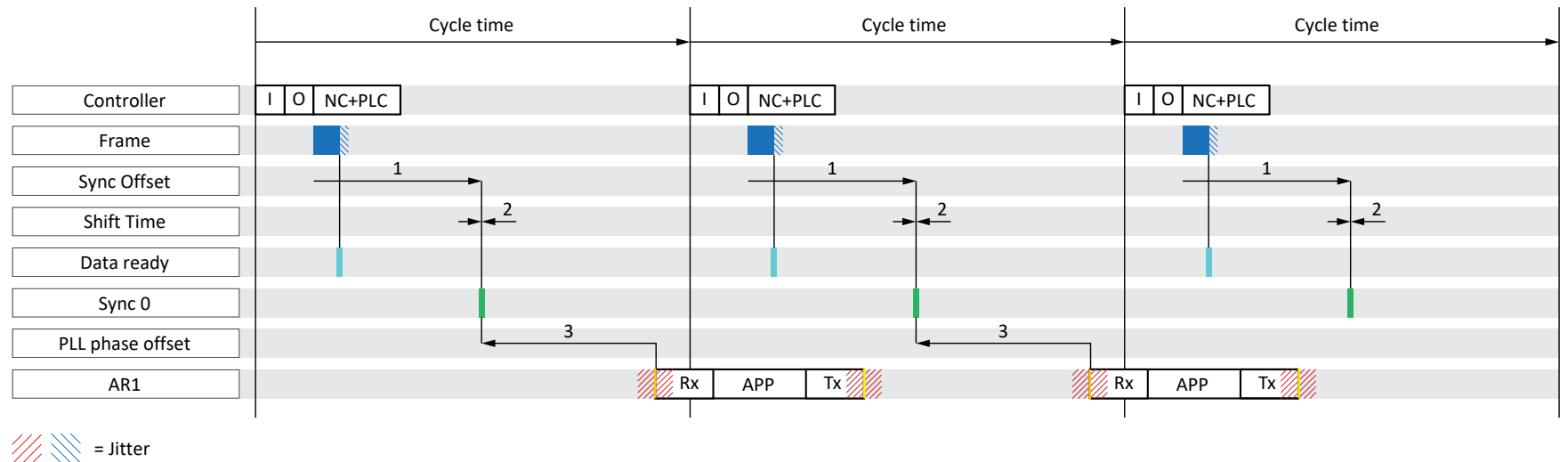


Fig. 16: CODESYS V3: DC-Sync – Settings

Blue	Frame	Represents the duration of the frame arriving in the drive controller
Light blue	Data	The incoming process data for the drive controller was stored in the ESC
Green	Sync 0	Synchronization signal for the DC synchronization
Orange	Rx	Beginning of the application in the drive controller; the process data to be calculated is read out from the ESC and calculated in the application
Yellow	Tx	End time of the application in the drive controller; the calculated process data was fully transmitted to the ESC

In the example, the data transfer (I/O) of the controller is set to task start in the EtherCAT configuration. For the application cycle sequence, the order is defined as RxPDO, graphical programming, TxPDO (A149 = 0).

Settings on the master's and slave's end

In general, the following settings are particularly significant for DC-Sync:

- ▶ Sync Offset
 - ... Specifies the time span between the release of the process data from the master and the Sync 0 signal of the slaves for the entire network simultaneously. Sync Offset is parameterized on the master's end.
- ▶ Shift Time
 - ... Shifts the time of the Sync 0 event in addition to Sync Offset for each slave individually. Shift Time is also parameterized on the master's end.
- ▶ PLL phase offset
 - ... Specifies the time span between the Sync 0 signal and the start of process data processing by the slave. The phase offset is parameterized on the slave's end, i.e. in the drive controller in parameter A292. A negative value pushes the beginning of processing to after the synchronization signal. A292 can only move the beginning of processing within the cycle time of the drive controller.
- ▶ Sync Unit Cycle
 - Permitted cycle times for a Sync 0 signal must be whole-number multiples of the slave cycle time A150 and must not exceed 8 ms. Non-permitted signal times result in a slave not changing from the pre-operational state to the safe operational state. Sync Unit Cycle is parameterized on the master's end.

Conditions for stable synchronization

If the master cycle time is equal to the slave cycle time, the following conditions must be met for stable synchronization:

- ▶ $\text{Sync Offset (1) + Shift Time (2) - PLL phase offset (3) + AR1 + Jitter} < \text{Cycle time}$

If the master cycle time is a multiple of the slave cycle time, the following condition also applies:

- ▶ $\text{Sync Offset (1) + Shift Time (2) - PLL phase offset (3)} < \text{Slave cycle time}$

Checking settings

If you want to check your settings, take the following values into consideration for AR1 and jitter:

- ▶ AR1:
 - The current utilization of the real-time task gives you parameter E191.
- ▶ Frame jitter (controller):
 - $\pm 5 \mu\text{s}$
- ▶ Application jitter (drive controller):
 - $\pm 10 \mu\text{s}$

9.5.2.2.1.1 Cycle time < 1 ms

For cycle times < 1 ms, quality defects can occur in the EtherCAT communication if the receipt of the PDO data from the controller and transmission of the process data from the drive controller overlap. The following graphic shows unstable synchronization using distributed clocks when using CODESYS V3.

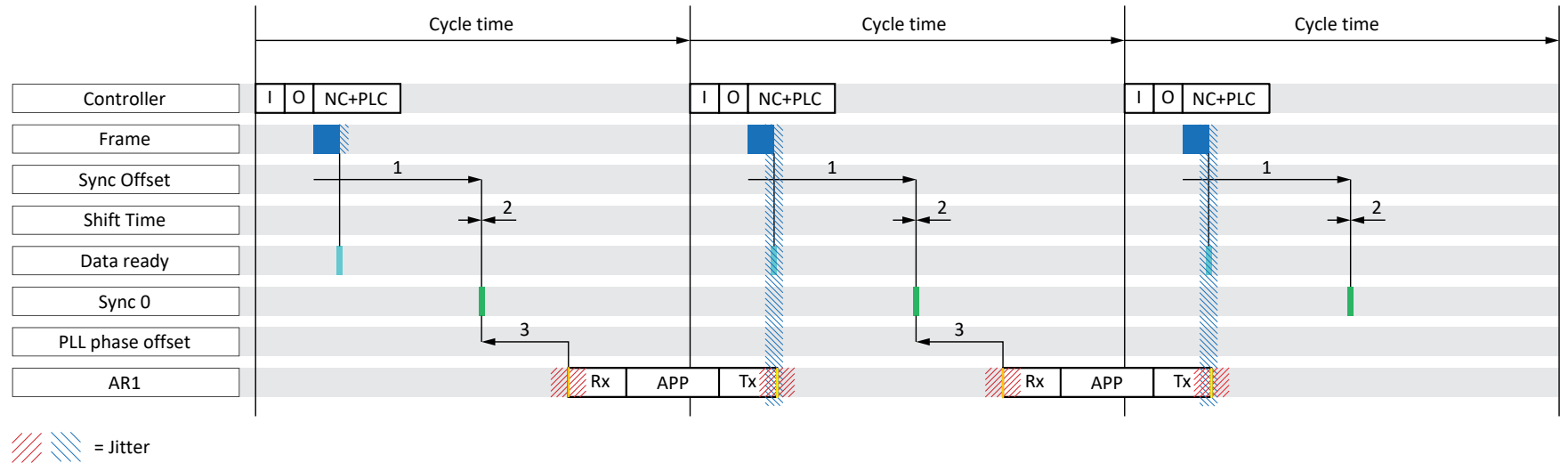


Fig. 17: CODESYS V3: DC-Sync – Unstable synchronization, cycle time < 1 ms

Blue	Frame	Represents the duration of the frame arriving in the drive controller
Light blue	Data	The incoming process data for the drive controller was stored in the ESC
Green	Sync 0	Synchronization signal for the DC synchronization
Orange	Rx	Beginning of the application in the drive controller; the process data to be calculated is read out from the ESC and calculated in the application
Yellow	Tx	End time of the application in the drive controller; the calculated process data was fully transmitted to the ESC

In the example, the data transfer (I/O) of the controller is set to task start in the EtherCAT configuration. For the application cycle sequence, the order is defined as RxPDO, graphical programming, TxPDO (A149 = 0).

Intern

Changing the cycle sequence

For cycle times < 1 ms, change the cycle sequence of the application to RxPDO, TxPDO, graphical programming (A149 = 1). The following example shows a synchronization with a changed cycle sequence. The frame jitter (controller) and the writing of the PDO data in the ESC of the drive controller (Tx) are separated from each other in time and synchronization is stable.

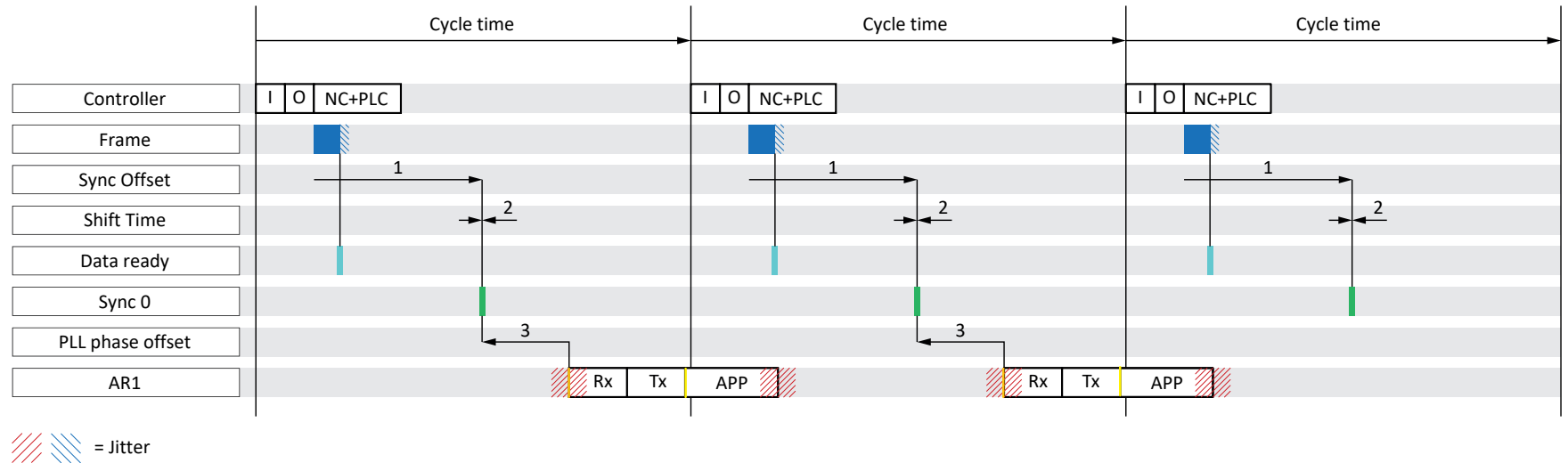


Fig. 18: CODESYS V3: DC-Sync – Stable synchronization, cycle time < 1 ms

Blue	Frame	Represents the duration of the frame arriving in the drive controller
Light blue	Data	The incoming process data for the drive controller was stored in the ESC
Green	Sync 0	Synchronization signal for the DC synchronization
Orange	Rx	Beginning of the application in the drive controller; the process data to be calculated is read out from the ESC and calculated in the application
Yellow	Tx	End time of the application in the drive controller; the calculated process data was fully transmitted to the ESC

9.5.2.2.1.2 Optimizing DC-Sync using DriveControlSuite

You can check your settings using DriveControlSuite and optimize the DC synchronization. The Optimize DC-Sync wizard suggests a suitable value range for you for the PLL phase offset. Stable DC synchronization is ensured if neither Rx nor Tx (both including jitter) are in the time range of the frame of the controller (including jitter). To achieve this, the beginning the application in the drive controller Rx can be shifted as desired using the PLL phase offset in A292.

- ✓ You are in DriveControlSuite.
- 1. Select the relevant drive controller in the project tree and click on the desired projected axis in the Project menu > Wizard area.
- 2. Select the EtherCAT wizard > Optimize DC-Sync.
 - ⇒ The current settings for cycle sequence, cycle time and PLL phase offset are displayed.
- 3. Click on Start measurement.
 - ⇒ The Start of Frame, End of Frame, Rx, Tx and Sync signals are measured and a suitable PLL phase offset is calculated (measurement duration: approx. $1000 \times$ cycle time).
 - ⇒ A287[1] indicates the progress:
The measured values of End of Frame to Sync, frame duration and Sync to Rx are displayed.
 - ⇒ A287[2] indicates the result:
After the measurement is successfully completed with result 7 of 8, the current situation of the signals is represented in a diagram.
 - ⇒ For the PLL phase offset, a minimum and maximum value are suggested.
- 4. In A292, enter a value that is between the two values.
 - ⇒ The optimization of DC synchronization is concluded.

If a measurement is not successful (result < 7), the value set in A292 is displayed as minimum and maximum recommended PLL phase offset.

Meaning of the measured values

- ▶ End of Frame to Sync: Time difference between frame end and synchronization signal
- ▶ Frame duration: Maximum duration of the frame, including jitter, which contains the process data for these drive controllers.
- ▶ Sync to Rx: Time difference between the synchronization signal and beginning of the application in the drive controller

9.5.2.2.2 Optimize values and correct problems

You have commissioned your EtherCAT network. If you need to optimize synchronization using distributed clocks after the fact due to insufficient EtherCAT communication quality, we recommend the following measures.

9.5.2.2.2.1 EtherCAT master: DC-Sync configured for EtherCAT slaves?

Check whether DC-Sync is configured for all EtherCAT slaves on the master's end. If this is not the case, change the configuration as described below.

- ✓ You are in CODESYS V3.
- 1. Navigate to the first of the added drive controllers in the device tree and double click to open it.
- 2. Distributed Clock:
 - Select DC: The list entry DC enabled (multiplier = 1) must be selected.
 - Sync 0: The option Enable Sync 0 must be activated.
 - Cycle Time and Sync Offset: Make sure that the presets match the corresponding values of A291 PLC Cycle time and A293 PLL gain in DriveControlSuite.
- 3. If you would like to change the presets, activate the option Additional > Enable Expert Settings and adjust the settings accordingly.
- 4. Repeat steps 2 and 3 for all of the slaves in your EtherCAT network.

9.5.2.2.2.2 EtherCAT slave: Check control

Check the status of control for all EtherCAT slaves and take one of the described measures if necessary.

- ✓ You are in DriveControlSuite.
- 1. Select the relevant drive controller in the project tree and click on the desired projected axis in the Project menu > Wizard area.
- 2. Select the PLL synchronization wizard.
 - ⇒ A298 shows the status of the synchronization between the controller and the drive controller in question.
- 3. Bit 0 – 1: PLL engaged
 - If one or both of the two associated LEDs lights up, the control range is working at $\geq 50\%$ capacity (frequency too high or too low).
 - In this case, adjust the cycle time of the Sync 0 signal on the master's end. Note that the cycle time of the Sync 0 signal must be a whole-number multiple of the cycle time A150 and must not exceed 8 ms.
- 4. Bit 2: Cycle time extended
 - If the associated LED lights up, the PLL has performed an extending control intervention on the task system.
- 5. Bit 3: Maximum control range reached
 - If the associated LED lights up, check whether the cycle times of the master and drive controller agree. Realign these to each other if necessary.
- 6. Bit 4: Cycle time for the synchronization signals is greater than the specification (A296 > A291)
 - If the associated LED lights up, check whether the cycle times of the master and drive controller agree. Realign these to each other if necessary.
- 7. Bit 5: Control/synchronization deactivated
 - If the associated LED lights up, set A290 to 1: Active.

9.5.2.2.2.3 EtherCAT slave: Synchronization – Read out diagnostic parameter

You can get information about the status of the EtherCAT synchronization using the A261 diagnostic parameter. It checks whether a frame arrives at an EtherCAT slave within a certain time period based on the Sync 0 signal.

✓ You are in DriveControlSuite.

1. Select the relevant drive controller in the project tree and click on the desired projected axis in the Project menu > Wizard area.
2. Select the EtherCAT wizard > Diagnostics.
⇒ A261[0] – [3] shows the state of the EtherCAT synchronization.
3. A261[0]:
Specifies the number of the error code.
4. A261[1]:
Specifies the time difference between the data provision and the Sync 0 signal in μs .
5. A261[2]:
If the process data from the master arrived at the slave after the Sync 0 signal in the slave, A261[2] is incremented.

9.6 Modular ESI files

An ESI file involves a device description file that is made available to the EtherCAT master, i.e. a controller, for the configuration of the EtherCAT network. Each controller accepts a maximum of one ESI file per drive controller series for configuring the corresponding EtherCAT network.

In order to guarantee maximum flexibility regarding PDO transmission options, Pilz ESI files have a modular structure.

A Pilz ESI file contains specified configurations for PDO transmission for every application in the form of default modules. You can add to the standard configurations of any application or configured PDO transmission freely as desired and add your Pilz ESI file as a new module. The number of expandable modules is unlimited.

9.6.1 Adding to a modular ESI file

- ✓ You have expanded the configuration for the RxPDO and/or TxPDO transmission specified on the system side.

In order to make this available to the controller, add a new module that contains your configuration to the ESI file.

1. Select the relevant drive controller in the project tree and click on the desired projected axis in the Project menu > Wizard area.
2. Select the EtherCAT wizard.
3. E72 User configuration identification:
Give the new module a descriptive name.
4. Click on Edit ESI.
 - ⇒ The Add to ESI file dialog box opens.
5. Navigate to the location where you saved the ESI file, highlight the file and click Open.
 - ⇒ The EsiModuleEdit dialog box opens.
In addition to standard modules (Modules of the ESI file column), the ESI contains the module previously created by you (New modules column).
6. New modules column:
In order to add the new module to the ESI file, click on the green arrow and confirm with OK.
 - ⇒ The Edit ESI dialog box opens.
7. Save the addition to the ESI file by clicking on Yes.
8. Repeat the steps for each additional module that you would like to add to the ESI in question.
 - ⇒ You have added your individual PDO configuration to the ESI file.



Information

All ESI modules are axis-dependent.

9.6.2 Deleting a module from the ESI file

You can delete a configuration of the PDO transmission you have added, i.e. the associated module, from an existing ESI file.



Information

We recommend against deleting the system-specified modules of an ESI file, even if these are not used.

1. Select the relevant drive controller in the project tree and click on the desired projected axis in the Project menu > Wizard area.
2. Select the EtherCAT wizard.
3. Click on Edit ESI.
 - ⇒ The Add to ESI file dialog box opens.
4. Navigate to the location where you saved the ESI file in question, highlight the file and click Open.
 - ⇒ The EsiModuleEdit dialog box opens.
5. Modules of the ESI file column:
 - Click on the red cross for the module that you would like to delete and confirm with OK.
 - ⇒ The Edit ESI dialog box opens.
6. Save the modified ESI file by clicking on Yes.
 - ⇒ The module is deleted from the ESI file.

9.7 Cycle times

Possible cycle times can be found in the following table.

Type	Cycle times	Relevant parameters
EtherCAT fieldbus, cyclical communication	250 μ s, 500 μ s, 1 ms, 2 ms, 4 ms, 8 ms	Adjustable in A150

Tab. 9: Cycle times

9.8 Activating and executing actions

To be able to activate and execute actions via fieldbus, you must first enable action activation in DriveControlSuite and extend the process data by the control byte and status word for actions.

Enabling action activation

1. Select the relevant drive controller in the project tree and click on the desired projected axis in the Project menu > Wizard area.
2. Select the CiA 402 application wizard > Additional functions.
3. Enable the Action activation option.
4. Repeat the steps for the 2nd axis (only for double-axis controllers).

Adjusting receive process data

1. Select the relevant drive controller in the project tree and click on the desired projected axis in the Project menu > Wizard area.
2. Select the EtherCAT wizard > Received process data RxPDO.
3. A225[0] – A225[23], A226[0] – A226[23]:
Add the control byte for activating actions to the receive process data.
Single-axis controller: Complete 1.A75.
Double-axis controller: Complete 1.A75 in column A and 2.A75 in column B.

Adjusting transmit process data

1. Select the relevant drive controller in the project tree and click on the desired projected axis in the Project menu > Wizard area.
2. Select the EtherCAT wizard > Transmitted process data TxPDO.
3. A233[0] – A233[23], A234[0] – A234[23]:
Add the status word for activating actions to the transmit process data.
Single-axis controller: Complete 1.A69.
Double-axis controller: Complete 1.A69 in column A and 2.A69 in column B.

Executing an action

Then execute the desired action. Here, take into account any prerequisites with regard to the device state as well as any further measures required after the start of the action. All prerequisites as well as more detailed information on the individual actions can be found in the corresponding parameter descriptions in DriveControlSuite.

Selecting an action	Establishing the device state	Starting an action	Executing the next step	Completing an action (by progress = 100%)
0001 bin = Save values (A00)	—	Execute (A75, bit 0 = 1)	—	Undo execute (A75, bit 0 = 0)
0011 bin = Reset memorized values (A37)				
0111 bin = Clear reference (I38)				
1000 bin = Delete limit switch memory (I52)				
0010 bin = Restart (A09)	E48 ≠ 4: Enabled + E48 ≠ 7: Quick stop	Execute (A75, bit 0 = 1)	—	Undo execute (A75, bit 0 = 0)
1101 bin = Test winding (B43)	E48 = 2: Ready for switch-on	Execute (A75, bit 0 = 1)	—	Undo execute (A75, bit 0 = 0)
1010 bin = Test phase (B40)	E48 = 2: Ready for switch-on	Execute (A75, bit 0 = 1)	Enable drive controller (E48 = 4: Enabled)	Undo execute (A75, bit 0 = 0) + Undo enable
1011 bin = Calibrate motor (B41)				
1100 bin = Optimize current controller (B42)				
1110 bin = Optimize current controller (standstill) (B49)				
0100 bin = Test brake (B300)				
0101 bin = Grind brake (B301)				
0110 bin = Brake 2 grind (B302)				
1001 bin = Test brake (S18)				

Tab. 10: Selecting and executing an action

9.9 Fieldbus scaling

Using parameter A213, you define the scaling for both cyclical transmission of process data objects as well as acyclical transmission of service data objects in the network in the DriveControlSuite commissioning software. The values are either converted and represented as an integer or transmitted as a raw value without scaling according to their data types.

Regardless of the settings selected in parameter A213, the configuration as well as the firmware both work exclusively with raw values. The following graphic shows an overview of fieldbus scaling.

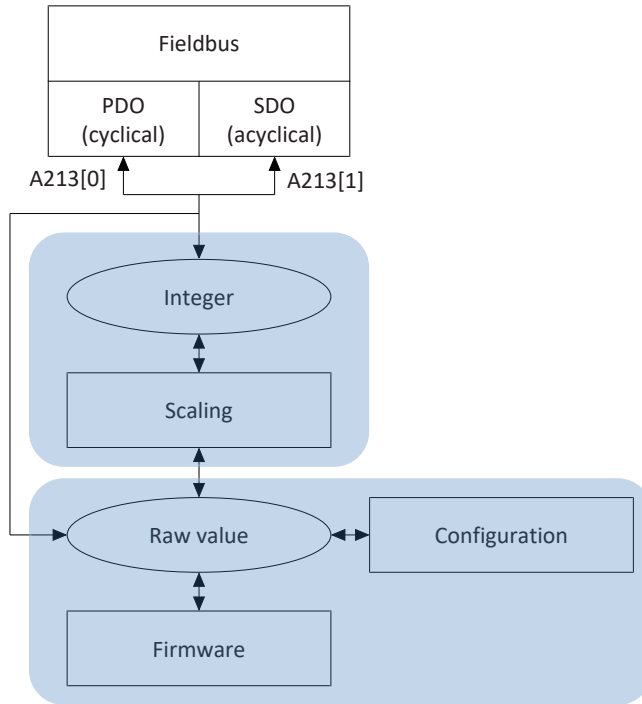


Fig. 19: Overview of fieldbus scaling

For transmission as an integer, the number of decimal places can be defined for all parameters that affect positions, velocities, accelerations, decelerations and jerk. For all other parameters, the number of decimal places is fixed. The values for scaling are output in DriveControlSuite with the properties of a parameter. The following table lists the parameters which you can use to define the number of decimal places for scaled transmission.

Scaling	Axis model	Master axis model
Position	I06	G46
Velocity (DB)	I66	G66
Velocity (CiA)	A310	—
Acceleration, deceleration, jerk (DB)	I67	G67
Acceleration, deceleration, jerk (CiA)	A311	—

Tab. 11: Fieldbus scaling for integer: Parameters for defining the decimal places

9.10 SDO Info service

Using the SDO Info service, the EtherCAT controller can read out the objects from the drive controller. During the read-out, all relevant object properties, such as data type, write and read access rights as well as mapping capabilities, are transmitted to the controller. You can define which objects are transmitted using the service in DriveControlSuite using parameter A268. The service is only supported if you have selected the EtherCAT Rx SDO Info template when creating the project in DriveControlSuite.



ATTENTION!

Change of addressing when changing the template

If you change the template from EtherCAT Rx to EtherCAT Rx SDO Info, the addressing of the elements of array and record parameters also changes. Note this in particular for existing configurations. For the templates, various ESI files are created. When changing the template, you must create a new ESI file using the wizards in DriveControlSuite and provide it to TwinCAT 3. A template change also causes a change to the revision number of the drive controller. Therefore, restart the drive controller after changing the template.

9.10.1 Setting SDO Info service in TwinCAT 3

- ✓ You have configured the drive controller in DriveControlSuite with the EtherCAT Rx SDO Info template.
 - ✓ The state of the drive controller in the EtherCAT network is Pre-Operational, Safe-Operational or Operational (display: A255).
 - ✓ The drive controller has already been created in the TwinCAT project.
1. Navigate to the drive controller from which the objects are to be read out in the solution explorer.
 2. Double-click on the drive controller.
 - ⇒ The settings open in the main window.
 3. In the main window, switch to the CoE – Online tab and click Advanced....
 - ⇒ The Advanced Settings window opens.
 4. Select Dictionary from the tree view on the left.
 5. Online - via SDO Information:
 - Activate this option and select the All Objects option from the list if all objects are to be read out. As an alternative, you can define that only Rx- or Tx-mapping-capable objects are to be read out.
 - The Backup Objects and Settings Objects options are not supported.
 6. Confirm the settings with OK.
 - ⇒ Reading out objects starts.
 - ⇒ After completing the read-out, the Advanced Settings window closes and all read-out objects are listed.

9.10.2 Access to objects

In the DriveControlSuite commissioning software, use parameter A268 to define the scope of the communication object list that is read out. By selecting the object groups, you define whether only the index area of standardized objects, only the index area of manufacturer-specific parameters or the entire index area is read out. Furthermore, you can define whether or not each parameter group from A to Z is part of the list in the manufacturer-specific parameters.

Using parameter A10[2], you define the access level. Only objects with an access level \leq the set access level are read out.

Note that, in addition to EtherCAT objects, only those objects are read out that are available through the configuration of the drive controller or depending on the application.

To enable changes to the values of objects in the solution explorer of TwinCAT 3 directly using the CoE - Online tab, set parameter A213[1] to 1: Native.

9.10.3 Check for conformity

In TwinCAT 3, the revision numbers are used to check whether the configuration in the drive controller and ESI file match. Therefore, in case of changes to the configuration of the drive controller, create a new ESI file using the wizard in DriveControlSuite and provide it to TwinCAT 3.



Information

The check for conformity happens only for ESI files that were created in DriveControlSuite commissioning software in version 6.5-D or later.

During the check, the revision numbers in the drive controller and ESI file are checked. If these do not match, you receive one of the error messages described below.

Error message when scanning the hardware environment

It is not possible to add a drive controller to the project. In TwinCAT XAE, you receive the error message `New device type found`.

Error message when starting the project configuration

After you have added a drive controller manually as a slave, or if you change the template in the drive controller in an existing TwinCAT project configuration, you receive the error message `Check revision number. Comparison failed when starting the project configuration in TwinCAT XAE`. The drive controller switches to the Init state.

9.11 Diagnosis History

With the help of the Diagnosis History object (10F3 hex), the EtherCAT diagnostic memory of the drive controller can be read out by the EtherCAT master. Up to 20 messages can be stored in the diagnostic memory of the drive controller. When the maximum number of 20 messages is reached, the oldest messages are overwritten. Diagnostic messages are stored in volatile memory. Each time the drive controller is restarted, the messages are deleted. A diagnostic message can be an Info, Warning or Error type. In addition, the time at which the event occurred in the drive controller is transmitted in the message. If A250 is added to the process data mapping in the DriveControlSuite of the parameters (EtherCAT wizard > Transmitted process data TxPDO), the automation software of the controller is able to determine that a new diagnostic message can be read out from the drive controller.

9.11.1 Reading out the Diagnosis History in TwinCAT 3

Diagnostic messages are displayed in TwinCAT 3 in the German, English or French language. The decisive factor is the language that you have set in TwinCAT XAE.

If you want to read out the Diagnosis History, proceed as follows:

1. Start TwinCAT XAE.
 2. In the solution explorer, navigate to the drive controller whose diagnosis history you want to read out.
 3. In the main window, switch to the Diag History tab.
 4. Click Update History.
- ⇒ The Diagnosis History is read out from the drive controller and displayed in the main window.



Information

If you enable the *Auto Update* option, new messages will be read out automatically. It is not necessary to click the *Update History* button. Activate the *Only New Messages* option if you want to hide already acknowledged messages. You can send messages via the button *Ack. Acknowledge messages*. In the *Flags* column, you can see which messages are new (N) and which have already been acknowledged (Q).

If required, in the advanced settings you can specify which messages should be stored in the Diagnosis History.

1. On the *Diag History* tab, click *Advanced...*
⇒ The *Advanced Settings* window opens.
2. In the section *Message Types*, define which messages should be stored in the *Diagnosis History*.
⇒ Deactivated message types are no longer stored in the *Diagnosis History*.
3. Confirm your selection with *OK*.



Information

Do not change the settings in the sections *Emergency* and *Overwrite/Acknowledge Mode*. Disabling these options will be ignored.


9.11.2 Determination of the system time

The system time in the drive controller can be determined in different ways:

Distributed Clocks

If the EtherCAT network is synchronized via Distributed Clocks, the current system time of the EtherCAT network is used as the time stamp of the diagnostic message.

SNTP server

If the EtherCAT network does not synchronize or if synchronization is done via SM-Sync, an SNTP server can be used to determine the current time stamp (see [Simple Network Time Protocol \(SNTP\)](#) [ 122]).

Without time stamp

If the current time stamp cannot be determined via Distributed Clocks or an SNTP server, the value 0 is transmitted as the time stamp. This value is also transmitted if an event occurs before the distributed clocks are synchronized or before the current timestamp is determined via an SNTP server.

10 Appendix

10.1 Supported communication objects

The following chapters provide an overview of the supported communication objects of the standardized ETG (EtherCAT Technology Group) profile as well as their mapping to the corresponding parameters of Pilz.

Information about the supported communication objects of the CiA 402 profile as well as about the standard mapping of the CiA 402 application and EtherCAT can be found in the corresponding application manual.

10.1.1 ETG.1000.6 EtherCAT specification: 1000 hex – 1FFF hex

The following table includes the supported communication objects for the standardized profile ETG.1000.6 EtherCAT specification – CANopen over EtherCAT (CoE) Communication Area as well as how the objects are mapped to the corresponding parameters of Pilz.

Index	Subindex	TxPDO	RxPDO	Name	Comment
1000 hex	0 hex	—	—	Device type	Constant value 20192 hex Bit 0 – 15: Device profile number, 192 hex = 402 Bit 16 – 23: Type, 2 hex = Servo drive Bit 24 – 31: Reserved
1001 hex	0 hex	—	—	Error register	
1008 hex	0 hex	—	—	Manufacturer device name	E50
1009 hex	0 hex	—	—	Manufacturer hardware version	E52[1]
100A hex	0 hex	—	—	Manufacturer software version	E52[3]
1018 hex				Identity object	Record with 4 elements
1018 hex	0 hex	—	—	Highest subindex supported	Constant value of 4 hex
1018 hex	1 hex	—	—	Vendor ID	manufacturer ID: B9 hex
1018 hex	2 hex	—	—	Product code	Nominal power in unit 0.1 kW
1018 hex	3 hex	—	—	Revision number	Software build number
1018 hex	4 hex	—	—	Serial number	E52[2]
1600 hex				1st RxPDO mapping parameter	Array with 24 elements
1600 hex	0 hex	—	✓	Number of mapped application objects in RxPDO	Constant value of 18 hex
1600 hex	1 hex – 18 hex	—	✓	Application objects	A225[0] – A225[23]
1601 hex				2nd RxPDO mapping parameter	Array with 24 elements

Intern

Index	Subindex	TxPDO	RxPDO	Name	Comment
1601 hex	0 hex	—	✓	Number of mapped application objects in RxPDO	Constant value of 18 hex
1601 hex	1 hex – 18 hex	—	✓	Application objects	A226[0] – A226[23]
1602 hex				3rd RxPDO mapping parameter	Array with 24 elements
1602 hex	0 hex	—	✓	Number of mapped application objects in RxPDO	Constant value of 18 hex
1602 hex	1 hex – 18 hex	—	✓	Application objects	A227[0] – A227[23]
1603 hex				4th RxPDO mapping parameter	Array with 24 elements
1603 hex	0 hex	—	—	Number of mapped application objects in RxPDO	Constant value of 18 hex
1603 hex	1 hex – 18 hex	—	—	Application objects	A228[0] – A228[23]
1A00 hex				1st TxPDO mapping parameter	Array with 24 elements
1A00 hex	0 hex	—	✓	Number of mapped application objects in TxPDO	Constant value of 18 hex
1A00 hex	1 hex – 18 hex	—	✓	Application objects	A233[0] - A233[23]
1A01 hex				2nd TxPDO mapping parameter	Array with 24 elements
1A01 hex	0 hex	—	✓	Number of mapped application objects in TxPDO	Constant value of 18 hex
1A01 hex	1 hex – 18 hex	—	✓	Application objects	A234[0] - A234[23]
1A02 hex				3rd TxPDO mapping parameter	Array with 24 elements
1A02 hex	0 hex	—	✓	Number of mapped application objects in TxPDO	Constant value of 18 hex
1A02 hex	1 hex – 18 hex	—	✓	Application objects	A235[0] - A235[23]
1A03 hex				4th TxPDO mapping parameter	Array with 24 elements
1A03 hex	0 hex	—	—	Number of mapped application objects in TxPDO	Constant value of 18 hex
1A03 hex	1 hex – 18 hex	—	—	Application objects	A236[0] - A236[23]
1C00 hex				Sync manager communication type	Record with 4 elements
1C00 hex	0 hex	—	—	Highest subindex supported	Constant value of 4 hex
1C00 hex	1 hex	—	—	Communication type sync manager 0	

Intern

Index	Subindex	TxPDO	RxPDO	Name	Comment
1C00 hex	2 hex	—	—	Communication type sync manager 1	
1C00 hex	3 hex	—	—	Communication type sync manager 2	
1C00 hex	4 hex	—	—	Communication type sync manager 3	
1C12 hex				Sync manager 2	Record with 4 elements
1C12 hex	0 hex	—	✓	Highest subindex supported	Constant value of 4 hex
1C12 hex	1 hex	—	✓	PDO receive assign 1st PDO	A252[0]
1C12 hex	2 hex	—	✓	PDO receive assign 2nd PDO	A252[1]
1C12 hex	3 hex	—	✓	PDO receive assign 3rd PDO	A252[2]
1C12 hex	4 hex	—	✓	PDO receive assign 4th PDO	A252[3]
1C13 hex				Sync manager 3	Record with 4 elements
1C13 hex	0 hex	—	✓	Highest subindex supported	Constant value of 4 hex
1C13 hex	1 hex	—	✓	PDO transmit assign 1st PDO	A253[0]
1C13 hex	2 hex	—	✓	PDO transmit assign 2nd PDO	A253[1]
1C13 hex	3 hex	—	✓	PDO transmit assign 3rd PDO	A253[2]
1C13 hex	4 hex	—	✓	PDO transmit assign 4th PDO	A253[3]
1C32 hex				Output SyncManager parameter	Record with 10 elements
1C32 hex	0 hex	—	—	Highest subindex supported	Constant value of 20 hex
1C32 hex	1 hex	—	—	Synchronization type	A264[0]
1C32 hex	2 hex	—	—	Cycle time	A264[1]
1C32 hex	3 hex	—	—	Shift time	A264[2]
1C32 hex	4 hex	—	—	Synchronization types supported	A264[3]
1C32 hex	5 hex	—	—	Minimum Cycle Time	A264[4]
1C32 hex	6 hex	—	—	Calc and Copy Time	A264[5]
1C32 hex	9 hex	—	—	Delay Time	A264[6]
1C32 hex	B hex	—	—	SM-Event missed Counter	A264[7]
1C32 hex	C hex	—	—	Cycle Time too small Counter	A264[8]
1C32 hex	20 hex	—	—	Sync Error	A264[9]

Intern

Index	Subindex	TxPDO	RxPDO	Name	Comment
1C33 hex				Input SyncManager parameter	Record with 10 elements
1C33 hex	0 hex	—	—	Highest subindex supported	Constant value of 20 hex
1C33 hex	1 hex	—	—	Synchronization type	A265[0]
1C33 hex	2 hex	—	—	Cycle time	A265[1]
1C33 hex	3 hex	—	—	Shift time	A265[2]
1C33 hex	4 hex	—	—	Synchronization types supported	A265[3]
1C33 hex	5 hex	—	—	Minimum Cycle Time	A265[4]
1C33 hex	6 hex	—	—	Calc and Copy Time	A265[5]
1C33 hex	9 hex	—	—	Delay Time	A265[6]
1C33 hex	B hex	—	—	SM-Event missed Counter	A265[7]
1C33 hex	C hex	—	—	Cycle Time too small Counter	A265[8]
1C33 hex	20 hex	—	—	Sync Error	A265[9]

Tab. 12: CiA 301 communication objects: 1000 hex – 1FFFF hex

10.1.2 ETG.1020 EtherCAT protocol enhancements

The following table contains the supported communication objects of the ETG.1020 EtherCAT Protocol Enhancements profile and their mapping to the corresponding parameters of Pilz. The listed extensions are part of the EtherCAT specification and may become part of the ETG.1000 series in the future.

Index	Subindex	TxPDO	RxPDO	Name	Comment
10F3 hex				Diagnosis History Object	
10F3 hex	1 hex	—	—	Maximum messages	
10F3 hex	2 hex	—	—	Newest Message	
10F3 hex	3 hex	—	—	Newest Acknowledged Message	
10F3 hex	4 hex	✓	—	New Messages Available	A250
10F3 hex	5 hex	—	—	Flags	
10F3 hex	6 hex	—	—	Diagnosis message	

Tab. 13: CiA 301 communication object: 10F3 hex

10.1.3 ETG.5000.1 Modular Device Profile: F000 hex – FFFF hex

The following table includes the supported communication objects of the standardized profile ETG.5000.1 Modular Device Profile.

Index	Subindex	TxPDO	RxPDO	Name	Comment
F050 hex				Detected module ident list	Record with 2 elements
F050 hex	0 hex	—	—	Highest subindex supported	Constant value of 2 hex
F050 hex	1 hex	—	—	Module ident axis A	
F050 hex	2 hex	—	—	Module ident axis B	

Tab. 14: ETG.5000.1 communication objects: F000 hex – FFFF hex

10.1.4 Manufacturer-specific parameters: 2000 hex – 53FF hex

Index, subindex and calculation example (axis A)



Information

Index and Subindex must be specified in the format required by the controller.



Information

The calculation described below is only valid for converting the manufacturer-specific parameters.

The index is calculated from the group and line of the parameter according to the following formula:
 $\text{Index} = 8192 + (\text{number of the group} \times 512) + \text{number of the line}$

The subindex for simple parameters is always 0.

For array or record parameters, the subindex for EtherCAT Rx corresponds to the element number of the parameter.

The subindex for EtherCAT Rx SDO Info corresponds to the element number of the parameter + 1 for array or record parameters.

	Simple parameters	Array or record parameter
Index	$8192 + (\text{number of the group} \times 512) + \text{number of the line}$	
Subindex for EtherCAT Rx	0	Element number
Subindex for EtherCAT Rx SDO Info	0	Element number + 1

Tab. 15: Index and subindex for manufacturer-specific parameters

Calculation example

Calculation for parameter E200[0]:

Number of the group = 4

Number of the line = 200

Index = $8192 + (4 \times 512) + 200 = 10440 = 28C8$ hex

Subindex for EtherCAT Rx = 0 = 0 hex

Subindex for EtherCAT Rx SDO Info = 1 = 1 hex

Communication objects (axis A)

The following table includes the manufacturer-specific communication objects of axis A and their mapping to the corresponding parameters of Pilz.

For information on the manufacturer-specific communication objects of axis B, see [Manufacturer-specific parameters: A000 hex – D3FF hex \[116\]](#). The axes are differentiated by an offset of 8000 hex.

Index	Group	Number	Parameters
2000 hex – 21FF hex	A: Drive controller	0	A00 – A511
2200 hex – 23FF hex	B: Motor	1	B00 – B511
2400 hex – 25FF hex	C: Machine	2	C00 – C511
2600 hex – 27FF hex	D: Set value	3	D00 – D511
2800 hex – 29FF hex	E: Show	4	E00 – E511
2A00 hex – 2BFF hex	F: Terminals	5	F00 – F511
2C00 hex – 2DFF hex	G: Technology	6	G00 – G511
2E00 hex – 2FFF hex	H: Encoders	7	H00 – H511
3000 hex – 31FF hex	I: Motion	8	I00 – I511
3200 hex – 33FF hex	J: Motion blocks	9	J00 – J511
3400 hex – 35FF hex	K: Control panel	10	K00 – K511
3600 hex – 37FF hex	M: Profile	12	M00 – M511
3E00 hex – 3FFF hex	P: Customer-specific parameters	15	P00 – P511
4000 hex – 41FF hex	Q: Customer-specific parameters, instance-dependent	16	Q00 – Q511
4200 hex – 43FF hex	R: Production data	17	R00 – R511
4400 hex – 45FF hex	S: Safety	18	S00 – S511
4600 hex – 47FF hex	T: Scope	19	T00 – T511
4800 hex – 49FF hex	U: Protection functions	20	U00 – U511
5200 hex – 53FF hex	Z: Fault counter	25	Z00 – Z511

Tab. 16: Manufacturer-specific communication objects: 2000 hex – 53FF hex

10.1.5 Manufacturer-specific parameters: A000 hex – D3FF hex

Index, subindex, and calculation example (axis B)



Information

Index and Subindex must be specified in the format required by the controller.



Information

The calculation described below is only valid for converting the manufacturer-specific parameters.

The index is calculated from the group and line of the parameter according to the following formula:
 $\text{Index} = 40960 + (\text{number of the group} \times 512) + \text{number of the line}$

The subindex for simple parameters is always 0.

For array or record parameters, the subindex for EtherCAT Rx corresponds to the element number of the parameter.

The subindex for EtherCAT Rx SDO Info corresponds to the element number of the parameter + 1 for array or record parameters.

	Simple parameters	Array or record parameter
Index	$40960 + (\text{number of the group} \times 512) + \text{number of the line}$	
Subindex for EtherCAT Rx	0	Element number
Subindex for EtherCAT Rx SDO Info	0	Element number + 1

Tab. 17: Index and subindex for manufacturer-specific parameters

Calculation example

Calculation for parameter E200[0]:

Number of the group = 4

Number of the line = 200

Index = $40960 + (4 \times 512) + 200 = 43208 = \text{A8C8 hex}$

Subindex for EtherCAT Rx = 0 = 0 hex

Subindex for EtherCAT Rx SDO Info = 1 = 1 hex

Intern

Communication objects (axis B)

The following table includes the vendor-specific communication objects supported by axis B and how they are mapped to the corresponding parameters of Pilz.

Index	Group	Number	Parameters
A000 hex – A1FF hex	A: Drive controller	0	A00 – A511
A200 hex – A3FF hex	B: Motor	1	B00 – B511
A400 hex – A5FF hex	C: Machine	2	C00 – C511
A600 hex – A7FF hex	D: Set value	3	D00 – D511
A800 hex – A9FF hex	E: Show	4	E00 – E511
AA00 hex – ABFF hex	F: Terminals	5	F00 – F511
AC00 hex – ADFF hex	G: Technology	6	G00 – G511
AE00 hex – AFFF hex	H: Encoders	7	H00 – H511
B000 hex – B1FF hex	I: Motion	8	I00 – I511
B200 hex – B3FF hex	J: Motion blocks	9	J00 – J511
B400 hex – B5FF hex	K: Control panel	10	K00 – K511
B600 hex – B7FF hex	M: Profile	12	M00 – M511
BE00 hex – BFFF hex	P: Customer-specific parameters	15	P00 – P511
C000 hex – C1FF hex	Q: Customer-specific parameters, instance-dependent	16	Q00 – Q511
C200 hex – C3FF hex	R: Production data	17	R00 – R511
C400 hex – C5FF hex	S: Safety	18	S00 – S511
C600 hex – C7FF hex	T: Scope	19	T00 – T511
C800 hex – C9FF hex	U: Protection functions	20	U00 – U511
D000 hex – D1FF hex	Z: Fault counter	25	Z00 – Z511

Tab. 18: Vendor-specific communication objects: A000 hex – D3FF hex

10.2 SDO transmission: Error codes

If an SDO frame cannot be processed, the slave sends an SDO Abort Domain Transfer and outputs one of the following errors – along with the error class, error code and additional information – over the Abort SDO Transfer Protocol in case of an error.

Error class	Error code	Additional code	Meaning
5 hex	3 hex	0 hex	Toggle bit not changed
5 hex	4 hex	0 hex	SDO protocol timeout expired
5 hex	4 hex	1 hex	SDO command specifier invalid or unknown
5 hex	4 hex	5 hex	Memory not sufficient
6 hex	1 hex	0 hex	Access to object is not supported
6 hex	1 hex	1 hex	Read attempt on a write-only parameter
6 hex	1 hex	2 hex	Write attempt on a read-only parameter
6 hex	2 hex	0 hex	Object not present in the object directory
6 hex	4 hex	41 hex	Object cannot be mapped to PDO
6 hex	4 hex	42 hex	Number and/or length of the object to be transmitted exceeds PDO length
6 hex	4 hex	43 hex	General parameter incompatibility
6 hex	4 hex	47 hex	General internal device incompatibility
6 hex	6 hex	0 hex	Access terminated due to hardware error
6 hex	7 hex	10 hex	Incorrect data type or parameter length
6 hex	7 hex	12 hex	Incorrect data type or parameter length too long
6 hex	7 hex	13 hex	Incorrect data type or parameter length too short
6 hex	9 hex	11 hex	Subindex not available
6 hex	9 hex	30 hex	Invalid parameter value (write process)
6 hex	9 hex	31 hex	Parameter value too large
6 hex	9 hex	32 hex	Parameter value too small
6 hex	9 hex	36 hex	Maximum value is less than minimum value
8 hex	0 hex	0 hex	General SDO error
8 hex	0 hex	20 hex	Access not possible
8 hex	0 hex	21 hex	Access not possible due to local controller
8 hex	0 hex	22 hex	Access not possible in current device state
8 hex	0 hex	23 hex	Dynamic generation of the object directory failed or no object directory available

Tab. 19: SDO: Error codes

10.3 EMCY message: Incorrect state transition error codes

Error code	Meaning
A000 hex	Incorrect transition from pre-operational to safe-operational state
A001 hex	Incorrect transition from safe-operational to pre-operational state

Tab. 20: EMCY: Transition error codes

Error register specifies the state of the EtherCAT state machine at the time the EMCY is sent.

Error register	State
1 hex	Initializing
2 hex	Pre-operational
3 hex	Safe-operational
4 hex	Operational

Tab. 21: EMCY: Error codes for transitions, error register (state of the EtherCAT state machine)

Diag code provides information about the cause of the error.

Diag code	Error cause	
0 hex	SyncManager at address that is not permitted	SyncManager 0 (write mailbox data from frame in mailbox)
1 hex	SyncManager at address that is not permitted	
2 hex	PDO length not correct	
3 hex	SyncManager parameterized incorrectly	
4 hex	SyncManager at address that is not permitted	SyncManager 1 (write mailbox data from mailbox to frame)
5 hex	SyncManager at address that is not permitted	
6 hex	PDO length not correct	
7 hex	SyncManager parameterized incorrectly	
8 hex	SyncManager at address that is not permitted	SyncManager 2 (write process data from frame in process data memory)
9 hex	SyncManager at address that is not permitted	
A hex	PDO length not correct	
B hex	SyncManager parameterized incorrectly	
C hex	SyncManager at address that is not permitted	SyncManager 3 (write process data to frame from process data memory)
D hex	SyncManager at address that is not permitted	
E hex	PDO length not correct	
F hex	SyncManager parameterized incorrectly	

Tab. 22: EMCY: Error codes for transitions, Diag code (cause of error)

10.4 EMCY message: Device fault error codes

Error code	Error register	Event (E82)
0 hex: No error	0 hex: No error	30: Inactive
2110 hex: Short-circuit earth	2 hex: Current	31: Short/ground
2230 hex: Intern short-circuit earth	2 hex: Current	32: Short/ground internal
2310 hex: Continuous overcurrent	2 hex: Current	33: Overcurrent
3110 hex: Mains overvoltage	4 hex: Voltage	36: High voltage
3120 hex: Mains undervoltage	4 hex: Voltage	46: Low voltage
3130 hex: Phase failure	1 hex: Generic error	83: Failure of one/ all phases (mains)
3180 hex: Mains failure	1 hex: Generic error	84: Drop in network voltage when power section active
4210 hex: Temperature	8 hex: Temperature	38: Temperature drive controller sensor
4280 hex: Temperature device I ² t	8 hex: Temperature	39: Overtemperature drive controller i ² t or 59: Overtemperature drive controller i ² t
4310 hex: Temperature drive	8 hex: Temperature	41: Temp.MotorTMP
4380 hex: Temperature drive I ² t	8 hex: Temperature	45: Overtemp.motor i ² t
5200 hex: Device hardware	1 hex: Generic error	34: Hardware fault
6010 hex: Internal software	1 hex: Generic error	35: Watchdog, 57: Runtime requirement or 71: Firmware
6320 hex: Loss of parameters	1 hex: Generic error	40: Invalid data or 70: Parameter consistency
6330 hex: Unknown Lean motor type	1 hex: Generic error	86: Unknown LeanMotor
7110 hex: Brake chopper	1 hex: Generic error	48: Brake release monitoring, 49: Brake, 72: Brake test timeout or 73: Axis 2 brake test timeout
	8 hex: Temperature	42: TempBrakeRes
7120 hex: Motor	1 hex: Generic error	69: Motor connection or 81: Motor allocation
7303 hex: Resolver 1 fault	1 hex: Generic error	37: Motor encoder
7304 hex: Resolver 2 fault	1 hex: Generic error	76: Position encoder, 77: Master encoder or 79: Motor/position encoder plausibility
7500 hex: Communication	10 hex: Communication	52: Communication
7580 hex: Communication control panel	1 hex: Generic error	88: Control panel
8311 hex: Excess torque	1 hex: Generic error	47: Torque/force-max. limit
8400 hex: Velocity speed control	1 hex: Generic error	56: Overspeed

Error code	Error register	Event (E82)
8500 hex: Position control	1 hex: Generic error	53: Limit switch
8510 hex: Excessive reference position jump	1 hex: Generic error	85: Excessive jump in reference value
8600 hex: Positioning controller	1 hex: Generic error	51: Virtual master software limit switch
8611 hex: Following error	1 hex: Generic error	54: Following error
8612 hex: Reference limit	1 hex: Generic error	78: Position limit cyclic
FF00 – FF07 hex: Manufacturer specific error	1 hex: Generic error	60: Application event 0 – 67: Application event 7
FF09 hex: Manufacturer specific error	1 hex: Generic error	44: External fault 1
FF0A hex: Manufacturer specific error	1 hex: Generic error	68: External fault 2

Tab. 23: EMCY: Device fault error codes

10.5 EMCY message: EoE-error error codes

Error code	Error register	Meaning
0 hex: No error	0 hex: No error	No error occurred
FFF0 hex: Manufacturer specific error	23 hex: EoE address occupied	IP address cannot be opened (IP address does not match the subnet mask, IP address already assigned, etc.)

Tab. 24: EMCY: Device fault error codes

10.6 Simple Network Time Protocol (SNTP)

An SNTP client is implemented in the drive controller in accordance with RFC4330. This client sets the internal clock of the drive controller to the current time, which it obtains from an external time server. The internal clock runs with an (inaccurate) controllable internal clock in the drive controller. As a result, the time is queried from the server in intervals. It is compared with the internal time and the clock for the internal clock is readjusted accordingly. You define the settings in parameter A199.

Two NTP servers can be defined as time sources, both of which are used as possible time servers. In the case of data traffic via the service interface, the computer through which the drive controller is connected via DriveControlSuite automatically counts as one of the possible time servers. The time servers must be accessible either via EoE, service interface X9 or terminals X200 and X201. Make sure that the time server is accessible from the drive controller. It may be necessary to set the gateway parameter A175 accordingly.

The time is always requested from the same NTP server and then cyclically repeated by the server to track the synchronicity control loop. If the current server fails, the next one in the list is used. A server that was once active is discarded only in case of connection failure to this server or server unavailability.

After the drive controller is switched on, it takes a random time of 1 to 5 minutes (in accordance with RFC4330) until the SNTP client sends its first request to one of the time servers.

The cyclic repetition of a request occurs approximately every 5 to 6 hours.

10.6.1 Setting up time service on the computer

On a Windows PC with DriveControlSuite, you can use the registry editor to set up the time service. You must stop the time server in advance and restart it after changing the registry. Proceed as follows:

1. Open the command prompt, e.g. as follows:
 - 1.1. Use the [WINDOWS]+[R] key combination to open the Run dialog box.
 - 1.2. Enter the command `cmd` and confirm with OK.
⇒ The command prompt opens.
2. Stop the time server by using the command `net stop w32time`.
3. Open the registry editor, e.g. as follows:
 - 3.1. Use the [WINDOWS]+[R] key combination to open the Run dialog box.
 - 3.2. Enter the command `regedit` and confirm with OK.
⇒ The registry editor opens.
4. Select HKEY_LOCAL_MACHINE > SYSTEM > CurrentControlSet > Services > W32Time > TimeProvider > NtpServer.
5. Set Enable to the value 1 and confirm with OK.
6. Close the registry editor.
7. Open the command prompt again.
8. Start the time server in the command prompt by using the command `net start w32time`.
⇒ The time service is set up on the PC.

Automation using a command script

If you want to change the registry on the PC via a command script, create a *.reg file by creating an empty text file and renaming the file extension. Then open the file and enter the following content:

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\W32Time\TimeProviders\ntpserver] "Enabled"=dword:00000001
```

Run the file in the command line of the command prompt.

More commands

If you want to query the status on the current PC, use the following command in the command prompt:

```
w32tm /query /status
```

To query the IP address by PC name, use the following command at the command prompt:

```
nslookup <name>
```

Example:

```
nslookup ptbtime1.ptb.de
Name: ptbtime1.ptb.de
Addresses: 2001:638:610:be01::108 192.53.103.108
```

The IP address is: 192.53.103.108.

10.7 Detailed information

The documentation listed below provides you with further relevant information on the drive controllers. The current status of the documentation can be found at:

<https://www.pilz.com/en-INT>.

Title	Documentation	Contents	ID
PMC SC6 drive controller	Manual	System design, technical data, project configuration, storage, installation, connection, commissioning, operation, service, diagnostics	1005343
Multi-axis drive system with PMC SI6 and PMC PS6	Manual	System design, technical data, project configuration, storage, installation, connection, commissioning, operation, service, diagnostics	1005342
CiA 402 application – PMC SC6, PMC SI6	Manual	Project planning, configuration, parameterization, function test, detailed information	1005347
PMC SY6 safety technology – STO and SS1 via FSoE	Manual	Technical data, installation, commissioning, diagnostics, detailed information	1005345

Additional information and sources that form the basis of this documentation or are referenced by the documentation:

Beckhoff Automation GmbH & Co. KG (publisher): *EtherCAT System Documentation*. Version 5.1. Verl, 2016.

A free basic version of the TwinCAT 3 automation software is available at <https://www.beckhoff.com/en-us/products/automation/twincat/te1xxx-twincat-3-engineering/te1000.html>.

EtherCAT Technology Group (ETG), 2012. *ETG.1300: EtherCAT Indicator and Labeling*. ETG.1300 S (R) V1.1.0. Specification. 2012-01-27.

10.8 Abbreviations

Abbreviation	Meaning
AT	Acknowledge Telegram
CiA	CAN in Automation
CNC	Computerized Numerical Control
CoE	CANopen over EtherCAT
EMCY	Emergency
EMC	Electromagnetic Compatibility
EoE	Ethernet over EtherCAT
ESC	EtherCAT Slave Controller
ESI	EtherCAT Slave Information
ESM	EtherCAT State Machine
ETG	EtherCAT Technology Group
EtherCAT	Ethernet for Control Automation Technology
FTP	File Transfer Protocol
HTTP	Hypertext Transfer Protocol
I/O	Input/Output
IP	Internet Protocol
LSB	Least Significant Bit
LSW	Least Significant Word
MDT	Master Data Telegram
MSB	Most Significant Bit
MSW	Most Significant Word
NC	Numerical Control
NTP	Network Time Protocol
PDO	Process Data Objects
RFC	Request For Comments
RxPDO	Receive PDO (receive process data)
S/FTP	Screened/Foiled Twisted Pair
SDO	Service Data Objects
SF/FTP	Screened Foiled/Foiled Twisted Pair
SF/UTP	Screened Foiled/Unshielded Twisted Pair
SNTP	Simple Network Time Protocol
PLC	Programmable Logic Controller
SYNC	Synchronization
TCP	Transmission Control Protocol
TxPDO	Transmit PDO (transmit process data)
UDP	User Data Protocol

Broadcast domain

Logical grouping of network devices within a local network that reaches all nodes via broadcast.

CiA 402

Application of the commissioning software, which includes both the controller-based and drive-based operating modes (csp, csv, cst, ip, pp, vl, pv, pt).

CiA 402 HiRes Motion

Application of the commissioning software, which includes both the controller-based and drive-based operating modes (csp, csv, cst, ip, pp, vl, pv, pt). The interface for the controller is tailored to the HiRes CODESYS device driver, meaning that set and actual values are represented and transmitted in units that can be defined by the user.

CoE

EtherCAT protocol that provides CANopen-compliant communication mechanisms, enabling the use of the entire CANopen profile family over EtherCAT.

DC-Sync

Also: Synchronization using distributed clocks. Method for EtherCAT network synchronization. Each EtherCAT slave with distributed clocks functionality has a local clock. Normally, the time from the first DC-Sync-capable EtherCAT slave downstream of the master in the network serves as the reference time: Both the master and the slaves synchronize with this reference clock when prompted by the master. The event belonging to a synchronization is labeled as a Sync 0 signal and is generated cyclically by the SyncManager of each slave.

EMCY

Communication objects in a CANopen or EtherCAT network that, in the event of incorrect state transitions or device-internal errors, transmit the associated error codes and causes.

EoE

Acyclical EtherCAT protocol that enables any data traffic between EoE-capable nodes of an EtherCAT network. The Ethernet frames are tunneled through the EtherCAT protocol; the EtherCAT real-time properties remain unimpaired. The EtherCAT master is used as a gateway to the Ethernet network.

IPv4 limited broadcast

Type of broadcast in a network with IPv4 (Internet Protocol version 4). The IP address 255.255.255.255 is entered as the destination. The content of the broadcast is not forwarded by a router, which limits it to the local network.

Jitter

Generally refers to slight fluctuations in the clock when transmitting digital signals or a slight accuracy fluctuation in the transmission clock. In network technology, it is also variance in the runtime of data packets.

Network Time Protocol (NTP)

Standard for synchronizing clocks in computer systems via packet-based communication networks. The protocol uses the connectionless transport protocol UDP or the connection-based TCP. It is specifically designed to provide reliable timing over networks with variable packet runtime.

Process Data Objects (PDO)

Communication objects in a CANopen or EtherCAT network that transmit data such as set and actual values, control commands or status information based on events or objectives, in cycles or in real time on request. PDOs are generally exchanged over the process data channel with high priority. Depending on the view of the respective node, a distinction is made between receive PDOs (RxPDO) and transmit PDOs (TxPDO).

RFC

Proposed and published Internet standards, reviewed by the Internet Engineering Task Force (IETF), as consensus-building body that facilitates discussion, and eventually a new standard is established.

SDO

Communication objects in a CANopen or EtherCAT network that grant access to the object directory and enable device configuration. SDOs are transmitted over the mailbox channel acyclically during ongoing cyclical CANopen or EtherCAT operation.

SDO Info

Service that enables the EtherCAT controller to read out objects from the drive controller. During the read-out, all relevant object properties, such as data type, write and read access rights as well as mapping capabilities, are transmitted to the controller.

Simple Network Time Protocol (SNTP)

Simplified version of the Network Time Protocol (NTP). The structure of the protocol is identical to that of NTP. SNTP clients can thus also obtain the time from NTP servers. The main difference lies in the algorithms used for time synchronization. While NTP usually uses multiple time servers for time synchronization, SNTP uses only one time server.

SM-Sync

Also: Synchronization using SyncManager event. Method for EtherCAT network synchronization where the EtherCAT slaves synchronize with an event from incoming data.

Startup list

Predefined list of CiA objects that is processed every time EtherCAT is started. The values it contains are sent to the corresponding EtherCAT slave with the defined state change.

Synchronization

Time synchronization of EtherCAT network nodes that allows the EtherCAT master and slaves to work synchronously with each other in the same cycle. EtherCAT provides two different methods for precisely synchronizing the master and slaves: a SyncManager event (SM-Sync) and distributed clocks (DC-Sync). If the master and slaves are not synchronized, they are in the FreeRun state.

Template

In the context of the DriveControlSuite commissioning software, a template for graphical programming. This template can be selected in the configuration dialog for device control, communication (fieldbus) or application in a certain version.

Fig. 1	EtherCAT: Network structure, using the PMC SI6 series as an example	12
Fig. 2	DS6: Program interface	15
Fig. 3	DriveControlSuite: Navigation using text links and symbols	17
Fig. 4	TwinCAT 3 (TwinCAT XAE): Program interface.....	18
Fig. 5	LEDs for the EtherCAT state	58
Fig. 6	LEDs for the state of the EtherCAT network connection.....	59
Fig. 7	EtherCAT: Communication protocols	67
Fig. 8	Network overview: Topology 1	69
Fig. 9	Network overview: Topology 2	70
Fig. 10	EtherCAT state machine: States and state changes.....	82
Fig. 11	SM-Sync: Synchronization using SyncManager event.....	85
Fig. 12	EtherCAT: Distributed clocks	86
Fig. 13	TwinCAT 3: DC-Sync – Settings	87
Fig. 14	TwinCAT 3: DC-Sync – Unstable synchronization, cycle time < 1 ms	89
Fig. 15	TwinCAT 3: DC-Sync – Stable synchronization, cycle time < 1 ms	90
Fig. 16	CODESYS V3: DC-Sync – Settings	93
Fig. 17	CODESYS V3: DC-Sync – Unstable synchronization, cycle time < 1 ms	95
Fig. 18	CODESYS V3: DC-Sync – Stable synchronization, cycle time < 1 ms	96
Fig. 19	Overview of fieldbus scaling.....	104

Tab. 1	X200 and X201 connection description.....	13
Tab. 2	Parameter groups.....	19
Tab. 3	Parameters: data types, parameter types, possible values.....	20
Tab. 4	Parameter types.....	21
Tab. 5	Meaning of the red LED (error).....	58
Tab. 6	Meaning of the green LED (Run).....	58
Tab. 7	Meaning of the green LEDs (LA).....	59
Tab. 8	Event 52 – Causes and actions.....	61
Tab. 9	Cycle times.....	101
Tab. 10	Selecting and executing an action.....	103
Tab. 11	Fieldbus scaling for integer: Parameters for defining the decimal places.....	104
Tab. 12	CiA 301 communication objects: 1000 hex – 1FFFF hex.....	109
Tab. 13	CiA 301 communication object: 10F3 hex.....	112
Tab. 14	ETG.5000.1 communication objects: F000 hex – FFFF hex.....	113
Tab. 15	Index and subindex for manufacturer-specific parameters.....	114
Tab. 16	Manufacturer-specific communication objects: 2000 hex – 53FF hex.....	115
Tab. 17	Index and subindex for manufacturer-specific parameters.....	116
Tab. 18	Vendor-specific communication objects: A000 hex – D3FF hex.....	117
Tab. 19	SDO: Error codes.....	118
Tab. 20	EMCY: Transition error codes.....	119
Tab. 21	EMCY: Error codes for transitions, error register (state of the EtherCAT state machine).....	119
Tab. 22	EMCY: Error codes for transitions, Diag code (cause of error).....	119
Tab. 23	EMCY: Device fault error codes.....	120
Tab. 24	EMCY: Device fault error codes.....	121

Intern

► Support

Technical support is available from Pilz round the clock.

Americas

Brazil

+55 11 97569-2804

Canada

+1 888 315 7459

Mexico

+52 55 5572 1300

USA (toll-free)

+1 877-PILZUSA (745-9872)

Asia

China

+86 400-088-3566

Japan

+81 45 471-2281

South Korea

+82 31 778 3300

Australia and Oceania

Australia

+61 3 95600621

New Zealand

+64 9 6345350

Europe

Austria

+43 1 7986263-444

Belgium, Luxembourg

+32 9 3217570

France

+33 3 88104003

Germany

+49 711 3409-444

Ireland

+353 21 4804983

Italy, Malta

+39 0362 1826711

Scandinavia

+45 74436332

Spain

+34 938497433

Switzerland

+41 62 88979-32

The Netherlands

+31 347 320477

Türkiye

+90 216 5775552

United Kingdom

+44 1536 462203

You can reach our international hotline on:

+49 711 3409-222

support@pilz.com

Pilz develops environmentally-friendly products using ecological materials and energy-saving technologies. Offices and production facilities are ecologically designed, environmentally-aware and energy-saving. So Pilz offers sustainability, plus the security of using energy-efficient products and environmentally-friendly solutions.



CECE[®], CHRE[®], CMSE[®], INDUSTRIAL P[®], Leansafe[®], Myzel[®], PAS4000[®], PASscal[®], PASconfig[®], Pilz[®], PIT[®], PMCprimo[®], PMCprotego[®], PMCiendo[®], PMD[®], PME[®], PNOZ[®], Primo[®], PSEN[®], PSEN[®], PSS[®], PVIS[®], SafetyBUS p[®], SafetyNET p[®], THE SPIRIT OF SAFETY[®] are registered and protected trademarks of Pilz GmbH & Co. KG in some countries. We would point out that product features may vary from the details stated in this document, depending on the status at the time of publication and the scope of the equipment. We accept no responsibility for the validity, accuracy and entirety of the text and graphics presented in this information. Please contact our Technical Support if you have any questions.

We are represented internationally. Please refer to our homepage www.pilz.com for further details or contact our headquarters.

Headquarters: Pilz GmbH & Co. KG, Felix-Wankel-Straße 2, 73760 Ostfildern, Germany
Telephone: +49 711 3409-0, E-Mail: info@pilz.com, Internet: www.pilz.com