rexroth
A Bosch Company

Application Manual

EtherCAT Master App

EtherCAT Master for ctrlX CORE 01VRS

R911403773, Edition 08

Liability

The specified data is intended for product description purposes only and shall not be deemed to be a guaranteed characteristic unless expressly stipulated in the contract. All rights are reserved with respect to the content of this documentation and the availability of the product.

DOK-XCORE*-ETHERCATV01-AP08-EN-P

574fb33b3d23cf8e0a347e8830009366, 8, en_US

Table of contents

1 About this documentation

Editions of this documentation

Edition	Release date	Note
01	2020-06	First edition for EtherCAT Master app (Version ECM-V-0102 & ECM-V-0102)
02	2020-12	Edition for EtherCAT Master app (Version ECM-V-0106) Revision: • ➦Terms and abbreviations • ➦EtherCAT Master – Features • ➦Configuring the EtherCAT master • ➦Configuring the EtherCAT field bus devices • ➦CoE SDO abort codes • ➦FoE error codes • ➦SoE error codes Removed: • Chapter 11 “Field bus-comprehensive features”
03	2021-06	Edition for EtherCAT Master app (Version ECM-V-0108) Revision: • ➦Technical data
04	2021-08	Edition for EtherCAT Master app (Version ECM-V-0110) Revision: • ➦Technical data
05	2022-02	Edition for EtherCAT Master app (Version ECM-V-0112) New: • Windows "EtherCAT Slave Statistics" (in version ECM-V-0116 converted to tab, see edition 06)
06	2022-08	Edition for EtherCAT Master app (Version ECM-V-0116) New: • ➦Window – “EtherCAT Master” • ➦Window – “EtherCAT Master” Detailed view • ➦Tab – “Slaves” • ➦Tab – “Distributed Clocks” • ➦Tab – “Slave statistics” Revision: • ➦EtherCAT Master – Features

Edition	Release date	Note
07	2023-01	Edition for EtherCAT Master app (Version ECM-V-0118) New: • ➦ Tab – “Master statistics” • ➦ Tab – “General” (EtherCAT Slave) • ➦ Tab – “AoE” (EtherCAT master instance) • ➦ Tab – “EoE” (EtherCAT master instance) • ➦ Header – “EtherCAT Master” • ➦ Header – “EtherCAT slave” Revision: • ➦ EtherCAT Master – Features
08	2023-05	Edition for EtherCAT Master app (Version ECM-V-0120) Revision: • ➦ EtherCAT Master – Features • ➦ General • ➦ WorkingCounter • ➦ Tab – “Distributed Clocks”

2 Important directions on use

2.1 Intended use

2.1.1 Introduction

Rexroth products are developed and manufactured to the state-of-the-art.

The products are tested prior to delivery to ensure operational safety and reliability.

⚠ WARNING	**Personal injury and damage to property due to incorrect use of products!** The products may only be used as intended. Failure to use the products as intended may cause situations resulting in property damage and personal injury.

NOTICE	**Damages resulting from unintended use** Rexroth As the manufacturer does not assume any warranty, liability or compensatory claims for damages resulting from unintended use of the products. The user alone shall bear the risks of an unintended use of the products. Before using Rexroth products, make sure that all the prerequisites for an intended use of the products are met: – Personnel that in any way, shape or form uses Rexroth products must first read and understand the relevant safety instructions and be familiar with their intended use – Leave hardware products in their original state, i.e., do not make any structural modifications. It its not permitted to decompile software products or alter source codes – Do not install damaged or defective products or commission them – It has to be ensured that the products have been installed as described in the relevant documentation

2.1.2 Areas of use and application

Products of the ctrlX series are suitable for Motion/Logic applications.

NOTICE	Products of the ctrlX series may only be used with the accessories, mounting parts, and other components specified in this documentation. Components that are not expressly mentioned must neither be attached nor connected. The same applies to cables and lines. Only to be operated with the hardware component configurations and combinations expressly specified and with the software and firmware specified in the corresponding documentations and functional descriptions.

Products of the ctrlX series are suitable for single-axis as well as for multi-axis drive and control tasks. Device types with different equipment and interfaces are available for using the system in specific applications.

Typical areas of application:

- Building automation
- IoT and Security Gateway or Device
- Handling & Robotic

Controls of the ctrlX CORE series may only be operated under the mounting and installation conditions, in the position of normal use and under the ambient conditions (temperature, degree of protection, humidity, EMC, etc.) specified in the related documentations.

2.2 Unintended use

"Unintended use" refers to using the ctrlX products outside of the above-mentioned areas of application or under operating conditions and technical data other than described and specified in the documentation.

ctrlX products must not be used if they are exposed to following conditions:

- Operating conditions that do not meet the specified ambient conditions. Operation under water, under extreme temperature fluctuations or under extreme maximum temperatures is prohibited
- Applications that have not been expressly authorized by Rexroth

3 Safety instructions

The Safety instructions contained in the available application documentation feature specific signal words (DANGER, WARNING, CAUTION or NOTICE) and, where required, a safety alert symbol (in accordance with ANSI Z535.6-2006).

The signal word is meant to draw the reader's attention to the safety instruction and identifies the hazard severity.

The safety alert symbol (a triangle with an exclamation point), which precedes the signal words DANGER, WARNING and CAUTION, is used to alert the reader to personal injury hazards.

The Safety instructions in this documentation are designed as follows:

⚠ DANGER	In case of non-compliance with this safety instruction, death or serious injury **will** occur.
⚠ WARNING	In case of non-compliance with this safety instruction, death or serious injury **could** occur.
⚠ CAUTION	In case of non-compliance with this safety instruction, minor or moderate injury could occur.
NOTICE	In case of non-compliance with this safety instruction, property damage could occur.

4 Introduction and overview

4.1 EtherCAT Master app – Basics

Install the “EtherCAT Master App” to add the EtherCAT Master functionality to the ctrlX CORE control:

- EtherCAT stack to operate the field bus on the control
- For additions on the field bus configuration and to display or control the operating state in the ctrlX CORE interface, refer to ➥ Chapter 8.1.1 Side navigation – node EtherCAT Master on page 43

License

The following license is required to operate the EtherCAT Master on the ctrlX CORE control:

Type code	Part number
SWL-XC*-ECM-ETHERCATMAS**-BANN	R911400508

Purchasing a license

➥ In the ctrlX App Store

Configuring the EtherCAT Slaves

Configure the EtherCAT Slaves in the software tool ctrlX I/O Engineering included in the ctrlX WORKS installation (optional installation on an Engineering PC).

Related topics

➥ App installation
➥ Side navigation – node EtherCAT Master
➥ Window – “EtherCAT Master”
➥ Window – “EtherCAT Master” Detailed view

Web links

➥ ctrlX CORE - Community
➥ ctrlX CORE - How to
➥ ctrlX CORE - Forum

4.2 Terms and abbreviations

“EtherCAT”	“Ethernet for Control Automation Technology” EtherCAT® is a registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany EtherCAT is an open, internationally standardized standard and developed further by the EtherCAT Technology Group (ETG).
“EC/ECAT” “ECM”	Abbreviations for EtherCAT (EC/ECAT) or EtherCAT master (ECM)
“ENI”	“EtherCAT Network Information”: Description of the bus configuration as XML file
“ESI”	“EtherCAT Slave Information”: Device description file as XML file
“ESC”	“EtherCAT slave controller”: Communication interface integrated into hardware (ASIC, FPGA) or as software solution
“ETG”	“EtherCAT Technology Group”: User organization

“EAP”	“EtherCAT Automation Protocol”: Protocol for networking on control level
“DC”	“Distributed Clocks” The procedure "Distributed Clocks" is used to synchronize bus devices. The Rexroth EtherCAT master supports different synchronization procedures.
“FSoE”	“FailSafe over EtherCAT”: Safety protocol for EtherCAT
“HC”	“Hot Connect”: Connecting and disconnecting slaves at runtime
“HC group”	“Hot Connect Group”: Slaves can be combined to one group. The group can be declared as optional (flexible topologies).
“Mailbox”	Acyclic communication over EtherCAT
“CoE”	“CAN application protocol over EtherCAT” (mailbox)
“SoE”	“Servo drive profile over EtherCAT”: Adapted Sercos II profile for servo drives (mailbox)
“FoE”	“File access over EtherCAT” (mailbox)
“EoE”	“Ethernet over EtherCAT”: Ethernet communication, e.g. TCP/IP (mailbox)
“VoE”	“Vendor-specific protocol over EtherCAT”: Manufacturer-specific mailbox protocols
“AoE”	“Automation Device Specification (ADS) over EtherCAT”: Device- and field bus-dependent interface from the Beckhoff company (mailbox)
“WC/WKC” “WcState”	Abbreviations for WorkingCounter (WC/WKC) or WorkingCounter status (WcState): Validity of command executions or data (WcState value 0 = valid, 1 = invalid).

The following figure shows the field bus-dependent names of the master or slave interfaces of the field buses:

Table 1: Terms of the field bus interfaces

Field bus	Master interface	Slave interface
EtherCAT	Master	Slave

Table 2: Terms of the data transfer mechanisms

Field bus	Cyclic transfer	Acyclic transfer
EtherCAT	Cyclic data/process data	Different mailbox protocols

Device description files

The device description file describes the properties of the field bus device. This file contains all relevant data important for the engineering as well as for the data exchange with the I/O device.

ESI file	
	Device description file used to implement and configure EtherCAT slaves in the ctrlX I/O Engineering (ESI: EtherCAT Slave Information).

Bus coupler	
	Field bus connection (slave, device, adapter), Compact • For a compact slave, the module structure is fixedly specified. The modules below the "Slave" object node are already present. No more modules are available in the library Modular • The module structure of the slave is subject to changes. The modules (terminals) can be arranged individually, but according to the device mounting specifications.
Module, terminal	
	Smallest I/O unit to assemble a field bus connection and/or the Inline bus.

4.3 Transmission modes

The transmission modes can be divided into the following categories:

- Cyclic transmission channel
- Acyclic transmission channel

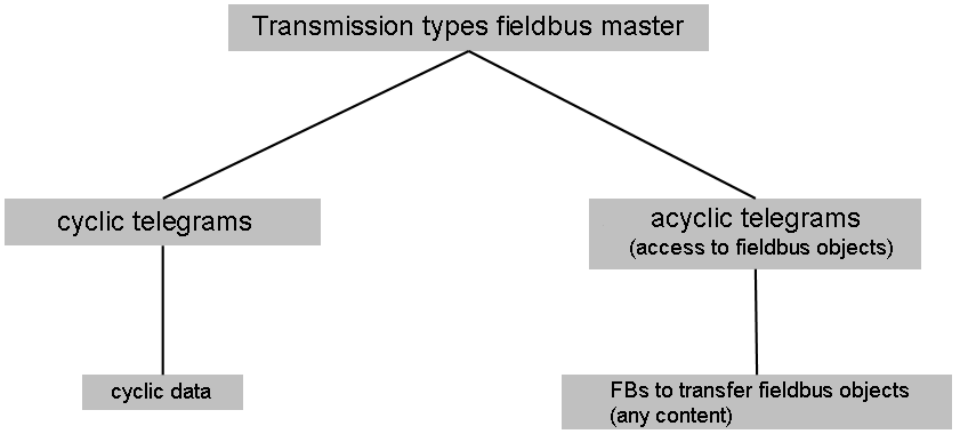

Fig. 1: Overview on the transmission modes for the field bus master

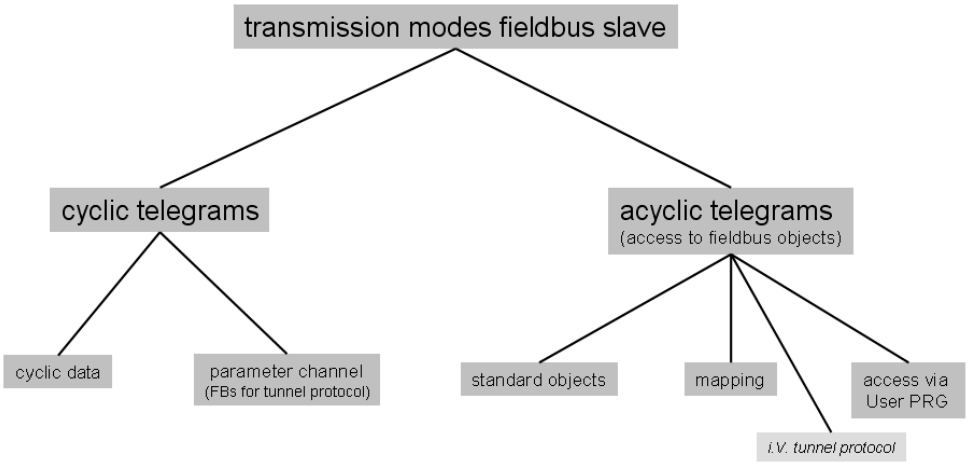

Fig. 2: Overview on the transmission modes for the field bus slaves

4.4 EtherCAT basics

4.4.1 Topology

An EtherCAT network consists of a master and of up to 65535 slaves.

Topologies can be implemented arbitrarily, e.g. by using junction terminals.

When cabling, note that slave devices are provided with exactly one defined input port. This is generally port 0, normally labeled with “IN”. Most of the slaves are provided with two ports, i.e. an “IN port” and an “OUT” port (port 1). In case of a bus coupler with E-BUS as backplane bus, the Ethernet telegrams are routed via this backplane bus (E-BUS), i.e. each module (terminal) is a slave with IN port and OUT port. The E-BUS bus coupler or junction terminals are thus provided with three or four ports (= multiple output ports).

The following figure shows an exemplary topology with different device types:

- Slave **(A)** is a modular bus coupler after the “Modular Device Profile” (e.g. Rexroth S20-EC-BK): The extension modules are connected to a device-specific backplane bus with the bus coupler. The bus coupler is a slave device in the network and contains all I/O data of the modules.
- Slave **(B)** is a modular system with E-BUS (e.g. Beckhoff EK1100): Each module, the bus coupler and possible junction terminals and extension terminals each represent an own slave device at the EtherCAT network.
- Slave **(C)** is a drive.

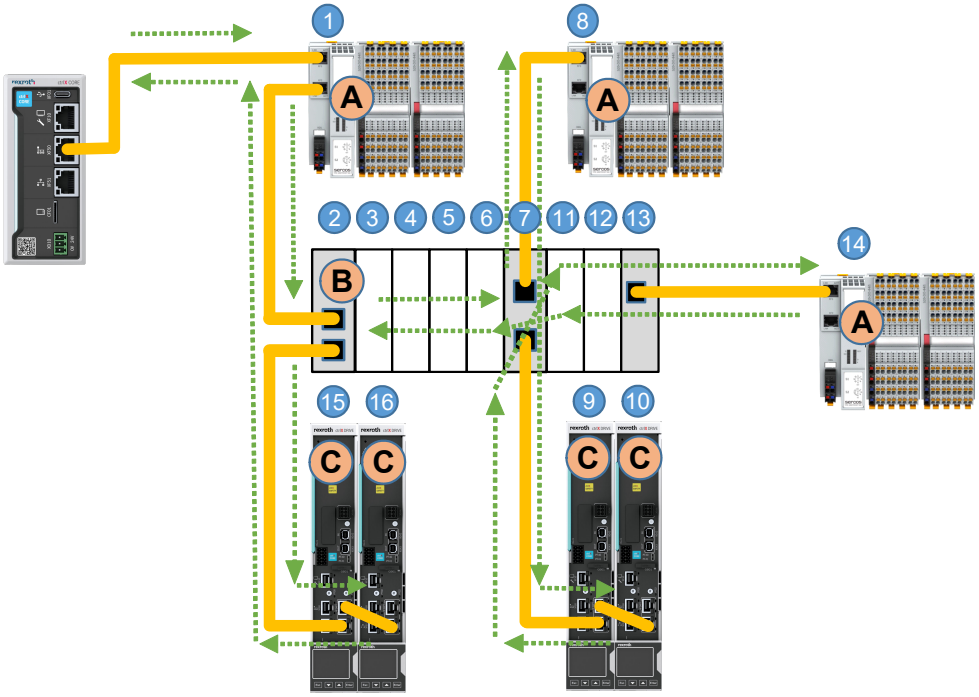

Fig. 3: Exemplary topology with different slave device types (A/B/C)

The EtherCAT telegram is sent by the master and runs through all slave devices. In the EtherCAT slave controllers (ESC), input/output data is only processed in forward direction. The last slave device sends the telegram back to the master. The telegram in the slaves is only routed, not processed. Logically, the EtherCAT represents a ring - irrespective of the topology. The previous figure shows the numbers from (1) to (16) in the logical sequence of the EtherCAT telegram passing the slave devices. For the sequence of the EtherCAT telegram passing the ports in case of junction terminals or devices with more than two ports, refer to the product documentation of the device.

Cable redundancy

There can only be a cable redundancy with a ring topology. The last slave device is connected to the master at its second Ethernet port.

The Rexroth EtherCAT master does **not** support cable redundancy, i.e. only the port XF50 can be used at the ctrlX CORE.

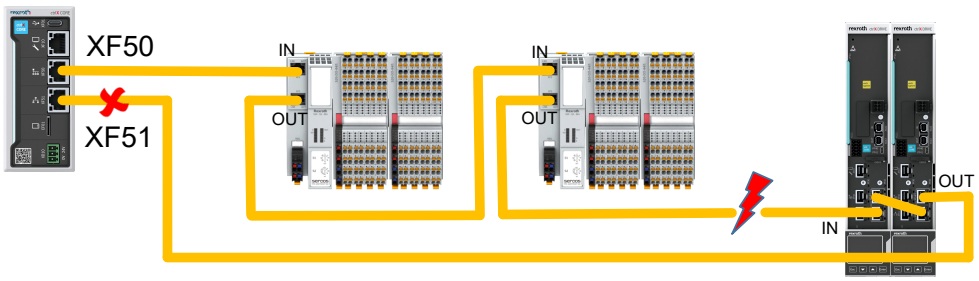

Fig. 4: Ring topology for cable redundancy

4.4.2 Addressing and identification procedure

The are different procedures to address and identify a slave device in an EtherCAT network.

Auto increment address

The auto increment address is a topological address, i.e. the slave is addressed using a physical position in the network.

The master mainly uses the auto increment address when initializing the bus communication. The EtherCAT address (“fixed physical address”) is used for communication purposes as of the state “Pre-Operational”.

The auto increment address is `0` for the first slave after the master. The value is decremented for each further slave, i.e. the second slave has the address `-1` (`0xFFFF`) and the third slave has the address `-2` (`0xFFFE`).

For the communication, each slave increments the auto increment address in the EtherCAT telegram. The slave reading the value “0” in the telegram is addressed correspondingly.

EtherCAT address

The EtherCAT address is also called “fixed physical address” in the ETG specification. Synonyms of other vendors are for example “Slave Address” or “Station Address”.

Upon the bus initialization, the EtherCAT address is not saved in the slave devices, but it is assigned to each slave by the master. The assignment which EtherCAT address obtains a certain slave device takes place in the EtherCAT configuration either via the auto increment address or - for Hot Connect groups (flexible topologies) via an identification value.

The EtherCAT address `0` is reserved for the master. Slaves can have an address from `1` to `65535` (16 bits). EtherCAT addresses for slaves are often used starting from the value `1001`.

Identification value

For Hot Connect groups (flexible topologies) or to exclude wiring errors, clearly identify slave devices irrespective of their topological position.

To always assign a specified EtherCAT address to a certain slave device, there are three different identification procedures. Which identification procedures are supported by a slave device (no support, one or more procedures are supported) is described in its device description or product documentation. The identification value is for example configured via the DIP switch or rotary switch at the slave device.

Identification procedure:

- (Configured) **Station alias**: The station alias is also often called “Second Slave Address” (SSA) or “Second Station Address”. The value can be retrieved from the ESC register `0x0012`. It is generally configured via the EtherCAT configuration tool to which the identification value is written to the slave EEPROM. The EtherCAT slave controller then applies this value to its register upon restart.
- **Explicit Device Identification**: The slave informs on its identification value in the AL status code register (“0x0134”) upon master request. In the “ETG.1020”, this procedure is also called “Requesting ID”.
- **Data Word (/ Input Word)**: The identification value is located in a specified memory range of the EtherCAT slave (e.g. process data area from ADO `0x1000`). For the address offset (ADO), refer to the device description (attribute “IdentificationAdo”) or set it if required. In the “ETG.1020”, this procedure is also called “Direct ID”.

Example

In the following example, the first two slaves (EtherCAT addresses `1001` and `1005`) do not have any identification value, i.e. the EtherCAT address is only specified using the topology position (AutoInc address). If the physical order of these two devices is exchanged (identical device type), e.g. wiring error or intended device change, the EtherCAT address is still assigned to the respective position.

For the following two slaves (EtherCAT addresses `1042` and `1043`), an identification procedure is configured, i.e. the master accepts the slaves in the cyclic communication only if the identification values (here 2 and 3) match the configured topology position (AutoInc address). Thus, there will be no wiring error between these two slaves. For an intended cross exchange or device exchange (spare device), the identification value has to be set accordingly.

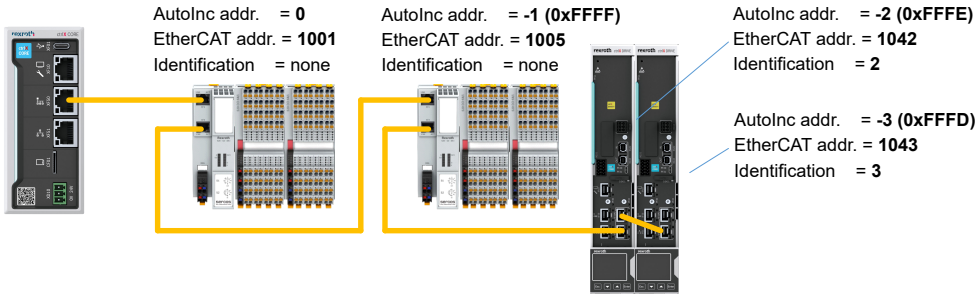

Fig. 5: Example of an EtherCAT addressing

4.4.3 State machine

There are the following communication states for the EtherCAT:

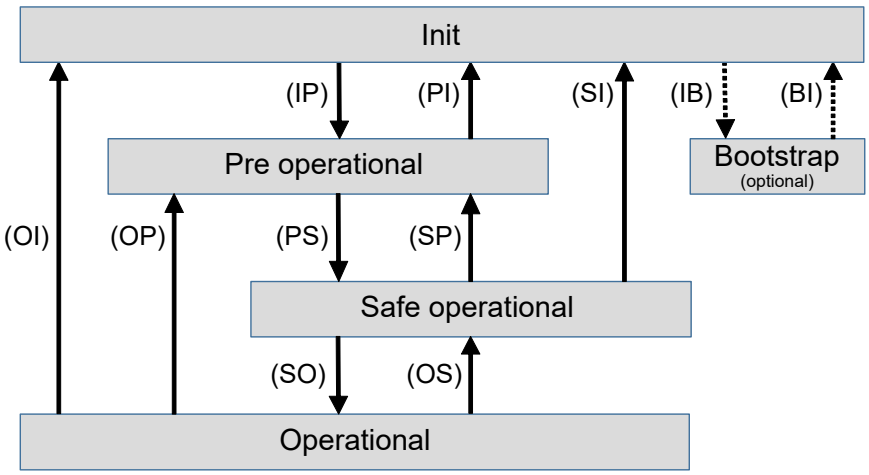

Fig. 6: EtherCAT state machine

State	Description
Init (I)	• No communication with application layer • Transition to “PreOP” (IP): Master configures slave register, e.g. address register and Sync manager for mailbox communication

Pre-Operational (P)	• The mailbox communication on the application layer is possible as of “PreOP” • Not yet any process data communication • Transition to “SafeOP” (PS): Master also configures process data mapping (FMMU) and Sync manager for process data communication
Safe-Operational (S)	• Process data communication. Only input data is analyzed. Output data remains in the "safe state" • DC synchronization is adjusted
Operational (O)	• Process data communication: Input and output data is valid • DC synchronization
Bootstrap (B)	• Optional state in EtherCAT slave devices (recommended for firmware update) • The state “Bootstrap” can only be reached from “Init” • No process data communication • Mailbox communication via FoE (FoE only possible in “Bootstrap” => device-specific)

Additional information:

- An EtherCAT slave device can be switched to a "lower" state than the master bus state, e.g. if the EtherCAT is in the bus state “OP”, a slave can be switched individually to “PreOP”.
- Only a master can initiate a change in state. Otherwise, a slave can only “downshift” its state in case of error.

4.4.4 Acyclic communication (mailbox)

For EtherCAT, acyclic communication is also called mailbox communication and only possible starting from the “Pre-Operational” state. There are different protocols that can be tunneled via EtherCAT. For information on which protocols are supported by a slave, refer to the documentation (or device description) of the slave device.

Overview on mailbox protocols

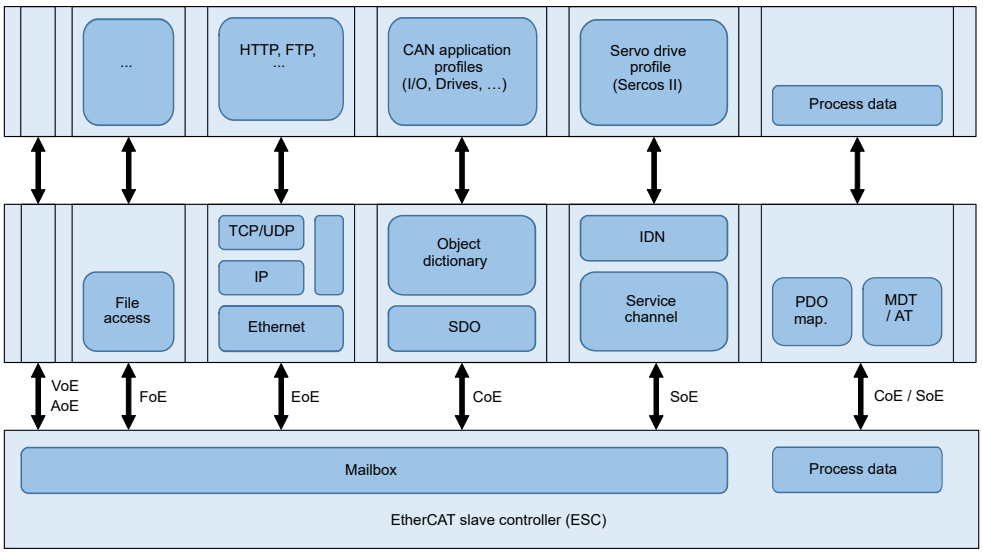

Fig. 7: Overview on mailbox protocols

CoE: CAN application protocol over EtherCAT

CoE objects are also called “Process Data Objects” (PDO) or “Service Data Objects” (SDO). An object is addressed with “Index” and “Subindex”, e.g. `0x1018:01`. If an object has more than one subindex, the number of subindices is written to the subindex `0`.

The object directory of a slave as well as the object descriptions and element descriptions of the subindices can be retrieved using the optional CoE service “SDO Information”.

Also refer to ➦ Chapter 9.3 CoE SDO abort codes on page 58.

SoE: Servo drive profile over EtherCAT

SoE is an adapted Sercos II profile for Servo drives, i.e. some Sercos II parameters do not have any or not the same meaning for SoE. One parameter is addressed with a 16-bit identification number (IDN: Parameter type S/P, parameter set, data block number).

In contrast to Sercos, multi-axis devices represent only one slave device for SoE. The subordinate drives differ by a drive number (DriveNo: 0 / 1 / ... / 7), optionally also referred to as channel (channel: A / B / ... / H).

Also refer to ➦ Chapter 9.5 SoE error codes on page 60.

The following IDNs are **not** relevant for SoE:	
S-0-0003	Minimum AT transmit starting time (T1min)
S-0-0004	Transmit/receive transition time (TATMT)
S-0-0005	Minimum actual value detection (T4min)
S-0-0009	Initial address in the master-data telegram
S-0-0010	Length of the master-data telegram
S-0-0088	Receive-to-receive recovery time TMTSY
S-0-0090	Command value proceeding time TMTSG
S-0-0127	C0100 Communication phase 3 transition check It is performed during the EtherCAT state transition from PreOP => SafeOP. If there is an error, the cause can be evaluated using S-0-0021.
S-0-0128	C5200 Communication phase 4 transition check It is performed during the EtherCAT state transition from SafeOP => OP. If there is an error, the cause can be evaluated using S-0-0022.

The following IDNs have an **adjusted** meaning at SoE:	
S-0-0006	Transmission time of the drive telegram (T1) T1 specifies the time shift from the EtherCAT Sync signal to the point in time when the application should provide new AT data to the ESC memory.
S-0-0014	Interface status This parameter is intended to display the DL status, AL status and the AL status code of the EtherCAT.

S-0-0028	MST error counter Counter for missing datagrams of cyclic data and RX error counter and lost link counter of the EtherCAT.
S-0-0089	T2 Transmit starting time MDT T2 specifies the time shift from the EtherCAT Sync signal to the point in time when new data is available to the master in the ESC memory.

FoE: File access over EtherCAT

FoE is a protocol, analog to TFTP, to transfer files. Normally, a firmware update is used for FoE. Thus, FoE is only possible in the EtherCAT state “Bootstrap” (device-specific).

Also refer to ➥ Chapter 9.4 FoE error codes on page 59.

EoE: Ethernet over EtherCAT

EoE can be used to tunnel the standard Ethernet configuration (e.g. TCP/IP) via EtherCAT. A “virtual Ethernet switch” is required in the EtherCAT master.

An IP-capable device (e.g. laptop) cannot be connected to any location in the EtherCAT EtherCAT network. A “switch port” is required (e.g. special terminal with “switch port” ⇔ is not an EtherCAT port).

Optionally, an IP port can also be configured for an EtherCAT slave device if the device supports EoE to allow for addressing of an integrated web server via a configured IP port.

In both cases, the EtherCAT slave requires a “virtual MAC address” in the EtherCAT network specified in the EtherCAT configuration. The EoE settings are then written from the master to the slave devices upon bus startup (transition Init => PreOP). EoE is only possible starting from the PreOP state like all mailbox protocols.

To use IP routing between the Engineering port and the EtherCAT master, enable IP forwarding at the Engineering port.

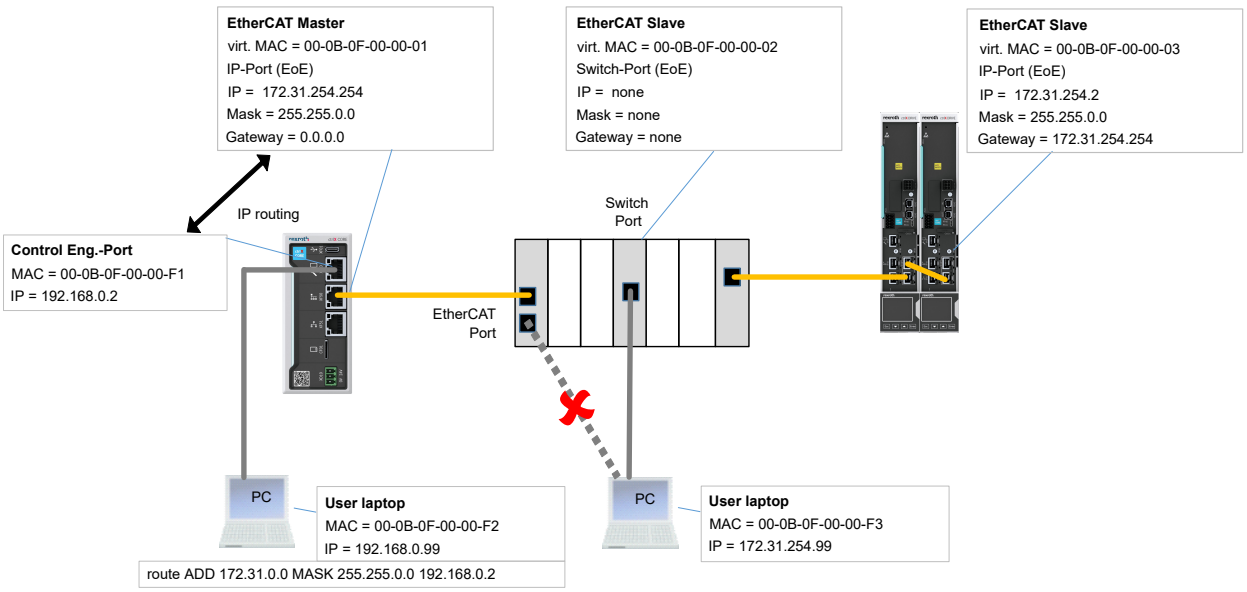

Fig. 8: EoE configuration example

VoE: Vendor specific protocol over EtherCAT

VoE is used to transmit vendor-specific protocols.

AoE: ADS over EtherCAT

AoE is a device- and field bus-independent interface by Beckhoff (ADS: Automation Device Specification).

Slaves without mailbox communication

EtherCAT slave device do not have to support a mailbox protocol. Simple digital input/output terminals do not support any mailbox protocol for example. For these slave devices, only the registers supported by EtherCAT slave can be read and written. This has to be supported by every EtherCAT slave controller (ESC).

4.4.5 Telegram structure

EtherCAT uses standard Ethernet frames (IEEE 802.3) with an own EtherType 0x88A4. Ethernet telegrams of another type are not supported by an EtherCAT slave controller and have to be tunneled by EtherCAT (e.g. EoE).

The EtherCAT frame consists of multiple subtelegrams or datagrams with different commands. Each SyncUnit is for example an own datagram and an acyclic communication (mailbox) or other commands are own datagrams.

Refer to ➦ Chapter 4.4.7 SyncUnits on page 22.

Up to 15 datagrams can be summarized int one EtherCAT frame. Multiple frames are required if there are more than 15 datagrams.

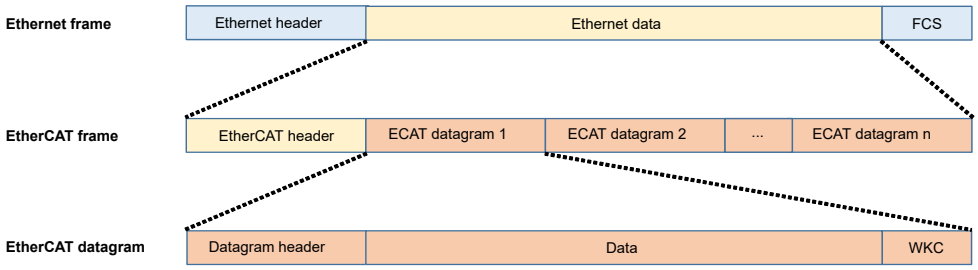

Fig. 9: EtherCAT telegram structure

4.4.6 WorkingCounter

The checksum of the Ethernet frame (FCS : Frame Check Sequence) ensures the correct data transfer to the bus.

To check the correct command execution in the slave devices, each datagram has a **WorkingCounter** (WKC). The WKC is sent with the value 0 by the master. Each addressed slave increments the WKC with regard from the command. After the frame ran through all slaves and reached the master again, the master can compare the expected value (command value) of the WKC to the actual value and thus check the correct command processing.

For the validity of cyclic process data, the status of the WorkingCounter check is considered.

Please also refer to ➦ General

4.4.7 SyncUnits

SyncUnit

A SyncUnit describes a logical grouping of cyclic data that should be exchanged consistently and synchronously. In the easiest case, all EtherCAT slaves are combined to a SyncUnit, but there can also be individual SyncUnits for each slave.

Generally, optional slaves (e.g. slaves at runtime that may fail) or machine modules are grouped in an own SyncUnit. Thus, these devices or this SyncUnit are provided with an own datagram and thus with an own WorkingCounter. A missing device for example results only in an invalid WorkingCounter of this SyncUnit. The data status of all other SyncUnits is still valid.

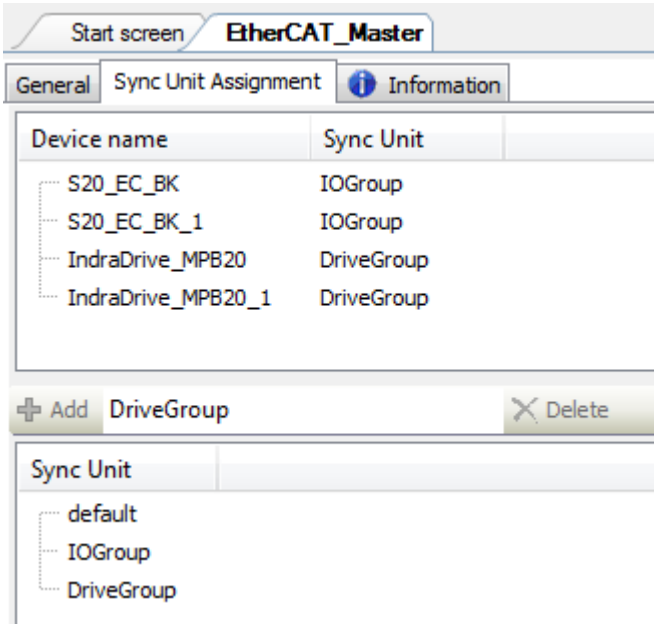

Fig. 10: Example of a SyncUnit configuration

Different EtherCAT cycle times

Different EtherCAT cycle times can also be executed with SyncUnits. The SyncUnits (= datagrams) are then sent in own frames with different cycle times. Fast I/O components and drive groups are for example grouped into individual SyncUnits with different cycle times.

The Rexroth EtherCAT master does not support different EtherCAT cycle times!

4.4.8 Hot Connect

Currently, "Hot Connect" is not supported by ctrlX CORE.

Hot Connect

Hot Connect (HC) generally means that slaves can be removed and connected at runtime (bus state "Operational"). This also includes use cases with multiple missing devices or machine modules or uses cases that already allow the commissioning of a part of the system.

HC group

The configuration of HC groups automatically groups the slaves into an own SyncUnit. An HC group can be declared as "optional". Thus, the EtherCAT master does not detect a missing HC group as error (flexible topologies). The first slave of an HC group is detected by an identification procedure (refer to ➥ Chapter 4.4.2 Addressing and identification procedure on page 15).

It can also be specified for an HC group whether this group may only be connected at a specific topology position (port of the predecessor EtherCAT device) or at any position.

The Rexroth EtherCAT master does not support the configuration of HC groups!

4.4.9 EtherCAT Slave controller

Structure

The following figure shows a schematic diagram of an EtherCAT slave controller (ESC).

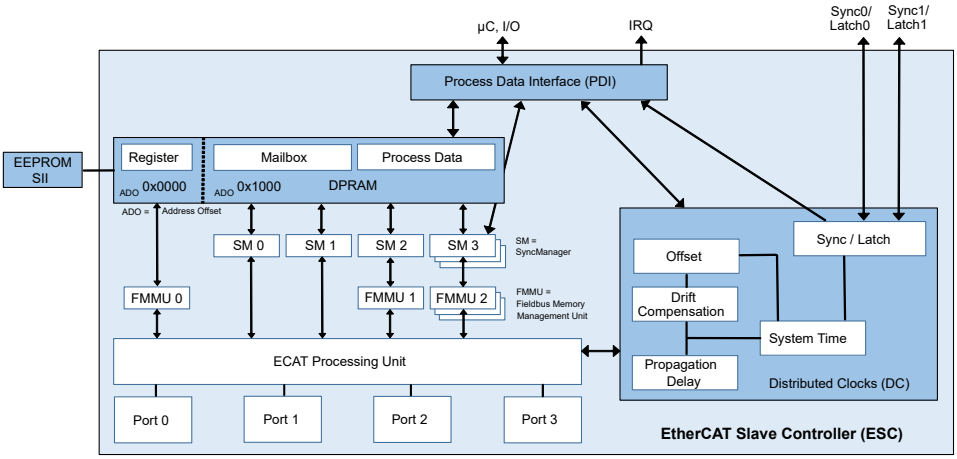

Fig. 11: EtherCAT slave controller structure

Register

The registers are in the local address range from 0x0000 to 0x0FFF of the ESC and are also used

- to configure the ESC, e.g. EtherCAT address (“fixed physical address“), SyncManager, FMMU, DC,
- to access the EEPROM/SII (Slave information Interface). It also includes the device identification and the default values for the ESC configuration,
- Control and status data of the EtherCAT state machine.

SyncManager (SM)

The SyncManagers ensure data consistency in the DPRAM of the ESC, i.e. simultaneous access to the memory ranges is inhibited, e.g. if the ECAT processing unit (telegram processing unit) and the local application try to access the memory range in the DPRAM at the same time.

SM configuration from the register ADO 0x0800.

There can be up to 16 independent SyncManager channels:

- Normally, a slave without mailbox has two SyncManager:
  - SM0 = Process data outputs (or inputs if there are no outputs),
  - SM1 = Process data inputs.
- Normally, a slave with mailbox has four SyncManagers:
  - SM0 = Mailbox-Out,
  - SM1 = Mailbox-In,
  - SM2 = Process data outputs (or inputs if there are no outputs),
  - SM3 = Process data inputs.
- Process data of one direction (inputs or outputs) can also be divided into multiple SyncManagers; e.g. if the data consistency of individual memory range is to be monitored separately. This is also called "SyncUnits" in the slave.

It is distinguished between different types:

- Mailbox (acyclic communication): 1 buffer system with handshake mechanism.
- Process data: 1- or 3-buffer system. With three buffers, there can be independent read and write accesses, e.g. for the Free Run communication if the local application and the ECAT telegram processing run independently from each other.

Fieldbus Memory Management Unit (FMMU)

Process data is assigned to a logic address space on the EtherCAT bus (up to 4 GB). The address space can be split on any number of slaves. The FMMU is responsible to map a global logical address to the local physical address of the ESC.

The FMMU configuration takes place from the register ADO 0x0600.

There can be up to 16 independent FMMU channels, e.g. separate FMMUs for data direction and memory ranges.

Distributed Clocks (DC)

The procedure "Distributed Clocks" with a specified global reference clock is used to synchronize EtherCAT devices.

The system time starting at 1/1/ 2000 at 00:00 o’clock (unit 1 ns/64 bits) is controlled in all devices, i.e. the drift and the offset of the local ESC clock is respectively compensated to the reference clock (accuracy < 100ns).

The DC configuration takes place from the register ADO 0x0900.

The DC unit generates respective events (Sync/Latch) for input/output data processing. The second Sync signal (Sync1) can be configured using a defined offset towards the Sync0 event. The latch signals are for example used for the "Time Stamping" of the system time.

Commands

There are different EtherCAT commands for all addressing procedures. Apart from separate read and write commands, there are also combined commands executing read and write accesses with a datagram. These combined accesses are more efficient, but not supported by all slave devices.

Value	Code	Name	Description
0	NOP	No Operation	Slave ignores command.
1	APRD	Auto Increment Read	Slave addressing with auto increment address. The slave - receiving address 0 - processes data. Each slave increments its address by 1.
2	APWR	Auto Increment Write	
3	APRW	Auto Increment Read Write	

Value	Code	Name	Description
4	FPRD	Fixed Physical Read	Slave addressing with a configured EtherCAT address (“fixed physical address”).
5	FPWR	Fixed Physical Write	
6	FPRW	Fixed Physical Read Write	
7	BRD	Broadcast Read	All slaves execute an OR operation of the receiving datagram data with the addressed memory range and write the result to the same location in the datagram.
8	BWR	Broadcast Write	Data is written to all slaves.
9	BRW	Broadcast Read Write	All slaves execute an OR operation of the receiving datagram data with the addressed memory range and write the result to the same location in the datagram and to the memory. (This command is normally not used).
10	LRD	Logical Memory Read	Slave addressing with logic memory address, i.e. all slaves whose FMMUs are configured with the address of the logic memory mapping from the datagram, processes data.
11	LWR	Logical Memory Write	
12	LRW	Logical Memory Read Write	
13	ARMW	Auto Increment Read Multiple Write	Slave addressing with auto increment address. The slave receiving address 0 adds input data to the datagram (“read”), all other slaves extract output data at the same location of the datagram (“Multiple Write”). Each slave increments its address by 1.
14	FRMW	Fixed Physical Read Multiple Write	The slave with the configured EtherCAT address (fixed physical address) adds input data to the datagram (“read”), all other slaves extract output data at the same location of the datagram (“Multiple Write”).
15-255	-	-	Reserved.

Depending on the command type, the WorkingCounter of the datagram is incremented:

Command type	Result	Incremention
Read	Reading not successful (or slave not addressed)	No
	Reading successful	+ 1
Write	Writing not successful (or slave not addressed)	No
	Writing successful	+ 1
Read + write	Reading and writing not successful (or slave not addressed)	No
	Reading successful	+ 1
	Writing successful	+ 2
	Reading and writing successful	+ 3

5 EtherCAT synchronization

5.1 General

An individual synchronization mode can be set for each EtherCAT slave. For the modes supported by a slave device, refer to the documentation or device description of the slave device. The following synchronization modes are distinguished:

- Free Run (no synchronization)
- Synchronization with SM event (SM: SyncManager)
  The main task of the SyncManager (SM) is to ensure data consistency in the DPRAM of the ESC. There can be up to 16 independent SyncManager channels. Four are used in the standard configuration: SM0 = Mailbox - Out, SM1 = Mailbox In, SM2 = Process data Out, SM3 = Process data In.)
- Synchronization with DC sync (DC: "Distributed Clocks")

The following example shows the DC configuration with the Rexroth IndraDrive MPx20 SoE. The configurator shows the predefined configurations from the device description (ESI). Apart from the "Free Run" mode, three DC modes are provided with different "Sync0Shift" (time shift in the slave to the Sync0 event).

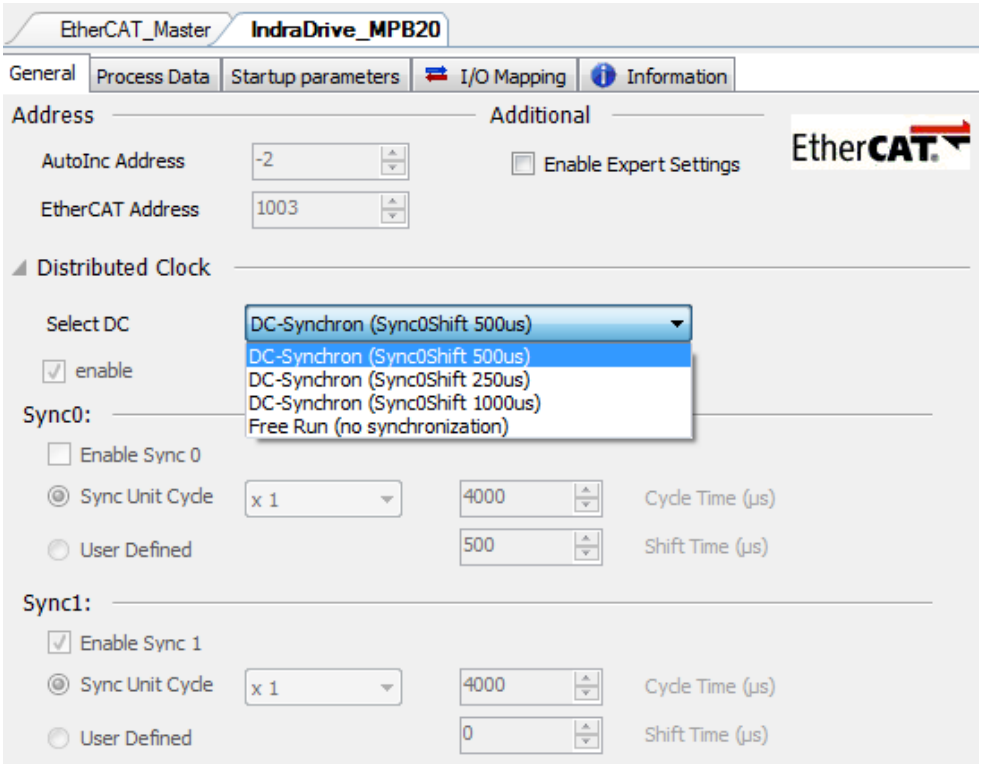

Fig. 12: DC configuration with Rexroth IndraDrive MPx20 SoE

5.2 Free run

The EtherCAT communication cycle and the input/output data processing run independently in the slave device. I.e. the local application cycle in the slave device can be slower or faster than the EtherCAT bus cycle.

The following figures show the copying and processing times in the slave devices using the “In”/“Out” bars.

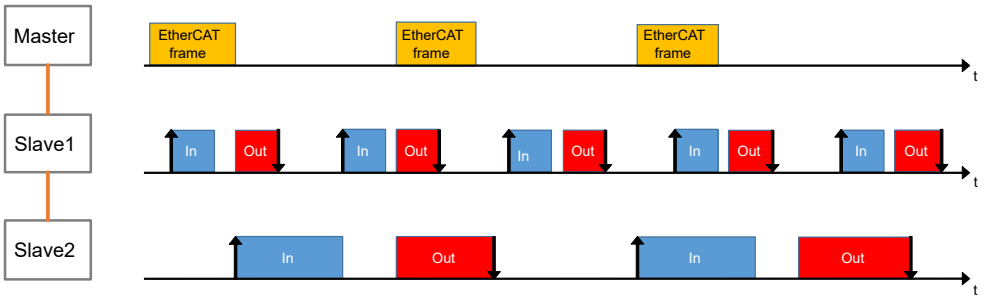

Fig. 13: “Free Run” mode

For CoE, the following parameters of the SyncManager object 0x1C3m are relevant (with the standard configuration "SM2/3". The following applies to process data: 0x1C32 for outputs and 0x1C33 for inputs):

SI	Description	Access	Usage	Explanation/default value
1	Synchronization type	R/RW	M	Synchronization type: 0x00: Free Run
2	Cycle time	R/RW	O	Local cycle time of the application controller in ns
4	Synchronization types Supported	R	M	Supported synchronization types: Bit 0 = 1: Free Run supported
5	Minimum cycle time	R	C	Minimum cycle time: Maximum runtime of the local application cycle (required if SI=2 can be written)

SI = Subindex, R = Read-only, RW = Read/Write, M = Mandatory , O = Optional, C = Conditional

5.3 Synchronizing with SM event

This synchronization is often called "SM2/3" event, as the SyncManagers normally use 0/1 for mailbox out/in and the SyncManagers use 2/3 for process data out/in.

The output data processing in the slave is triggered when receiving the EtherCAT telegram by the SyncManager. When output data becomes valid depends on the jitter at the sending point in time of the master, on the duration of the telegram during transfer and on the processing times in the slaves. To apply input data, an "Input Shift" can optionally be configured (per device) to have the most "current" input data possible upon the next SM event (next telegram run).

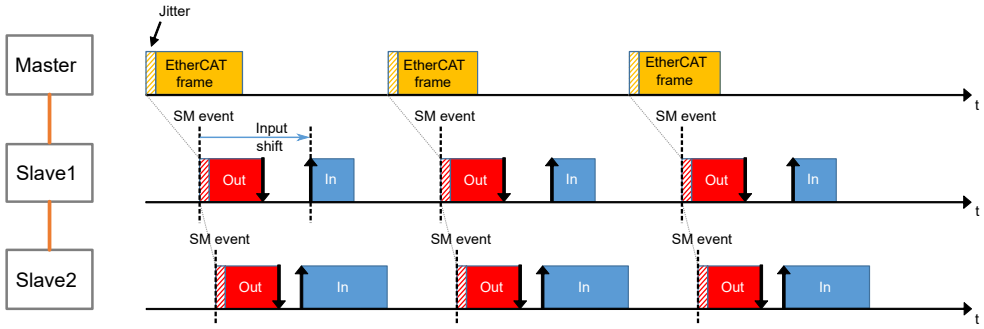

Fig. 14: Mode "Synchronization with SM event"

For CoE, the following parameters of the SyncManager object 0x1C3m are relevant (with the standard configuration "SM2/3". The following applies to process data: 0x1C32 for outputs and 0x1C33 for inputs):

SI	Description	Access	Usage	Explanation/default value
1	Synchronization type	RW	M	Synchronization type: Outputs (0x1C32): 0x01: Synchronous with SM2 event Inputs (0x1C33): 0x01: Synchronous with SM3 event (if no outputs are available) 0x22: Synchronous with SM2 event (if outputs are available)
2	Cycle time	R/RW	O	EtherCAT communication cycle in ns (Time between two SM events)
3	Shift Time	RW	-/ M	Time shift, input data application (Shift Input Latch) (only inputs / 0x1C33) in ns
4	Synchronization types Supported	R	M	Supported synchronization types: Bit 1 = 1: Synchronous SM supported
5	Minimum cycle time	R	M	Minimum cycle time supported by the slave (Maximum time of the local application cycle in ns)
8	Get Cycle Time	RW	C	Used in the SM and DC mode for a variable cycle time: Bit 0: Local time measurement started Bit 1: Reset of error counter
11	SM-event missed	R	O	This error counter is incremented if the SM event is not received within the expected time (consequently, data cannot be copied anymore)
12	Cycle time too small	R	M	This error counter is incremented if the cycle time is too short to completely execute the local application cycle (input data cannot be prepared until next SM event)

SI	Description	Access	Usage	Explanation/default value
14	RxPDO toggle failed	R	O	This error counter is incremented if the slave supports the RxPDO toggle and if no change in status was received anymore by the master
32	Sync error	R	C	Synchronization error: Should be supported if SI = 11 is available. Can be mapped in TxPDO. Value 0: No Sync error or Sync not supported Value 1: Sync error

SI = Subindex, R = Read-only, RW = Read/Write, M = Mandatory , O = Optional, C = Conditional

5.4 Synchronizing with DC sync

For the synchronization procedure "Distributed Clocks" (DC), there are multiple options to specify the reference clock:

- **Master Shift**
  The first DC slave at the bus is used as reference clock and as re-adjustment of the timer of the EtherCAT master.
- **Bus Shift**
  The EtherCAT master sets the system time and writes it to the first DC slave at the bus (shifting the bus time).
- **External reference clock** (IEEE 1588)

Based on the reference clock, a global system time is set in all devices. Use this system time to generate corresponding "DC Sync" events in the EtherCAT slave controllers to process input/output data.

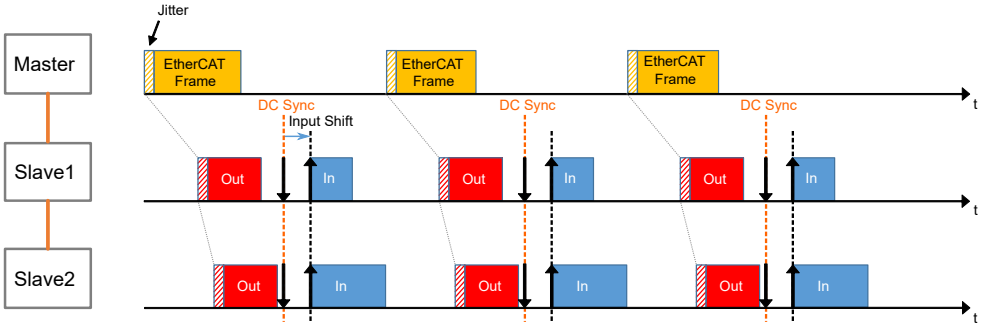

Fig. 15: Mode "Synchronization with DC sync" (simplified representation)

In addition to the Sync event "Sync0", another subordinate Sync event "Sync1" is generally supported (functionality of a standard EtherCAT slave controller (ESC)). Which Sync modes are supported by a slave device should be included in its device description file (ESI) or in the product documentation. The following DC modes are distinguished:

- Synchronous with Sync0 event
- Synchronous with Sync0 event, input shift with Sync1 event
- Synchronous with Sync0 event, output shift with Sync1 event
- Synchronous with Sync1 event
- DC with subordinate cycles (Sync1 event only every nth Sync0 cycle)

The point in time of the Sync event is globally set in the EtherCAT master as "Sync Shift Time" (or "Sync Offset") related to the master application cycle. One Sync0 and Sync1 cycle time can be set individually for each slave. This Sync cycle time is a multiple of the EtherCAT communication cycle. Additionally, a shift ("Shift Time") related to the global Sync point in time is possible.

The DC settings are normally predefined configurations from the device description (ESI) of a slave and should only be changed by experts!

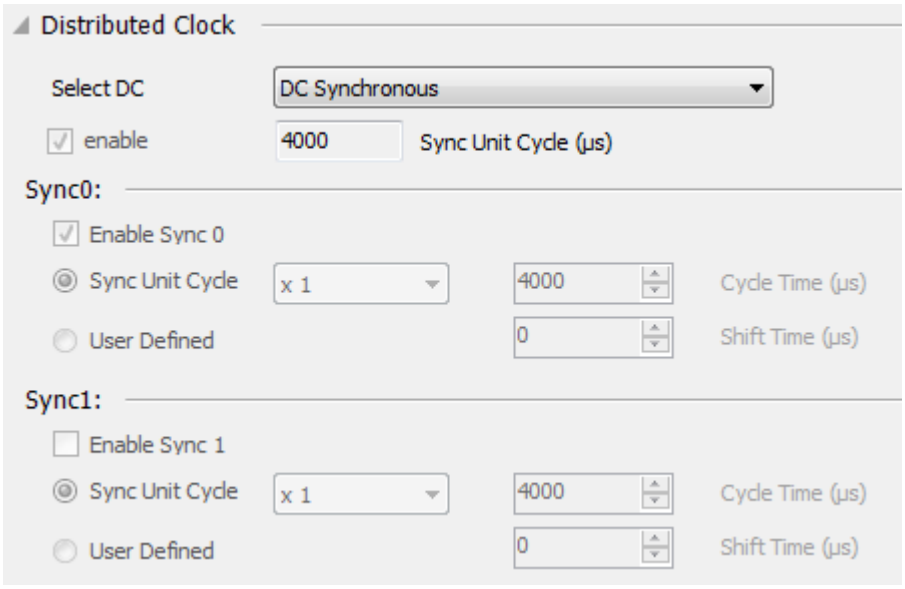

Fig. 16: Slave configuration of the DC settings

For CoE, the following parameters of the SyncManager object 0x1C3m are relevant (with the standard configuration "SM2/3". The following applies to process data: 0x1C32 for outputs and 0x1C33 for inputs):

SI	Description	Access	Usage	Explanation/default value
1	Synchronization type	RW	M	Synchronization type: 0x02: DC synchronous with Sync0 event 0x03: DC Synchronous with Sync1 event
2	Cycle time	R	O	Cycle time Sync0 event, time between two Sync0 events in ns (register 0x09A3:0x09A0) For subordinate cycles: Time between two Sync1 events in ns (register 0x09A7:0x09A4) The cycle times Sync0/1 and, where applicable, the shift time of Sync0/1 are specified in the configurator.
3	Shift Time	RW	M	Time shift in ns: Outputs: Time between Sync0 event and "Outputs valid" (initialized with 0x1C32:09) Inputs: Time between Sync0 event and "Input Latch" (initialized with 0x1C33:05 – 0x1C33:06)

SI	Description	Access	Usage	Explanation/default value
4	Synchronization types Supported	R	M	Supported synchronization types: Bit 4-2: Supported DC modes 001 = DC Sync0 010 = DC Sync1 100 = Subordinate cycle Bits 5-6: Shift settings 00 = Output Shift not supported 01 = Input/Output Shift with local timer 10 = Input/Output Shift with Sync1
5	Minimum cycle time	R	M	Minimum cycle time supported by the slave (Maximum time of the local application cycle in ns)
6	Calc and copy time	R	M	Computation and copying time in ns: Minimum time between SM2 event and Sync0 event. For Sync1: Minimum time between Sync0 and setting of outputs
8	Get Cycle Time	RW	C	Is used in the SM and DC mode for a variable cycle time Bit 0: Local time measurement started Bit 1: Reset of error counter
9	Delay time	R	M	Hardware delay time in ns: Time between trigger (Sync0 or Sync1 event) and validity of data in the process (e.g. electric signals)
11	SM-event missed	R	O	This error counter is incremented if the SM event is not received within the expected time (consequently, data cannot be copied anymore)
12	Cycle time too small	R	M	This error counter is incremented if the cycle time is too short to completely execute the local application cycle (input data cannot be prepared until next SM event)
13	Shift Time Too Short	R	O	This error counter is incremented if the SM event was received too late so that the Sync0 event is received before "Calc+Copy Time" is complete (master telegram too late; configuration problem)
14	RxPDO toggle failed	R	O	This error counter is incremented if the slave supports the RxPDO toggle and if no change in status was received anymore by the master.
15	Minimum cycle distance	R	O	Minimum distance between two SM events in ns (to monitor jitter)
16	Maximum cycle distance	R	O	Maximum distance between two SM events in ns (to monitor jitter)
17	Minimum SM Sync distance	R	O	Minimum distance between SM event and Sync0 event in ns (to monitor jitter)

SI	Description	Access	Usage	Explanation/default value
18	Maximum SM Sync distance	R	O	Maximum distance between SM event and Sync0 event in ns (to monitor jitter)
32	Sync error	R	C	Synchronization error: Should be supported if SI = 11 or SI = 13 is available. Can be mapped in TxPDO. Value 0: No Sync error or Sync not supported Value 1: Sync error

SI = Subindex, R = Read-only, RW = Read/Write, M = Mandatory , O = Optional, C = Conditional

The following boundary conditions apply to the DC modes:

(1) Synchronous with Sync0 event

Optional time shifts of "Outputs valid" and "Input Latch" by Shift Time (0x1C3m:03).

Synchronization type:

- Outputs (0x1C32:01) = 2 (Sync0 event)
- Inputs (0x1C33:01) = 2 (Sync0 event)

Cycle time:

- Sync0 cycle time (0x1C3m:02) > Min. cycle time (0x1C3m:05)

Shift time:

- Output Shift Time (0x1C32:03) > Output Delay Time (0x1C32:09)
- Input Shift Time (0x1C33:03) > [Min Cycle time (0x1C32:05) – Input Calc and Copy Time (0x1C33:06)]
- Input Shift Time (0x1C33:03) < [Sync0 Cycle Time (0x1C33:02) – Input Calc and Copy Time (0x1C33:06)]

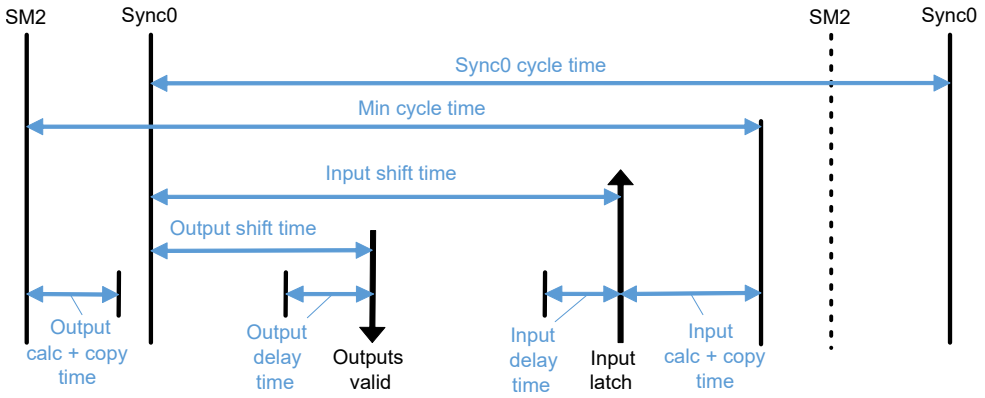

Fig. 17: Boundary conditions (1) Synchronous with Sync0 event

(2) Synchronous with Sync0 event, Input Shift with Sync1 event

Time shift from "Input Latch" with Sync1 event (in contrast to "Shift Time", the "Start Latch" point in time is described, i.e. without "Input Delay Time").

Synchronization type:

- Outputs (0x1C32:01) = 2 (Sync0 event)
- Inputs (0x1C33:01) = 3 (Sync1 event)

Cycle time:

- Sync0 cycle time (0x1C3m:02) > Min. cycle time (0x1C3m:05)

Shift time:

- Shift Sync0 <> Sync1 (config.) **>=** [Min Cycle Time (0x1C3m:05) – Output Calc and Copy Time (0x1C32:06) – Input Calc and Copy Time (0x1C33:06) – Input Delay Time (0x1C33:09)]

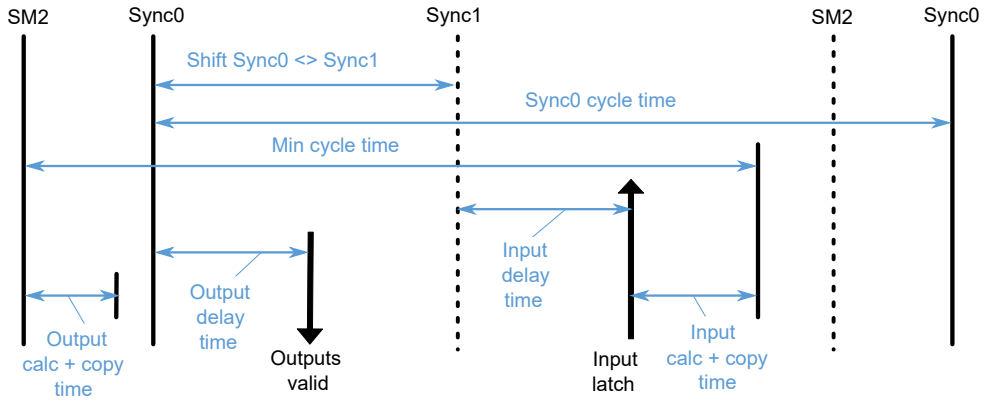

Fig. 18: Boundary conditions (2) Synchronous with Sync0 event, Input Shift with Sync1 event

(3) Synchronous with Sync0 event, output shift with Sync1 event

Time shift from "Outputs valid" with Sync1 event (in contrast to "Shift Time", the "Start Outputs" point in time is described, i.e. without "Output Delay Time").

Synchronization type:

- Outputs (0x1C32:01) = 3 (Sync1 event)
- Inputs (0x1C33:01) = 2 (Sync0 event)

For more detailed information on the configuration "Output shift with Sync1 Event", contact the ETG.

(4) Synchronous with Sync1 event

Time shifts of "Outputs valid" and "Input Latch" by Sync1 event.

Synchronization Type

- Outputs (0x1C32:01) = 3 (Sync1 event)
- Inputs (0x1C33:01) = 3 (Sync1 event)

Cycle Time

- Sync0 cycle time (0x1C3m:02) **>** Min. cycle time (0x1C3m:05)

Shift Time

- Shift Sync0 <> Sync1 (Config.) **>=** Output Calc and Copy Time (0x1C32:06)

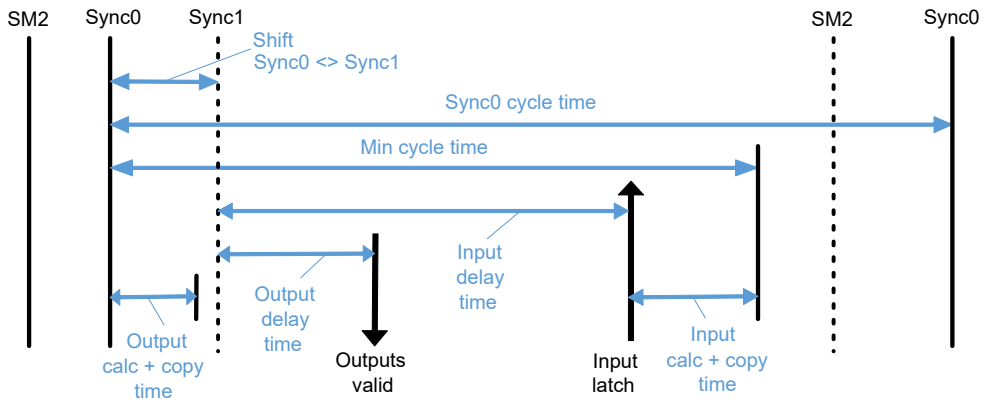

Fig. 19: Boundary conditions (4) Synchronous with Sync1 event

(5) DC with subordinate cycles

The Sync0 event is used as trigger for the local application cycle (e.g. for fast controls) while the Sync1 event is used for a (slower) communication cycle. I.e. the Sync1 cycle time is a multiple of the Sync0 cycle time (config.).

Synchronization type:

- Outputs (0x1C32:01) = 3 (Sync1 event)
- Inputs (0x1C33:01) = 3 (Sync1 event)

Cycle time:

- Cycle time (0x1C3m:02) is Sync1 cycle time

Shift time:

- The time shifts of "Outputs valid" and "Input Latch" can only be specified by "Input / Output Shift Time" (0x1C3m:03).

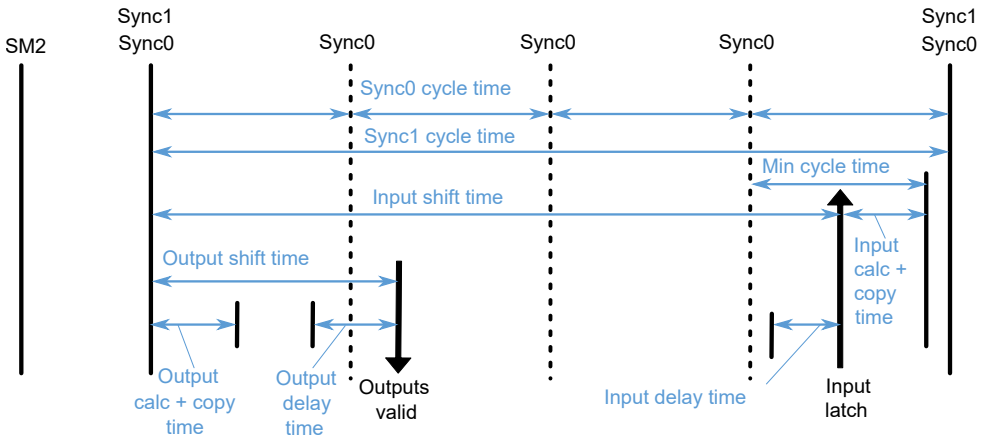

Fig. 20: Boundary conditions (5) DC with subordinate cycles

6 EtherCAT configuration

6.1 Overview

Connections

The Rexroth EtherCAT master does not support cable redundancy. Only one Ethernet port can be used for the EtherCAT master cabling at the EtherCAT master:

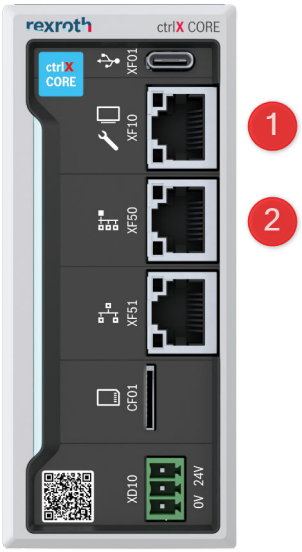

Fig. 21: Connections of a ctrlX CORE control
① XF10 engineering port
② XF50 EtherCAT port

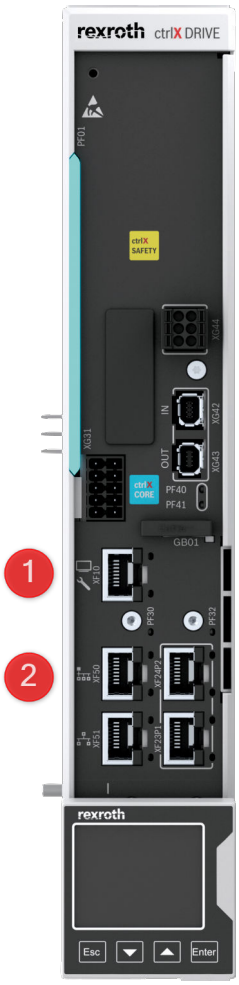

Fig. 22: Connections at a ctrlX DRIVE with integrated ctrlX CORE control
① XF10 engineering port
② XF50 output of the first EtherCAT slave (ctrlX DRIVE)

Configuring the EtherCAT master

Refer to ➦ Chapter 7.2 Configuring the EtherCAT master on page 40

Configuring the EtherCAT field bus devices

Refer to ➦ Chapter 7.3.1 Configuring the EtherCAT field bus devices on page 41

7 EtherCAT master

7.1 EtherCAT Master – Features

Licensing

The EtherCAT Master app and the use of the available functions are subject to license!

The following license is required for each EtherCAT Master:

Type code: **SWL-XC*-ECM-ETHERCATMAS**-BANN**

Ordering number: **R911400508**

Functional scope

The implementation of the EtherCAT Master includes the following functionality:

- Cyclic communication “Free Run”
- Synchronization mode (“Distributed Clocks”)
- Import of the device description file (ESI) in the ctrlX I/O Engineering device database
- Offline configuration of the slaves
- Manual configuration of SyncUnits
- Online diagnostics in the ctrlX I/O Engineering interface
- Bus scan and configuration in the ctrlX I/O Engineering device tree
- PLC function blocks (CXA_Ethercat library)
  - Function blocks for state machine
  - Function blocks for acyclic communication
  - Function blocks for diagnostics (e.g. master status, slave status, configuration/online info)

Control hardware

Supported control types:

- ctrlX CORE
- ctrlX COREplus
- ctrlX DRIVE with integrated ctrlX CORE
- ctrlX IPC with integrated ctrlX CORE

Technical data

ctrlX CORE device variant	X3 X3plus DRIVEplus	X7
Max. number of devices (slaves)	128	256
Max. number of Sync Units	Unlimited (Limited to one cyclic frame)	
Max. number of cyclic input data of the master (a cyclic frame)	1 kB	1.5 kB
Max. number of cyclic output data of the master (a cyclic frame)	1 kB	1.5 kB
Baud rate	100 MBit/s	
Auto negotiation/autocrossing	Yes	

ctrlX CORE device variant	X3 X3plus DRIVEplus	X7
Cycle time Adjustable in 1ms steps	Min: 2 ms Max: 20 ms	Min: 500µs, (or multiples of 1 ms) Max: 20 ms
Support of multiple different cycle times	No	
Synchronization mode (Distributed Clocks)	Yes	
Hot Connect groups (flexible topologies)	No	
Mailbox CoE (CAN application layer over EtherCAT)	Yes	
Mailbox SoE (Sercos drive profile over EtherCAT)	Yes	
Mailbox EoE (Ethernet over EtherCAT)	Yes	
Mailbox FoE (File access over EtherCAT)	Yes	
Mailbox VoE (Vendor-specific protocol over EtherCAT)	No	
Mailbox AoE (ADS over EtherCAT)	Yes	
Slave-to-slave communication	No	Yes (routed via master)
Cable redundancy	No	
Disabling slaves or modules in the project configuration (Maximal project)	No	
Disabling slaves or modules in the runtime (Maximal project)	No	
Engineering and diagnostics	ctrlX I/O Engineering	
IEC APII (programming interface)	Basic Subset	
Data Layer Interface NRT	Yes	
Data Layer Interface RT	Yes	
Data Layer Interface RT additional Information	No	
EtherCAT interface configuration	Default configuration (template)	

7.2 Configuring the EtherCAT master

After installing the EtherCAT Master app, the EtherCAT bus system on the ctrlX device is not yet active!

To activate the system, the EtherCAT Master on the ctrlX device has to be activated first.

Subsequently, the bus topology has to set in the ctrlX I/O Engineering engineering tool.

ctrlX I/O Engineering is a part of the ctrlX WORKS installation for your Engineering PC.

To activate the EtherCAT Master on the ctrlX Core device, please execute the following steps:

1. In the ctrlX CORE side navigation, navigate to the window *“EtherCAT → EtherCAT Master”*.

2. In the window click on the button ⊕ to add the EtherCAT Master.
   ➡ The dialog “Add EtherCAT-Master” opens.
3. Keep the default settings in the input fields “Name” and “Port” and confirm the dialog with “OK”.
   ➡ The EtherCAT Master is added and displayed in the table.

Further information

➦ Configuring the EtherCAT field bus devices

7.3 EtherCAT Slave

7.3.1 Configuring the EtherCAT field bus devices

The EtherCAT field bus device configuration is executed in the ctrlX I/O Engineering Engineering tool. ctrlX I/O Engineering is part of the ctrlX WORKS installation for your engineering PC.

To configure the EtherCAT field bus devices, proceed as follows:

1. Open “ctrlX I/O Engineering”:
   Open the “EtherCAT Master” window via the ctrlX CORE side navigation
2. In the window header, click on the I/O button
   ➡ The “ctrlX I/O Engineering” tool is opened on the engineering PC and displayed at the device tree is displayed at the left edge of the screen.
   The device tree contains three objects in its initial state:
   - : Project node
   - : Device node of the control
   - : Device nodes of the EtherCAT master
3. Select the device node of the EtherCAT master node and right-click to open its context menu
4. Execute the command “Scan For Devices...”
   ➡ The “Bus scan” dialog opens and shows all slaves available at the bus
5. To apply the found slaves to the project, select the button and confirm the dialog using the button “Apply”
   ➡ The EtherCAT Slave is added to the device tree below the EtherCAT master
6. Apply and enable the field bus configuration on the control. Select the button in the command bar
   ➡ The field bus configuration is applied to the control and the bus is switched to the “OP” state
   ℹ If there are more components at the bus such as a bus coupler, execute the command “Scan For Devices...” a second time. The modules connected to the bus coupler are then detected at the bus and can be applied to the configuration.
7. Save the configuration to ctrlX I/O Engineering via the button

8 ctrlX UI – Elements

8.1 Navigation

8.1.1 Side navigation – node EtherCAT Master

By installing the “EtherCAT Master” app on the control, the “EtherCAT Master” node is added to the ctrlX CORE side navigation, see:

➦ Window – “EtherCAT Master”

EtherCAT widget

In addition to the “EtherCAT Master” in the side navigation, an EtherCATwidget is displayed after the app installation in the window“Home”, see ➦ documentation.

The widget provides information and contains the following contents:

- Information on the current EtherCAT operating status
- Several links to call EtherCAT-relevant settings and Engineering tools to configure master and slaves

8.2 Windows

8.2.1 Window – “EtherCAT Master”

The Window is used to configure the EtherCAT Master on the ctrlX device.

After installing the EtherCAT Master app, the EtherCAT bus is not yet ready for operation.

The following steps are still necessary to activate the EtherCAT Master on the ctrlX device and to configure the bus topology:

- ➦ Configuring the EtherCAT master
- ➦ Configuring the EtherCAT field bus devices

For the current version only one EtherCAT Master on the ctrlX device is supported.

Window call

In the ctrlX CORE web interface:

“side navigation → EtherCAT Master”

Window description

GUI element	Description
Header	Call to start ctrlX I/O Engineering to configure the bus devices, see ➦ Header – “EtherCAT Master”
Command bar	• “[x] items” Number of EtherCAT master instances • ⊕ Adding the EtherCAT master instance, see ➦ Configuring the EtherCAT master
Table (in case of enabled EtherCAT Master)	“Name” Name of the EtherCAT Master instance. Clicking on the name opens the detailed view of the Master, see: ➦ Window – “EtherCAT Master” Detailed view

GUI element	Description
	“Port” Bus connection on the ctrlX device via which the EtherCAT master communicates
	“State” EtherCAT Master operating status display: • “Init” = Init • “Pre-OP” = Pre-Operational • “Safe-OP” = Safe-Operational • “OP” = Operational Refer to ➦ State machine
	“Diagnostics” Diagnostic information EtherCAT master
	“Actions” •  Delete EtherCAT master on the control

Related topics

➦ EtherCAT Master app – Basics

➦ Side navigation – node EtherCAT Master

➦ Header – “EtherCAT Master”

8.2.2 Window – “EtherCAT Master” Detailed view

The “EtherCAT Master” detailed view is an extended view of the EtherCAT Master and provides the following functions and displays:

- **Header**
  Shows the current operating status of the EtherCAT master and provides buttons to toggle the operating state and to invoke ctrlX I/O Engineering, see ➦ Header – “EtherCAT Master”
- **Tab** “Slaves”
  Contains a tabular display of the connected slaves and their operating states, see:
  ➦ Tab – “Slaves”
- **Tab** “Distributed Clocks”
  Shows information about the cycle time setting and the Distributed Clocks, see:
  ➦ Tab – “Distributed Clocks”
- **Tab** “Master statistics”
  Provides information about the transmitted data on the EtherCAT bus, see:
  ➦ Tab – “Master statistics”
- **Tab** “Slave statistics”
  Contains error counters of the configured EtherCAT slaves (e.g. for error analysis) that respond, see:
  ➦ Tab – “Slave statistics”
- **Tab** “AoE” **(EtherCAT Master instance)**
  Overview of the AoE configuration (ADS over EtherCAT) in the EtherCAT network, see:
  ➦ Tab – “AoE” (EtherCAT master instance)
- **Tab** “EoE” **(EtherCAT Master instance)**
  Overview of the EoE configuration (Ethernet over EtherCAT), see:

➦ Tab – “EoE” (EtherCAT master instance)

Window call

In the ctrlX CORE web interface:

“side navigation → EtherCAT Master → Click on the name of the EtherCAT instance in the overview table”

Related topics

➦ EtherCAT Master app – Basics

➦ Side navigation – node EtherCAT Master

8.3 Header

8.3.1 Header – “EtherCAT Master”

The header is used to control the EtherCAT master on the ctrlX device and is integrated into the EtherCAT master detailed view of the web interface, see: ➦ Window – “EtherCAT Master” Detailed view

Functions

- Display of the master operating status
- Display and button for changing the master operating state
- Link to the call of ctrlX I/O Engineering (tool for the configuration of the bus devices)

Call

In the ctrlX CORE web interface:

“side navigation → EtherCAT Master → Click on the EtherCAT master in the overview table”

Description

GUI element	Description
Status display (graphical/textual)	• = bus in OP / RUN • = bus interference or incorrect configuration The graphical status message is supplemented by a textual status message. The text output provides more detailed information on the bus operating status, for example in the event that the bus cannot be commissioned. In this case, the closest criterion is displayed, which is not met, so that the bus can be commissioned. By clicking on the text output, a box with all criteria is expanded and their status is displayed as graphics. Required criteria: • EtherCAT port exists • License to operate the bus system is available • Settings are valid • Links available • Bus topology OK • Master in required operating state • Slaves in master operating state • Slaves without error • Master in operating state OP • DC in sync • Bus running
Operating state	Display of the current operating status and possibility to switch the status. Operating states: • “Init” = Init • “Pre-OP” = Pre-Operational • “Safe-OP” = Safe-Operational • “OP” = Operational Refer to ➦ State machine
I/O	Button for calling ctrlX I/O Engineering to configure the bus devices

Related topics

➦ EtherCAT Master app – Basics
➦ Side navigation – node EtherCAT Master
➦ Header – “EtherCAT slave”

8.3.2 Header – “EtherCAT slave”

The header is used to control the EtherCAT slaves and is integrated in the slave detail views of the web interface, see: ➦ Tab – “General” (EtherCAT Slave)

Functions

- Display of the slave operating status
- Display and switching option of the slave operating state
- Notes, e.g. if the configuration has been changed and a manual update is required

Call

In the ctrlX CORE web interface:

“side navigation → EtherCAT Master → Click on the EtherCAT master instance in the overview table → Click on tab "Slaves" → Click on the desired slave”

Description

GUI element	Description
Slave name	To be named individually
Visual status display Textual status display	• = Slave in OP / RUN • = The slave reports an error or is not in OP Shows the current device status (operating status, connection status, pending errors). Via the button [ ], a table with the most important criteria is displayed: Possible messages: • Slave in OP • Slave not in OP • Connected • Not connected • No error • If an error is pending, the error code is displayed Diagnostic codes see: ➦AL status codes
Operating state (Display/switching)	Display of the current operating status and possibility to switch the status. Operating states: • “Init” Transition to the state Init is executed: – No communication on the application layer – Master has access to the DL information registers • “Pre-OP” Transition to the state Pre-Operational is executed. • “Safe-OP” Transition to the state Safe-Operational is executed. • “OP” Transition to the state Operational is executed. Refer to ➦State machine

Related topics

➦EtherCAT Master app – Basics

➦Header – “EtherCAT Master”

8.4 Tabs – EtherCAT Master

8.4.1 Tab – “Slaves”

The tab is used to display the configured slaves in tabular form. In the table you can open a detailed view of the respective slave by clicking on one of the slave names. The detailed view provides further information about the slave and a specific header for diagnostics and for controlling the slave operating state, see:

➦ Tab – “General” (EtherCAT Slave)
➦ Header – “EtherCAT slave”

Call

In the ctrlX CORE web interface:
“side navigation → EtherCAT Master → Overview table → Click on the name of the EtherCAT master instance → Click on the "Slaves“ tab”

Related topics

➦ Window – “EtherCAT Master”
➦ Window – “EtherCAT Master” Detailed view
➦ EtherCAT Master app – Basics
➦ Side navigation – node EtherCAT Master

8.4.2 Tab – “Distributed Clocks”

The tab displays information about the cycle time setting and the Distributed Clocks, see below.

Call

In the ctrlX CORE web interface:
“side navigation → EtherCAT Master → Click on the EtherCAT master instance in the overview table → Click on tab "Distributed Clocks"”

Displayed information

Table entry	Description
“DC mode”	set and active (in brackets) DC mode: • “Auto” – Default setting – Automatic selection of the synchronization mode • “Free Run” – Cyclic communication without clock synchronization – There is no regulation of the distributed clocks • “Bus Shift” – Distributed Clocks active – Slaves are synchronized with the EtherCAT Master – The EtherCAT Master provides the system time and writes it into the bus reference clock (moving the bus time) • “Link Layer Reference Clock” – Distributed Clocks active – Slaves are synchronized with the EtherCAT Master – The clock integrated in the EtherCAT Master link layer is used as a reference clock for synchronization. The setting guarantees the highest synchronization precision but requires special hardware. **WARNING** Do NOT use the setting “Link Layer Reference Clock” if you are simultaneously synchronizing to an external clock, such as an NTP server. This can result in unpredictable behavior!
“Cycle time”	set cycle time

Table entry	Description
“Sync Offset”	Display of the Sync Offset in percent and as absolute effective value
“Sync window monitoring”	Indicates whether Sync window monitoring is active, including the active monitoring window. If this option is enabled, the synchronization of the Distributed Clocks is monitored in all slaves. The "system time difference" (register 0x092C) is read cyclically by each slave. If the system time difference is smaller than the monitoring window for all slaves, synchronization is considered to be adjusted ("DC in Sync" status).
“Continuous propagation compensation”	Indicates whether continuous runtime compensation is active
Slave overview table	“Status” • **Green**: The slave is connected correctly and does not report an error • **Red**: The slave is not connected or reports an error
	“Name” Configured device name of the slave
	“Address” EtherCAT address
	“State” Operating state of the EtherCAT slave
	“System time difference” System time difference in ns (ESC register 0x09C2) The value is only displayed if Sync Window Monitoring is active and is only valid if „Distributed Clocks“ is configured. The first slave at the bus, for which the option "Distributed Clocks" is activated, provides the reference clock.
	“DC support” Indicates whether the controller of the EtherCAT slave supports „Distributed Clocks“ (32 / 64 bit).

Related topics

➦ Window – “EtherCAT Master”

➦ EtherCAT Master app – Basics

➦ Side navigation – node EtherCAT Master

8.4.3 Tab – “Master statistics”

The tab displays information about the transferred data on the bus EtherCAT-Bus is displayed.

The tab is divided into two table sections:

- **Frames**
  Shows sent and lost frames, divided into cyclic and acyclic communication
- **Mailbox requests**
  Shows all sent mailbox requests, subdivided by mailbox and transmission direction

Reset counter readings

Via the “Reset counters” button, the counters in the respective table areas can be reset separately.

Copy/save counter readings

Via the button, the counter values of the respective table area are transferred to the clipboard of the engineering PC, for example to save the values locally on the PC.

Notes on the evaluation

Lost frames are a first indicator for communication problems, but can also occur by plugging or unplugging a bus cable or by switching off/on a slave or the controller.

If the number of lost frames increases continuously during regular operation, check the Slave statistics, see:

➦ Tab – “Slave statistics”

For further notes and details on the procedure, please refer to the documentation of the "EtherCAT Technology Group":

➦ https://www.ethercat.org/.../EtherCAT_Diagnosis_For_Users.pdf

Calling up the tab via the navigation bar

In the ctrlX CORE web interface:

“side navigation → EtherCAT Master → Click on the name of the EtherCAT master instance in the overview table → Click on tab Master statistics”

Related topics

➦ EtherCAT Master app – Basics

➦ Side navigation – node EtherCAT Master

➦ Header – “EtherCAT Master”

8.4.4 Tab – “Slave statistics”

Error counters to the projected and accessibleEtherCAT slaves are displayed in the tab. By analyzing error counters and the underlying bus topology, a potential point of failure in the bus system can be detected.

References for troubleshooting

- Reset the error counters when the system is in the "Master in OP“ operating state , see:
  ➦ Header – “EtherCAT Master”
- Subsequently, monitor the error counters over a longer time interval
- Analyze the error counter according to the current topology (order of the frame run)
- If all error counters in a slave report 0 and if errors were detected in the subsequent slave, analyze the connection between the slaves

Reset error counter

The error counters count up to a maximum value of 255 (max. register width in a slave) and freeze upon reaching this value.

The error counters can be reset to 0 by using the “Reset error counter” button.

The error counters also detect faulty frames which can occur due to connecting or disconnecting cables or by switching a slave or the control on or off.

Call

In the ctrlX CORE web interface:

“side navigation → EtherCAT Master → Overview table → Click on the name of the EtherCAT master instance → Click on the "Slave statistics“ tab”

Description

For details on the procedure and the meaning of the displayed registers, please refer to the documentation of the "EtherCAT Technology Group", which can be accessed via the EtherCAT diagnostics for users button.

The following information is provided in a table:

- **Status**
  Operating state of slave
- **Name**
  Slave name
- **Address**
  Slave address
- **Invalid Frames / Port**
  Content of the ESC register 0x0300, 0x0302, 0x0304, 0x0306
- **RX Errors / Port**
  Content of the ESC register 0x0301, 0x0303, 0x0305, 0x0307
- **Forwarded RX Errors / Port**
  Content of the ESC register 0x0308, 0x0309, 0x030A, 0x030B
- **Proc. Unit** (processing unit error counter)
  Content of the ESC register 0x030C
- **PDI Errors**
  Content of the ESC register 0x030D

Related topics

➦ Window – “EtherCAT Master”
➦ EtherCAT Master app – Basics

8.4.5 Tab – “AoE” (EtherCAT master instance)

The tab is used for the overview of the AoE configuration (ADS over EtherCAT) in the EtherCAT network.

Call

In the ctrlX CORE web interface:

“side navigation → EtherCAT Master → Click on the name of the EtherCAT master instance in the overview table → Click on tab "AoE"”

Description

GUI element	Description
“Master configuration”	Net ID of the master
Tabular listing of the slaves	The table lists all slaves that are AoE and for which AoE is active in the configuration. If no slave supports AoE, the list remains empty. • “Name”: Slave name • “Address”: Slave address • “Net ID”: Net ID of the slave

Related topics

➦ EtherCAT Master app – Basics
➦ Side navigation – node EtherCAT Master
➦ Header – “EtherCAT Master”

8.4.6 Tab – “EoE” (EtherCAT master instance)

The tab gives an overview of the EoE configuration (Ethernet over EtherCAT).

Call

In the ctrlX CORE web interface:

“side navigation → EtherCAT Master → Click on the name of the EtherCAT master instance in the overview table → Click on tab "EoE"”

Description

GUI element	Description
“Master configuration”	EoE configuration of the master: • “IP address”: IP address of the master (empty, if EoE is not active at the master). • “Subnet mask”: Subnet mask of the master (empty, if EoE is not active at the master)
Tabular listing of the slaves	In the table, all slaves are listed which support EoE If no slave supporting EoE is configured, the list remains empty. • “Name”: Slave name • “Address”: Slave address • “MAC address”: MAC address of the slave (empty if EoE is not supported) • “IP address”: IP address of the slave (empty if EoE is not active or is configured as a switch port) • “Subnet mask”: Subnet mask of the slave (empty if EoE is not active or is configured as a switch port) • “Default gateway”: Default gateway of the slave (empty if EoE is not active or is configured as a switch port) • “DNS name”: DNS name of the slave (empty, if EoE is not active or is configured as a switch port)

Related topics

➦ EtherCAT Master app – Basics
➦ Side navigation – node EtherCAT Master
➦ Header – “EtherCAT Master”

8.5 Tabs – EtherCAT Slave

8.5.1 Tab – “General” (EtherCAT Slave)

The tab is used to display general configuration information of the slave.

Call

In the ctrlX CORE web interface:

“side navigation → EtherCAT Master → Click on the EtherCAT master instance in the overview table → Click on tab "Slaves" → Click on the desired slave → Tab "General"”

Displayed information

GUI element	Description
“Identity”	Displays the identity information of the slave • “Name”: Name of the slave from the configuration • “Vendor”: Name of the manufacturer derived from the VendorID • “Product code”: Product code of the slave • “Revision”: Slave revision
“Mailbox”	Displays information about the mailbox configuration. If no mailbox is supported, the following fields are empty. • “Protocols”: List of supported protocols • “Standard Out / Standard In”: Mailbox size in normal operation • “Bootstrap Out / Bootstrap In”: Mailbox size in bootstrap (e.g. for firmware download)

Related topics

➦ EtherCAT Master app – Basics
➦ Side navigation – node EtherCAT Master
➦ Header – “EtherCAT slave”

9 EtherCAT diagnostics

9.1 General

EtherCAT diagnostics are available on different communication layers.

A cycle-accurate diagnostics is generated in the master. It comprises the master bus state and the analysis of the WorkingCounter (WKC). By analyzing the WKC, there is a cycle-accurate detection whether a datagram (e.g. cyclic process data) was processed correctly and whether the data status is valid. All slaves in a datagram (e.g. SyncUnit) addressed with a faulty WKC do thus have an invalid data status.

NOTICE

In case of an invalid WorkingCounter, the EtherCAT Master App reacts with a "zero-setting" of the cyclic inputs of the affected SyncUnit. The validity of the cyclic outputs is not affected.

In order to be able to check the status of the slaves and, derived from this, the status of the cyclic input data in the applications, the EtherCAT Master App provides the following options.

- Datalayer NRT node
- PLC function blocks e.g.: IL_ECATMasterState, IL_ECATRemoteSlaveState.

A detailed diagnostics whether a slave has an EtherCAT state error for example or whether it is not connected anymore or whether there is a link problem (port status) is then acyclically determined by the master.

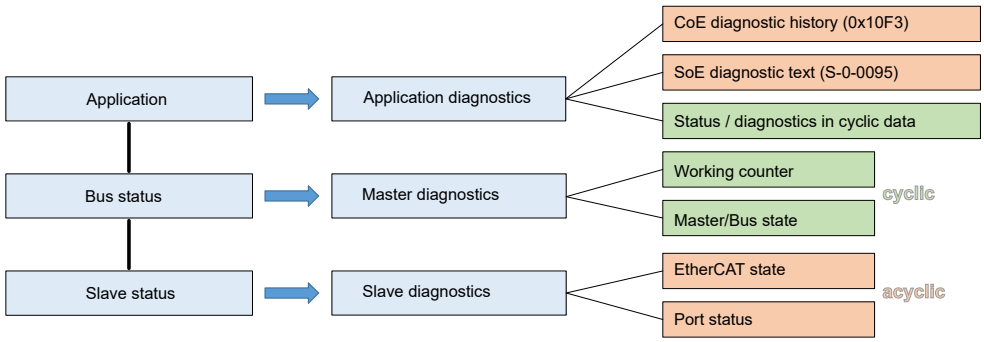

Fig. 23: Overview on EtherCAT diagnostics

On application level, many slaves are already provided with status messages or simple diagnostic data in the cyclic data. Detailed diagnostics can also be retrieved acyclically, e.g. there is a diagnostic history object (0x10F3) for CoE slaves or a diagnostic text (S-0-0095) for SoE slaves.

To analyze the diagnostics in the PLC program, the Rexroth EtherCAT master provides different function blocks in the CXA_Ethercat library.

Some CoE slaves also support the CoE emergency messages. These emergency messages mostly contain only device-specific information. Acknowledging them is not required. Thus, only these are generally entered into the control log.

9.2 AL status codes

The "AL status code" (AL: Application Layer) describes the error code of a slave device if a requested change in state cannot be executed or if a slave detects an internal error.

In case of error, the Rexroth EtherCAT master automatically retrieves the AL status code (ESC register 0x0134:0x0135) and acknowledges the error indication in the slave. Upon the next change in slave state, the read AL status code is reset in the master.

If there is an unspecified error or a configuration is detected as invalid, refer to the product documentation of the slave device or contact the vendor of the slave device.

The following abbreviations are used in the following table:

- I = State “Init”
- B = State “Bootstrap”
- P = State “Pre-Operational”
- S = State “Safe-Operational”
- O = State “Operational”

AL status code	Description	Current state or change in state	Resulting state
0x0000	No error	All	Current state
0x0001	Unspecified error	All	All
0x0002	No memory	All	All
0x0003	Invalid device configuration	P → S	P
0x0004	Invalid revision	P → S	P
0x0005	Reserved due to compatibility reasons	-	-
0x0006	SII/EEPROM data does not match the firmware	I → P	I
0x0007	Firmware update not successful	All	All
0x000E	License errors	All	I
0x0011	Invalid change in state requested	I → S, I → O, P → O, P → B, S → B, O → B	Current state
0x0012	Unknown state requested	All	Current state
0x0013	“Bootstrap” state not supported	I → B	I
0x0014	No valid firmware	I → P	I
0x0015	Invalid mailbox configuration (Bootstrap)	I → B	I
0x0016	Invalid mailbox configuration (PreOP)	I → P	I
0x0017	Invalid Sync manager configuration	P → S, S → O	Current state
0x0018	No valid inputs available	O, S → O	S
0x0019	No valid outputs possible	O, S → O	S
0x001A	Synchronization error (different causes)	O, S → O	S
0x001B	Sync manager watchdog	O, S	S

AL status code	Description	Current state or change in state	Resulting state
0x001C	Invalid Sync manager types	O, S, P → S	S
0x001D	Invalid Sync manager configuration (outputs)	O, S, P → S	S
0x001E	Invalid Sync manager configuration (inputs)	O, S, P → S	P
0x001F	Invalid watchdog configuration	O, S, P → S	P
0x0020	Slave requires cold start	All	Current state
0x0021	Slave requires the state “Init”	B, P, S. O	Current state
0x0022	Slave requires the state “Pre-operational”	S, O	S, O
0x0023	Slave requires the state “Safe-operational”	O	O
0x0024	Invalid mapping input data	P → S	P
0x0025	Invalid mapping output data	P → S	P
0x0026	Inconsistent settings (general)	P → S	P
0x0027	“Free Run” not supported	P → S	P
0x0028	Sync mode not supported	P → S	P
0x0029	“Free Run” requires “Sync Manager” in 3 buffer mode	P → S	P
0x002A	Background watchdog	S, O	P
0x002B	No valid input and output data	O, S → O	S
0x002C	Fatal sync error Sync0 or Sync1 signal not received anymore.	O	S
0x002D	No sync signal Slave waits for Sync0/Sync1 signal (S → O timeout) in “Safe-Operational”.	S → O	S
0x002E	Cycle time too small The set EtherCAT cycle time is not supported by the slave.	S → O	S
0x0030	Invalid DC sync configuration	O, S → O, P → S	S, P
0x0031	Invalid DC latch configuration	O, S → O, P → S	S, P
0x0032	PLL errors, Master is not synchronized, but at least one DC event is received.	O, S → O	S
0x0033	DC sync I/O error Multiple synchronization errors; master and slave are not synchronized anymore.	O, S → O	S
0x0034	DC sync timeout error Multiple synchronization error, too many SM events omitted	O, S → O	S
0x0035	Invalid DC cycle time	P → S	P
0x0036	DC Sync0 cycle time does not run with application	P → S	P

AL status code	Description	Current state or change in state	Resulting state
0x0037	DC Sync1 cycle time does not run with application	P → S	P
0x0041	Mailbox AoE	B, P, S, O	Current state
0x0042	Mailbox EoE	B, P, S, O	Current state
0x0043	Mailbox CoE	B, P, S, O	Current state
0x0044	Mailbox FoE	B, P, S, O	Current state
0x0045	Mailbox SoE	B, P, S, O	Current state
0x004F	Mailbox VoE	B, P, S, O	Current state
0x0050	EEPROM no access	All	All
0x0051	EEPROM errors	All	All
0x0052	External hardware not ready for operation	All	All
0x0060	Slave executed restart	All	I
0x0061	Device identification value updated	P	P
0x0070	Invalid module configuration For a slave with modular device profile, the configured modules (0xF030) do not match the physical modules (0xF050).	P → S	P
0x00F0	Application control available	I	I
< 0x8000	Reserved	-	-
0x8000 - 0xFFFF	Vendor-specific	-	-

9.3 CoE SDO abort codes

For an acyclic CoE mailbox communication, the following standardized SDO abort codes can result. These are mapped on the error codes of the control (→ ECM error code).

SDO abort code	Description	ECM error code
n/a	Ecat CoE: Protocol not supported	0x0C850032
0x0503 0000	Ecat SDO: Toggle bit not alternated	0x0C850040
0x0504 0000	Ecat SDO: SDO protocol timeout	0x0C850041
0x0504 0001	Ecat SDO: Client/server command specifier not valid	0x0C850042
0x0504 0002	Ecat SDO: Invalid block size (block mode only)	0x0C850043
0x0504 0003	Ecat SDO: Invalid sequence number (block mode only)	0x0C850044
0x0504 0004	Ecat SDO: CRC error (block mode only)	0x0C850045
0x0504 0005	Ecat SDO: Out of memory	0x0C850046
0x0601 0000	Ecat SDO: Unsupported access to an object	0x0C850047
0x0601 0001	Ecat SDO: Attempt to read a write only object	0x0C850048
0x0601 0002	Ecat SDO: Attempt to write a read only object	0x0C850049
0x0601 0003	Ecat SDO: Subindex cannot be written, SI0 must be 0	0x0C870004
0x0601 0004	Ecat SDO: Complete access not supported	0x0C870005
0x0601 0005	Ecat SDO: Object length exceeds mailbox size	0x0C870006
0x0601 0006	Ecat SDO: Object mapped to RxPDO, SDO Download blocked	0x0C870007

SDO abort code	Description	ECM error code
0x0602 0000	Ecat SDO: Object does not exist in object dictionary	0x0C85004A
0x0604 0041	Ecat SDO: Object cannot be mapped to PDO	0x0C85004B
0x0604 0042	Ecat SDO: PDO length exceeded	0x0C85004C
0x0604 0043	Ecat SDO: General incompatibility (parameter)	0x0C85004D
0x0604 0047	Ecat SDO: General incompatibility (device internal)	0x0C85004E
0x0606 0000	Ecat SDO: Access failed (hardware error)	0x0C85004F
0x0607 0010	Ecat SDO: Data type and length code do not match	0x0C850050
0x0607 0012	Ecat SDO: Wrong data type (parameter too long)	0x0C850051
0x0607 0013	Ecat SDO: Wrong data type (parameter too short)	0x0C850052
0x0609 0011	Ecat SDO: Subindex does not exist	0x0C850053
0x0609 0030	Ecat SDO: Parameter value out of range	0x0C850054
0x0609 0031	Ecat SDO: Parameter value out of high limit	0x0C850055
0x0609 0032	Ecat SDO: Parameter value out of low limit	0x0C850056
0x0609 0033	Ecat SDO: Module Ident List mismatch	0x0C85005E
0x0609 0036	Ecat SDO: Maximum value less than minimum value	0x0C850057
0x0800 0000	Ecat SDO: General error	0x0C850058
0x0800 0020	Ecat SDO: Data cannot be transferred or stored	0x0C850059
0x0800 0021	Ecat SDO: Data cannot be transferred (local control)	0x0C85005A
0x0800 0022	Ecat SDO: Data cannot be transferred (device state)	0x0C85005B
0x0800 0023	Ecat SDO: Object dictionary does not exist	0x0C85005C
n/a	Ecat SDO: Unknown error code	0x0C85005D

9.4 FoE error codes

For acyclic FoE mailbox communication, the following standardized FoE error codes can result. These are mapped on the error codes of the control (→ ECM error code).

FoE error code	Description	ECM error code
n/a	Ecat FoE: Protocol not supported	0x0C850034
n/a	Ecat FoE: Protocol not supported in Bootstrap	0x0C85010F
0x8000	Ecat FoE: Vendor specific FoE error	0x0C850060
0x8001	Ecat FoE: Not found	0x0C850061
0x8002	Ecat FoE: Access denied	0x0C850062
0x8003	Ecat FoE: Disk full	0x0C850063
0x8004	Ecat FoE: Illegal	0x0C850064
0x8005	Ecat FoE: Wrong packet number	0x0C850065
0x8006	Ecat FoE: Already exists	0x0C850066
0x8007	Ecat FoE: User missing	0x0C850067
0x8008	Ecat FoE: Bootstrap only	0x0C850068
0x8009	Ecat FoE: File name not valid in Bootstrap	0x0C850069
0x800A	Ecat FoE: No rights	0x0C85006A
0x800B	Ecat FoE: Program error	0x0C85006B
0x800C	Ecat FoE: Wrong checksum	0x0C85006C
0x800D	Ecat FoE: Firmware does not fit for Hardware	0x0C85006D

FoE error code	Description	ECM error code
0x800E	Reserved	n/a
0x800F	Ecat FoE: File not found	0x0C85006F
0x8010	Ecat FoE: File header missing	0x0C870001
0x8011	Ecat FoE: Flash problem	0x0C870002
0x8012	Ecat FoE: File incompatible	0x0C870003
n/a	Ecat FoE: File bigger than max file size	0x0C85017A

9.5 SoE error codes

For acyclic SoE mailbox communication, the following standardized SoE error codes can result. These are mapped on the error codes of the control (→ ECM error code).

Sercos error code	Description	ECM error code
n/a	Ecat SoE: Protocol not supported	0x0C850035
n/a	Ecat SoE: Invalid access to element 0	0x0C850078
0x1001	Ecat SoE: Does not exist	0x0C850079
0x1009	Ecat SoE: Invalid access to element 1	0x0C85007A
0x2001	Ecat SoE: No name	0x0C85007B
0x2002	Ecat SoE: Name transmission too short	0x0C85007C
0x2003	Ecat SoE: Name transmission too long	0x0C85007D
0x2004	Ecat SoE: Name cannot be changed (read only)	0x0C85007E
0x2005	Ecat SoE: Name is write-protected at this time	0x0C85007F
0x3002	Ecat SoE: Attribute transmission too short	0x0C850080
0x3003	Ecat SoE: Attribute transmission too long	0x0C850081
0x3004	Ecat SoE: Attribute cannot be changed (read only)	0x0C850082
0x3005	Ecat SoE: Attribute is write-protected at this time	0x0C850083
0x4001	Ecat SoE: No unit	0x0C850084
0x4002	Ecat SoE: Unit transmission too short	0x0C850085
0x4003	Ecat SoE: Unit transmission too long	0x0C850086
0x4004	Ecat SoE: Unit cannot be changed (read only)	0x0C850087
0x4005	Ecat SoE: Unit currently write-protected	0x0C850088
0x5001	Ecat SoE: No minimum input value	0x0C850089
0x5002	Ecat SoE: Min. input value transmission too short	0x0C85008A
0x5003	Ecat SoE: Min. input value transmission too long	0x0C85008B
0x5004	Ecat SoE: Min. input value cannot be changed (read only)	0x0C85008C
0x5005	Ecat SoE: Min. input value is write-protected at this time	0x0C85008D
0x6002	Ecat SoE: Max. input value transmission too short	0x0C85008F
0x6003	Ecat SoE: Max. input value transmission too long	0x0C850090
0x6004	Ecat SoE: Max. input value cannot be changed (read only)	0x0C850091
0x6005	Ecat SoE: Max. input value is write-protected at this time	0x0C850092
0x7001	Ecat SoE: No operation data	0x0C850093
0x7002	Ecat SoE: Operation data transmission too short	0x0C850094
0x7003	Ecat SoE: Operation data transmission too long	0x0C850095
0x7004	Ecat SoE: Operation data cannot be changed (read only)	0x0C850096

Sercos error code	Description	ECM error code
0x7005	Ecat SoE: Operation data is write-protected at this time	0x0C850097
0x7006	Ecat SoE: Operation data is smaller than the min value	0x0C850098
0x7007	Ecat SoE: Operation data is greater than the max value	0x0C850099
0x7008	Ecat SoE: Invalid operation data	0x0C85009A
0x7009	Ecat SoE: Operation data write-protected by a password	0x0C85009B
0x700A	Ecat SoE: Operation data is write-protected (cycl. conf)	0x0C85009C
0x700B	Ecat SoE: Invalid indirect addr., (data container, list)	0x0C85009D
0x700C	Ecat SoE: Operation data is write-protected (other)	0x0C85009E
0x7010	Ecat SoE: Procedure command already active	0x0C85009F
0x7011	Ecat SoE: Procedure command not interruptible	0x0C850100
0x7012	Ecat SoE: Procedure command at this time not executable	0x0C850101
0x7013	Ecat SoE: Procedure command not executable	0x0C850102
n/a	Ecat SoE: Response drive number mismatch	0x0C850103
n/a	Ecat SoE: Response IDN mismatch	0x0C850104
n/a	Ecat SoE: At least one fragment lost	0x0C850105
n/a	Ecat SoE: RX buffer full	0x0C850106
n/a	Ecat SoE: No data state	0x0C850107
n/a	Ecat SoE: No default value	0x0C850108
n/a	Ecat SoE: Default value transmission too long	0x0C850109
n/a	Ecat SoE: Default value cannot be changed	0x0C85010A
n/a	Ecat SoE: Invalid drive number	0x0C85010B
n/a	Ecat SoE: General error	0x0C85010C
n/a	Ecat SoE: No element addressed	0x0C85010D

10 FAQs

10.1 ctrlX DRIVE: How to configure SoE using Free Run?

- Adding **ctrlX DRIVE as slave device**
  ctrlX DRIVE create IndraDrive with a respective version below the EtherCAT master in the Project Explorer, for example MPB20:
- **Settings in the slave dialog “General”**
  For the “Free Run” operation, select “Free Run” in the “Distributed Clock” setting. This is a setting value from the device description file of the slave (here: from IndraDrive MPB20).
  Additionally, set the checkmark in front of ”Enable expert settings” in this dialog to show the additional dialog “Expert mode process data”.
- **Settings in the slave dialog “Expert process data”**
  The requested process data for the drive can be configured manually in this dialog.
  Select either S-0-0016 (AT) or S-0-0024 (MDT) in the field “PDO list” on the upper right. Then, the currently configured cyclic parameters are shown in the lower right “PDO content”. Use the functions “Add/Edit/Delete” in the bottom dialog on the right to manually configure the parameters for AT (drive telegram) or MDT (master data telegram).
  If there are no predefined parameters in the device description of the slave (as shown in the following screenshot), name, parameter number (IDN) and data type have to be entered correctly and manually.
- **Settings in the slave dialog “Startup parameters”**
  For the Free Run operation, create the starting parameter S-0-0002 (“Sercos cycle time”) with the same value as S-0-0001 (“NC cycle time”). While S-0-0001 has already been predefined as starting parameter via the device description, S-0-0002 has to be set manually as 16-bit parameter.
  The value of S-0-0001 corresponds to the EtherCAT cycle time and it is automatically set by the configurator. Set the same value for S-0-0002.
- **Expert settings “SyncManager”**
  The SyncManager configuration of SoE is set to “ 1 buffer system” by default.
  For the “Free Run” mode, the setting “3 buffers” is recommended. Refer to ➥ Configuring the EtherCAT field bus devices.

10.2 ctrlX DRIVE: How to connect ctrlX DRIVE Engineering via EtherCAT?

The ctrlX DRIVE Engineering can set up a communication to the ctrlX DRIVE at the Rexroth EtherCAT master either with SoE or EoE using the ctrlX Data Layer.

Variant A

Connecting the ctrlX DRIVE Engineering to the EtherCAT master via the ctrlX Data Layer

In the ctrlX DRIVE Engineering in the “Selecting the connection” dialog, under the Control tab, select “ctrlX CORE” as type.

Enter the engineering port of the ctrlX CORE as IP address. ctrlX DRIVE Engineering uses a ctrlX Data Layer connection to the ctrlX CORE control that communicates acyclically using the mailbox SoE.

The variant can only be used for drives with SoE profile. An EoE configuration or IP routing settings on the engineering PC are not required.

The following variants B.) and C.) are recommended for the ctrlX DRIVE with EoE.

Variant B

ctrlX DRIVE Engineering with EoE and switch port

With a switch port, e.g. Beckhoff EL6601, Ethernet devices can be connected decentrally to an EtherCAT network. An IP communication is thus possible with the EtherCAT slave.

For ctrlX DRIVE, an EoE configuration has to be set. In this configuration, the IP settings have to be in the same network as those of the engineering PC, refer to Fig. 8.

In IndraWorks ctrlX DRIVE Engineering, a connection can be set up via the network or IP address search. An IP routing is not required on the engineering PC.

Variant C

ctrlX DRIVE Engineering with EoE and IP routing in the EtherCAT master

The engineering PC with ctrlX DRIVE Engineering is connected to the engineering port of the control and the control executes an IP routing on EoE to the EtherCAT network.

To set up a valid IP configuration in the EtherCAT network, set an EoE configuration for ctrlX DRIVE and the EtherCAT master, also refer to Fig. 8.

Before setting up a connection in ctrlX DRIVE Engineering via the network or IP address search, add a respective routing entry manually into the IP routing table of the engineering PC (e.g. refer to the Windows command line, command “route help”).

11 Related documentation

11.1 Overview

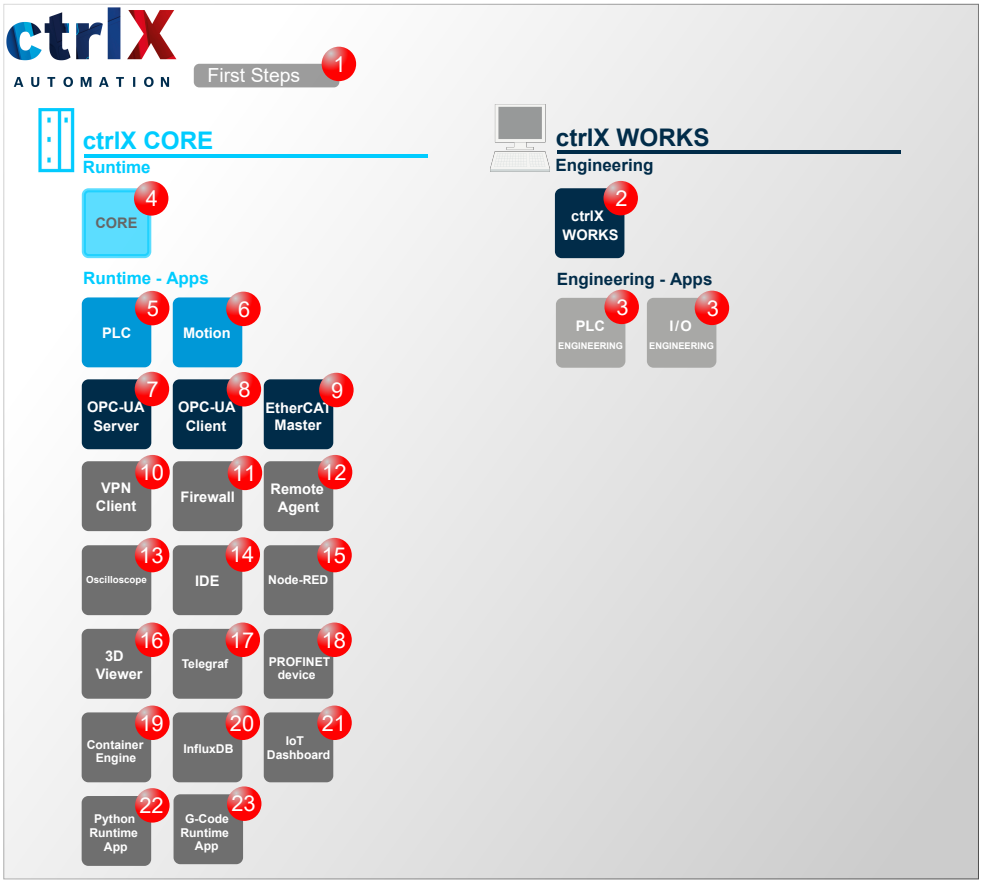

Fig. 24: Overview on further documentations

11.2 ctrlX AUTOMATION

No.	Documentation
1	**ctrlX WORKS First Steps 01VRS** Quick Start Guide ➦ Web documentation link Ordering information: • DOK-XWORKS-F*STEP**V01-QURS-EN-P • R911403760

11.3 ctrlX WORKS

No.	Documentation
2	**ctrlX WORKS Basic System 01VRS** Application Manual ➦ Web documentation link Ordering information: • DOK-XWORKS-********V01-APRS-EN-P • R911403761
3	**ctrlX PLC Engineering - PLC Programming System 01VRS** Application Manual ➦ Web documentation link Ordering information: • DOK-XPLC**-ENG*****V01-APRS-EN-P • R911403764
3	**ctrlX PLC Engineering - PLC Libraries 01VRS** Reference ➦ Web documentation link Ordering information: • DOK-XPLC**-LIBRARY*V01-RERS-EN-P • R911403766

11.4 ctrlX CORE

Nr.	Dokumentation
4	**ctrlX CORE - Runtime 01VRS** Application Manual ➦ Web documentation link Ordering information: • DOK-XCORE*-BASE****V01-APRS-EN-P • R911403768
	ctrlX CORE - Nodes of the Data Layer 01VRS Reference ➦ Link zur Web-Dokumentation Bestellinformationen: • DOK-XCORE*-BASE*DL*V01-RERS-EN-P • R911420072
	ctrlX CORE - Diagnostics 01VRS Reference ➦ Web documentation link Ordering information: • DOK-XCORE*-DIAG****V01-RERS-EN-P • R911403770

11.5 ctrlX CORE Apps

Nr.	Dokumentation
5	**PLC App - PLC Runtime Environment for ctrlX CORE 01VRS** Application Manual ➦ Web documentation link Ordering information: • DOK-XCORE*-PLC*****V01-APRS-EN-P • R911403787
6	**Motion App - Motion Runtime Environment for ctrlX CORE 01VRS** Application Manual ➦ Web documentation link Ordering information: • DOK-XCORE*-MOTION**V01-APRS-EN-P • R911403791
7	**OPC UA Server App - OPC UA Server for ctrlX CORE 01VRS** Application Manual ➦ Link zur Web-Dokumentation Ordering information: • DOK-XCORE*-OPCSERV*V01-APRS-EN-P • R911403778
8	**OPC UA Client App - OPC UA Client for ctrlX CORE 01VRS** Application Manual ➦ Web documentation link Ordering information: • DOK-XCORE*-OPCCLIENV01-APRS-EN-P • R911403781
9	**EtherCAT Master App - EtherCAT Master for ctrlX CORE 01VRS** Application Manual ➦ Web documentation link Ordering information: • DOK-XCORE*-ETHERCATV01-APRS-EN-P • R911403773
10	**VPN Client App - Remote Support Software for ctrlX CORE 01VRS** Application Manual ➦ Web documentation link Ordering information: • DOK-XCORE*-VPN*****V01-APRS-EN-P • R911403775
11	**Firewall App - Security Functions for ctrlX CORE 01VRS** Application Manual ➦ Web documentation link Ordering information: • DOK-XCORE*-FIREWALLV01-APRS-EN-P • R911403783

Nr.	Dokumentation
12	**Remote Agent App - ctrlX Device Portal Connection for ctrlX Devices 01VRS** Application Manual ➥ Web documentation link Ordering information: • DOK-XCORE*-REMOTE**V01-APRS-EN-P • R911403785
13	**Oscilloscope App - Oscilloscope Function for ctrlX Devices 01VRS** Application Manual ➥ Web documentation link Ordering information: • DOK-XCORE*-OSCI****V01-APRS-EN-P • R911409806
14	**IDE App - Integrated Development Environment 01VRS** Application Manual ➥ Web documentation link Ordering information: • DOK-XCORE*-IDE*****V01-APRS-EN-P • R911410625
15	**Node RED App - Graphic Programming for ctrlX CORE 01VRS** Application Manual ➥ Web documentation link Ordering information: • DOK-XCORE*-NODERED*V01-APRS-EN-P • R911403789
16	**3D Viewer App - Browser-based 3D Kinematic Simulation for ctrlX CORE 01VRS** Application Manual ➥ Web documentation link Ordering information: • DOK-XCORE*-3D*VIEW*V01-APRS-EN-P • R911416124
17	**Telegraf App - Server Agent for Collecting Data in the Data Layer 01VRS** Application Manual ➥ Web documentation link Ordering information: • DOK-XCORE*-TELEGRAFV01-APRS-EN-P • R911416836
18	**PROFINET Device App - PROFINET Device for ctrlX CORE 01VRS** Application Manual ➥ Web documentation link Ordering information: • DOK-XCORE*-PROFINETV01-APRS-EN-P • R911417857

Nr.	Dokumentation
19	**Container Engine App - Use of Docker® Images on ctrlX CORE 01VRS** Application Manual ➦ Web documentation link Ordering information: • DOK-XCORE*-DOCKER**V01-APRS-EN-P • R911417855
20	**InfluxDB App - Influx Database Connection for ctrlX CORE 01VRS** Application Manual ➦ Web documentation link Ordering information: • DOK-XCORE*-INFLUX**V01-APRS-EN-P • R911418738
21	**IoT Dashboard App - Data Visualization in Dynamic, Interactive Dashboards 01VRS** Application Manual ➦ Web documentation link Ordering information: • DOK-XCORE*-GDB*****V01-APRS-EN-P • R911420426
22	**Python Runtime App - Python Runtime Environment for ctrlX CORE 01VRS** Application Manual ➦ Link zur Web-Dokumentation Ordering information: • DOK-XCORE*-PYR*****V01-APRS-EN-P • R911420430
23	**G-Code Runtime App - G-Code Interpreter for ctrlX CORE 01VRS** Application Manual ➦ Link zur Web-Dokumentation Ordering information: • DOK-XCORE*-GCO*****V01-APRS-EN-P • R911420428

12 Service and support

Our worldwide service network provides an optimized and efficient support. Our experts provide you with advice and assistance. You can contact us **24/7**.

Service Germany

Our technology-oriented Competence Center in Lohr, Germany, is responsible for all your service-related queries for electric drive and controls.

Contact the **Service Hotline** and **Service Helpdesk** under:

Phone:	**+49 9352 40 5060**
Fax:	**+49 9352 18 4941**
Email:	➦ service.svc@boschrexroth.de
Internet:	➦ http://www.boschrexroth.com

Additional information on service, repair (e.g. delivery addresses) and training can be found on our internet sites.

Service worldwide

Outside Germany, please contact your local service office first. For hotline numbers, refer to the sales office addresses on the internet.

Preparing information

To be able to help you more quickly and efficiently, please have the following information ready:

- Detailed description of malfunction and circumstances
- Type plate specifications of the affected products, in particular type codes and serial numbers
- Your contact data (phone and fax number as well as your e-mail address)

13 Index

rexroth
A Bosch Company

Bosch Rexroth AG
Bgm.-Dr.-Nebel-Str. 2
97816 Lohr a.Main
Germany
Tel. +49 9352 18 0
Fax +49 9352 18 8400
www.boschrexroth.com/electrics

R911403773