EtherCAT® Protocol Enhancements

Document: ETG.1020 S (R) V1.2.0

Nomenclature:

ETG-Number	ETG.1020
Type	S (Standard)
State	R (Release)

Created by:	ETG
Contact:	info@ethercat.org
Filename:	ETG1020_v1i2i0_S_R_ProtocolEnhancements.docx
Date:	01.12.2015

EtherCAT® Technology Group

LEGAL NOTICE

DOCUMENT HISTORY

Version	Comment
1.0.0.	First Release of PDF version
1.1.0	Changes according to ETG1020_CommentResolution_V1i0i0R.zip
1.2.0	Changes according to ETG1020_CommentFormV1i1i0D_2015-12-01.doc

CONTENTS

TABLES

FIGURES

ABBREVIATIONS

µC	Microcontroller
C	Conditional
CMD	Command
CoE	CANopen over EtherCAT
DC	Distributed Clock
DPRAM	Dual-Ported RAM
ENI	EtherCAT Network Information (EtherCAT XML Master Configuration)
EoE	Ethernet over EtherCAT
ESC	EtherCAT Slave Controller
ESI	EtherCAT Slave Information (EtherCAT Devices Description)
ESM	EtherCAT State Machine
ETG	EtherCAT Technology Group
FMMU	Fieldbus Memory Management Unit
FoE	File Access over EtherCAT
FPMR	Configured Address Physical Read Multiple Write
FPRD	Configured Address Physical Read
FPRW	Configured Address Physical ReadWrite
FPWR	Configured Address Physical Write
I/O	Input/Output
IDN	Identification Number (Servo Profile Identifier)
IEC	International Electrotechnical Commission
INT	Integer
IRQ	Interrupt Request
LRD	Logical Read
LRW	Logical ReadWrite
LSB	Least Significant Bit
LWR	Logical Write
M	Mandatory
MAC	Media Access Controller
MI	(PHY) Management Interface
MII	Media Independent Interface
MSB	Most Significant Bit
NIC	Network Interface Card
NOP	No Operation
ns	nanoseconds (10^{-9} seconds)
O	Optional
OD	Object Dictionary
OS	Oversampling
PDO	Process Data Object
PREOP	Pre-Operational
RO	Read Only
RD	Read
SDO	Service Data Object
SI	SubIndex
SM	SyncManager
SoE	Servo Profile over EtherCAT
SOF	Start of Frame
SPI	Serial Peripheral Interface
SU	Sync Unit
WD	Watchdog
WKC	Working Counter
WO	Write Only
WR	Write
XML	eXtensible Markup Language

1 Scope

This specification contains enhancements of the EtherCAT Protocol. These enhancements are part of the EtherCAT specification and may become part of the ETG.1000 series in the future.

2 References

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ETG Standards

[1] ETG.1000.2: Physical Layer service definition and protocol specification

[2] ETG.1000.3: Data Link Layer service definition

[3] ETG.1000.4: Data Link Layer protocol specification

[4] ETG.1000.5: Application Layer service definition

[5] ETG.1000.6: Application Layer protocol specification

[6] ETG.2000: EtherCAT Slave Information specification

[7] ETG.5001: EtherCAT Modular Device Profiles

[8] ETG.5100: Safety over EtherCAT specification

Other References

[9] IEC 61158-x-12 (all parts for type 12): Industrial communication networks – Fieldbus specifications

[10] IEC 61784-2: Industrial communication networks – Profiles – Part 2: Additional fieldbus profiles for real-time networks based on ISO/IEC 8802-3

[11] IEC 61800-7: Adjustable speed electrical power drives systems – Part 7: Generic interface and use of profiles for power drive systems

3 Terms, Definitions and Word Usage

3.1 Terms and Definitions

The terms and definitions of ETG.1000 series shall be fully valid, unless otherwise stated.

3.2 Word usage: shall, should, may, can

The word *shall* is used to indicate mandatory requirements strictly to be followed in order to conform to the standard and from which no deviation is permitted (*shall* equals *is required to*).

The word *should* is used to indicate that among several possibilities one is recommended as particularly suitable, without mentioning or excluding others; or that a certain course of action is preferred but not necessarily required; or that (in the negative form) a certain course of action is deprecated but not prohibited (*should* equals *is recommended that*).

The word *may* is used to indicate a course of action permissible within the limits of the standard (*may* equals *is permitted to*).

The word *can* is used for statements of possibility and capability, whether material, physical, or causal (*can* equals *is able to*).

4 Description of AL Status Codes

AL Status Codes are defined in ETG.1000.6, [5]. This chapter defines additional codes and provides additional information for already existing AL Status Codes, see Table 1.

Table 1: Description of AL Status Codes usage

Code	Meaning	Description	Reference
0x0000	No error	No error	ETG.1000.6
0x0001	Unspecified error	General Error which is not defined in the following list	ETG.1000.6
0x0002	No Memory	Local Application runs out of memory (e.g. dynamic memory allocation for emergency messages)	ETG.1000.6
0x0003	Invalid Device Setup	This AL Status code should be used if the EtherCAT Slave has no valid setup of application specific settings. E.g. if a bus-coupler has no physical modules connected PreOp->SafeOp, final state ErrPreop	Additional code
0x0006	SII/EEPROM information does not match firmware	if the slave checks if the SII/EEPROM content matches the firmware, e.g. process data description or revision number, and detects a mismatch. Init -> PreOp, final state ErrInit	Additional code
0x0007	Firmware update not successful. Old firmware still running	Error occurred during firmware update	Additional code
0x000E	License error	HW/SW license invalid or evaluation period expired Current State: Any Resulting State: I + E Use case: if ESC IP-Core runs in a time-limited evaluation version, e.g. with a license chip. Using this in a product should be very carefully and noticeably documented because it may stop a EtherCAT system in some applications.	Additional code
0x0011	Invalid requested state change	Requested state change is invalid (e.g. IS, IO,PO,PB,SB,OB)	ETG.1000.6
0x0012	Unknown requested state	Requested state change is unknown (e.g. if the requested state code is not 1,2,3,4,8)	ETG.1000.6
0x0013	Bootstrap not supported	The device doesn’t support Bootstrap state	ETG.1000.6
0x0014	No valid firmware	The downloaded file is no valid firmware file	ETG.1000.6
0x0015	Invalid mailbox configuration	The mailbox SyncManger configuration is not valid in Bootstrap state	ETG.1000.6
0x0016	Invalid mailbox configuration	The mailbox SyncManger configuration is not valid in PREOP state	ETG.1000.6
0x0017	Invalid sync manager configuration	Sync manager configuration is not valid	ETG.1000.6
0x0018	No valid inputs available	Slave application cannot provide valid inputs	ETG.1000.6
0x0019	No valid outputs	Slave application cannot receive valid outputs	ETG.1000.6
0x001A	Synchronization error	Multiple synchronization errors. Device is not synchronized any more (shall be used if no specific Synchronization error can be distinguished)	ETG.1000.6
0x001B	Sync manager watchdog	No process data received yet (S->O) or not received within a specified timeout value	ETG.1000.6
0x001C	Invalid Sync Manager Types		ETG.1000.6
0x001D	Invalid Output Configuration	SyncManger configuration for output process data is invalid	ETG.1000.6
0x001E	Invalid Input Configuration	SyncManger configuration for input process data is invalid	ETG.1000.6

Code	Meaning	Description	Reference
0x001F	Invalid Watchdog Configuration	Watchdog Settings are invalid (e.g. SyncManger watchdog trigger is enabled but no watchdog timeout is defined)	ETG.1000.6
0x0020	Slave needs cold start	Slave device requires a power off – power on reset	ETG.1000.6
0x0021	Slave needs INIT	Slave application requests INIT state	ETG.1000.6
0x0022	Slave needs PREOP	Slave application requests PREOP state	ETG.1000.6
0x0023	Slave needs SAFEOP	Slave application requests SAFEOP state	ETG.1000.6
0x0024	Invalid Input Mapping	Input process data mapping do not match to expected mapping	ETG.1000.6
0x0025	Invalid Output Mapping	Output process data mapping do not match to expected mapping	ETG.1000.6
0x0026	Inconsistent Settings	General settings mismatch	ETG.1000.6
0x0027	Freerun not supported		ETG.1000.6
0x0028	Synchronization not supported		ETG.1000.6
0x0029	Freerun needs 3 Buffer Mode	FreeRun Mode, sync manager has to run in 3-Buffer Mode	ETG.1000.6
0x002A	Background Watchdog		ETG.1000.6
0x002B	No Valid Inputs and Outputs		ETG.1000.6
0x002C	Fatal Sync Error	Fatal Sync Error: Sync0 or Sync1 are not received any more	ETG.1000.6
0x002D	No Sync Error	Sync not received: In SAFEOP the slave waits for the first Sync0/Sync1 events before switching to OP, if these events were not received during the SAFEOP to OP-Timeout time the slave should refuse the state transition to OP with this AL Status Code (SystemTimeOffset too big, no DC event received)	ETG.1000.6
0x002E	Cycle time too small	EtherCAT cycle time is smaller than Minimum Cycle Time supported by slave SafeOp-> Op, final state SafeOpErr	Additional Code
0x0030	Invalid DC SYNC Configuration	Distributed Clocks configuration is invalid due to application requirements	ETG.1000.6
0x0031	Invalid DC Latch Configuration	DC Latch configuration is invalid due to application requirements	ETG.1000.6
0x0032	PLL Error	Master not synchronized, at least one DC event received	ETG.1000.6
0x0033	DC Sync IO Error	Multiple synchronization errors. IO is not synchronized any more	ETG.1000.6
0x0034	DC Sync Timeout Error	Multiple synchronization errors. Too much SM Events missed	ETG.1000.6
0x0035	DC Invalid Sync Cycle Time		ETG.1000.6
0x0036	DC Sync0 Cycle Time	DC Sync0 Cycle time does not fit to application requirements	ETG.1000.6
0x0037	DC Sync1 Cycle Time	DC Sync1 Cycle time does not fit to application requirements	ETG.1000.6
0x0041	MBX_AOE		ETG.1000.6
0x0042	MBX_EOE		ETG.1000.6
0x0043	MBX_COE		ETG.1000.6
0x0044	MBX_FOE		ETG.1000.6
0x0045	MBX_SOE		ETG.1000.6

Code	Meaning	Description	Reference
0x004F	MBX_VOE		ETG.1000.6
0x0050	EEPROM No Access	EEPROM not assigned to PDI	ETG.1000.6
0x0051	EEPROM Error	EEPROM access error	ETG.1000.6
0x0052	External Hardware not ready	This AL Status Code should be used if the EtherCAT-Slave refused the state transition due to an external connection to another device or signal is missing	Additional code
0x0060	Slave Restarted Locally		ETG.1000.6
0x0061	Device Identification value updated	The device identification value was updated and is now valid	ETG.1000.6
0x0070	Detected Module Ident List does not match	Configured Module Ident List (0xF030) and Detected Module Ident list (0xF050) does not match PreOp->SafeOp, final state ErrPreop Should be used for devices in which the physical hardware configuration (0xF030) determines the expected configuration	Additional code
0x00F0	Application Controller available	The local application releases the application controller which is now available and services the EtherCAT state machine and all other device features This optional AL Status Code shall only be used for devices that have a different power supply for the ESC and the application controller and which cannot define a maximum timeout value for transition I→P NOTE: Use case can be if ESC is powered before application controller. Err Indication can be used by Master to indicate that slave is ready for boot-up (instead of cyclic polling)	Additional code

5 EEPROM Access

By default, EtherCAT master side has the access rights to the EEPROM. The EtherCAT master shall give the EEPROM access rights to the application controller by writing 0x0500.0 = 1 in the following transitions and states:

- in transition INIT – PREOP
- in transition INIT – BOOT and in state BOOT.
- If the Element “AssignToPdi” is set in the ESI-File the master shall grant the access right to the PDI in all states except INIT.

During power-up the ESC loads the first seven words from the EEPROM to configure its PDI. ESC is always accessible via EtherCAT, even if PDI Configuration Data (or even the whole EEPROM content), is filled with wrong data. That means the master/configuration logic can still rewrite EEPROM with updated information.

During production the EEPROM shall be loaded with the correct values; at least the first seven words of the EEPROM.

NOTE: To allow the application controller to check EEPROM content for correct content it might be useful to connect the two EEPROM I2C-signals to the controller. The controller can use the access to the EEPROM during power-up, before releasing the reset signal to the ESC.

6 Mailbox InitCmds

During boot-up of a device the master can send Init commands to slave’s register (Register InitCmd) and to the mailbox (Mailbox InitCmd).

Sequence of Mailbox InitCmds and Register InitCmds depends on state transition:

- State transition from Init:
  Register InitCmds shall be send first and Mailbox InitCmds afterwards.
  Mailbox InitCmds can be sent in state transition IP and IB if the sequence is kept. This allows configuration of Mailbox-Register, e.g. to send EoE InitCmds in transition IP.
- other transitions:
  Mailbox InitCmds shall be send first and afterwards the corresponding Register Init Cmds. This allows the slave to verify the configuration.

7 ADS over EtherCAT (AoE)

An acyclic service to a fieldbus slave normally needs request and response data so that a SDO Download/SDO Upload combination would be necessary. This could be done with a command object but there is always an extended communication overhead because at least two complete SDO services have to be transmitted to get a successful acyclic service transmission.

To reach a good enough performance parallel services are needed that acyclic service requests to different fieldbus slaves can be transmitted at the same time, but the SDO service cannot be used in parallel.

Therefore it makes sense to use the AoE services (ADS over EtherCAT) for accessing fieldbus slaves (or modules) with acyclic services. AoE services can be transmitted in parallel because every AoE command contains an Invoke ID which identifies the request.

For AoE services the mailbox type 1 is used.

7.1 AoE Mailbox Frame

The mailbox frame coding is defined in Table 2:

Table 2: Mailbox frame with AoE

Frame part	Data Field	Data Type	Value / Description
Mailbox Header	Length	WORD	Length of the mailbox data
	Address	WORD	Station Address of the source, if a master is client, Station Address of the destination, if a slave is client
	Channel	UNSIGNED6	0x00 (reserved for future)
	Priority	UNSIGNED2	0x00 (reserved for future)
	Type	UNSIGNED4	0x01: AoE
	Cnt	UNSIGNED3	Counter of the mailbox services (0 reserved, 1 is start value, next value after 7 is 1)
	Reserved	UNSIGNED1	0x00
AoE Header		OCTET-STRING[32]	see Table 3
AoE Command		OCTET-STRING	Command structure according to AoE Command in AoE Header

The AoE Header is defined in Table 3:

Table 3: AoE Header

AoE Header	Offset (in Bytes)	Data Type	Description
Target Net ID[1]	0	OCTET-STRING[6]	Destination Net-ID
Target Port[2]	6	UNSIGNED16	Destination Port
Sender Net ID	8	OCTET-STRING[6]	Source Net-ID
Sender Port	14	UNSIGNED16	Source Port
Command ID	16	UNSIGNED16	Command ID of the AoE service, see Table 4
State Flags	18	UNSIGNED16	Bit 0: 0 = Request, 1 = Response Bit 2: 1 Bit 1, 3…15: 0
Data Size	20	UNSIGNED32	size of the data following the AoE header
Error Code	24	UNSIGNED32	error code (only for responses)
Invoke ID	28	UNSIGNED32	Invoke ID (will be copied unchanged from request to response)

[1]The Net ID defines a unique identification of a device used for the AoE communication.

[2]The Port number shall be unique within the device.

The AoE Commands are defined in Table 4.

Table 4: AoE Commands

AoE Command	Commmand ID	Description
AoE Read	2	AoE Read command to read data
AoE Write	3	AoE Write command to write data
AoE WriteControl	5	AoE Write Control command change ADS state and device state and write data
AoE ReadWrite	9	AoE ReadWrite command to exchange data in both directions (read and write)
AoE Fragmentation	0x902	AoE Fragmentation command (is used if the AoE command does not fit in the mailbox)

7.2 AoE Commands

7.2.1 AoE Read

The AoE Read Request Header is defined in Table 5.

Table 5: AoE Read Request Header

AoE Read Request Header	Offset (in Bytes)	Data Type	Description
Index Group	0	UNSIGNED32	Index Group to address the requested data see clause 7.5
Index Offset	4	UNSIGNED32	Index Offset to address the requested data see clause 7.5
Read Length	8	UNSIGNED32	length of the data to be read

The AoE Read Response Header is defined in Table 6.

Table 6: AoE Read Response Header

AoE Read Response Header	Offset (in Bytes)	Data Type	Description
Result	0	UNSIGNED32	Result (0 = Success) of the read operation
Read Length	4	UNSIGNED32	length of the following data
Read Data	8	OCTET-STRING	read data

7.2.2 AoE Write

The AoE Write Request Header is defined in Table 7.

Table 7: AoE Write Request Header

AoE Write Request Header	Offset (in Bytes)	Data Type	Description
Index Group	0	UNSIGNED32	Index Group to address the requested data see clause 7.5
Index Offset	4	UNSIGNED32	Index Offset to address the requested data see clause 4.4
Write Length	8	UNSIGNED32	length of the following data
Write Data	12	OCTET-STRING	data to be written

The AoE Write Response Header is defined in Table 8.

Table 8: AoE Write Response Header

AoE Write Response Header	Offset (in Bytes)	Data Type	Description
Result	0	UNSIGNED32	Result (0 = Success) of the write operation

7.2.3 AoE WriteControl

The AoE WriteControl Request Header is defined in Table 9.

Table 9: AoE WriteControl Request Header

AoE WriteControl Request Header	Offset (in Bytes)	Data Type	Description
AoE State	0	UNSIGNED16	AoE State (standardized)
Device State	2	UNSIGNED16	device specific state
Write Length	4	UNSIGNED32	length of the following data
Write Data	8	OCTET-STRING	data to be written

The AoE WriteControl Response Header is defined in Table 10.

Table 10: AoE WriteControl Response Header

AoE WriteControl Response Header	Offset (in Bytes)	Data Type	Description
Result	0	UNSIGNED32	Result (0 = Success) of the write operation

7.2.4 AoE ReadWrite

The AoE ReadWrite Request Header is defined in Table 11.

Table 11: AoE ReadWrite Request Header

AoE ReadWrite Request Header	Offset (in Bytes)	Data Type	Description
Index Group	0	UNSIGNED32	Index Group to address the requested data
Index Offset	4	UNSIGNED32	Index Offset to address the requested data
Read Length	8	UNSIGNED32	length of the data to be read
Write Length	12	UNSIGNED32	length of the following data
Write Data	16	OCTET-STRING	data to be written

The AoE ReadWrite Response Header is defined in Table 12.

Table 12: AoE ReadWrite Response Header

AoE ReadWrite Response Header	Offset (in Bytes)	Data Type	Description
Result	0	UNSIGNED32	Result (0 = Success) of the read operation
Read Length	4	UNSIGNED32	length of the following data
Read Data	8	OCTET-STRING	read data

7.3 AoE Fragmentation

The AoE Fragmentation command uses the Invoke ID for the fragmentation of the AoE command to be transmitted (see Table 13).

Table 13: AoE Header for AoE Fragmentation command

AoE Header	Offset (in Bytes)	Data Type	Description
Target Net ID	0	OCTET-STRING[6]	Destination Net-ID
Target Port	6	UNSIGNED16	Destination Port
Sender Net ID	8	OCTET-STRING[6]	Source Net-ID
Sender Port	14	UNSIGNED16	Source Port
Command ID	16	UNSIGNED16	**0x902**
State Flags	18	UNSIGNED16	0x0004
Data Size	20	UNSIGNED32	size of the data following the AoE header
Error Code	24	UNSIGNED32	error code (only for responses)
Fragment Number	28	UNSIGNED16	increasing number with every fragment starting with 0
Packet Number	30	UNSIGNED16	increasing number with every new fragmented AoE command (shall be the same for all fragments of one AoE command)

The AoE command will be split in several AoE fragments depending on the mailbox and AoE command size. The first 16 bytes of the AoE command shall not be transmit (because they are included in the AoE fragmentation header). So an AoE command with size 500 and mailbox size 100 will be transmitted with 8 fragments as shown in Table 14.

Table 14: AoE Fragmentation Example

Fragment Number	Mailbox header	AoE Fragmentation header	AoE command
0	6 bytes	32 bytes	62 bytes (offset 16-77)
1	6 bytes	32 bytes	62 bytes (offset 78-139)
2	6 bytes	32 bytes	62 bytes (offset 140-201)
3	6 bytes	32 bytes	62 bytes (offset 202-263)
4	6 bytes	32 bytes	62 bytes (offset 264-325)
5	6 bytes	32 bytes	62 bytes (offset 326-387)
6	6 bytes	32 bytes	62 bytes (offset 388-449)

7.4 AoE Initialization

The AoE Net ID for the EtherCAT Slave will be transported with a special AoE Write command in state transition to PREOP as shown in Table 15 and Table 16.

Table 15: AoE Header for the AoE Initialization Write Command

AoE Header	Offset (in Bytes)	Data Type	Description
Target Net ID	0	OCTET-STRING[6]	AoE Net-ID of the EtherCAT Master
Target Port	6	UNSIGNED16	EtherCAT Address of the EtherCAT Slave (Register 0x10-0x11)
Sender Net ID	8	OCTET-STRING[6]	**AoE Net-ID of the EtherCAT Master**
Sender Port	14	UNSIGNED16	EtherCAT Address of the EtherCAT Slave (Register 0x10-0x11)
Command ID	16	UNSIGNED16	3 = AoE Write
State Flags	18	UNSIGNED16	Bit 0: 0 = Request, 1 = Response Bit 2: 1 Bit 1, 3…15: 0
Data Size	20	UNSIGNED32	12+6
Error Code	24	UNSIGNED32	0
Invoke ID	28	UNSIGNED32	Invoke ID (will be copied unchanged from request to response)

Table 16: AoE Write Request Header for the AoE Initialization Write Command

AoE Write Request Header	Offset (in Bytes)	Data Type	Description
Index Group	0	UNSIGNED32	**1**
Index Offset	4	UNSIGNED32	**3**
Write Length	8	UNSIGNED32	6
Write Data	12	OCTET-STRING[6]	AoE Net-ID of the EtherCAT Slave

7.5 Mailbox Services with AoE

AoE can embed other mailbox protocols. With AoE the mailbox services can be transmitted from the EtherCAT master to the EtherCAT Slave. The EtherCAT Slave can transmit the mailbox services to the underlying modules by the following alternatives:

- via a profile specific protocol (e.g. fieldbus gateways)

- via the mailbox protocol on the module's communication protocol
- by transmitting AoE to modules

The underlying modules can have their own object dictionary that can be accessed via embedded CoE protocol or a new firmware can be downloaded via embedded FoE.

A module will be addressed by its own port; the mailbox service will be selected by the index group. Details about the own dictionary and the own ports can be found in section "Modular Object Dictionary", which is part of the Modular Device Profiles, see [7].

7.5.1 CoE Services

For the CoE services the index groups shown in Table 17 are defined.

Table 17: Index Group of the CoE Services

CoE Service	AoE Service	AoE Port	AoE IndexGroup	AoE IndexOffset	AoE Data
SDO Upload	Read	EtherCAT Slave Address or 0xFFFF for Master	0xF302	Bit 16-31: Index Bit 8: Complete Access Bit 0-7: SubIndex	Response: read data
SDO Download	Write	EtherCAT Slave Address or 0xFFFF for Master	0xF302	Bit 16-31: Index Bit 8: Complete Access Bit 0-7: SubIndex	Request: data to be written
SDO Information Get Object List	Read	EtherCAT Slave Address or 0xFFFF for Master	0xF3FC	Bit 16-31: List Type Request	Response: Word 0: List Type Word 1..n: List
SDO Information Get Object Description	Read	EtherCAT Slave Address or 0xFFFF for Master	0xF3FD	Bit 16-31: Index Request	Response: Object Description see [5]
SDO Information Get Entry Description	Read	EtherCAT Slave Address or 0xFFFF for Master	0xF302	Bit 16-31: Index Bit 8-15: ValueInfo Bit 0-7: SubIndex	Response: Entry Description see [5]

7.5.2 FoE Services

The mapping of the FoE services in the AoE protocol is defined in Table 18

Table 18: AoE Mapping of FoE Services

FoE Service	AoE Service	AoE Port	AoE IndexGroup	AoE IndexOffset	AoE Data
FoE OpenRead	ReadWrite	EtherCAT Slave Address or 0xFFFF for Master	0xF401	Password	Request: File name Response: File handle (4 bytes and optional error text)
FoE Read	ReadWrite	EtherCAT Slave Address or 0xFFFF for Master	0xF404	File handle	Request: - Response: File data
FoE OpenWrite	ReadWrite	EtherCAT Slave Address or 0xFFFF for Master	0xF402	Password	Request: File name Response: File handle (4 bytes and optional error text)
FoE Write	ReadWrite	EtherCAT Slave Address or 0xFFFF for Master	0xF405	File handle	Request: File data Response: -

7.5.3 SoE Services

The mapping of the SoE services in the AoE protocol is defined in Table 19

Table 19: AoE Mapping of SoE Services

SoE Service	AoE Service	AoE Port	AoE IndexGroup	AoE IndexOffset	AoE Data
SoE Read	Read	EtherCAT Slave Address	0xF420	Bit 0-15: IDN Bit 16: Data State Bit 17: Name Bit 18: Attribute Bit 19: Unit Bit 20: Min Value Bit 21: Max Value Bit 22: Value Bit 23: Default value Bit 24-26: Drive Bit 27-31: 0	Request: - Response: data
SoE Write	Write	EtherCAT Slave Address	0xF420	Bit 0-15: IDN Bit 16: Data State Bit 17: Name Bit 18: Attribute Bit 19: Unit Bit 20: Min Value Bit 21: Max Value Bit 22: Value Bit 23: Default value Bit 24-26: Drive Bit 27-31: 0	-

7.5.4 VoE Services

The mapping of the VoE services in the AoE protocol is defined in Table 20

Table 20: AoE Mapping of VoE Services

VoE Service	AoE Service	AoE Port	AoE IndexGroup	AoE IndexOffset	AoE Data
VoE Read / Write	ReadWrite	EtherCAT Slave Address or 0xFFFF for Master	0xF430	0x00	DWORD: Vendor ID WORD: Vendor protocol Type Protocol specific data

8 CoE Abort Codes (extension)

CoE Abort Codes are defined in ETG.1000.6, [5]. Additional codes listed in Table 21 are defined here.

Table 21: CoE Abort Codes (extension)

Value	Meaning
0x06 01 00 03	Subindex cannot be written, SI0 must be 0 for write access
0x06 01 00 04	SDO Complete access not supported for objects of variable length such as ENUM object types
0x06 01 00 05	Object length exceeds mailbox size
0x06 01 00 06	Object mapped to RxPDO, SDO Download blocked This optional Abort Code is used only in states SafeOp and Op.
0x06 09 00 33	configured module list does not match detected module list It shall be used if Object 0xF03x is written but does not fit to object 0xF05x

9 Command Object

The command object should be used for actions or services (commands in the following) which cannot be adapted to a single SDO Upload or SDO Download service; for example if the command needs request and response data or if the processing of the command needs time so that the SDO timeout would expire with a standard transfer.

With the command object the command consists of at least two SDO services:

- By writing the SubIndex 1 with SDO Download of the command object the command is started.
- By reading the SubIndex 3 with SDO Upload the response is fetched. If the response is not available when reading SubIndex 3 the first byte of the response data should give information about the progress.

The SubIndex 1 is not writable until the command is finished and the response is read, so that multiple accesses from different applications cannot overwrite each other. In this case CoE Abort Code 0x08 00 00 21 “Data cannot be transferred or stored to the application because of local control” shall be used.

The SubIndex 2 is only defined for compatibility reasons to the corresponding CANopen DS301 definition. The data type of the command object's object description shall be 0x0025.

The command object structure is shown in Table 22.

Table 22: Command object structure

SubIndex	Description	Data Type	Value
1	Command	OCTET_STRING	Byte 0-n: Service Request Data A write access to the command data will execute the command
2	Status	UNSIGNED8	0: last command completed, no errors, no response data 1: last command completed, no errors, response data available 2: last command completed, error, no response data 3: last command completed, error, response data available 4-99: reserved for future use 100-200: indicates how much of the command has been executed (in %, 100 = 0 %, 200 = 100 %) 201-254: reserved for future use 255: command is executing (if the percentage display is not supported)
3	Response	OCTET_STRING	Byte 0: see SubIndex 2 Byte 1: unused 2-n: Service Response Data

10 Change of Mapping Objects

Some objects contain mapping information that might be variable. For example

- 0x1600…0x17FF Receive PDO Mapping
- 0x1A00…0x1BFF Transmit PDO Mapping
- 0x1C10…0x1C2F Sync manager Channel PDO Assignment
- 0xF01x, 0xF02x, 0xF03x defined in ETG.5001 [7].

To change the content of these objects the following procedure shall be done, if the entries are changed separately, i.e. without CompleteAccess:

1) set SubIndex 0 = 0
   The object is considered disabled when SubIndex 0 has the value 0.
2) configure the mapping information in the mapping entries SubIndex 1 … n
3) set SubIndex 0 = number of used mapping entries

A write access to the SubIndex 1…n while SubIndex 0 does not have the value 0 should be handled in the following way:

- if value = stored value:
  write access is allowed if object is not readonly.
  This can be used to check the device configuration against the master configuration
- if value <> stored value:
  Abort write access with Abort Code 0x06 01 00 03 “Subindex cannot be written, SI0 must be 0 for write access”

With CompleteAccess the whole object can be written at once (if not readonly)

The write-access to the entries might be restricted to a state, e.g. PreOp, of the EtherCAT State Machine.

11 SDO Services

11.1 SDO access single entry with flexible length

Entries with flexible length, as specified in Table 93, may have shorter length than given as the maximum size of the entry.

For entries with data type VISIBLE_STRING the following applies:

Write access

- Master can send a VISIBLE_STRING without 0-terminator:
  slave shall add the 0-terminator if the written VISIBLE_STRING is shorter than the maximum entry size
- Master can also write VISIBLE_STRING with 0-terminator

Read access:

- If the master reads a VISIBLE_STRING the slave should return the VISIBLE_STRING with its actual length.
- Bytes following the 0-terminator shall be zero.

11.2 SDO Complete Access

With the Complete Access all SubIndixes of an object are uploaded or downloaded with a single SDO service. Complete Access should be supported to minimize boot-up time of the device.

For the data of the object there are the following rules for the correct alignment:

1. SubIndex 0 is padded to 16 bit
2. SubIndixes of bit data type (BOOLEAN and BIT1 to BIT7) will be filled over the byte border, the next non-bit data type will start at the next BYTE address
   e.g. object 0x10F3 SI4 = BOOL, SI5 = UINT16 →Bit-Offset of SI5 = 48 (not 41)
3. To define gaps in the byte stream of a complete access, it is allowed to define gap entries (Data_Type = 0, Bit_Length = gap size in bits)
   The value of gaps shall be 0.
4. SubIndixes that don't exist will not need space
5. The Complete Access can start with SubIndex 0 or SubIndex 1, other subindixes are not allowed
6. If the slave supports complete access all objects which fit in the mailbox should be accessible by SDO with complete access
7. If the slave supports Complete Access and Segmented SDO Transfer all objects should be accessible by SDO with Complete Access
8. Objects with dynamic entries (i.e. "OCTET_STRING", "UNICODE_STRING", "ARRAY_OF_*" or "VISIBLE_STRING") cannot be accessed by Complete access.
9. Objects that cannot be accessed shall return the error code 0x06010000 (Unsupported Access) or 0x6010004 (Complete access not supported).
10. For ENUM objects complete access may not be supported
11. The length of the responded Complete Access data shall either match to the current byte length of the object or to the Bit Size of the object description.
12. The current maximum subindex may exceed the number of entry descriptions of the offline object dictionary.
13. The SDO Download Complete Access data length shall always match the full current object size (defined by SubIndex0).

Entries of data type string must have a fixed length in the entry description to be used for complete access. The string itself can be shorter. Unused Bytes shall be filled with 0.

Example: EntryDescription Length = 240 Byte, actual String value = "Test" = 5 Byte with 0-Termination.

Even when the slave supports complete access there still may be objects that cannot be accessed via complete access. For descriptions about complete access in ESI, see ETG2000.

- If Object/Flags@SdoAccess=CompleteAccess in ESI, the device shall support complete access to the object.

An example of SDO Complete Access starting with SubIndex 1 is shown in Table 23.

Table 23: SDO Complete Access Example with SubIndex 1

SubIndex	Data Type	Bit Offset
1	BOOLEAN	0
2	BIT2	1
3	BIT3	3
4	0 = Gap, Length = 2	6
10	UNSIGNED8	8
11	UNSIGNED16	16
12	UNSIGNED32	32
13	UNSIGNED8	64
14	OCTET-STRING[4]	72
15	VISIBLE-STRING[7]	104
16	INTEGER32	160
17	UNSIGNED64	192

An Example of SDO Complete Access starting with SubIndex 0 is shown in Table 24.

Table 24: SDO Complete Access Example with SubIndex 0

SubIndex	Data Type	Bit Offset
0	UNSIGNED8	0
1	BOOLEAN	16
2	BIT2	17
3	BIT3	19
4	0 = Gap, Length = 2	22
10	UNSIGNED8	24
11	UNSIGNED16	32
12	UNSIGNED32	48
13	UNSIGNED8	80
14	OCTET-STRING[4]	88
15	VISIBLE-STRING[7]	120
16	INTEGER32	176
17	UNSIGNED64	208

For a download CompleteAccess to an object with RW and RO entries, the RO objects shall ignore the write access. Download data should be Zero for the RO entries.

11.3 SDO Information services

SDO Information services are defined in ETG.1000.6, [5].

Some objects support a higher maximum subindex than it is given in subindex 0, e.g. PDO assign object, Diagnosis object.

In SDO Information Service “Get Entry Description Response” the ValueInfo“, „Data Type“, „Bit Length“ and “Object Access” shall be 0 for those objects.

An SDO Upload/Download service shall be aborted with SDO Abort Code “Subindex not exist” (0x06090011) for those objects.

An SDO Information request to an Index <0x1000 is not allowed and shall be aborted with Abort code “Unsupported Access” (0x06010000).

12 ENUM Definition

For the textual description of the allowed values of an entry, the data type of the entry description shall be a device specific data type enumeration definition in the object area 0x0800-0x0FFF. The data type enumeration definition itself is defined as a record of the ENUM type (0x28) which is read-only in the object dictionary.

Values (Byte0-3) shall be sorted in ascending order.

Table 25: ENUM Definition - Textual description of allowed values

SubIndex	Description	Data type	Value
0	Number of entries	UNSIGNED8	n
1	Allowed Value 1	Octet String	Byte 0-3: Value 1 Byte 4-n1: textual description of Value 1
2	Allowed Value 3	Octet String	Byte 0-3: Value 2 Byte 4-n2: textual description of Value 2
…			
n	Allowed Value n	Octet String	Byte 0-3: Value n Byte 4-nn: textual description of Value n

NOTE: Byte 0-3 define the values in the range 2^{32}. It is not the subindex number. Up to 255 enumeration values can be defined per object

In the EtherCAT Slave Information (ESI) file the ENUM definition is included in the Data Type definition of the Dictionary.

The SDO Upload service for a Data Type Index used for ENUMs is conditional (mandatory if ENUM is used in an Entry-Description). The SDO-Info service is optional for this Data Type object.

An ENUM may also be used instead of a specified entry data type. In that case the bit size shall remain as specified.

Use case: if a data type is defined for an object/entry, a device manufacturer can use an ENUM instead of the defined data type, to provide a list of possible/allowed values.

13 Diagnosis Handling

13.1 General

This clause describes a unique handling for diagnosis data in an EtherCAT slave. The Diagnosis Handling supports:

- a diagnosis history with up to 250 diagnosis messages
- different types of diagnosis messages: Info, Warning, Error
- status bit “New message available” which can be mapped to the cyclic data
- a Text ID for each message as a reference to a detailed diagnosis description stored in the ESI file
- two operation modes: Overwrite Mode and Acknowledge Mode

The Diagnosis History Object is described by 0x10F3.

13.2 Diagnosis History Object

There should be a history of diagnosis messages in the EtherCAT slave that an EtherCAT master can always access the latest diagnosis messages. With the Diagnosis History object 0x10F3 up to 250 diagnosis messages can be accessed.

The object is defined in Table 26, the entry description is given in Table 27.

Table 26: 0x10F3 Diagnosis History Object

Attribute	Value
Index	0x10F3
Name	Diagnosis History
Object Code	RECORD
Max SubIndex	6-255

Table 27: Entries of 0x10F3 Diagnosis History Object

Sub-Index	Description	Data Type	Access	PDO Mapping	M/O/C	Description / Default value
1	Maximum Messages	UNSIGNED8	R or RW	no	M	Read: Number of diagnosis messages which can be stored in the diagnosis history (SubIndex 6 onwards) Write: For devices with dynamic memory the number of stored diagnosis messages may be adapted by the EtherCAT master
2	Newest Message	UNSIGNED8	R	no	M	SubIndex of the newest diagnosis message (6-255) Default value = 0
3	Newest Acknowledged Message	UNSIGNED8	RW	no	M	Default value = 0 Overwrite Mode (SI5, Bit4 = 0): Read = 0: When the message queue will be overwritten, the slave shall set SI3 to 0 Writing = 0: (support optional) the slave will clear all messages, i.e. resetting SI2, SI3, SI4 and Si5 Bit 5 Writing = 1…5: The slave shall return SDO-Abort with codes 0x06090030 (value range of parameter exceeded) or 0x06090032 (value of parameter written too low) Writing = 6…[SI0] [1]: SI3 = Written value without checking Writing > [SI0]…255: SDO-Abort with codes 0x06090030 or 0x06090031(value of parameter written too high) Acknowledge Mode (SI5, Bit4 =1): Read = 0: No messages have been acknowledge so far Read <> 0: SubIndex of latest acknowledged diagnosis message (6-255) Writing = 0: (support optional) All acknowledged messages will be deleted. Writing = 1…5: The slave shall return SDO-Abort with codes 0x06090030 (value range of parameter exceeded) or 0x06090032 (value of parameter written too low) Writing = 6…[SI0]: Messages are acknowledged Writing > [SI0]…255: SDO-Abort with codes 0x06090030 or 0x06090031(value of parameter written too high)
4	New Messages Available	BOOLEAN	R	TxPDO	M	Overwrite Mode: 0: newest message was read 1: newest message was not read Acknowledge Mode: 0: no unacknowledged message 1: diagnosis messages are available which can be acknowledged (SI2 <> SI3)

[1] [SI0] means content of subindex 0

Sub-Index	Description	Data Type	Access	PDO Mapping	M/O/C	Description / Default value	
5	Flags	UNSIGNED16	R(W)*	no	M	Flags to control the sending and storing of diagnosis messages Which functionality the slave supports is described by the ESI / SII. When writing SI5, the read-only bits should match the current values. Bit5 shall be "don't care". The slave should send an abort with 0x6090030 Value exceeded in case the readonly bits differ from current values	
						Bit 0: *Write support is optional	Enable Emergency sending (according to ETG.1000-6) 0: default if device does not support Emergency sending 1: new diagnosis messages shall be sent as Emergency message
						Bit 1: *Write support is mandatory	Disable info messages 0: Info messages are stored in the diagnosis message queue (default) 1: Info messages will not be stored in the diagnosis message queue
						Bit 2 *Write support is mandatory	disable warning messages 0: Warning messages are stored in the diagnosis message queue (default) 1: Warning messages will not be stored in the diagnosis message queue
						Bit 3: *Write support is optional	disable error messages 0: Error messages are stored in the message queue (default) 1: Error messages will not be stored in the diagnosis message queue
						Bit 4: *Write support is optional	Mode selection for diagnosis history handling 0: Overwrite Mode old messages are overwritten by new ones when buffer is full 1: Acknowledge Mode New messages do only overwrite messages which were acknowledged before

Sub-Index	Description	Data Type	Access	PDO Mapping	M/O/C	Description / Default value	
						Bit 5: *readonly	Overwrite/Discard Information In Overwrite Mode: 1: unacknowledged messages have been overwritten (=buffer overrun) (SI3 is set to 0, too) In Acknowledge Mode: 1: message buffer is full with unacknowledged messages and a new massage is discarded.
						Bit 6-15	reserved for future use
6-255	Diagnosis message	OCTET-STRING	R	no	SI6 M SI 7-255 O	Diagnosis message buffer Depending on SI1 the EtherCAT slave can store up to 250 diagnosis messages; the first message is stored in SubIndex 6, the second in SubIndex 7 and so on. When the queue is full, the EtherCAT slave shall overwrite SubIndex 6 and so on, that always the latest maximum messages (SI1) shall be accessible by the EtherCAT master	

13.3 Diagnosis Message

The Diagnosis Message shall contain the parameters shown in Table 28.

Table 28: Diagnosis Message

Parameter	Data type	USE	Description		
Diag Code	UNSIGNED32	M	Diagnosis code to identify the diagnosis message		
			Bit 0-15 = 0x0000-0xDFFF	not used	
			Bit 0-15 = 0xE000-0xE7FF	Bit 16-31: can be used manufacturer specific	
			Bit 0-15 = 0xE800	Bit 16-31: Emergency Error Code as defined in DS301 or DS4xxx	
			Bit 0-15 = 0xE801-0xEDFF	reserved for future standardization	
			Bit 0-15 = 0xEE00-0xEFFF	Bit 16-31: Profile specific	
			Bit 0-15 = 0xF000-0xFFFF	not used	
Flags	UNSIGNED16	M	Bit 0-3	Diag type:	
				0	Info message
				1	Warning message
				2	Error message
				3-15	reserved for future standardization
			Bit 4	Time Stamp is a local time stamp, the global time stamp can be calculated by reading the actual time stamp (object 0x10F8) and calculating the age of the message	
			Bit 5-7	reserved for future standardization	
			Bit 8-15	Number of parameters in this Diagnosis Message	

Parameter	Data type	USE	Description	
Text ID	UNSIGNED16	M	Text ID as reference to Diagnosis text as defined in the ESI file	
			0	no Text ID
			1-65535	Text ID reference to ESI file Text ID shall be unique in combination with VendorID, Product Code, Revision Number
Time Stamp	UNSIGNED64	M	Time Stamp in ns (from the DC unit) Or local time stamp (object 0x10F8) if DCs are not supported or DC are only supported in 32 bit mode	
			0	no time stamp
			<> 0	time stamp
Flags Parameter 1	UNSIGNED16	O	Describes the type of Parameter 1	
			Bit 12-15 = 0	Bit 0-11 = Data type Index of the data type of parameter 1 0x0001: BOOLEAN 0x0002: INTEGER8 0x0003: INTEGER16 0x0004: INTEGER32 0x0005: UNSIGNED8 0x0006: UNSIGNED16 0x0007: UNSIGNED32 0x0008 : REAL32 0x0011 : REAL64 0x0015 : INTEGER64 0x001B : UNSIGNED64 The corresponding text parameters and formatting are specified in the ETG.2000
			Bit 12-15 = 1	Bit 0-11 = size of BYTE-Array in bytes, parameter 1 is a BYTE-Array shall be used for following Data Type: OCTET_STRING (index: 0x00A), specifier %s
			Bit 12-15 = 2	Bit 0-11 = size of ASCII-String (without ending 0) in bytes, parameter 1 is an ASCII-string shall be used for following Data Type: VISIBLE_STRING (index: 0x009), specifier %s
			Bit 12-15 = 3	Bit 0-11 = size of Uni-Code-String in bytes, parameter 1 is an Uni-Code-string shall be used for following Data Type: UNICODE_STRING (index: 0x00B), specifier %s
			Bit 12-15 = 4	Bit 0-11 = 2 (size of parameter 1 in bytes), parameter 1 number of TextID as referenced in ESI specifier %s
			Bit 12-15 = 5-15	reserved for future standardization
Parameter 1	depend on Flags Parameter 1	C	value of parameter 1 (Mandatory if Flags Parameter 1 exist)	
Flags Parameter 2	UNSIGNED16	O	see flags parameter 1	
...				

13.4 Diagnosis History Operation Modes

The Diagnosis Message buffer (0x10F3:6-255) works like a ring buffer. This ring buffer can be handled in Overwrite Mode or Acknowledge Mode.

The mode can be read/ written via 0x10F3:SI5:Bit4.

13.4.1 Overwrite Mode

In Overwrite Mode new messages overwrite older messages once the buffer is full even when they were not acknowledged before.

13.4.1.1 Acknowledging Messages in Overwrite Mode

When writing the Newest Acknowledged Message (0x10F3:SI3) all messages up to the age of the message which is in the written SubIndex are acknowledged. The slave does not check if those messages had been read before.

Indication:
0x10F3:SI5:BIT5 = 0

The slave shall return SDO-Abort with codes 0x06090030 (value range of parameter exceeded) or 0x06090031 (value of parameter written too high) in the following case:

- If SI3 is written with a value of a Subindex which does not hold a message

13.4.1.2 Clearing Messages in Overwrite Mode

Write 0 to Newest Acknowledged Message (0x10F3:SI3) will delete all messages even if the messages were not acknowledged or read before.

Indication:
0x10F3:SI2 = 0; 0x10F3:SI3 = 0; 0x10F3:SI4 = 0 and 0x10F3:SI5:BIT5 = 0

13.4.2 Acknowledge Mode

Messages which have not been acknowledged are not overwritten within the ring buffer. New messages are not stored and go lost.

13.4.2.1 Acknowledging Messages in Acknowledge Mode

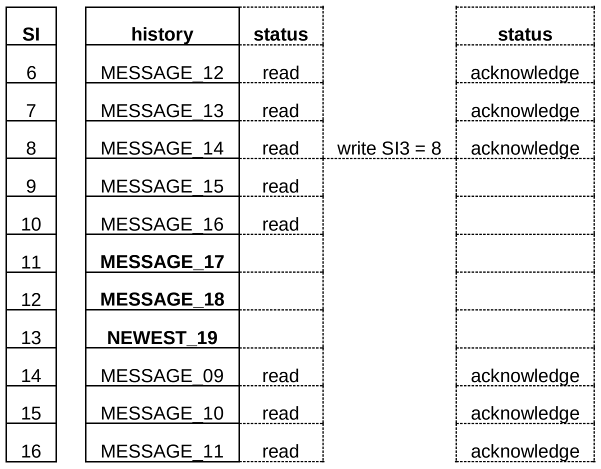
When writing the Newest Acknowledged Message (0x10F3:SI3) all messages up to the age of the message which is in the written SubIndex are acknowledged. The slave does not check if those messages had been read before.

Indication:
0x10F3:SI5:BIT5 = 0

The slave shall return SDO-Abort with codes 0x06090030 (value range of parameter exceeded) or 0x06090031 (value of parameter written too high) in the following case:

- If SI3 is written with a value of a Subindex which does not hold a message

The following examples with writing different values to the Acknowledge (SI3) show the behaviour.



SI	history	status		status
6	MESSAGE_12	read		acknowledge
7	MESSAGE_13	read		acknowledge
8	MESSAGE_14	read	write SI3 = 8	acknowledge
9	MESSAGE_15	read		
10	MESSAGE_16	read		
11	**MESSAGE_17**			
12	**MESSAGE_18**			
13	**NEWEST_19**			
14	MESSAGE_09	read		acknowledge
15	MESSAGE_10	read		acknowledge
16	MESSAGE_11	read		acknowledge

Figure 1: Acknowledge Mode: Example 1

SI	history	status		status
6	MESSAGE_12	read		acknowledge
7	MESSAGE_13	read		acknowledge
8	MESSAGE_14	read		acknowledge
9	MESSAGE_15	read		acknowledge
10	MESSAGE_16	read		acknowledge
11	**MESSAGE_17**			acknowledge
12	**MESSAGE_18**		write SI3 = 12	acknowledge
13	**NEWEST_19**			
14	MESSAGE_09	read		acknowledge
15	MESSAGE_10	read		acknowledge
16	MESSAGE_11	read		acknowledge

Figure 2: Acknowledge Mode: Example 2

13.4.2.2 Clearing Message History in Acknowledge Mode

To gain a better overview within the diagnosis message history all messages which have been acknowledged before can be deleted from the message history by writing “0” to the Acknowledge (SI3). Messages which had not been acknowledged before are shifted to the beginning of the message history queue.

Figure 3 shows an example.

SI	history	status		history
6	MESSAGE_12	acknowledged		**MESSAGE_17**
7	MESSAGE_13	acknowledged		**MESSAGE_18**
8	MESSAGE_14	acknowledged		**NEWEST_19**
9	MESSAGE_15	acknowledged		
10	MESSAGE_16	acknowledged	write SI3 = 0 →	
11	**MESSAGE_17**	Read		
12	**MESSAGE_18**	Read		
13	**NEWEST_19**			
14	MESSAGE_09	acknowledged		
15	MESSAGE_10	acknowledged		
16	MESSAGE_11	acknowledged		

All acknowledged messages are deleted

Figure 3: Clearing Message History

13.5 Timestamp Object

The Timestamp object holds the current local Time of the slave

The Timestamp object is defined in Table 29.

Table 29: 0x10F8 Timestamp Object

Attribute	Value
Index	0x10F8
Name	Timestamp Object
Object Code	VAR
DataType	UNSIGNED64
Category	optional
Access	ro
PDO Mapping	Yes
Value	Local Timestamp of the device in ns

14 PDO Parameter Description

14.1 General

The PDO Parameter object contains information about the PDOs. Each PDO Mapping object (0x1600-0x17FF, 0x1A00-0x1BFF) is related to a PDO Parameter object (0x1400-0x15FF, 0x1800-0x19FF). The subindexes 1-5 of the PDO Parameter object are reserved to keep compatibility to CANopen.

14.2 RxPDO Parameter

Table 30 defines the RxPDO Parameter objects. Table 31 describes the object entries.

Table 30: RxPDO Parameter Objects

Attribute	Value
Index	0x1400…0x15FF
Name	RxPDO Parameter
Max SubIndex	9
Category	Optional
Access	Read Write
PDO Mapping	Yes

Table 31: RxPDO Parameter

SubIndex	Description	Data Type	Description
6	RxPDO Exclude RxPDOs	OCTET_STRING	Includes the Indexes of the RxPDO Mapping object Indexes which could not be assigned together with this RxPDO
7	RxPDO State*	BOOLEAN	Shall be set (TRUE) from the slave if the output data of this RxPDO could not be put to the hardware, mappable in a TxPDO
8	RxPDO Control	BOOLEAN	Shall be set (TRUE) from the master if the output data of this RxPDO do not have valid values, mappable in a RxPDO
9	RxPDO Toggle* Optional: OutputUpdateCounter	BOOLEAN Optional: BIT2	Shall toggle/increment with every update of the corresponding RxPDO (written by the master and mappable in a RxPDO)

* RxPDO State and RxPDO Toggle/OutputUpdateCounter might be defined module specific in a corresponding device profile, such as ETG.5001 [7] . If such a profile is used, the profile specific entries should be mapped to the process data.

14.3 TxPDO Parameter

Table 32 defines the TxPDO Parameter objects. Table 33 describes the object entries.

Table 32: TxPDO Parameter Objects

Attribute	Value
Index	0x1800…0x19FF
Name	TxPDO Parameter
Max SubIndex	9
Category	Optional
Access	Read Write
PDO Mapping	Yes

Table 33: TxPDO Parameter

SubIndex	Description	Data Type	Description
6	TxPDO Exclude TxPDOs	OCTET_STRING	Includes the Indexes of the TxPDO Mapping object Indexes which could not be assigned together with this TxPDO
7	TxPDO State*	BOOLEAN	Shall be set (TRUE) from the slave if the input data of this TxPDO is not valid, mappable in a TxPDO Should be used, if the input data from the application are not valid, e.g. because a measurement hardware is not connected properly or it can break. The slave stays in OP.
9	TxPDO Toggle* Optional: InputUpdateCounter	BOOLEAN Optional: Bit2	Shall toggle/increment with every update of the corresponding TxPDO (written by the slave and mappable in a TxPDO)

* TxPDO State and TxPDO Toggle/InputUpdateCounter might be defined module specific in a corresponding device profile, such as ETG.5001 [7] . If such a profile is used, the profile specific entries should be mapped to the process data.

14.4 PDO Mapping entries larger than 31 Byte

Since the bit length of a mapping entry is restricted to 255 bit (UINT8), PDO Mapping entries >31 Byte shall be separated into several parts with maximum 30 Byte (240 Bit) length. The first PDO Mapping entry shall contain the entries' index and subindex information with a bit size of 240. Following parts shall be used as padding entries with index = 0x0000 and subindex = 0x00 and with bit size of 240, expect the last block with a bit size <240.

To distinguish between padding entries used for such kind of large mapping entries and alignment information (also using index = 0x0000 and subindex = 0x00) the configuration tool shall evaluate the data type of the PDO.

Following rules shall be fulfilled:

1. Bitsize=240 and DataType:Bitlength <= 240
   - all following 0000:00:yy are alignment information
2. Bitsize=248 and DataType:Bitlength = 248
   - all following 0000:00:yy are alignment information
3. Bitsize=240 and DataType:Bitlength > 240
   - Pdo entry needs padding information
     All padding Pdo mapping entries must be "0000:00:240",
     except the last Pdo entry may be "0000:00:nn" where nn should be <=240
4. The PDO entry bit size and data type may be smaller than the definition in the OD.

Examples:

32 byte entry (0x7000:01) + no alignment + 2 byte entry (0x7000:02):

0x1600.1	0x7000: 01: 240	Data type ARRAY[0..**31**] OF BYTE
0x1600.2	0x0000: 00: 16	**last two Bytes**
0x1600.3	0x7000: 02: 16	Data type e.g. UINT

30 byte entry (0x7000:01) + 2 byte alignment + 2 byte-entry (0x7000:02):

0x1600.1	0x7000: 01: 240	Data type ARRAY[0..**29**] OF BYTE
0x1600.2	0x0000: 00: 16	**2 Byte alignment**
0x1600.3	0x7000: 02: 16	Data type e.g. UINT

32 byte entry (0x7000:01) + 2 byte alignment + 2 byte entry (0x7000:02):

0x1600.1	0x7000: 01: 240	Data type ARRAY[0..**31**] OF BYTE
0x1600.2	0x0000: 00: 16	**last two bytes**
0x1600.3	0x0000: 00: 16	**2 Byte alignment**
0x1600.4	0x7000: 02: 16	Data type e.g. UINT

Map only 30 byte of a 40 byte entry (ARRAY[0..**39**] OF BYTE) + 2 byte alignment + 2 byte entry (0x7000:02):

0x1600.1	0x7000: 01: 240	Data type ARRAY[0..**29**] OF BYTE
0x1600.2	0x0000: 00: 16	2 Byte alignment
0x1600.3	0x7000: 02: 16	Data type e.g. UINT

60 Byte-entry (fits exactly in two 240 bit entry) (0x7000:01) + no alignment + 2 Byte entry (0x7000:02)

0x1600.1	0x7000:01:240	Data type ARRAY[0..59] OF BYTE
0x1600.2	0x0000:00:240	last 30 Byte
0x1600.3	0x7000:02:16	Data type e.g. UINT

60 Byte-entry (fits exactly in two 240 bit entry) (0x7000:01) + alignment + 2 Byte entry (0x7000:02)

0x1600.1	0x7000:01:240	Data type ARRAY[0..59] OF BYTE
0x1600.2	0x0000:00:240	last 30 Byte
0x1600.3	0x0000:00:16	**2 Byte alignment**
0x1600.4	0x7000:02:16	Data type e.g. UINT

The configuration tool always has to take care of the corresponding data type.

15 FoE Extension

15.1 FoE read sequence with busy

FoE services are defined in ETG.1000.5, corresponding protocols in ETG.1000.6.

For an FoE read request with Busy the sequence shown in Figure 4 shall be used.

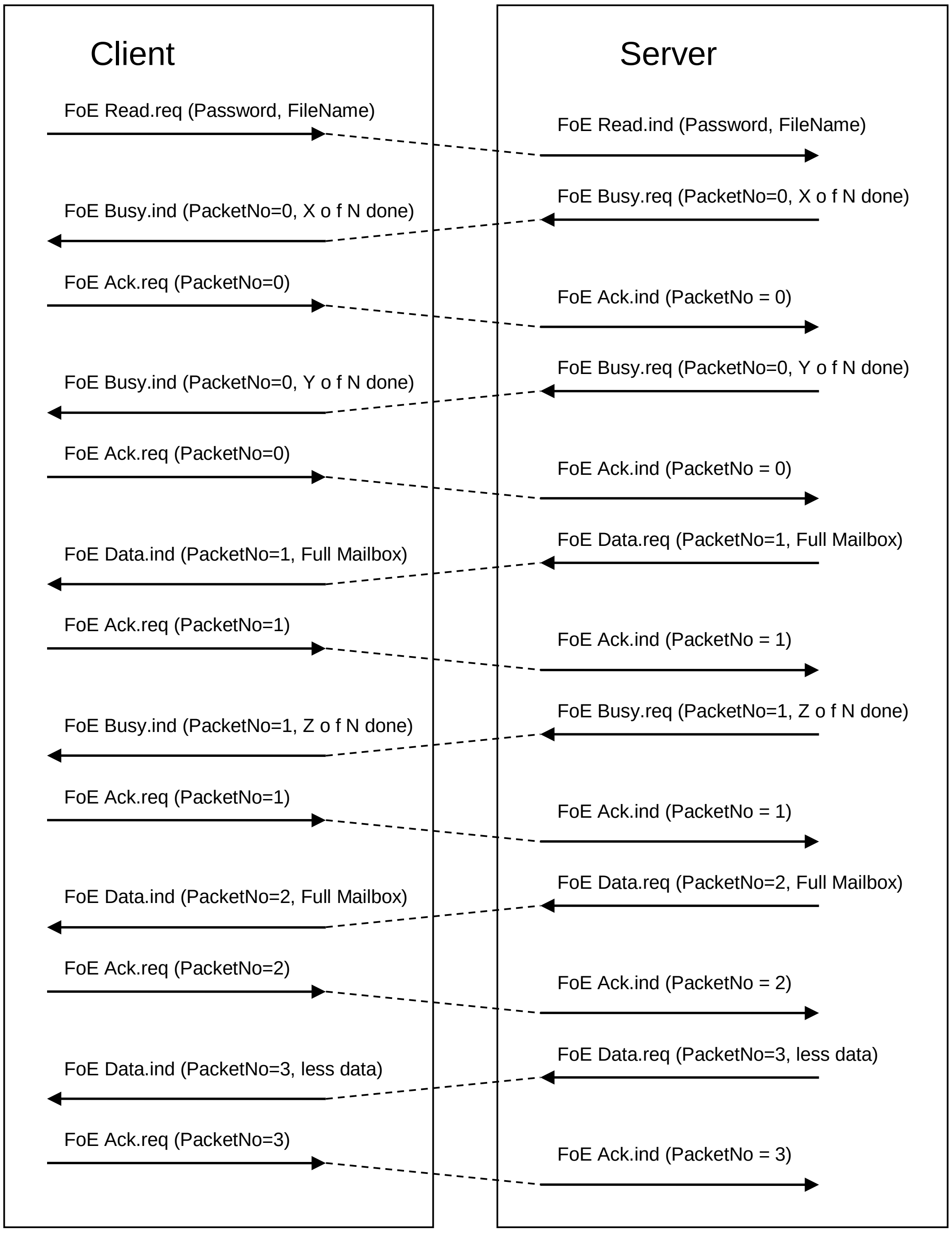

Client
Server
FoE Read.req (Password, FileName)
FoE Read.ind (Password, FileName)
FoE Busy.ind (PacketNo=0, X o f N done)
FoE Busy.req (PacketNo=0, X o f N done)
FoE Ack.req (PacketNo=0)
FoE Ack.ind (PacketNo = 0)
FoE Busy.ind (PacketNo=0, Y o f N done)
FoE Busy.req (PacketNo=0, Y o f N done)
FoE Ack.req (PacketNo=0)
FoE Ack.ind (PacketNo = 0)
FoE Data.ind (PacketNo=1, Full Mailbox)
FoE Data.req (PacketNo=1, Full Mailbox)
FoE Ack.req (PacketNo=1)
FoE Ack.ind (PacketNo = 1)
FoE Busy.ind (PacketNo=1, Z o f N done)
FoE Busy.req (PacketNo=1, Z o f N done)
FoE Ack.req (PacketNo=1)
FoE Ack.ind (PacketNo = 1)
FoE Data.ind (PacketNo=2, Full Mailbox)
FoE Data.req (PacketNo=2, Full Mailbox)
FoE Ack.req (PacketNo=2)
FoE Ack.ind (PacketNo = 2)
FoE Data.ind (PacketNo=3, less data)
FoE Data.req (PacketNo=3, less data)
FoE Ack.req (PacketNo=3)
FoE Ack.ind (PacketNo = 3)

Figure 4: FoE read sequence with busy

If the last data segment of the file exactly fits into the mailbox an additional FoE Data.req (PacketNo=4, less data=0) without data has to be sent.

A FoE_Write request with Busy shall always be handled as defined in ETG.1000.5. That means that the slave shall always send the FoE_ACK.req as response to the FoE_Write.ind. Even if the internal procedure needs additional time to check the Write request from the master. After FoE_Data.ind the slave can send FoE_Busy.req or FoE_Error.req in case of an error (e.g. if the file is not known).

15.2 FoE error Codes

FoE Error Codes are defined in ETG.1000.6, [5].

The FoE error codes are derived from TFTP error codes. Several implementations use the TFTP error codes without the Offset of 0x8000. Both codes can be used arbitrarily.

Table 34 shows already existing error codes and additional error codes that should be used for FoE.

Table 34: Error codes for FoE

Error code	Meaning	Reference
0x8000 or 0x0000	Not defined	ETG.1000.6
0x8001 or 0x0001	Not found	ETG.1000.6
0x8002 or 0x0002	Access denied	ETG.1000.6
0x8003 or 0x0003	Disk full	ETG.1000.6
0x8004 or 0x0004	Illegal	ETG.1000.6
0x8005 or 0x0005	Packet number wrong	ETG.1000.6
0x8006 or 0x0006	Already exists	ETG.1000.6
0x8007 or 0x0007	No user	ETG.1000.6
0x8008 or 0x0008	Bootstrap only	ETG.1000.6
0x8009 or 0x0009	Not Bootstrap	ETG.1000.6
0x800A or 0x000A	No rights	ETG.1000.6
0x800B or 0x000B	Program Error	ETG.1000.6
0x800C or 0x000C	Checksum wrong	Additional code
0x800D or 0x000D	Firmware does not fit for Hardware	Additional code
0x800E or 0x000E	reserved	Additional code
0x800F or 0x000F	No file to read	Additional code
0x8010 or 0x0010	File header does not exist	Additional code
0x8011 or 0x0011	Flash problem	Additional code
0x8012 or 0x0012	File incompatible	Additional code

16 EoE Extension

16.1 General

EoE Timestamp information can be used for synchronization protocols like IEEE 1588.

Slave → Master: Time Stamp contains exact receive time

Master → Slave: Time Stamp contains desired send time

16.2 Get IP Parameter

With this service the IP parameter could be read, the service response is similar to the Set IP parameter request as shown in Table 35.

Table 35: EoE Get IP Parameter service request

Frame part	Data Field	Data Type	Description
Mailbox Header	Length	WORD	Length of Mailbox Service Data = 4
	Address	WORD	Station Address of the source, if a master is client, Station Address of the destination, if a slave is client
	Channel	Unsigned6	Reserved for future use
	Priority	Unsigned2	Reserved for future use
	Type	Unsigned4	EoE = 2
	Counter	Unsigned3	Counter for Mailbox Data Link layer
	Reserved	Unsigned1	Reserved for future use
EoE Header	Frame Type	Unsigned4	0x06
	Port	Unsigned4	0x00: send to no specific port 0x01-0x0F: specific ports selected
	Last Fragment	Unsigned1	0x01: last Data part
	Time Appended	Unsigned1	0x00: no time stamp value will be appended
	Time Request	Unsigned1	0x00: no time stamp value of the send time requested
	Reserved	Unsigned5	reseerved for future use
	Fragment number	Unsigned6	0x00
	Offset	Unsigned6	0x00
	Frame number number	Unsigned4	0x00

The Set IP parameter response service is shown in Table 36.

Table 36: EoE Get IP Parameter service response

Frame part	Data Field	Data Type	Description
Mailbox Header	Length	WORD	Length of Mailbox Service Data >= 10
	Address	WORD	Station Address of the source, if a master is client, Station Address of the destination, if a slave is client
	Channel	Unsigned6	Reserved for future use
	Priority	Unsigned2	Reserved for future use
	Type	Unsigned4	EoE = 2
	Counter	Unsigned3	Counter for Mailbox Data Link layer
	Reserved	Unsigned1	Reserved for future use
EoE Header	Frame Type	Unsigned4	0x07
	Port	Unsigned4	0x00: send to no specific port 0x01-0x0F: specific ports selected
	Last Fragment	Unsigned1	0x01: last Data part
	Time Appended	Unsigned1	0x00: no time stamp value will be appended
	Time Request	Unsigned1	0x00: no time stamp value of the send time requested
	Reserved	Unsigned5	reseerved for future use
	Fragment number	Unsigned6	0x00
	Offset	Unsigned6	0x00
	Frame number number	Unsigned4	0x00
Get IP parameter Header	MAC included	Unsigned1	MAC address according to ISO/IEC 8802-3
	IP address included	Unsigned1	IP address according to IETF RFC 791
	Subnet Mask included	Unsigned1	Subnet mask according to IETF RFC 791
	Default Gateway included	Unsigned1	Default Gateway address according to IETF RFC 791
	DNS Server IP Address included	Unsigned1	IP address of DNS server according to IETF RFC 791
	DNS Name included	Unsigned1	DNS name according to IETF RFC 791
	reserved	Unsigned26	reserved for future use
(conditional)	MAC	BYTE[6]	MAC address according to ISO/IEC 8802-3
(conditional)	IP address	BYTE[4]	IP address according to IETF RFC 791
(conditional)	Subnet Mask	BYTE[4]	Subnet mask according to IETF RFC 791
(conditional)	Default Gateway	BYTE[4]	Default Gateway address according to IETF RFC 791
(conditional)	DNS Server IP Address	BYTE[4]	IP address of DNS server according to IETF RFC 791
(conditional)	DNS Name	char[32]	DNS name according to IETF RFC 791

16.3 Get Address Filter

With this service the Address Filter can be read, the service response is similar to the Set Address Filter request. Table 37 defines the service request and Table 38 the service response.

Table 37: EoE Get Address Filter service request

Frame part	Data Field	Data Type	Description
Mailbox Header	Length	WORD	Length of Mailbox Service Data = 4
	Address	WORD	Station Address of the source, if a master is client, Station Address of the destination, if a slave is client
	Channel	Unsigned6	Reserved for future use
	Priority	Unsigned2	Reserved for future use
	Type	Unsigned4	EoE = 2
	Counter	Unsigned3	Counter for Mailbox Data Link layer
	Reserved	Unsigned1	Reserved for future use
EoE Header	Frame Type	Unsigned4	0x08
	Port	Unsigned4	0x00: send to no specific port 0x01-0x0F: specific ports selected
	Last Fragment	Unsigned1	0x01: last Data part
	Time Appended	Unsigned1	0x00: no time stamp value will be appended
	Time Request	Unsigned1	0x00: no time stamp value of the send time requested
	Reserved	Unsigned5	reserved for future use
	Fragment number	Unsigned6	0x00
	Offset	Unsigned6	0x00
	Frame number	Unsigned4	0x00

Table 38: EoE Get Address Filter service response

Frame part	Data Field	Data Type	Description
Mailbox Header	Length	WORD	Length of Mailbox Service Data >= 6
	Address	WORD	Station Address of the source, if a master is client, Station Address of the destination, if a slave is client
	Channel	Unsigned6	Reserved for future use
	Priority	Unsigned2	Reserved for future use
	Type	Unsigned4	EoE = 2
	Counter	Unsigned3	Counter for Mailbox Data Link layer
	Reserved	Unsigned1	Reserved for future use
EoE Header	Frame Type	Unsigned4	0x09
	Port	Unsigned4	0x00: send to no specific port 0x01-0x0F: specific ports selected
	Last Fragment	Unsigned1	0x01: last Data part
	Time Appended	Unsigned1	0x00: no time stamp value will be appended
	Time Request	Unsigned1	0x00: no time stamp value of the send time requested
	Reserved	Unsigned5	reserved for future use
	Fragment number	Unsigned6	0x00
	Offset	Unsigned6	0x00
	Frame number	Unsigned4	0x00
Get Address Filter Header	MAC filter count	Unsigned4	Count of MAC address according to ISO/IEC 8802-3 which are accepted by Ethernet Ports on this slave
	MAC filter mask	Unsigned2	Count of MAC address masks according to ISO/IEC 8802-3 which are combined with filter by Ethernet Ports on this slave
	Reserved	Unsigned1	reserved for future use
	Inhibit Broadcast	Unsigned1	Filter Broadcast messages
	Reserved	Unsigned8	reserved for future use
(conditional, used if MAC filter count <> 0)	List of MAC Address	List of BYTE [6]	MAC address according to ISO/IEC 8802-3
(conditional, used if MAC filter mask <> 0)	List of MAC Address Filter	List of BYTE [6]	MAC address according to ISO/IEC 8802-3 A set bit means that this address bit of Destination MAC Address is compared with the corresponding entry in the List of MAC Address.

17 VoE Coding

17.1 PDU structure

The general attribute types of VoE are described in Figure 5.

```
typedef struct
{
  unsigned32        VendorID;
  unsigned16        VendorType;
} TVOEHEADER;

typedef struct
{
  TMBXHEADER        MbxHeader;
  TVOEHEADER        VoeHeader;
  BYTE              Data[MBX_DATA_SIZE-2];
} TVOEMBX;
```

Figure 5: VoE general structure

The VoE coding is specified in Table 39.

Table 39: VoE elements

Frame part	Data Field	Data Type	Value/Description
Mailbox Header	Length	WORD	Length of the Mailbox Service Data
	Address	WORD	Station Address of the source, if a master is client, Station Address of the destination, if a slave is client
	Channel	Unsigned6	0x00 (Reserved for future)
	Priority	Unsigned2	0x00: lowest priority … 0x03: highest priority
	Type	Unsigned4	0x0F: VoE
	Cnt	Unsigned3	Counter of the mailbox services (0 reserved, 1 is start value, next value after 7 is 1)
	Reserved	Unsigned1	0x00
VoE Header	VendorID	Unsigned32	Vendor ID
	Vendor Type	Unsigned16	Vendor specific Protocol Type
	Data		Protocol specific data

18 Explicit Device Identification

18.1 Intention

The use of EtherCAT Device identification is to identify an EtherCAT slave explicitly. This is necessary for the following use cases:

- Hot Connect applications
  Within some applications it might be useful to connect or disconnect parts of the network. In this case the master must have the possibility to identify which part of the network is available.
- Prevention against cable swapping
  If at least two identical devices are used in one application it might be necessary to prevent the mix-up of these devices by cable swapping.
  Example Scenario: Within a machining center there might be two identical drives to work in X and Y direction. To avoid that the drives receive wrong process data, for example after a device replacement, an explicit identification of the devices can be used.

The Device Identification value can be used optionally for unique addressing.

18.2 Terms and Definitions

Device Identification Value
Explicit Device Identification value that is set locally on the device

ID-Selector
Local non-volatile selector to set the Device Identification Value

SII
Slave Information Interface – content of EEPROM

SII Configured Station Alias
Parameter Configured Station Alias stored in the SII and loaded during boot-up to register 0x0012 of the ESC

Configured Station Alias Register – referred to as Register 0x0012
Register 0x0012 in the ESC that is loaded from EEPROM during boot-up of the ESC

18.3 Device Identification Value

The Device Identification value should be set uniquely within a network configuration.

The following properties shall be fulfilled by the Device Identification value:

- Device Identification Value = 0 is invalid
  This shall be the default value of the SII Configured Station Alias
- The Device Identification Value shall have a length of 2 Byte

The Device Identification value can be set

- by an **ID-Selector**, locally on the slave
  This can be a Switch (e.g. Rotary- or DIP-switch) or set via a display control stored non-volatile. The ID-Selector should have a length of 16 Bit.
- by a configuration tool in the SII Configured Station Alias
  The value of the SII is automatically loaded during boot-up of the ESC to the Configured Station Alias Register (0x0012).

18.4 Slave implementation

Three different device types for setting the Device ID are defined in this specification:

- Complex Slave with application microcontroller and ID-Selector
  → Requesting ID mechanism
- Simple Slave w/o microcontroller and with ID-Selector
  → Direct ID mechanism
- Set Device Identification by Configuration Tool
    - Slave w/o ID-Selector
    - Remote settable Device ID

ETG.2000 defines the elements “IdentificationAdo” and “IdentifiationReg134” to distinguish between Requesting ID and Direct ID mechanism.

18.4.1 Requesting ID – Complex Slave with application microcontroller and ID-Selector

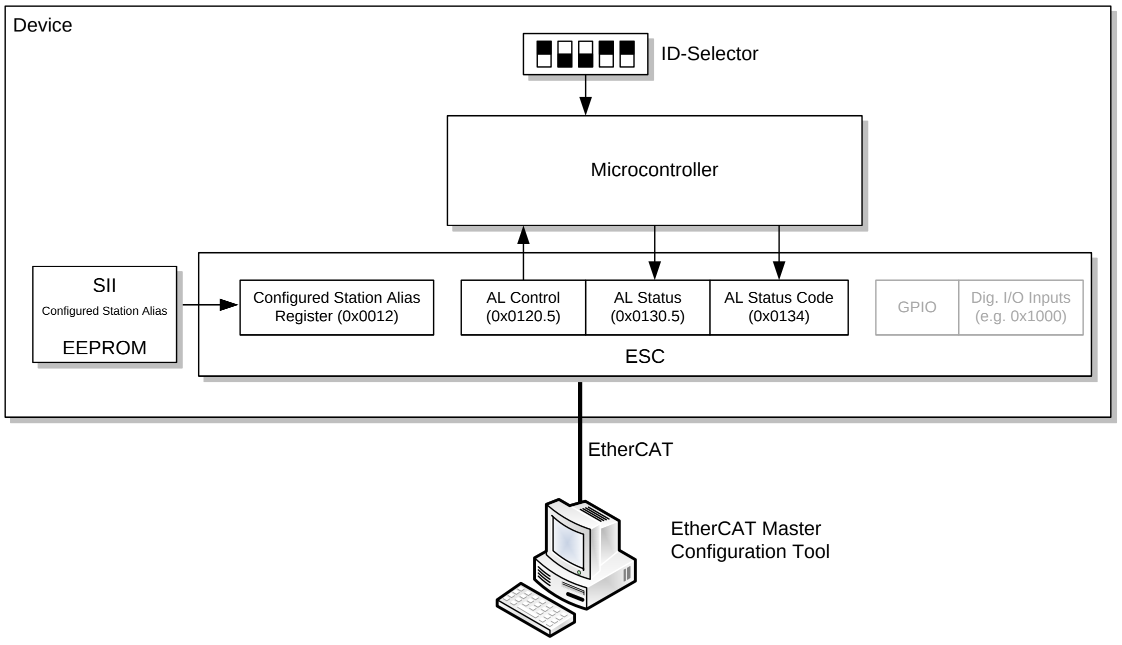


Figure 6: Complex slave with application microcontroller

Handling of Requesting ID mechanism within EtherCAT state machine (ESM) is specified in ETG.1000.5 and ETG.1000.6. Following clauses are for description only.

A complex Slave with application Microcontroller and ID-Selector should use AL Status Code register 0x0134 for Device Identification value.

The EtherCAT Master can request the Device Identification value by setting Bit 5 of AL Control register (0x0120.5 = *ID Request*). The slave shall load the current Device Identification Value to the AL Status Code register and set Bit 5 of AL Status register (0x0130.5 = *ID loaded*) to indicate that its Device Identification Value is loaded. A changed Device Identification value, e.g. because the ID-Selector is set to a different value, shall be loaded only with a new ID Request by the EtherCAT Master.

Figure 7 describes a timing example of an ID request by the master.

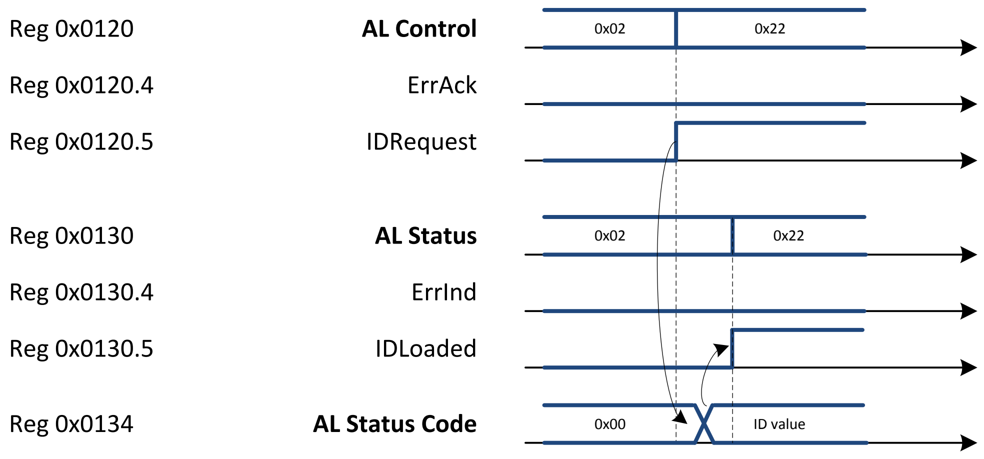


Figure 7: Timing Example: Request Device ID

If the slave has to indicate an Error at the same time at which the ID Value is requested the slave shall reset *ID Loaded* Flag (0x0130.5), load the *Error code* to the AL Status Code register and set the *Error Indication* Flag (0x0130.4), see Figure 8.

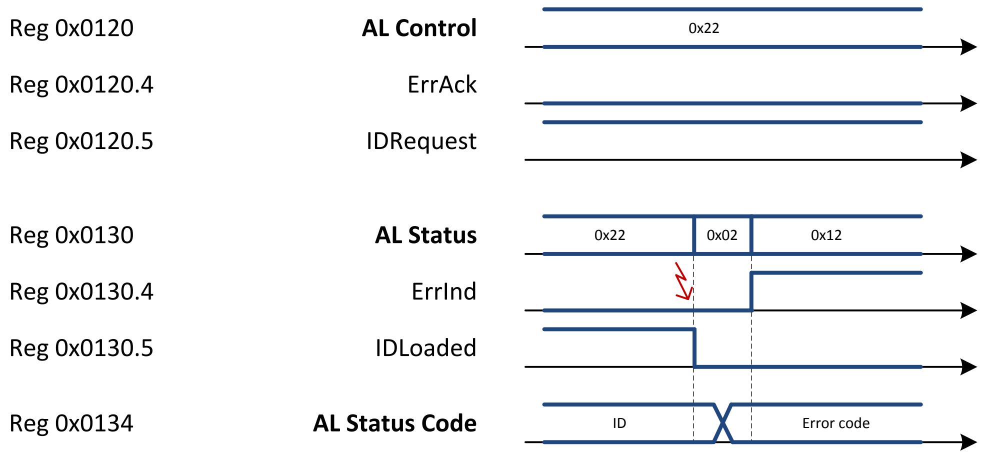


Figure 8: Timing Example: Error Indication while Device ID loaded

If the slave has an active *Error Indication* but the Error is not pending and the master sends an *Err Acknowledge* and *ID Request* at the same time, the slave shall reset the *Error Indication* Flag and confirm noError, than load the Device Identification Value in the AL Status Code register and set *ID Loaded* again, see Figure 9.

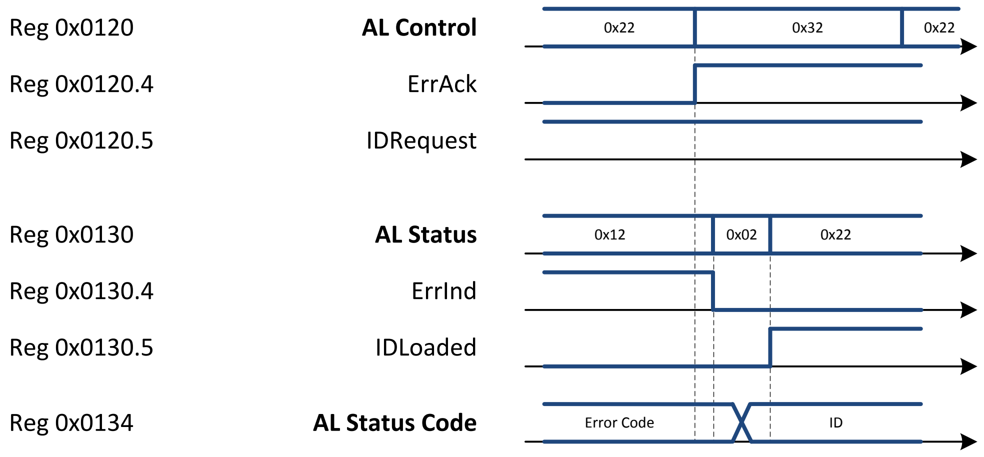


Figure 9: Timing Example: Error Acknowledge with Device ID Request

If the device supports the Requesting ID mechanism it shall support this mechanism in all states except Boot. A request of the Device ID in Boot state shall be ignored.

Timeouts for the Requesting ID mechanism are the same as used for ESM transitions.

For simple identification process The master can use following command sequence (for example as part of the ENI file) to identify the slave:

- Write register 0x0120 = 0x0021 (Bit 0 → Init; Bit 5 →Load Device ID)
- Poll registers 0x0130…0x0135 and mask to compare with expected Device ID.
  Master shall use one datagram for data consistency to read register 0x0130 and 0x0134.
- Write register 0x0120 = 0x0001 (Bit 0 → Init)

NOTE: The Requesting ID mechanism might be ignored by the Master / Configuration Tool and Configured Station Alias may be used alternatively. The ID-Selector should be zero in that case.

NOTE: Configured Station Alias Register 0x0012 should be used for a complex slave with microcontroller only for compatibility reasons, since this register is also loaded during power-on from SII Configured Station Alias. This might lead to a race condition.
If a slave uses Register 0x0012 for ID-Selector values, following functionality shall be used:

- If ID-Selector value == 0 nothing shall be done (value in register 0x0012 is loaded from SII, independent if "0" or different value)
- If ID-Selector value != 0 AND SII value == 0: value from ID-Selector shall be copied to register 0x0012
- If ID-Selector value != 0 and SII value != 0: a race condition failure procedure shall be started:
  - copy ID-Selector value to register 0x0012
  - set internal error flag (ERRint = 1)
  - overwrite SII value with "0" during state change from Init to PreOp (SII access to PDI)
  - depending on further master action:
    a) if state change to SafeOp is requested: reject state change (Al Status = 0x12) and set error "Device Identification value updated" (Al Status Code = 0x0061)
    b) if state Init is requested OR power cycle OR successful SDO write to Reload object: delete internal error, proceed normal

18.4.2 Direct ID – Simple Slave w/o microcontroller and with ID-Selector

The ID-Selector can be connected directly to I/O Inputs of the ESC. There is no need for a microcontroller.

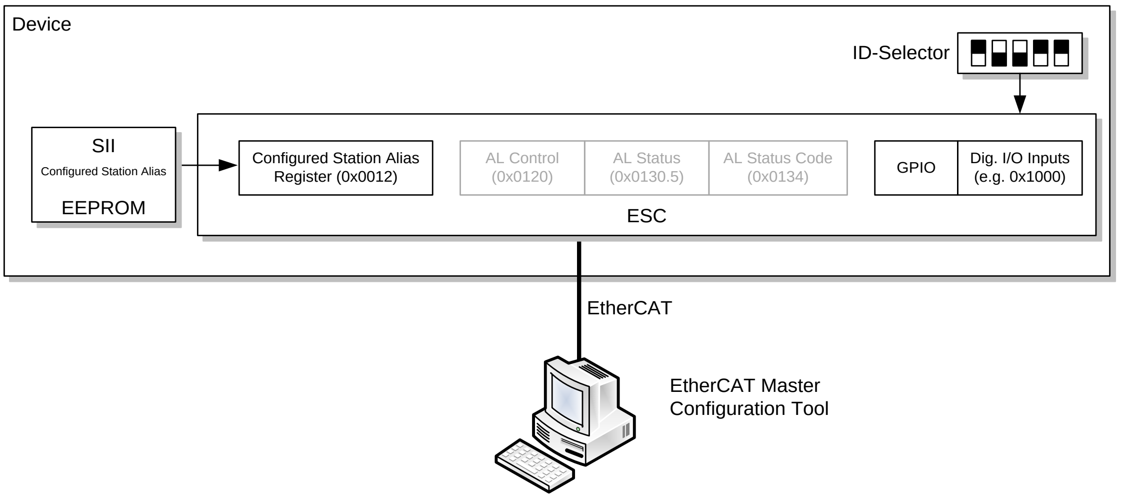


Figure 10: Simple slave w/o microcontroller

The ID-Selector value is copied to the address given in the IdentificationAdo ESI /SII element and can directly be read by the Master (Direct ID mechanism). A change of the ID-Selector is directly copied to the Device Identification value.

The local address of the Device Identification Value shall be within one of the following memory ranges in the ESC:

- General Purpose Inputs (e.g. 0x0F18 … 0x0F1F)
- Digital I/O Input Data (PDI Configuration: Digital I/O Interface, 0x1000…0x1003)

NOTE: availability of memory ranges depends on ESC type

NOTE: Do not use the User RAM area (0x0F80…0x0FFF) or Process Data RAM (0x1000…0xFFFF, if not initialized by hardware on power-up) since RAM comes up with undefined values after boot-up.

NOTE: The Direct ID mechanism might be ignored by the Master / Configuration Tool and Configured Station Alias may be used alternatively. The ID-Selector should be zero in that case.

18.4.3 Set Device Identification by Configuration Tool

If no local ID-Selector is available or the ID-Selector value is Zero the configuration tool can use Register 0x0012 for explicit device identification.

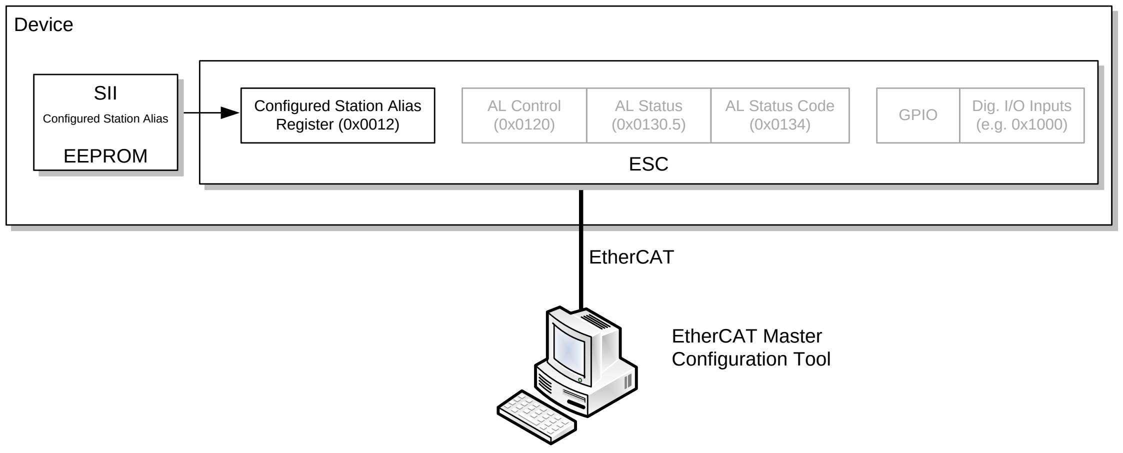


Figure 11: Set Device Identification by Configuration Tool

The configuration tool must write the SII Configured Station Alias and the device needs a power cycle to load the new value to the Register 0x0012.

NOTE: a reload command to the ESC does not reload the Register 0x0012

To load Register 0x0012 remotely by the master the device should implement the Device Identification Reload Object, see chapter 18.6. This avoids the power cycle after change of Configured Station Alias.

18.5 Master behaviour

18.5.1 Configuration Tool

The configuration tool should use the ESI /SII elements IdentificationAdo/IdentificationReg134 for configuration.

If no such information is available the Configured Station Alias Register 0x0012 can be used for Explicit Device Identification.

18.5.2 Master

Before reading the device ID the master should identify the slave, to determine which Identification mechanism shall be used. For example:

- read VendorID and ProductCode first
- then read Explicit Device ID

18.6 Device Identification Reload Object

The Device Identification Reload object should be used to explicitly reload registers in the EtherCAT Slave Controller that are available from the PDI.

The Device Identification Reload object is defined in Table 40 the entry description is given in Table 41.

Table 40: 0x10E0 Device Identification Reload Object

Attribute	Value
Index	0x10E0
Name	Device Identification Reload Object
Object Code	RECORD
Max SubIndex	3

Table 41: Entries of 0x10E0 Device Identification Reload Object

SubIndex	Description	Data Type	Value
0	Maximum supported Subindex	UINT 8	
1	Configured Station Alias register value	UINT16	Write: write value into register 0x0012 Read: read current value of register 0x0012
2	Write Configured Station Alias persistent	BOOL	FALSE: Write access to SI 1 will write value to 0x0012 only TRUE: Write access to SI 1 will write value to 0x0012 and to SII
3	Reload ID-selector value	UINT16	Write: writing 0x0000 to the subindex updates the current ID-selector value into register 0x0012 Read: read current value of ID-selector

The subindex 1 “Configured Station Alias register value” offers the possibility to explicitly reload the register 0x0012.

The purpose of this entry is to handle following use case:

- Remote change of Devcie Identification value in Configured Station Alias register (0x0012) without power-cycle of device

  NOTE: A “Reload EEPROM” command to the ESC register 0x0502 is not sufficient, since all registers except register 0x0012 and 0x0140.9 are reloaded.

If subindex 2 “Write Configured Station Alias persistent” is supported and set to TRUE, the slave shall also write the “Configured Station Alias” in the SII. The slave needs access to the EEPROM, therefore it shall set the Flag “AssignToPdi” in the ESI/SII.

Write access to Index 3 shall only be supported if the device uses Register 0x0012 for explicit device identification with ID-Selector.

19 Error Handling

The motivation of a standard EtherCAT Error Handling is a unique behaviour of EtherCAT devices (Masters and Slaves) in cases of communication problems.

In the following there will be a systematic classification of:

- Error Reasons
- Error Detection
- Error Reaction
- Restart Behaviour

19.1 Error Reasons

There are the following Slave specific Error Reasons:

- Inputs or part of Inputs are invalid (e.g. I/O-Error)
- Inputs are not updated
- No communication of the Application Controller with the ESC
- Application Controller restarted
- Cycle Time exceeded
- Slave powered off
- Slave disconnected
- Inputs are OK, but Outputs cannot be set (e.g. I/O-Error)

The following Master specific Error Reasons can be distinguished:

- Outputs or part of Outputs are invalid (e.g. PLC-Stop)
- Outputs are not updated (e.g. Application exception)
- Master Restart
- Timing Error (e.g. sending of cyclic datagram was not finished when the next cycle started)
- Master disconnected
- Master powered off
- Slave Timing Error
- Distributed Clocks were not started correctly (e.g. System Time Offset is not written)

The following bus specific Error Reasons can be distinguished:

- Single bus faults
- Multiple bus faults
- Bus disconnected

19.2 Error Detection

19.2.1 Mechanism

19.2.1.1 EtherCAT State

The Master should check the actual EtherCAT state of the Slave by reading the Slave’s AL Status Register (0x0130). If the read value does not match with the expected value(s) an error is detected.

The Slave shall check the requested EtherCAT state from the Master by reading the Slave’s AL Control Register (0x0120). If the read value does match with the expected value(s) an error is detected.

Table 42: AL Status Codes used for Synchronization

AL Status Code	Description
0x1A	Multiple synchronization errors. Device is not synchronized any more (shall be used if the following causes cannot be distinguished)
0x2C	Fatal Sync Error: Sync0 or Sync1 are not received any more
0x2D	Sync not received: In SAFEOP the slave waits for the first Sync0/Sync1 events before switching to OP, if these events were not received during the SAFEOP to OP-Timeout time the slave should refuse the state transition to OP with this AL Status Code
0x32	Multiple synchronization errors: PLL is not synchronized any more
0x33	Multiple synchronization errors. IO is not synchronized any more
0x34	Multiple synchronization errors. Too much SM Events missed

19.2.1.1.1 SM Change

An error is detected when the SM Change Event is received and as consequence of the event the SM-Settings are not correct (e.g. SyncManager was disabled).

19.2.1.1.2 Watchdog

The Master should initialize the Slave’s watchdog before switching the Slave to the OPERATIONAL state (usually done in transition Init-to-PreOp) by writing the Slave’s watchdog registers (0x400 and 0x420, the default values represent a 100 ms watchdog). If the watchdog expires an error is detected.

The watchdog will be retriggered every time the SM2-Event is received (SM3-Event if the output size is 0).

19.2.1.1.3 PDO State

If the Slave supports to send the TxPDO State (Index 0x1800+PDO Number-1, SubIndex 7, Data Type BOOLEAN) in a TxPDO (in that case the TxPDO State shall be included in the PDO Mapping of this TxPDO). The Master should check the TxPDO State. If the TxPDO State changes to TRUE (set by the Slave), the data of the related TxPDO is invalid and an error is detected.

If the Slave supports to send the RxPDO State (Index 0x1400+PDO Number-1, SubIndex 7, Data Type BOOLEAN) in a TxPDO (in that case the RxPDO State shall be included in the PDO Mapping of this TxPDO). The Master should check the RxPDO State. If the RxPDO State changes to TRUE (set by the Slave), the data of the related RxPDO could not be put to the hardware and an error is detected.

19.2.1.1.4 PDO Control

If the Slave supports to receive the RxPDO Control (Index 0x1400+PDO Number-1, SubIndex 8, Data Type BOOLEAN) in a RxPDO (in that case the RxPDO Control shall be included in the PDO Mapping of this RxPDO) the Slave should check the RxPDO Control. If the RxPDO Control changes to TRUE (set by the Master), the data of the related RxPDO is invalid and an error is detected.

19.2.1.1.5 PDO Toggle

If the Slave supports to send the TxPDO Toggle (Index 0x1800+PDO Number-1, SubIndex 9, Data Type BOOLEAN) in a TxPDO (in that case the TxPDO Toggle shall be included in the PDO Mapping of this TxPDO) the Master should check the TxPDO Toggle. If the TxPDO Toggle does not toggle with every sync cycle (toggled by the Slave), the data of the related TxPDO is old and an error is detected.

If the Slave supports to receive the RxPDO Toggle (Index 0x1400+PDO Number-1, SubIndex 9, Data Type BOOLEAN) in a RxPDO (in that case the RxPDO Toggle shall be included in the PDO Mapping of this RxPDO) the Slave should check the RxPDO Toggle. If the RxPDO Toggle does not toggle (done by the Master), the data of the related RxPDO is old and an error is detected.

A Profile may support a counter similar to the PDO Toggle (1-Bit Counter) which could be used instead of the PDO Toggle if Master and Slave support this Profile.

19.2.1.1.6 Synchronization Monitoring

1. In SM-synchronous mode an error is detected if the AL-Event is received again before the local cycle is finished.
2. In SM-synchronous mode an error is detected if the SM2-event jitters too much and the application cannot synchronize itself to the SM2 Event any more.
3. In DC Mode an error is detected if the Sync0 event is received and no SM2 Event was received before.
4. In DC Mode an error is detected if the Sync1 event is received or the Output Shift Time is expired before the Output CalcAndCopy is finished.
5. In DC Mode an error is detected if the Sync0 is not received within the Sync0 Cycle Time.
6. In DC Mode an error is detected if the Sync1 is not received within the Sync1 Cycle Time.
7. In DC Mode an error is detected if the Sync0 and/or Sync1 event is never detected

19.2.1.1.7 Working Counter

The Master should check the Working Counter of a received EtherCAT datagram. If the Working Counter does not match with the expected value an error is detected.

19.2.1.1.8 Lost Slaves

The Master may check the number of connected slaves with a BRD datagram for example. If the number of counted slaves does not match with the expected value an error is detected.

19.2.1.1.9 Lost Frames

The Master may use the Index (IDX) of the EtherCAT datagram header to check if all sent EtherCAT datagrams will be received. If EtherCAT datagrams are lost an error is detected.

19.2.1.1.10Local Error

The Master and Slave may detect a local error like a stop of the application or a fault in the connected hardware.

19.2.1.1.11Error Counter in the slave

An error might be detected by the master if an error counter of the slave increments.

19.2.1.1.12Sync Error

The Master may detect a synchronization error in the slave when the Sync Error Flag (0x1C33:20) is TRUE.

19.2.2 Error Settings

Table 43 defines the Error Settings object used for Error Reaction behaviour of the slave. Table 44 describes object entries.

Table 43: 0x10F1 Error Settings Object

Attribute	Value
Index	0x10F1
Name	Error Settings
Max SubIndex	2
Category	Optional
Access	Read Write
PDO Mapping	No

Table 44: 0x10F1 Error Settings

SubIndex	Description	Data Type	Access	Description / Default value
1	Local Error Reaction	UNSIGNED32	RW	0 = PDO State Slave changes the value of the PDO State Bit (either in the PDO Parameter or if it is part of the process data description then there. Slave remains in OP 1 = Disable SyncManager (outputs) from PDI side and go to ErrSafeOp 2 = Device specific state Input PDOs contains a device specific value, e.g. profile/device specific error code
2	Sync Error Counter Limit	UNSIGNED16	RW	Limit set by the master when slave shall change its EtherCAT State to SAFEOP with AL Status Code 0x1A. The Slave shall reset its internal error counter when it is synchronized again and confirms the state transition from SAFEOP to OP.

19.2.2.1 0x10F1:1 Local Error Reaction

If a slave supports several error reactions it shall support the object 0x10F1. It describes which error reaction shall be used in case of an error. The master can write this object.

Default error reaction shall be the setting of the Rx/Tx PDO State if supported. A device that cannot detect a local error may not support the entry 0x10F1:01.

19.2.2.2 0x10F1:2 Sync Error Counter Limit

The SM Event Missed Error Counter increments every time by one when an SM Event was missed. But it does not give any information about the relation between received and missed SM Events. While a single fault can be tolerated multiple errors shall reset the communication until the error is removed.

A weighted "internal" SM Event Missed Counter (in the following Internal Sync Error Counter) is used to count missed and received SM Events. The slave shall increment its Internal Sync Error Counter by 3 in case of a missed event. Every time a SM Event is received the Internal Sync Error Counter shall be decremented by 1.

The smallest value for the Internal Sync Error Counter is 0.

A Sync Error Counter Limit can be set by the master via 0x10F1:2 Sync Error Counter Limit. If the Internal Sync Error Counter exceeds the Sync Error Counter Limit a multiple error is detected and the slave changes its EtherCAT state to SAFEOP with AL Status Code 0x1A. The Sync Error Counter should be reset by the slave when the AL Error was acknowledged.

In Table 45 two examples show the use of the weighted Sync Error Counter.

Table 45: Example for weighted Error Counter

Example	SM Event	1	0	1	0	1	0	1	0	1	0	1
1	Internal Sync Error Counter (incremented by 1) Error Counter Limit = 3	0	1	0	1	0	1	0	1	0	1	0
2	Internal Sync Error Counter (incremented by 3) Error Counter Limit = 9	0	3	2	5	4	7	6	9	9	9	9

In example 1 the Internal Sync Error Counter is incremented by 1 in case of a missed SM2 Event and decremented by 1 in case of a received SM2 Event. Even though only every second SM2 Event is received the Sync Error Counter Limit will not be reached to indicate those multiple errors.

In example 2 the Internal Sync Error Counter is incremented by 3 in case of a missed SM2 Event and decremented by 1 in case of a received SM2 Event. After 8 bus cycles the Sync Error Counter Limit is reached and the multiple error is indicated to the master by an EtherCAT state change to SAFEOP with AL Status Code 0x1A.

Example 2 explains how the error counter shall be implemented for EtherCAT devices that support the Sync Error.

If the Sync Error Counter Limit is set to 0 the device does not change its EtherCAT State to SAFEOP in case of a Sync Error. This might be used for devices with dynamic Minimum Cycle Time.

19.2.3 Error Detection in the Slave

The Slave has the Error Detection mechanisms shown by Table 46.

Table 46: Error Detection Mechanisms by Slave

Mechanism	Use
EtherCAT State	M
SM Change	M
Watchdog	M
PDO Control	O
PDO Toggle	O
Synchronization Monitoring	C, mandatory if object 0x1F01:2 is supported
Local Error	C, mandatory if object 0x1F01:1 is supported

The Master specific Error Reasons will be detected with the mechanisms listed in Table 47.

Table 47: Error Detection of Master Errors

Master Error Reason	Slave Error Detection
Outputs or part of Outputs are invalid (e.g. PLC-Stop)	PDO Control, PDO Toggle
Outputs are not updated (e.g. Application exception)	Watchdog, PDO Toggle, Synchronization Monitoring
Master Restart	EtherCAT State, Watchdog, Synchronization Monitoring
Timing Error (e.g. sending of cyclic telegrams was not finished when the next cycle started)	PDO Toggle, AL Event, Synchronization Monitoring

19.2.4 Error Detection in the Master

Table 48 shows the Error Detection Mechanisms that might be supported by the master.

Table 48: Error Detection Mechanisms by Master

Mechanism	Use
EtherCAT State	M
PDO State	O
PDO Toggle	O
Working Counter	M
Lost Slaves	M
Lost Frames	M
Local Error	C, mandatory if Local error is supported

The Slave specific Error Reasons will be detected with the mechanisms listed in Table 49.

Table 49: Error Detection of Slave Errors

Slave Error Reason	Msaster Error Detection
Inputs or part of Inputs are invalid (e.g. I/O-Error)	PDO State, Working Counter
Inputs are not updated	PDO Toggle, Working Counter
No communication of the Application Controller with the ESC	PDO Toggle, Working Counter
Application Controller restarted	PDO Toggle, Working Counter, EtherCAT State
Cycle Time exceeded	PDO Toggle, Working Counter, Error Counters
Slave powered off	Lost Slaves, Lost Frames
Slave disconnected	Lost Slaves, Lost Frames
Inputs are OK, but Outputs cannot be set (e.g. I/O-Error)	PDO State, Working Counter
Slave Timing Error	Error Counters, Sync Error

19.3 Error Reaction

The Error Reactions are distinguished between local reactions and bus reactions.

19.3.1 Error Reactions in the Slave

19.3.1.1 Local Reactions

Table 50 shows the local reactions.

Table 50: Local Reactions Slave Errors

Local Reactions	Explanation	Use
Use default output data	The outputs will be set to a default value which may be settable by the Master.	M
Use old output data	The outputs will not be changed.	M
Use expected output data	The outputs will be adapted to the expected values.	O
Slave specific output reaction	The outputs will be modified according to a slave specific reaction like a function. This reaction may be configurable by the Master.	O

19.3.1.2 Bus Reactions

Table 51 shows the bus reactions.

Table 51: Bus Reactions Slave Errors

Bus Reactions	Explanation	Use
No reaction	The Slave will not change his EtherCAT behaviour	
TxPDO State	The Slave will set the TxPDO State to TRUE (only if TxPDO State is supported).	C Mandatory if TxPDO State is supported
RxPDO State	The Slave will set the RxPDO State to TRUE (only if RxPDO State is supported).	C Mandatory if RxPDO State is supported
EtherCAT State	The Slave will change the EtherCAT State.	M
Disable Sync Manager Channel	The Slave will disable Sync Manager channel(s) that the Working Counter will not increment any more when accessing the related memory areas.	O used if Local Error (e.g. Input error) would lead to this case. Indication mechanism. Alternative is use of PDO State
Error Counter	The Slave will increment of the error counters of object 0x1C3x	C Mandatory if Error Counters in Object 0x1C3x are supported (SI 11, 12, 13, 14)
Sync Error	The Slave will set the Sync Error bit (0x1Cx:20)	C Mandatory if Sync Error Bit in 0x1C3x:20 is supported

19.3.1.3 Relation between Error Detection and Error Reaction

Table 52 shows the relation between Error Detection and Error Reaction in the Slave.

Table 52: Relation Error Detection and Error Reaction in the Slave

Slave Error Detection	Local Error Reaction	Bus Error Reaction
EtherCAT State, Watchdog, SM-Change	Use default output data or Slave specific output reaction	EtherCAT State
PDO Control	Use default output data or Slave specific output reaction	No reaction
RxPDO Toggle (only single error)	Use old output data or use expected output data	Increment Error Counter (0x1C3x:0E RxPDO Toggle Failed Counter) and set Sync Error = 1
RxPDO Toggle (multiple error = Internal Error Counter Limit exceeded)	Use default output data or Slave specific output reaction	Increment Error Counter (0x1C3x:0E RxPDO Toggle Failed Counter) and set Sync Error = 1 and set EtherCAT State to SAFEOP with AL Status Code 0x33 (or 0x1A). See also 0x10F1:2 Sync Error Counter Limit.
Local Output Error	Use default output data or Slave specific output reaction	RxPDO State or (Disable Sync Manager Output Channel and set EtherCAT state to SAFEOP) (Error behaviour set in 0x10F1 Error Settings)
Local Input Error	Use default output data or Slave specific input reaction	TxPDO State or (Disable Sync Manager Input Channel and set EtherCAT State to SAFEOP) (Error behaviour set in 0x10F1 Error Settings)

Slave Error Detection	Local Error Reaction	Bus Error Reaction
Synchronization Monitoring 1 (only single error)	No reaction	Increment Error Counter (0x1C3x:0B Cycle Time Too Small) and acknowledge the AL Event but do not start a new cycle but dismiss the outputs. Master detects the Error by the TxPDO Toggle if it is supported by the slave.
Synchronization Monitoring 1 (multiple error = Internal Sync Error Counter Limit exceeded)	Use default output data or Slave specific output reaction	set EtherCAT State to SAFEOP with AL Status Code 0x33 (or 0x1A). See also 0x10F1:2 Sync Error Counter Limit.
Synchronization Monitoring 2 (only single error)	Use old output data or use expected output data	set Sync Error = 1
Synchronization Monitoring 2 (multiple error = Internal Sync Error Counter Limit exceeded)	Use default output data or Slave specific output reaction	set EtherCAT State to SAFEOP with AL Status Code 0x32 (or 0x1A). See also 0x10F1:2 Sync Error Counter Limit.
Synchronization Monitoring 3 (only single error)	Use old output data or use expected output data	Increment Error Counter (0x1C3x:0C SM-Event Missed) and set Sync Error = 1
Synchronization Monitoring 3 (multiple error = Internal Sync Error Counter Limit exceeded)	Use default output data or Slave specific output reaction	Increment Error Counter (0x1C3x:0C SM Event Missed) and set EtherCAT State to SAFEOP with AL Status Code 0x34 (or 0x1A). See also 0x10F1:2 Sync Error Counter Limit.
Synchronization Monitoring 4 (only single error)	Use old output data or use expected output data	Increment Error Counter (0x1C3x:0D Shift Time Too Short) and set Sync Error = 1
Synchronization Monitoring 4 (multiple error = Internal Sync Error Counter Limit exceeded)	Use default output data or Slave specific output reaction	Increment Error Counter (0x1C3x:0C Shift Time Too Short) and set EtherCAT State to SAFEOP with AL Status Code 0x32 (or 0x1A). See also 0x10F1:2 Sync Error Counter Limit.
Synchronization 5	Use default output data or Slave specific output reaction	Set EtherCAT state to SAFEOP and AL Status Code to 0x2C (or 0x1A)
Synchronization 6	Use default output data or Slave specific output reaction	Set EtherCAT state to SAFEOP and AL Status Code to 0x2C (or 0x1A)
Synchronization 7	Use default output data or Slave specific output reaction	Refuse state transition to OP and set AL Status Code to 0x2D (or 0x1A)

19.3.2 Error Reactions in the Master

19.3.2.1 Local Reactions

The local reactions may be settable in the Master's configuration tool. They are listed in Table 53.

Table 53: Local Reactions Master Errors

Local Reactions	**Explanation**	**Use**
Status	The error will be signalized in a status field to the application.	M
Use default input data	The inputs will be set to a default value which may be settable by the Master's configuration tool with a default value from the Slave's device description file.	O
Use old input data	The inputs will not be changed.	M
Use expected data	The inputs will be will be adapted to the expected values (in case of a control loop for example).	O
Master specific input reaction	The inputs will be modified according to a master specific reaction like a function.	O

19.3.2.2 Bus Reactions

The bus reactions may be settable in the Master's configuration tool. They are listed in Table 54.

Table 54: Bus Reactions Master Errors

Bus Reactions	**Explanation**	**Use**
No reaction	The Master will not change his EtherCAT behaviour	
RxPDO Control	The Master will set the RxPDO Control to TRUE (only if RxPDO Control is supported).	O
Use default output data	The outputs will be set to a default value which may be settable by the Master's configuration tool with a default value from the Slave's device description file.	O
SyncUnit to SAFEOP	The Master will change the EtherCAT State of the Slave(s) with an application controller and will disable the Sync Manager channel(s) of the Slave(s) which only support the EtherCAT state machine emulation.	O

19.3.2.3 Relation between Error Detection and Error Reaction

Table 55 shows the relation between Error Detection and Error Reaction in the Master.

Table 55: Relation Error Detection and Error Reaction in the Master

Slave Error Detection	Local Error Detection	Bus Error Detection
EtherCAT State	Status and one of the other local reactions (Handling of Input data)	SyncUnit to SAFEOP (for SyncUnit definition see 19.4.2)
RxPDO State	Status	No reaction
TxPDO State	Status and one of the other local reactions (Handling of Input data)	No reaction
TxPDO Toggle	Status and one of the other local reactions (Handling of Input data)	No reaction
Working Counter Single Fault	Status and one of the other local reactions (Handling of Input data)	No reaction
Working Counter Multiple Faults	Status and one of the other local reactions (Handling of Input data)	SyncUnit to SAFEOP
Single Lost Frame	Status and one of the other local reactions (Handling of Input data)	No reaction
Multiple Lost Frames	Status and one of the other local reactions (Handling of Input data)	SyncUnit to SAFEOP
Lost Slaves	Status and one of the other local reactions (Handling of Input data)	SyncUnit to SAFEOP
Local Error	Status	PDO Control if supported or EtherCAT State (of Slave(s) belonging to erroneous application)

19.4 Restart Behaviour

19.4.1 Restart Behaviour in the Slave

When the error is gone the Slave will use the outputs sent from the master instead of the values according to the local reaction.

The Restart Behaviour of the different bus reactions is shown in Table 56.

Table 56: Slave Restart Behaviour

Bus Error Reaction	Restart Behaviour
EtherCAT State	EtherCAT State according to the requested EtherCAT state from the master
RxPDO State	Set to FALSE
TxPDO State	Set to FALSE
Disable Sync Manager Channel(s)	Will be enabled
Error Counter	will not increment any more
SyncError	Set to FALSE

Figure 12 shows the state transitions in SAFEOP and OP of the slave (changes to PREOP or INIT (AL Control Event, SM Change Event) are not shown in this figure to get a better overview). In the following figure the local output error and the local input error shall be ignored if the Error behaviour (0x10F1) is set to PDO State or Device Specific State.

The restart behaviour of the watchdog is settable. If the “watchdog enable bit” of the Sync Manager 2 (3 if no outputs are transmitted in SAFEOP or OP) Control Byte is set, “WD okay” is only TRUE, if new outputs were received (inputs are read), otherwise “WD okay” is always TRUE (master legacy mode). The master legacy mode is optional the other watchdog behaviour is mandatory.

If the slave has to be synchronized in OP, it should delay the state transition confirmation from SAFEOP to OP until it is synchronized or the SAFEOP2OP-Timeout Time (defined in the slave's device description) has expired. It could happen that only one SM event is received before the state transition is requested by the master.

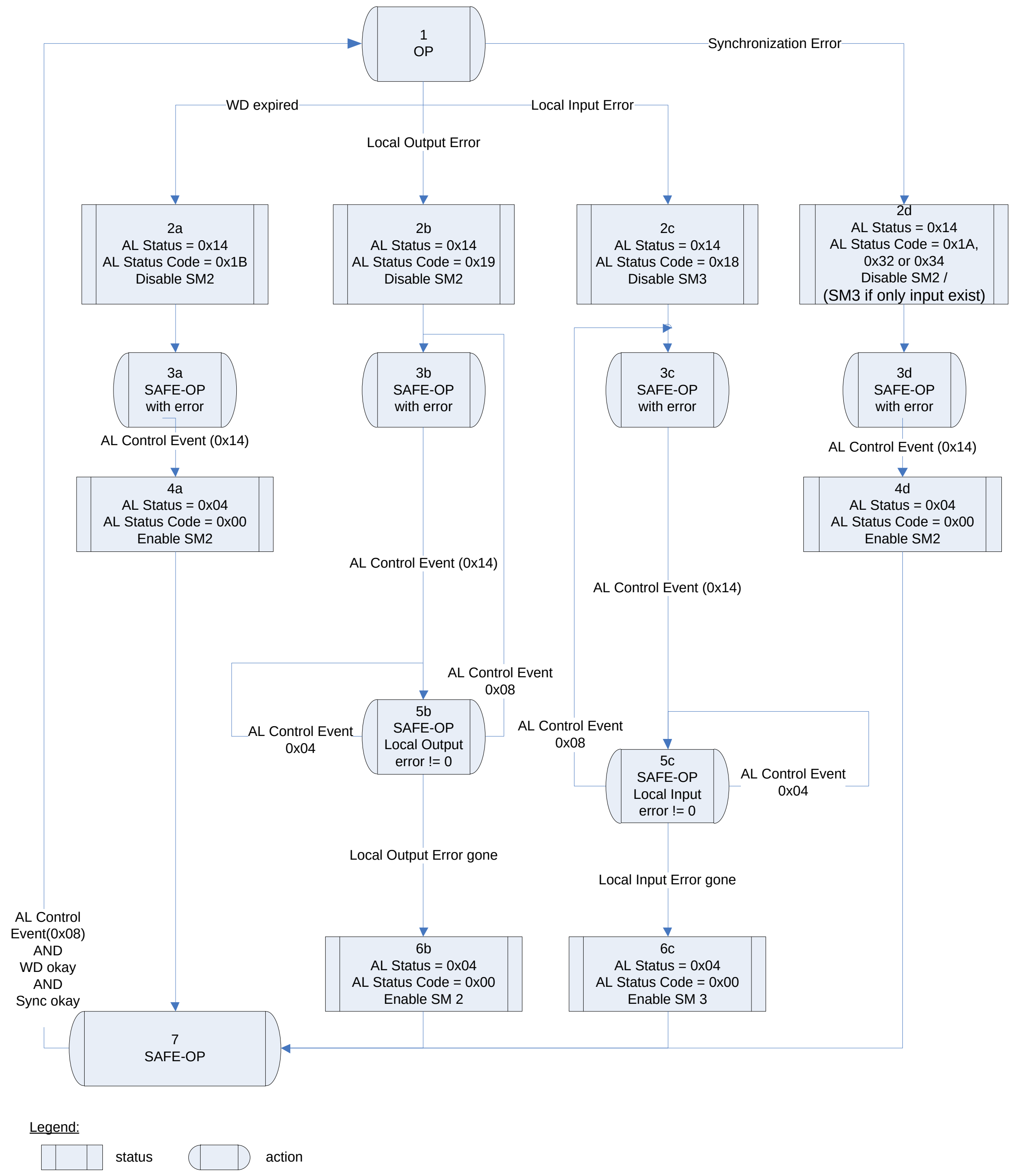

1 OP
Synchronization Error
WD expired
Local Input Error
Local Output Error
2a AL Status = 0x14 AL Status Code = 0x1B Disable SM2
2b AL Status = 0x14 AL Status Code = 0x19 Disable SM2
2c AL Status = 0x14 AL Status Code = 0x18 Disable SM3
2d AL Status = 0x14 AL Status Code = 0x1A, 0x32 or 0x34 Disable SM2 / (SM3 if only input exist)
3a SAFE-OP with error
3b SAFE-OP with error
3c SAFE-OP with error
3d SAFE-OP with error
AL Control Event (0x14)
AL Control Event (0x14)
4a AL Status = 0x04 AL Status Code = 0x00 Enable SM2
4d AL Status = 0x04 AL Status Code = 0x00 Enable SM2
AL Control Event (0x14)
AL Control Event (0x14)
AL Control Event 0x08
AL Control Event 0x04
5b SAFE-OP Local Output error != 0
AL Control Event 0x08
5c SAFE-OP Local Input error != 0
AL Control Event 0x04
Local Output Error gone
Local Input Error gone
AL Control Event(0x08) AND WD okay AND Sync okay
6b AL Status = 0x04 AL Status Code = 0x00 Enable SM 2
6c AL Status = 0x04 AL Status Code = 0x00 Enable SM 3
7 SAFE-OP
Legend:
status
action

Figure 12: Restart Behaviour of the Slave

19.4.2 Restart Behaviour in the Master

When the error is gone the Master will use the inputs sent from the slave(s) and reset the status to the application.

The Restart Behaviour of the different bus reactions is shown in Table 57.

Table 57: Master Restart Behaviour

Bus Error Reaction	Restart Behaviour
EtherCAT State of Sync Unit	See Figure 13
Use default output data	Set to process data values
PDO Control	Set to FALSE

Figure 13 shows the optional bus error reaction and the restart behaviour in the master when changing the EtherCAT state of all slaves belonging to a Sync Unit. A Sync Unit consists at least of all slaves whose process data will be transferred with the same EtherCAT datagram. In that case a local error in one of those slaves could not be assigned exactly so that all slaves related to that EtherCAT datagram should be set to SAFEOP.

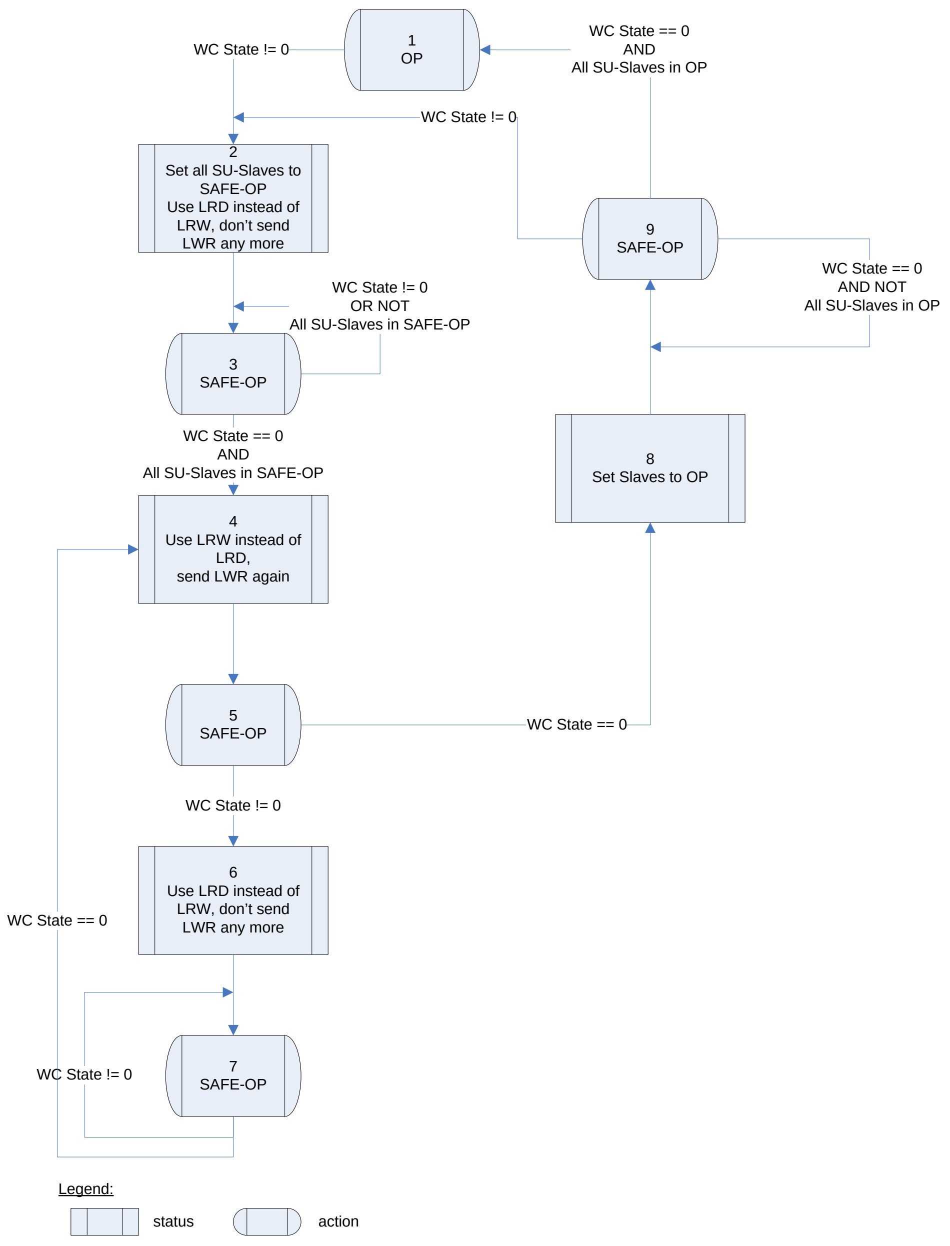

1 OP
WC State != 0
WC State == 0 AND All SU-Slaves in OP
WC State != 0
2 Set all SU-Slaves to SAFE-OP Use LRD instead of LRW, don't send LWR any more
9 SAFE-OP
WC State == 0 AND NOT All SU-Slaves in OP
WC State != 0 OR NOT All SU-Slaves in SAFE-OP
3 SAFE-OP
WC State == 0 AND All SU-Slaves in SAFE-OP
8 Set Slaves to OP
4 Use LRW instead of LRD, send LWR again
5 SAFE-OP
WC State == 0
WC State != 0
6 Use LRD instead of LRW, don't send LWR any more
WC State == 0
7 SAFE-OP
WC State != 0
Legend:
status
action

Figure 13: Restart Behaviour of a Sync Unit in the Master

If the master detects a working counter failure in a process data frame (LRW, LWR, LRD) while the master is in state OP (1), all slaves belonging to this Sync Unit should be set to state SAFEOP (2). All LRW frames belonging to the Sync Unit should be changed to LRD frames, LWR frames should not be sent anymore, so that no invalid outputs will be sent to the slaves.

The master is now in state SAFEOP (3) and waits until that the process data frame(s) (sent with LRD) return(s) with a correct working counter and all slaves belonging to the Sync Unit are in SAFEOP.

If the working counter is correct and all slaves belonging to the Sync-Unit are in SAFEOP, the master should use the original process data frames again (4). If the working counter is still not okay the master should change to LRD-frames again that inputs could be updated (6).

When the working counter of the original process data frame(s) is correct, the error is gone and all slaves belonging to the Sync Unit should be set back to OP state (8, 9). Depending on the master's application an acknowledge is necessary to switch to state 8 and 9.

The master waits until all slaves are in OP (9) and is back in OP (1).

20 Synchronization

The motivation of a standard EtherCAT Synchronization Handling is a unique behavior of EtherCAT devices (Masters and Slaves) for synchronized applications.

The following synchronization modes are distinguished.

Synchronization in the Slave

- Free Run
- Synchronization with SM Event
- Synchronization with Sync0 Event

Synchronization in the Master

- Cyclic Mode
- DC Mode

Configuration of Synchronization

The supported synchronization modes, times and diagnostic features are described with the objects 0x1C32, 0x1C33, 0x1C02, 0x1400-0x15FF and 0x1800-0x19FF. The supported synchronization modes are described in the element OpModes in the EtherCAT Slave Information (ESI) file.

NOTE: The SyncManager Communication Type is described with the Object 0x1C00. The following SyncManager assignment shall be used. If process data in both directions input and output is used then the output data shall be handled in a SyncManager with the lower index and input data shall be handled by the SyncManager with the higher index.

If mailboxes are used the output mailbox shall be assigned to the SyncManager with the lower index (usually SM0) and the input mailbox shall be assigned to the SyncManager (usually SM1) with the higher index.

Table 58 shows when mailbox and process data is used.

Table 58: SyncManger Assignment with Mailbox

SM	Description
0	MailboxOut
1	MailboxIn
2	Output Process Data
3	Input Process Data

If no process output data is used but input process data also SM2 might be used for input process data.

Table 59 shows when only process data is used and no mailbox.

Table 59: SyncManger Assignment without Mailbox

SM	Description
0	Output Process Data
1	Input Process Data

If no process output data is used but process input data also SM0 might be used for process input data.

For description of the SyncManagers the objects 0x1C3y are used (with y = number of SM).

For the following description of the Synchronization Modes the SynManager assignment as described in Table 58 SyncManger Assignment with Mailbox is assumed.

20.1 Synchronization in the Slave

The motivation of a standard EtherCAT Synchronization Handling is a unique behavior of EtherCAT devices (Masters and Slaves) for synchronized applications.

20.1.1 Synchronization Modes in the slave

The following synchronization modes are standardized in this document:

- Free Run
  Slave's application is not synchronized to EtherCAT.
- Synchronous with SM Event
  Slave's application is synchronized to the SM2 event (if cyclic outputs are transmitted) or the SM3 Event (if only cyclic inputs are transmitted). SM events are based on the time an EtherCAT frame is received. This time can jitter in the range of a few microseconds due to the EtherCAT Master implementation (delay in Stack, PHY & MAC Delay, etc).
- Synchronous with DC SYNC Event
  Slave's application is synchronized to the Sync0 or Sync1 event, which are based on the distributed clocks (DC) unit. The jitter could be reduced to a few nanoseconds.

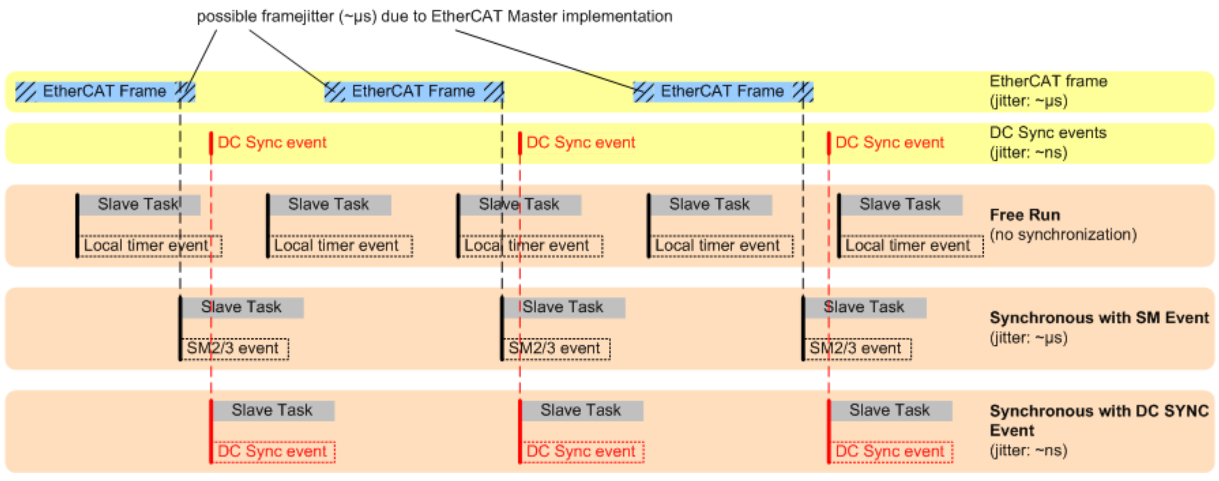

possible framejitter (~µs) due to EtherCAT Master implementation
EtherCAT Frame
EtherCAT Frame
EtherCAT Frame
EtherCAT frame (jitter: ~µs)
DC Sync event
DC Sync event
DC Sync event
DC Sync events (jitter: ~ns)
Slave Task
Local timer event
Free Run (no synchronization)
SM2/3 event
Synchronous with SM Event (jitter: ~µs)
DC Sync event
Synchronous with DC SYNC Event (jitter: ~ns)

Figure 14: Differences between synchronization in the slaves (simplified)

20.1.1.1 Synchronization Mode Identification

The different synchronization types can be identified by the different combinations of the Subindexes of 0x1C32 and 0x1C33 as shown in Table 60.

Clause 20.1.2 specifies the SyncManager Parameters generically.

NOTE: "--" within one table field means that the SubIndex is not used and can either be 0 or not existing.

Table 60: Synchronization Mode Identification

Sync Mode	Synchro-nization Type 0x1C32 SI 01	Synchro-nization Type 0x1C33 SI 01	Output Shift Time 0x1C32 SI 03	Input Shift Time 0x1C33 SI 03	Calc and Copy Time 0x1C32 SI 06	Calc and Copy Time 0x1C33 SI 06	Delay Time 0x1C32 SI 09	Delay Time 0x1C33 SI 09	Fixed Sync0 Cycle Time
Free Run Mode									
Free Run[2]	0x00	0x00	--	--	--	--	--	--	--
SM Event Mode									
SM2[2]	0x01	0x22	--	--	--	--	--	--	--
SM3[2]	--	0x01	--	--	--	--	--	--	--
SM2, Shift Input Latch[2]	0x01	0x22	--	!=0[3]	--	!=0	--	--	--
SM3, Shift Input Latch[2]	---	0x01	--	!=0[3]	--	!=0	--	--	--
DC Mode									
DC Sync0	0x02	0x02	--	--	>=0	!=0	>=0	--	--
DC Sync0, Shift Outputs Valid and Input Latch	0x02	0x02	>=0[3]	>=0[3]	!=0	!=0	>=0	--	--
DC, Shift of Input Latch with Sync1	0x02	0x03	--	--	!=0	!=0	>=0	>=0	--
DC Sync1	0x03	0x03	--	--	!=0	!=0	>=0	--	--
Subordinated Application Controller Cycles									
DC, Shift Outputs Valid/ Input Latch	0x03	0x02 or 0x03	>=0[3]	>=0[3]	>=0	!=0	>=0	>=0	!=0

20.1.1.2 Terminology

20.1.1.2.1 Copy and Prepare Outputs

With a trigger event (local timer event, SM2/3 event Sync0/1 event) output data is read from the SyncManager Output area and, if necessary, mathematical operations are performed on those values. Then, the physical output signal is generated. With the Outputs Valid mark the output signal is available at the process.

The Copy and Prepare Outputs time sums up the times for copying process data from the SyncManager to the local memory, mathematical operations if necessary and the hardware delays (depending on implementation including some software run time). The single times are not further

[2] If Outputs are available, the slave is synchronized always on the SM 2 event. If no outputs are available, the slave is synchronized on the SM 3 event.
[3] Shall be writeable if the related action can be shifted

determined. They would comply with the values described in SyncManager Object 0x1C32 (see Table 61).

Table 61: 0x1C32 Copy and Prepare Outputs

Described Times	SyncManager Object 0x1C32
Copy process data from SyncManager and mathematical operations	Calc and Copy Time (0x1C32:06)
Hardware delay time	Delay Time (0x1C32:09) Hardware delay time shall be kept short and fixed

20.1.1.2.2 Get and Copy Inputs

The Get and Copy Inputs time sums up the hardware delay of reading in the signal, the execution of mathematical operations, if necessary, and the copying of the input process data to the SyncManager 3 area. The single times are not further determined. They would comply with the values described in SyncManager Object 0x1C33 (see Table 62).

Table 62: 0x1C33 Get and Copy Inputs

Described Times	SyncManager Object 0x1C33
Mathematical operations and copy data from local memory to SyncManager	Calc and Copy Time (0x1C33:06)
Hardware delay to prepare the Input Latch	Delay Time (0x1C33:09) Hardware delay time shall be kept short and fixed

The Inputs are available at the SyncManager 3 area after the Min Cycle Time (0x1C32:05).

20.1.1.2.3 Outputs Valid

At the Outputs Valid the outputs are available for the process (e.g. as electrical signal).

20.1.1.2.4 Input Latch

With the Input Latch the input data is acquired but not mathematical operations have been performed on the data and have not been copied to the SyncManager area.

20.1.1.2.5 User Shift Time

The User Shift Time considers jitter of the master.

20.1.1.2.6 Sync1 Cycle Time

The Sync1 Cycle Time might be used for shifting the Start Input Latch or Start Driving Outputs. The Sync1 Cycle Time is described with the register 0x0984:0x0987. It describes the shift time between the Sync0 and Sync1 signal while the Sync0 is always the reference signal.

20.1.1.2.7 Shift Time

Shift Time describes the time between the sync event (SM2 event, SM3 event, SYN0, Sync1) and the Outputs Valid or Input Latch. If the Outputs Valid or Input Latch could be shifted this time should be writable.

20.1.1.3 Free Run

In Free Run Mode the local cycle is started by a local timer interrupt of the application controller. The cycle time may be changed by the Master (optional behaviour of the slave) to change the timer. In Free Run mode the local cycle runs independent from the communication cycle and/ or master cycle.

Optional features

The slave may support a flexible Cycle Time (0x1C32:02 is writable). In this case the Minimum Cycle Time (0x1C32:05) shall be supported by the slave, too.

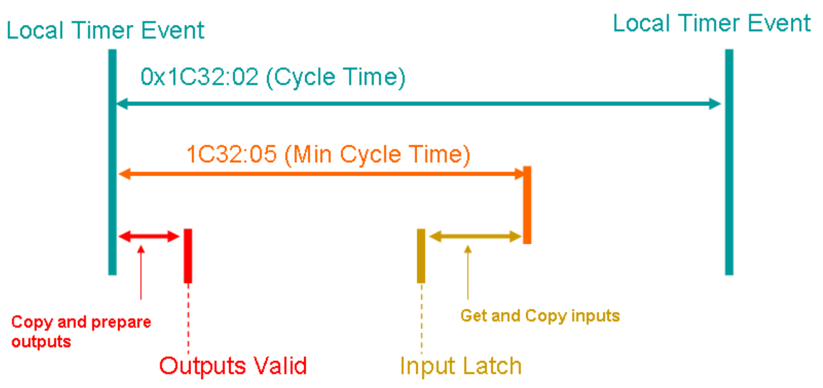


Figure 15: Local Cycle with Free Run synchronization

Table 63 and Table 64 explain how those objects shall be used in Free Run Mode.

Table 63: 0x1C32 Free Run

SubIndex	Description	Access	Use	Description / Default value
1	Synchronization Type	R or RW	M	0x00: Free Run
2	Cycle Time	R or RW	O	Local Cycle Time of application controller
3	Shift Time	--	--	
4	Synchronization Types supported	R	M	Bit 0: Free Run supported
5	Minimum Cycle Time	R	C	Mandatory if 0x1C32:02 is writable
6	Calc and Copy Time	--	--	
7	Minimum Delay Time	--	--	
8	Get Cycle Time	--	--	
9	Delay Time	--	--	
10	Sync0 Cycle Time	--	--	
11	SM-Event missed	--	--	
12	Cycle Time Too Small	--	--	
13	Shift Time Too Short	--	--	
14	RxPDO Toggle Failed	--	--	
15	Minimum cycle distance	--	--	
16	Maximum cycle distance	--	--	
17	Minimum SM SYNC distance	--	--	
18	Maximum SM SYNC distance	--	--	
31:19	--	--	--	
32	Sync Error	--	--	

Table 64: 0x1C33 Free Run

SubIndex	Description	Access	Use	Description / Default value
1	Synchronization Type	R or RW	M	0x00: Free Run
2	Cycle Time	R or RW	O	Same value as 0x1C32:02
3	Shift Time	--	--	
4	Synchronization Types supported	R	M	Same value as 0x1C32:04
5	Minimum Cycle Time	R	C	Same value as 0x1C32:05
6	Calc and Copy Time	--	--	
7	--	--	--	
8	Get Cycle Time	--	--	
9	Delay Time	--	--	
10	Sync0 Cycle Time	--	--	
11	SM-Event missed	--	--	
12	Cycle Time Too Small	--	--	
13	Shift Time Too Short	--	--	
14	RxPDO Toggle Failed	--	--	
31:15	--	--	--	
32	Sync Error	--	--	

20.1.1.4 Synchronous with SM Event

The local cycle is started when the SM2 (with cyclic outputs) or SM3 (without cyclic outputs) event is received.

If outputs are available the slave is synchronized always on the SM 2 event. If no outputs are available, the slave is synchronized on the SM 3 event.

20.1.1.4.1 Synchronous with SM2/3 Event

The local cycle is started when the SM2/3 event is received.

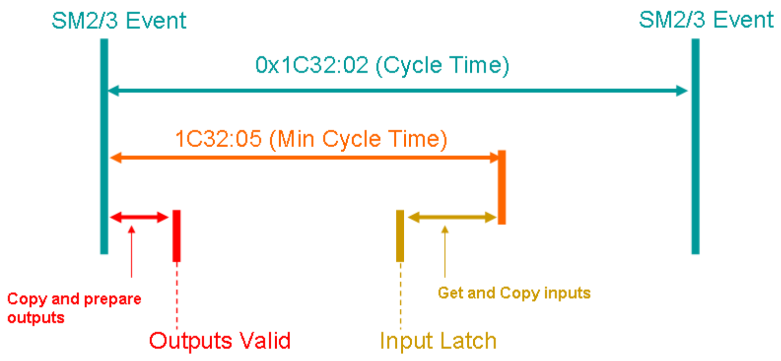


Figure 16: Local Cycle synchronous with SM2 Event

Table 65 and Table 66 explain how those objects shall be used in Synchronous with SM 2/3 Event Mode.

Table 65: 0x1C32 Synchronous with SM 2/3 Event

SubIndex	Description	Access	Use	Description / Default value
1	Synchronization Type	RW	M	0x01: Synchronous – synchronized with SM 2 Event
2	Cycle Time	R or RW	O	Communication cycle time

SubIndex	Description	Access	Use	Description / Default value
3	Shift Time	--	--	
4	Synchronization Types supported	R	M	Bit 1: Synchronous supported
5	Minimum Cycle Time	R	M	
6	Calc and Copy Time	--	--	
7	Minimum Delay Time	--	--	
8	Get Cycle Time	RW	C	Used in Synchronous Mode or in -DC Mode with variable cycle time
9	Delay Time	--	--	
10	Sync0 Cycle Time	--	--	
11	SM-Event Missed	R	O	
12	Cycle Time Too Small	R	M	
13	Shift Time Too Short	--	--	
14	RxPDO Toggle Failed	R	O	
15	Minimum cycle distance	R	O	
16	Maximum cycle distance	R	O	
17	Minimum SM SYNC distance	--	--	
18	Maximum SM SYNC distance	--	--	
31:19	--	--	--	
32	Sync Error	R	C	Shall be supported if SM-Event Missed Counter is supported

Table 66: 0x1C33 Synchronous with SM 2/3 Event

SubIndex	Description	Access	Use	Description / Default value
1	Synchronization Type	RW	M	0x01: Synchronous - synchronized with SM 3 Event (used when only inputs are transmit in SAFEOP and OP) 0x22: Synchronous with SM2 Event (used when outputs are transmit in SAFEOP and OP)
2	Cycle Time	R or RW	O	Same value as 0x1C32:02
3	Shift Time	--	--	
4	Synchronization Types supported	R	M	Same value as 0x1C32:04
5	Minimum Cycle Time	R	M	Same value as 0x1C32:05
6	Calc and Copy Time	--	--	
7	--	--	--	
8	Get Cycle Time	RW	C	Same value as 0x1C32:08
9	Delay Time	--	--	
10	Sync0 Cycle Time	--	--	
11	SM-Event missed	R	O	Same value as 0x1C32:0B
12	Cycle Time Too Small	R	M	Same value as 0x1C32:0C
13	Shift Time Too Short	--	--	
14	RxPDO Toggle Failed	R	O	Same value as 0x1C32:0E
31:15	--	--	--	
32	Sync Error	R	C	Same value as 0x1C32:20

20.1.1.4.2 Synchronous with SM2/3 Event, Shift Input Latch

Input data should be latched from the process as close as possible to the next SM2/3 event to provide the latest input data to the control system (master). To obtain this the input Latch is shifted closer to the SM2/3 event using the Shift Time (0x1C33:03).

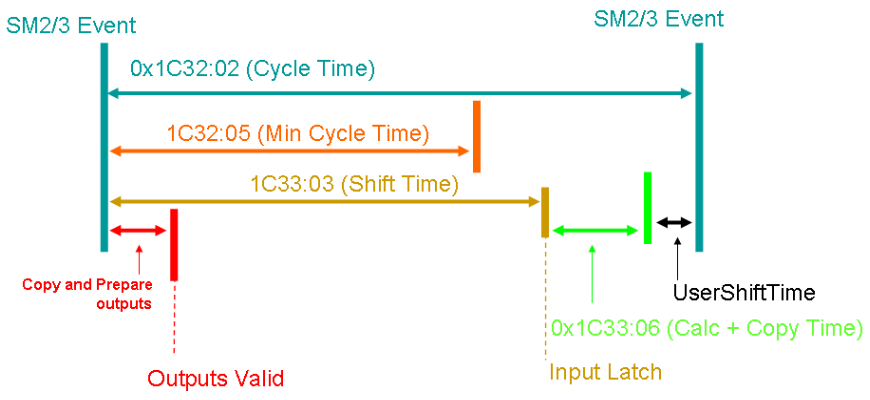


Figure 17: Local Cycle synchronous with SM2/3 Event, Shift Input Latch

Table 67 and Table 68 explain how those objects shall be used in Synchronous with SM 2/3 Event Mode and shifted Input Latch.

Table 67: 0x1C32 Synchronous with SM 2/3 Event, Shift Input Latch

SubIndex	Description	Access	Use	Description / Default value
1	Synchronization Type	RW	M	0x01: Synchronous – synchronized with SM 2/3 Event
2	Cycle Time	R or RW	O	Communication cycle time
3	Shift Time	--	--	
4	Synchronization Types supported	R	M	Bit 1: Synchronous supported
5	Minimum Cycle Time	R	M	
6	Calc and Copy Time	--	--	
7	Minimum Delay Time	--	--	
8	Get Cycle Time	RW	C	Used in Synchronous Mode or in -DC Mode with variable cycle time
9	Delay Time	--	--	
10	Sync0 Cycle Time	--	--	
11	SM-Event missed	R	O	
12	Cycle Time Too Small	R	M	
13	Shift Time Too Short	--	--	
14	RxPDO Toggle Failed	R	O	
15	Minimum cycle distance	R	O	
16	Maximum cycle distance	R	O	
17	Minimum SM SYNC distance	--	--	
18	Maximum SM SYNC distance	--	--	
31:19	--	--	--	
32	Sync Error	R	C	Shall be supported if SM-Event Missed Counter is supported

Table 68: 0x1C33 Synchronous with SM 2/3 Event, Shift Input Latch

SubIndex	Description	Access	Use	Description / Default value
1	Synchronization Type	RW	M	0x01: Synchronous - synchronized with SM 3 Event (used when only inputs avaialble) 0x22: Synchronous with SM2 Event (used when outputs available)
2	Cycle Time	R or RW	O	Same value as 0x1C32:02
3	Shift Time	RW	M	
4	Synchronization Types supported	R	M	Same value as 0x1C32:04
5	Minimum Cycle Time	R	M	Same value as 0x1C32:05
6	Calc and Copy Time	R	M	
7	--	--	--	
8	Get Cycle Time	RW	C	Same value as 0x1C32:08
9	Delay Time	--	--	
10	Sync0 Cycle Time	--	--	
11	SM-Event missed	R	O	Same value as 0x1C32:0B
12	Cycle Time Too Small	R	M	Same value as 0x1C32:0C
13	Shift Time Too Short	--	--	
14	RxPDO Toggle Failed	R	O	Same value as 0x1C32:0E
31:15	--	--	--	
32	Sync Error	R	C	Same value as 0x1C32:20

20.1.1.5 DC Mode

20.1.1.5.1 DC Mode (Synchronous with Sync0 Event)

20.1.1.5.1.1 General synchronization with Sync0 Event

The local cycle is started when the Sync0 event is received. The Process Data Frame (Frame) has to be processed completely in the slave before the next Sync0 Event is received. The Calc + Copy Time contains the minimum time difference between receiving of the frame (SM2 Event) and the Sync0 Event.

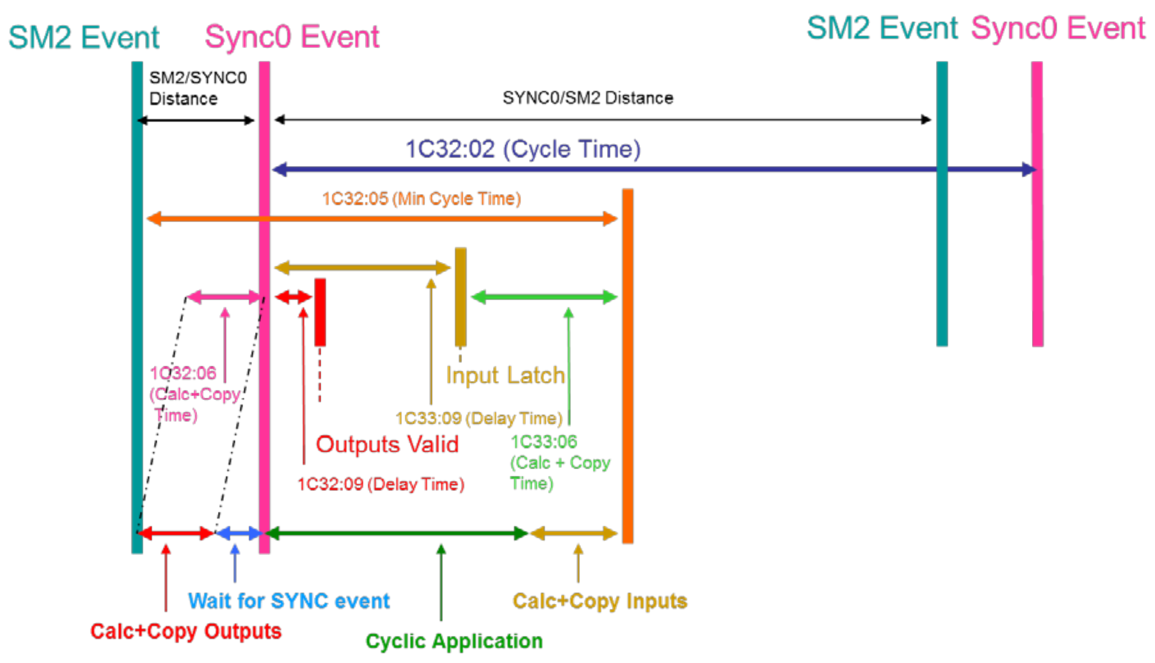


Figure 18: Local Cycle synchronous to Sync0 Event

Table 69 and Table 70 explain how those objects shall be used in DC Mode.

Table 69: 0x1C32 DC Mode

SubIndex	Description	Access	Use	Description / Default value
1	Synchronization Type	RW	M	0x02: DC Sync0 - synchronized with Sync0 Event
2	Cycle Time	R	O	Sync0 Cycle Time (register 0x09A3:0x09A0) Time between two Sync0 events. Sync0 Cycle Time shall be copied to this entry.
3	Shift Time	--	--	
4	Synchronization Types supported	R	M	Bit 4:2 : DC Type supported: 001 = DC Sync0
5	Minimum Cycle Time	R	M	
6	Calc and Copy Time	R	M	Minimum Time between the frame and the Sync0 event
7	Minimum Delay Time	--	--	
8	Get Cycle Time	RW	C	Used in Synchronous Mode or in -DC Mode with variable cycle time
9	Delay Time	R	M	
10	Sync0 Cycle Time	--	--	
11	SM-Event missed Counter	R	O	
12	Cycle Time Too Small	R	M	
13	Shift Time Too Short Counter	R	O	it could be incremented if the SM-event is received too late that the SYNC0-event is received before the calc+copy time ends (master telegram too late)
14	RxPDO Toggle Failed	R	O	
15	Minimum cycle distance	R	O	
16	Maximum cycle distance	R	O	
17	Minimum SM SYNC distance	R	O	
18	Maximum SM SYNC distance	R	O	

SubIndex	Description	Access	Use	Description / Default value
31:19	--	--	--	
32	Sync Error	R	C	Shall be supported if SM-Event Missed Counter (SI12) or Shift Time Too Short Counter (SI13) are supported

Table 70: 0x1C33 DC Mode

SubIndex	Description	Access	Use	Description / Default value
1	Synchronization Type	RW	M	0x02: DC Sync0 – synchronized with Sync0 Event
2	Cycle Time	R	O	Same value as 0x1C32:02
3	Shift Time	--	--	
4	Synchronization Types supported	R	M	Same value as 0x1C32:04
5	Minimum Cycle Time	R	M	Same value as 0x1C32:05
6	Calc and Copy Time	R	M	
7	--	--	--	
8	Get Cycle Time	RW	C	Same value as 0x1C32:08
9	Delay Time	--	--	
10	Sync0 Cycle Time	--	--	
11	SM-Event missed	R	O	Same value as 0x1C32:0B
12	Cycle Time Too Small	R	M	Same value as 0x1C32:0C
13	Shift Time Too Short	R	O	Same value as 0x1C32:0D
14	RxPDO Toggle Failed	R	O	Same value as 0x1C32:0E
31:15	--	--	--	
32	Sync Error	R	C	Same value as 0x1C32:20

20.1.1.5.1.2 Shift of Outputs Valid and/or Input Latch

The Outputs Valid can be delayed by use of the Shift Time (0x1C32:03). The Shift Time describes the time between Sync0 and Outputs Valid.

The Input Latch can be delayed by the use of the Shift Time (0x1C33:03). In this case the Shift Time describes the time between Sync0 and the Input Latch.

SM2 Event shall be received prior to the Sync0 Event.

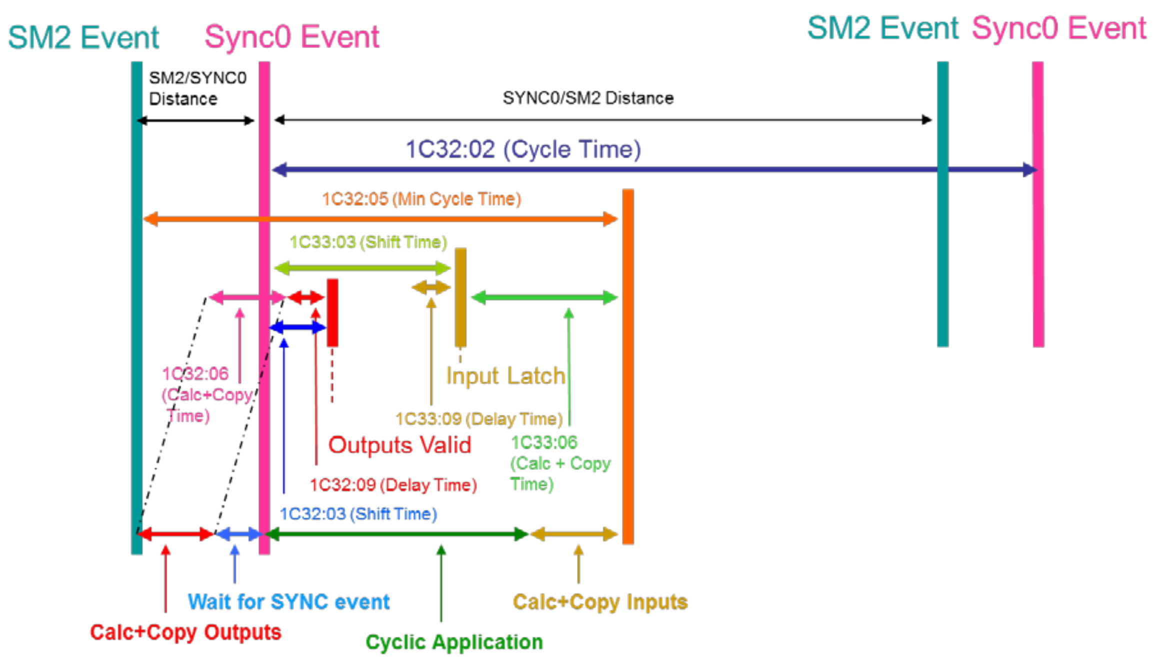


Figure 19: Local Cycle synchronous to Sync0 Event, Shift of Outputs/ Inputs

Table 71 and Table 72 explain how those objects shall be used in DC Mode, Shift of Outputs/ Inputs.

Table 71: 0x1C32 DC Mode, Shift of Outputs/ Inputs

SubIndex	Description	Access	Use	Description / Default value
1	Synchronization Type	RW	M	0x02: DC Sync0 - synchronized with Sync0 Event
2	Cycle Time	R	M	Sync0 Cycle Time (register 0x09A3:0x09A0) Time between two Sync0 events. Sync0 Cycle Time shall be copied to this entry.
3	Shift Time	RW	M	Time between Sync0 event and Outputs Valid (initialized with 0x1C32:09).
4	Synchronization Types supported	R	M	Bit 4:2 : DC Type supported: 001 = DC Sync0 Bit 5-6: Shift Settings 01 = Output Shift with local timer (Shift Time)
5	Minimum Cycle Time	R	M	
6	Calc and Copy Time	R	M	Minimum Time between frame and Sync0
7	Minimum Delay Time	--	--	
8	Get Cycle Time	RW	C	Used in Synchronous Mode or in -DC Mode with variable cycle time
9	Delay Time	R	M	
10	Sync0 Cycle Time	--	--	
11	SM-Event missed	R	O	
12	Cycle Time Too Small	R	M	
13	Shift Time Too Short	R	O	
14	RxPDO Toggle Failed	R	O	
15	Minimum cycle distance	R	O	
16	Maximum cycle distance	R	O	
17	Minimum SM SYNC distance	R	O	
18	Maximum SM SYNC distance	R	O	

SubIndex	Description	Access	Use	Description / Default value
31:19	--	--	--	
32	Sync Error	R	C	Shall be supported if SM-Event Missed or the Shift Time Too Short Counter is supported

Table 72: 0x1C33 DC Mode, Shift of Outputs/ Inputs

SubIndex	Description	Access	Use	Description / Default value
1	Synchronization Type	RW	M	0x02: DC Sync0 – synchronized with Sync0 Event
2	Cycle Time	R	M	Same value as 0x1C32:02
3	Shift Time	RW	M	Time between Sync0 and Input Latch (initialized with 0x1C33:05-0x1C33:06)
4	Synchronization Types supported	R	M	Bit 4:2 : DC Type supported: 001 = DC Sync0 Bit 5-6: Shift Settings 01 = Input Shift with local timer (Shift Time)
5	Minimum Cycle Time	R	M	Same value as 0x1C32:05
6	Calc and Copy Time	R	M	
7	--	--	--	
8	Get Cycle Time	RW	C	Same value as 0x1C32:08
9	Delay Time	--	--	
10	Sync0 Cycle Time	--	--	
11	SM-Event missed	R	O	Same value as 0x1C32:0B
12	Cycle Time Too Small	R	M	Same value as 0x1C32:0C
13	Shift Time Too Short	R	O	Same value as 0x1C32:0D
14	RxPDO Toggle Failed	R	O	Same value as 0x1C32:0E
31:15	--	--	--	
32	Sync Error	R	C	Same value as 0x1C32:20

20.1.1.5.1.3 Shift of Input Latch with Sync1

The Input Latch can be delayed by the use of the Sync1 event (0x1C33:01 = 0x03). In this case the Sync1 Cycle Time describes the time between Sync0 and Start Latch. The Delay Time (1C33:09) contains the hardware delay of the latching.

The Calc + Copy Time contains the minimum time difference between receiving of the frame (SM2 Event) and the Sync0 Event.

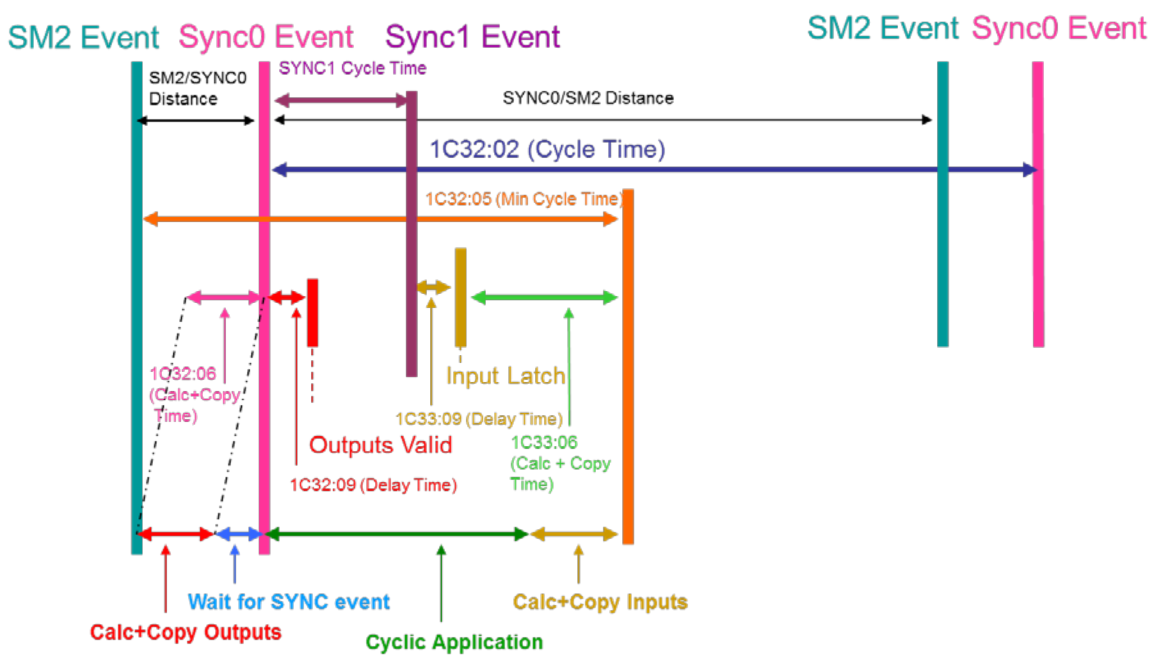


Figure 20: Local Cycle synchronous to Sync0 Event, Shift of Input Latch with Sync1

Table 73 and Table 74 explain how those objects shall be used in DC Mode, Shift of Input Latch with Sync1.

Table 73: 0x1C32 DC Mode, Shift of Input Latch with Sync1

SubIndex	Description	Access	Use	Description / Default value
1	Synchronization Type	RW	M	0x02: DC Sync0 - synchronized with Sync0 Event
2	Cycle Time	R	M	Sync0 Cycle Time (register 0x09A3:0x09A0) Time between two Sync0 events. Sync0 Cycle Time shall be copied to this entry.
3	Shift Time	--	--	
4	Synchronization Types supported	R	M	Bit 4:2 : DC Type supported: 001 = DC Sync0 Bit 5: Shift Settings 0 = No Output Shift supported
5	Minimum Cycle Time	R	M	
6	Calc and Copy Time	R	M	Minimum Time between frame and Sync0
7	Minimum Delay Time	--	--	
8	Get Cycle Time	RW	C	Used in Synchronous Mode or in -DC Mode with variable cycle time
9	Delay Time	R	M	
10	Sync0 Cycle Time	--	--	
11	SM-Event missed	R	O	
12	Cycle Time Too Small	R	M	
13	Shift Time Too Short	R	O	
14	RxPDO Toggle Failed	R	O	
15	Minimum cycle distance	R	O	
16	Maximum cycle distance	R	O	
17	Minimum SM SYNC distance	R	O	

SubIndex	Description	Access	Use	Description / Default value
18	Maximum SM SYNC distance	R	O	
31:19	--	--	--	
32	Sync Error	R	C	Shall be supported if SM-Event Missed or the Shift Time Too Short Counter is supported

Table 74: 0x1C33 DC Mode, Shift of Input Latch with Sync1

SubIndex	Description	Access	Use	Description / Default value
1	Synchronization Type	RW	M	0x03: DC Sync1 - synchronized with Sync1 Event
2	Cycle Time	R	M	Same value as 0x1C32:02
3	Shift Time	--	--	
4	Synchronization Types supported	R	M	Bit 4:2 : DC Type supported: 010 = DC Sync1 Bit 6:5: Shift Settings 10 = Input Shift with Sync1
5	Minimum Cycle Time	R	M	Same value as 0x1C32:05
6	Calc and Copy Time	R	M	
7	--	--	--	
8	Get Cycle Time	RW	C	Same value as 0x1C32:08
9	Delay Time	R	M	Time difference between Start Latch and Latch
10	Sync0 Cycle Time	--	--	
11	SM-Event missed	R	O	Same value as 0x1C32:0B
12	Cycle Time Too Small	R	M	Same value as 0x1C32:0C
13	Shift Time Too Short	R	O	Same value as 0x1C32:0D
14	RxPDO Toggle Failed	R	O	Same value as 0x1C32:0E
31:15	--	--	--	
32	Sync Error	R	C	Same value as 0x1C32:20

20.1.1.5.2 DC Mode (Synchronous with Sync1 Event)

The local cycle is started when the Sync0 event is received. The Process Data Frame (Frame) has to be received before the Sync0 event. The Sync1 is used for Outputs Valid (or Input Latch if no outputs are transmitted), so the Sync1 Cycle Time defines the time difference between Sync0 and Start Driving Outputs. The Calc+Copy Time (1C32:06) is the minimum allowed time for the Sync1 Cycle Time, the Delay Time (1C32:09) defines the hardware delay for driving the outputs.

SM2 Event shall be received prior to the Sync0 Event.

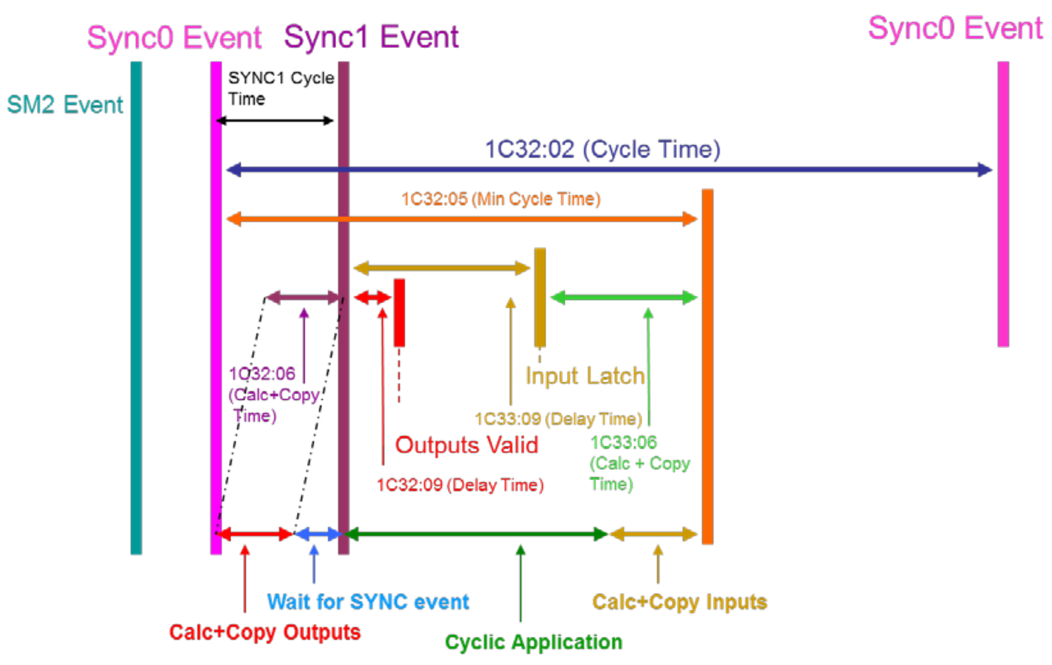


Figure 21: Local Cycle synchronous to Sync1 Event

Table 75 and Table 76 explain how those objects shall be used in DC Mode (Sync1).

Table 75: 0x1C32 DC Mode (Sync1)

SubIndex	Description	Access	Use	Description / Default value
1	Synchronization Type	RW	M	0x03: DC Sync1 - synchronized with Sync1 Event
2	Cycle Time	R	O	Sync0 Cycle Time (register 0x09A3:0x09A0) Time between two Sync0 events. Sync0 Cycle Time shall be copied to this entry.
3	Shift Time	--	--	
4	Synchronization Types supported	R	M	Bit 4:2 : DC Type supported: 010 = DC Sync1
5	Minimum Cycle Time	R	M	
6	Calc and Copy Time	R	M	Minimum Time between Sync0 and Start Driving Outputs (minimum Sync1 Cycle Time)
7	Minimum Delay Time	--	--	
8	Get Cycle Time	RW	C	Used in Synchronous Mode or in -DC Mode with variable cycle time
9	Delay Time	R	M	
10	Sync0 Cycle Time	--	--	
11	SM-Event missed Counter	R	O	
12	Cycle Time Too Small	R	M	
13	Shift Time Too Short Counter	R	O	Counter could be incremented if the distance between SYNC0-event and SYNC1-event is too short (configuration problem)
14	RxPDO Toggle Failed	R	O	
15	Minimum cycle distance	R	O	
16	Maximum cycle distance	R	O	
17	Minimum SM SYNC distance	R	O	
18	Maximum SM SYNC distance	R	O	

SubIndex	Description	Access	Use	Description / Default value
31:19	--	--	--	
32	Sync Error	R	C	Shall be supported if SM-Event Missed Counter (SI12) or Shift Time Too Short Counter (SI13) are supported

Table 76: 0x1C33 DC Mode (Sync1)

SubIndex	Description	Access	Use	Description / Default value
1	Synchronization Type	R	M	0x03: DC Sync1 - synchronized with Sync1 Event
2	Cycle Time	R	O	Same value as 0x1C32:02
3	Shift Time	--	--	
4	Synchronization Types supported	R	M	Same value as 0x1C32:04
5	Minimum Cycle Time	R	M	Same value as 0x1C32:05
6	Calc and Copy Time	R	M	
7	--	--	--	
8	Get Cycle Time	RW	C	Same value as 0x1C32:08
9	Delay Time	--	--	
10	Sync0 Cycle Time	--	--	
11	SM-Event missed	R	O	Same value as 0x1C32:0B
12	Cycle Time Too Small	R	M	Same value as 0x1C32:0C
13	Shift Time Too Short	--	--	
14	RxPDO Toggle Failed	R	O	Same value as 0x1C32:0E
31:15	--	--	--	
32	Sync Error	R	C	Same value as 0x1C32:20

20.1.1.6 DC, subordinated application controller cycles

20.1.1.6.1 General

For slaves with fast local cycle times (e.g. control loops) the cycle time of the application controller might be much faster than the communication cycle time. In this case two synchronization features are distinguished:

1. Shift of Outputs Valid
2. Shift of Input Latch

If a subordinated drive cycle is used with DC, Sync1 shall be used for the communication cycle time of the master. Output Shift and Input Shift shall be related to Sync1. Sync0 can be used for the subordinated internal cycle.

20.1.1.6.2 DC, subordinated µC cycles, Shift of Outputs Valid and/or Input Latch

The Sync0 signal is only used to trigger the local microcontroller cycle. The Outputs Valid and Input Latch are triggered by the Sync1 Event and can only be shifted by the Output Shift Time and Input Shift Time.

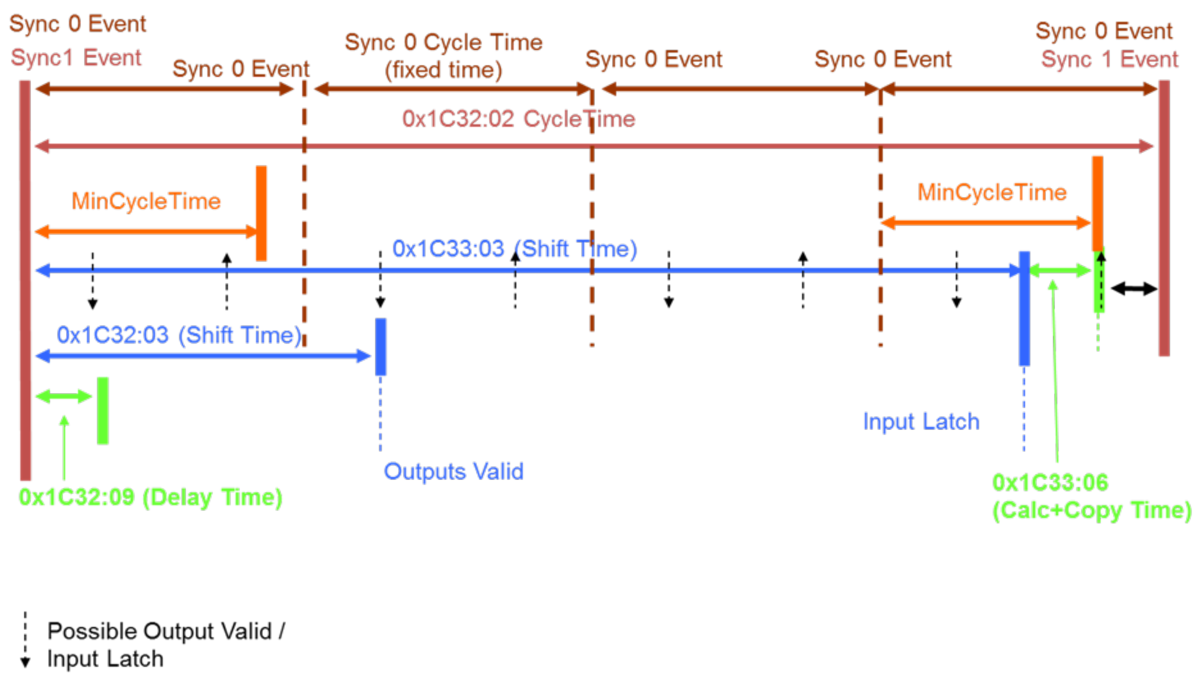


Figure 22: DC, subordinated µC cycles, Outputs/ Inputs delayed

Table 77 and Table 78 explain how those objects shall be used in DC Mode, subordinated µC cycles, Outputs Valid/ Input Latch shifted.

Table 77: 0x1C32 DC Mode, subordinated µC cycles, Shift Outputs/ Inputs

SubIndex	Description	Access	Use	Description / Default value
1	Synchronization Type	RW	M	0x03: DC Sync1 – synchronized with Sync1 Event
2	Cycle Time	R	M	Sync1 Cycle Time (register 0x09A7:0x09A4) Time between two Sync1 events Sync1 Cycle Time shall be copied by the slave application.
3	Shift Time	RW	O	
4	Synchronization Types supported	R	M	Bit 4:2 : DC Type supported: 100 = Subordinated Application Bit 6:5: Shift Settings 00 = No Output Shift supported 01 = Output Shift with local timer (Shift Time)
5	Minimum Cycle Time	R	M	
6	Calc and Copy Time	R	M	
7	--	--	--	
8	Get Cycle Time	RW	C	Used in Synchronous Mode or in -DC Mode with variable cycle time
9	Delay Time	R	M	
10	Sync0 Cycle Time	R	M	
11	SM-Event missed	R	O	
12	Cycle Time Too Small	R	M	
13	Shift Time Too Short	R	O	
14	RxPDO Toggle Failed	R	O	
15	Minimum cycle distance	R	O	
16	Maximum cycle distance	R	O	

SubIndex	Description	Access	Use	Description / Default value
17	Minimum SM SYNC distance	R	O	
18	Maximum SM SYNC distance	R	O	
31:19	--	--	--	
32	Sync Error	R	C	Shall be supported if SM-Event Missed or the Shift Time Too Short Counter is supported

Table 78: 0x1C33 DC Mode, subordinated µC cycles, Shift Outputs/ Inputs

SubIndex	Description	Access	Use	Description / Default value
1	Synchronization Type	RW	M	0x03: DC Sync1 – synchronized with Sync1 Event
2	Cycle Time	R	M	Same value as 0x1C32:02
3	Shift Time	R	O	
4	Synchronization Types supported	R	M	Bit 4:2 : DC Type supported: 100 = Subordinated Application Bit 6:5: Shift Settings 00 = No Output Shift supported 01 = Output Shift with local timer (Shift Time)
5	Minimum Cycle Time	R	M	Same value as 0x1C32:05
6	Calc and Copy Time	R	M	
7	--	--	--	
8	Get Cycle Time	RW	C	Same value as 0x1C32:08
9	Delay Time	R	M	
10	Sync0 Cycle Time	R	M	Same value as 0x1C32:0A
11	SM-Event missed	R	O	Same value as 0x1C32:0B
12	Cycle Time Too Small	R	M	Same value as 0x1C32:0C
13	Shift Time Too Short	R	O	Same value as 0x1C32:0D
14	RxPDO Toggle Failed	R	O	Same value as 0x1C32:0E
31:15	--	--	--	
32	Sync Error	R	C	Same value as 0x1C32:20

20.1.2 SyncManager Parameter

Table 79 defines the SyncManager Parameter object.

Table 79: 0x1C32 / 0x1C33 SyncManager Parameter objects

Attribute	Value
Index	0x1C32, 0x1C33
Name	SyncManager Parameter
Max SubIndex	32
Category	Conditional Optional only, if 0x1C32:01 = 0x00 Free run mode
PDO Mapping	No

The SyncManager Parameter Objects are defined in Table 80 and Table 81.

If no outputs are available the object 0x1C32 does not exist.

Table 80: Output SyncManager Parameter (0x1C32)

Sub-Index	Description	Data Type	Access	Use	Description / Default value
1	Synchronization Type	UNSIGNED16	R or RW	M	0x00: Free Run (not synchronized) 0x01: Synchronous – synchronized with SM Event 0x02: DC Sync0 – synchronized with Sync0 Event 0x03: DC Sync1 – synchronized with Sync1 Event NOTE: Entry only writable with new value in SafeOp or OP if Synchronization can be changed in those states
2	Cycle Time	UNSIGNED32	R or RW	O	Free Run (Synchronization Type = 0x00): Time between two local timer events in ns Synchronous with SM2 (Synchronization Type = 0x01): Minimum time between two SM2 events in ns DC Sync0 (Synchronization Type = 0x02): Sync0 Cycle Time (Register 0x9A3-0x9A0) in ns NOTE: Entry only writable with new value in SafeOp or OP if Synchronization can be changed in those states
3	Shift Time	UNSIGNED32	R or RW	O	Time between related event and the associated action (outputs valid hardware) in ns Shift of Output valid equal or greater than 0x1C32:09 NOTE: Entry only writable with new value in SafeOp or OP if Synchronization can be changed in those states

Sub-Index	Description	Data Type	Access	Use	Description / Default value
4	Synchronization Types supported	UNSIGNED16	R	M	Bit 0: Free Run supported Bit 1: Synchronous supported Bit 4:2 : DC Type supported: 000 = No DC 001 = DC Sync0 010 = DC Sync1 100 = Subordinated Application with fixed Sync0 Bit 6:5: Shift Settings 00 = No Output Shift supported 01 = Output Shift with local timer (Shift Time) 10 = Output Shift with Sync1 Bit 9…6: Reserved for future use Bit 10: Delay Times should be measured (because they depend on the configuration) Bit 11: Delay Time is fix (synchronization is done by hardware) Bit 13…11: Reserved for future use Bit 14: Dynamic Cycle Times Times described in 0x1C32 are variable (depend on the actual configuration) This is used for e.g. EtherCAT gateway devices. The slave shall support cycle time measurement in OP state. The cycle time measurement is started by writing 1 to 0x1C32:08. If this bit is set, the default values of the times to be measured (Minimum Cycle Time, Calc And Copy Time, Delay Time) could be 0. The default values could be set in INIT and PREOP state. Bit 14 should only be set, if the slave cannot calculate the cycle time after receiving all Start-up SDO in transition PS. Bit 15: Reserved for future use
5	Minimum Cycle Time	UNSIGNED32	R	C	Minimum cycle time supported by the slave (maximum duration time of the local cycle) in ns It might be necessary to start the Dynamic Cycle Time measurement SI04, Bit 14 and SI08, Bit 0 to get a valid value used in Synchronous or DC Mode
6	Calc and Copy Time	UNSIGNED32	R	C	Time needed by the application controller to copy the process data from the Sync Manager to the local memory and perform calculations if necessary before the data is sent to the process. Minimum time for Outputs to SYNC-Event used in DC mode
7	Minimum Delay Time	UNSIGNED32	R	O	Only important for DC Sync0/1 (Synchronization type = 0x02 or 0x03): Minimum Hardware delay time of the slave. because of software synchronization there could be a distance between the minimum and the maximum delay time Distance between minimum and maximum delay time (Subindex 9) shall be smaller than 1 µs. if SubIndex 4, Bit 11 is 1, this SubIndex contains the value of SubIndex 9
8	Get Cycle Time	UNSIGNED16	RW	C	Bit 0: 0: Measurement of local cycle time stopped 1: Measurement of local cycle time started. If written again, the measured values are reset Used in Synchronous or (DC Mode with variable Cycle Time) Bit 1: 0: -- 1: Reset the error counters Bit 2…15: reserved for future use

Sub-Index	Description	Data Type	Access	Use	Description / Default value
9	Delay Time	UNSIGNED32	R	C	Only important for DC Sync0/1 (Synchronization type = 0x02 or 0x03): Hardware delay time of the slave. Time from receiving the trigger (Sync0 or Sync1 Event) to drive output values to the time until they become valid in the process (e.g. electrical signal available). if Subindex 7 Minimum Delay Time is supported and unequal 0, this Delay Time is the Maximum Delay Time
10	Sync0 Cycle Time	UNSIGNED32	RW	O	Only important for DC Sync0 (Synchronization type = 0x03) and subordinated local cycles: Time between two Sync0 signals if fixed Sync0 Cycle Time is needed by the application
11	SM-Event Missed	UNSIGNED16	R	C	This error counter is incremented when the application expects a SM event but does not receive it in time and as consequence the data cannot be copied any more. used in DC Mode
12	Cycle Time Too Small	UNSIGNED16	R	C	This error counter is incremented when the cycle time is too small so that the local cycle cannot be completed and input data cannot be provided before the next SM event. used in Synchronous or DC Mode
13	Shift Time Too Short	UNSIGNED16	R	C	This error counter is incremented when the time distance between the trigger (Sync0) and the Outputs Valid is too short because of a too short Shift Time or Sync1 Cycle Time. used in DC Mode
14	RxPDO Toggle Failed	UNSIGNED16	R	O	This error counter is incremented when the slave supports the the RxPDO Toggle and does not receive new RxPDO data from the master (RxPDO Toggle set to TRUE)
15	Minimum Cycle Distance	UNSIGNED32	R	O	Minimum Cycle Distance in ns used in conjunction with SI16 to monitor the jitter between two SM-events
16	Maximum cycle distance	UNSIGNED32	R	O	Maximum cycle distance in ns used in conjunction with SI15 to monitor the jitter between two SM-events
17	Minimum SM SYNC Distance	UNSIGNED32	R	O	Minimum SM SYNC Distance in ns used in conjunction with SI18 to monitor the jitter between the SM-event and SYNC0-event in DC-SYNC0-Mode
18	Maximum SM SYNC Distance	UNSIGNED32	R	O	Maximum SM SYNC Distance in ns used in conjunction with SI17 to monitor the jitter between the SM-event and SYNC0-event in DC-SYNC0-Mode
31:19	--	--	--		Reserved for future use.
32	Sync Error	BOOL	R	C	Shall be supported if SM-Event Missed or Shift Time Too Short Counter is supported Mappable in TxPDO 0: no Synchronization Error or Sync Error not supported 1: Synchronization Error

Table 81 describes the settings for the Input SyncManager.

If no inputs are available the object 0x1C33 does not exist.

Table 81: Input SyncManager Parameter (0x1C33)

Sub-Index	Description	Data Type	Access	Use	Description / Default value
1	Synchronization Type	UNSIGNED16	R or RW	M	Synchronization Type according to 0x1C32:01 0x00: Free Run (not synchronized) 0x01: Synchronous with SM3 Event 0x02: DC Sync0 – synchronized with Sync0 Event 0x03: DC Sync1 – synchronized with Sync1 Event 0x22: Synchronous with SM2 Event NOTE: Entry only writable with new value in SafeOp or OP if Synchronization can be changed in those states
2	Cycle Time	UNSIGNED32	R	O	Same time as for 0x1C32:02
3	Shift Time	UNSIGNED32	R or RW	O	Time between related event and the associated action (inputs latched from hardware) in ns shift of Input Latch equal or greater than 0x1C33:09 NOTE: Entry only writable with new value in SafeOp or OP if Synchronization can be changed in those states
4	Synchronization Types supported	UNSIGNED16	R	M	Bit 0: Free Run supported Bit 1: Synchronous supported Bit 4:2 : DC Type supported: 000 = No DC 001 = DC Sync0 010 = DC Sync1 100 = Subordinated Application with fixed Sync0 Bit 6:5: Shift Settings 00 = No Input Shift supported 01 = Input Shift with local timer (Shift Time) 10 = Input Shift with Sync1 Bit 13:6: Reserved for future use Bit 14: Dynamic Cycle Times Same meaning as in 0x1C32:04 Bit 15: Reserved for future use
5	Minimum Cycle Time	UNSIGNED32	R	C	Same time as 0x1C32:05 used in Synchronous or DC Mode
6	Calc and Copy Time	UNSIGNED32	R	C	Time in ns needed by the application controller to perform calculations on the input values if necessary and to copy the process data from the local memory to the Sync Manager before the data is available for EtherCAT. Minimum time for Inputs after Input Latch used in DC mode
7	--	UNSIGNED32	--		Reserved for future use
8	Get Cycle Time	UNSIGNED16	RW	C	Same meaning as 0x1C32:08 used in Synchronous or (DC Mode with variable Cycle Time)
9	Delay Time	UNSIGNED32	R	C	Only important for DC Sync1 (Synchronization type = 3), if the Input Latch is started by the Sync1-event Hardware delay time of the slave. used in DC mode
10	Sync0 Cycle Time	UNSIGNED32	R	C	Same value as 0x1C32:0A used in Synchronous or DC Mode
11	SM-Event Missed	UNSIGNED16	R	O	Same value as 0x1C32:0B
12	Cycle Time Too Small	UNSIGNED16	R	C	Same value as 0x1C32:0C used in SM Synchronous or DC Mode

Sub-Index	Description	Data Type	Access	Use	Description / Default value
13	Shift Time Too Short	UNSIGNED16	R	O	Same value as 0x1C32:0D
14	reserved	UNSIGNED16			reserved
31:15	--	--	--		Reserved for future use.
32	Sync Error	BOOL	R	C	Same value as 0x1C32:20

20.1.3 Diagnosis of the Synchronization Mode

A cycle synchronous diagnosis can be done by setting the sync manager channels 2 and 3 in 1-buffer mode. In that case the working counter would not match if a sync error had happened.

The disadvantage of the 1 buffer mode is that all data of an EtherCAT datagram is invalid if a synchronization failure happens. Alternatively the PDO-Toggle can be used:

With the PDO-Toggle the sync manager should work in 3-buffer mode that a synchronization failure will not influence the data of other slaves. The corresponding toggle bit can be checked by the master and slave.

If the slave detects a synchronization error this information shall be reported in the Sync Error Entry (0x1C32:20, 0x1C33:20).

So the master could detect synchronization problems cycle synchronous by checking the PDO Toggle and the Sync Error.

20.1.3.1 Error Behaviour of the Slave

Table 82 shows the error behavior of the slave.

Table 82: Error Behaviour of the Slave

Error Type	Slave Behaviour
Communication Cycle Time (0x1C32:02) < Minimum Cycle Time (0x1C32:05)	After the master has written the cycle time via an SDO service the slave could check with the Minimum Cycle Time. If the Cycle Time is smaller than the Minimum Cycle Time the slave should return an Abort (Abort Code 0x06090032, Value too short) on this SDO service. In DC Mode the Communication Cycle Time will be overwritten with the Sync0 Cycle Time (Register 0x9A3:0x9A0). The slave should refuse the state transition from PREOP to SAFEOP with AL Status Code 0x36 (or 0x35) if the Sync0 Cycle Time (Register 0x9A3:0x9A0) is too small (or too great). If the Minimum Cycle Time is variable the slave might only check the timing during SAFEOP or OP. In that case the slave behaves like described for Error Type "too short interval between two sync events"
Sync1 used for Outputs Valid and Sync1 Cycle Time < Output Calc And Copy Time (0x1C32:06) or Sync1 used for Input Latch and Sync1 Cycle Time < (Input Shift Time (0x1C33:03) - Input Delay Time (0x1C33:09))	The slave should refuse the state transition from PREOP to SAFEOP with AL Status Code 0x37 (or 0x35) if the Sync1 Cycle Time (Register 0x9A7:0x9A4) is too small (or too great). If the Times are variable the slave might only check the timing during SAFEOP or OP. In that case the slave behaves like described for Error Type "distance between Sync0 and Sync1 signal too short"
Invalid sync or latch configuration	The slave should refuse the state transition from PREOP to SAFEOP with AL Status Code 0x30 or 0x31 if the sync or latch configuration does not match.
Interval between two sync events too short	If the slave detects a too short interval between two SM2/3 Events it should increment the Cycle Time Too Small Counter (0x1C3x:0C) and acknowledge the SM2/(3) event and wait for the next SM2/(3) event. The master could detect this error by checking the PDO Toggle. If the Internal Error Counter Limit exceeds, the slave shall leave its state to SAFEOP and set the AL Status Code to 0x33 (or 0x1A) (See also 0x10F1:02 in Table 44). For the SAFEOP2OP-transition the AL-Status-Code 0x32 should be generated in this case.
No SM2 Event before Sync0 event (only for DC Modes)	If the slave detects that no SM2 event was received when the Sync0 event was received, the SM-Event Missed Counter (0x1C3x:0B) shall be incremented and the Sync Error Flag (0x1C32:20) shall be set. If the Internal Error Counter Limit exceeds, the slave shall leave its state to SAFEOP and set the AL Status Code to 0x34 (or 0x1A) (See also 0x10F1:02 in Table 44).
Distance between Sync0 and Sync1 signal too short (only for DC Modes)	If the slave detects a too short distance between Sync0 and Sync1 signal, the Shift Too Short Counter (0x1C3x:0D) shall be incremented and the Sync Error Flag (0x1C32:20) shall be set. If the Internal Error Counter Limit exceeds, the slave shall leave its state to SAFEOP and set the AL Status Code to 0x33 (or 0x1A) (See also 0x10F1:02 in Table 44).
PLL could not be synchronized any more	If the slave has a PLL is used to synchronize its local cycle to the communication cycle and the PLL be could be synchronized any more, the slave shall leave its state to PREOP and set the AL Status Code to 0x32 (or 0x1A).
no Sync0/Sync1 signal any more (only for DC Modes) If the slave detects in SAFEOP or OP that no Sync0 or Sync1 event is generated any more it shall leave its state to SAFEOP and set the AL Status Code to 0x2C (or 0x1A).	no Sync0/Sync1 signal any more (only for DC Modes) If the slave detects in SAFEOP or OP that no Sync0 or Sync1 event is generated any more it shall leave its state to SAFEOP and set the AL Status Code to 0x2C (or 0x1A).
Timeout by waiting for first Sync0/Sync1 signal (only for DC Modes)	If the slave gets a timeout by waiting for the Sync0/Sync1-event in SAFEOP it shall refuse the state transition to OP and set the AL Status Code to 0x2D (or 0x1A).

20.2 Synchronization in the Master

The master normally has two different synchronization modes:

- Cyclic Mode
- DC Mode

20.2.1 Cyclic Mode

In Cyclic Mode the master should send the process data frames cyclically. The process data frames may be sent with different cycle times. The master cycle is usually controlled by a local timer on the master. The slaves may be operated in Free Run or synchronous with the SM event mode. For slaves running in synchronous mode the master should check that the cycle time of the corresponding process data frames is greater than the Minimum Cycle Time of the slaves (0x1C3x:05 Sync Manager Parameter).

20.2.2 DC Mode, Master follows Reference Clock

In DC Mode the master is behaving like in Cyclic Mode but the local cycle should be synchronized with the Master Clock. The local timer of the master should be adjusted with the ARMW datagram which distributes the master clock.

In DC Mode usually all slaves supporting DC should be synchronized to the DC Base Time. The master should synchronize its cycle to this DC Base Time, too.

Figure 23 shows how the local cycle of the master may be synchronized to the DC Base Time.

DC Base

The DC Base Time is a virtual time that is used to set the master cycle and communication cycle in to relation to the reference clock. The DC Base Time and the reference time have a fixed relation.

Local Timer

Time of Master that is adjusted to the master clock (reference time)

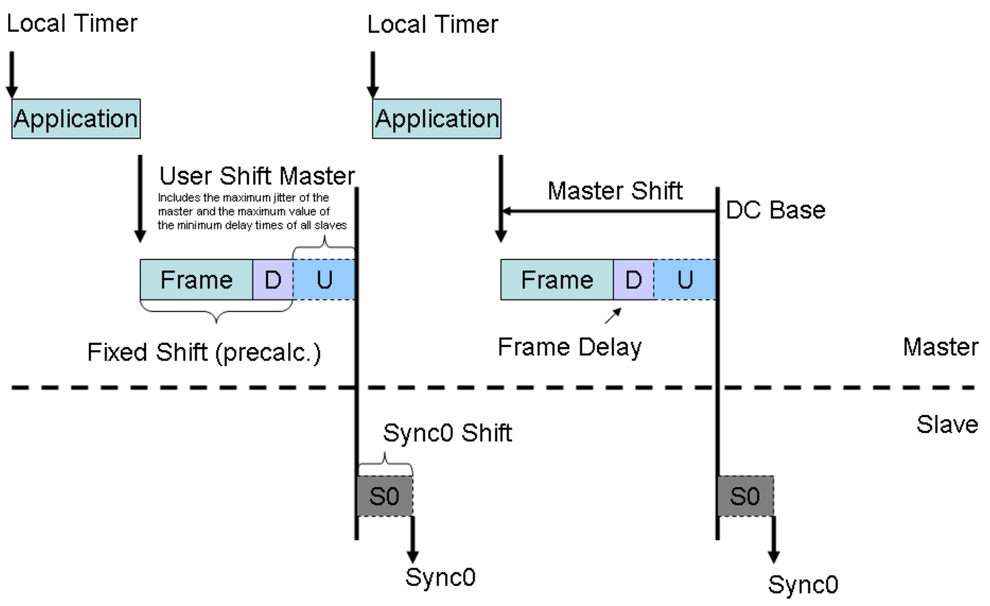


Figure 23: Master synchronized to DC Base

The local application is started with a local timer. The local timer is shifted to the DC Base Time by the sum of the following times:

- Duration of the application execution time (Application)
- Frame transmission time (Frame)
- Frame transmission delay (D)

- User Shift (U) which shall include the maximum of the minimum delay times of the slaves and the maximum jitter of the execution of the application.
  U+: positive User Shift as shown in Figure 24
  U-: negative User Shift as shown in Figure 24

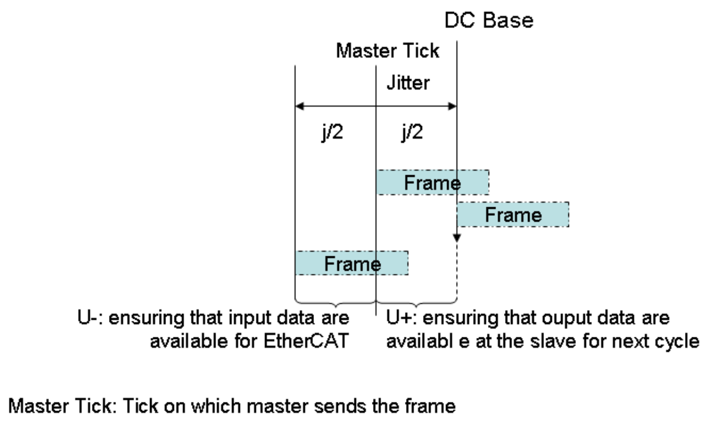

DC Base
Master Tick
Jitter
j/2
j/2
Frame
Frame
Frame
U-: ensuring that input data are available for EtherCAT
U+: ensuring that ouput data are availabl e at the slave for next cycle
Master Tick: Tick on which master sends the frame

Figure 24: User Shift

The cycle time of the local timer will be controlled by reading the system time (register 0x910-0x913) of the Master Clock with an ARMW command cyclically.

The Sync0 event of each slave may be shifted to the DC Base Time by writing the Start Time of the cyclic Operation (Register 0x990-0x993) for the slaves.

Master Settings for use of DCs

The DC Base signal is a virtual signal used as timing reference. Usually this should be synchronous with the Start time of the DC Reference Clock.

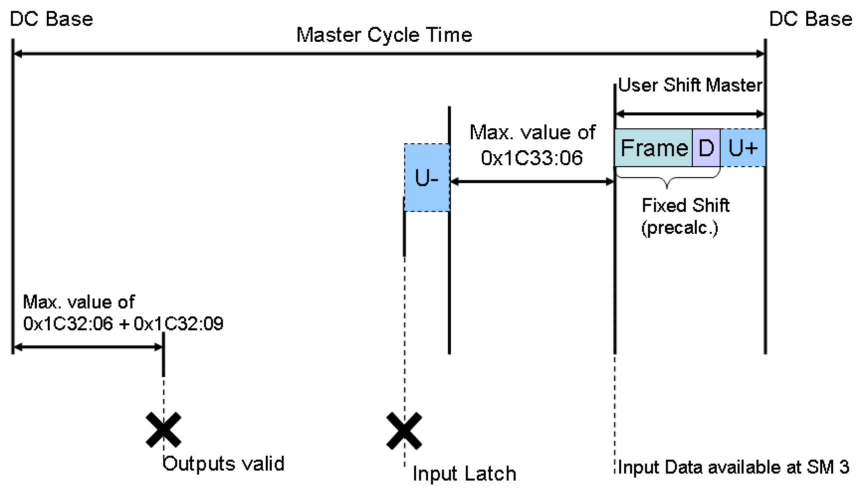


Figure 25: Master Settings for use of DCs

Max value of Calc and Copy Time (0x1C32:06) plus Output Delay Time (0x1C32:09) of all DC slaves shall be used as reference to set the Outputs Valid.

The maximum Shift Time for each slave can be MaxShiftTime = Master Cycle Time - User Shift Master - Calc And Copy Time - U-

Master Settings for use of DCs with slave related Settings

Figure 26 and Table 83 describes how the settings handled by the master are used for two different synchronization cases (Slv 1, Slv 2).

Table 83: Master Settings for use of DCs with slave related settings

Slave	Description
Slv1	Fixed Input Shift Time (0x1C33:03), no exact synchronization possible This device has the maximum value of 0x1C32:03 plus 0x1C32:06 in this example. Its Sync0 Signal comes with the DC Base signal. The Shift Time is fixed and thus, cannot be used to shift the Input Latch to the system Input Latch time. Synchronization of the Input Latch cannot be achieved in this case.
Slv 2	Sync0 and Shift Times (0x1C32:03, 0x1C33:03) used for synchronization in DC Mode The (Calc and Copy (0x1C32:06) + Output Delay Time (0x1C32:09)) of slave 2 is smaller than the Max value of (0x1C32:06 + 0x1C32:09). To synchronize the Outputs Valid of slave 2 with the system's Outputs Valid the Sync0 cycle of slave 2 is set that it is (0x1C32:06 + 0x1C32:09) away. The Input Shift Time measures from the Sync0 to the Input Latch. The local application should calculate an internal Shift Time from Shift Time minus Input Delay Time so that inputs can be read in physically with the system Input Latch.

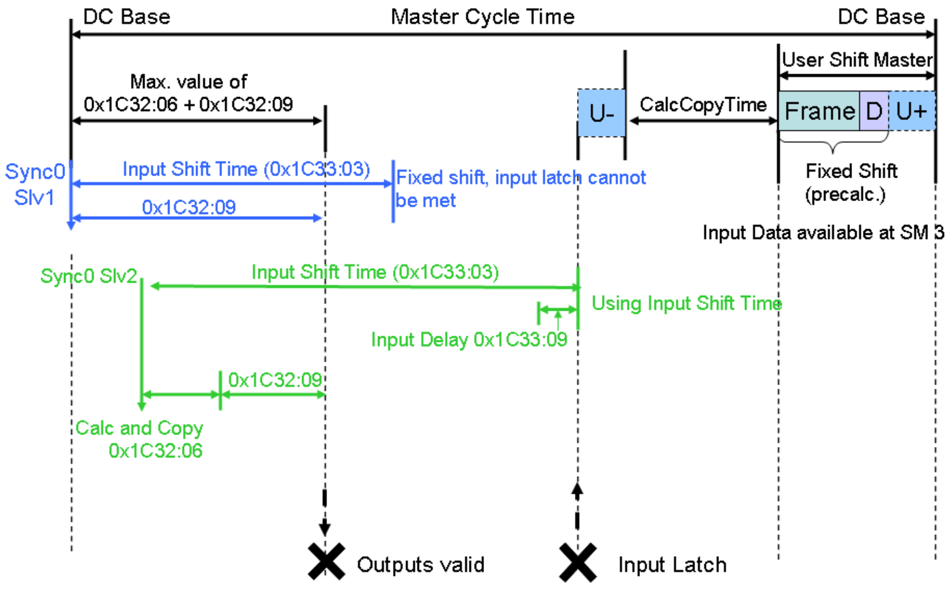


Figure 26: Master Settings

20.2.3 Sync-Unit: DcOffset correction

To prevent that the setting of DcOffset register (0x0920…0x0927) during active SyncOut-Unit (0x0981.0 == 1) will jump over an active Sync-Event the following sequence shall be used:

- deactivate Sync-Unit (0x0981 = 0)
- Set DcOffset (0x0920…0x0927)
- Set StartTimeCycle (0x0990…0x09978) with new value
- activate Sync-Unit (0x0981 = 1)

21 External Synchronization

21.1 External Synchronization of EtherCAT Networks, Reference Clock follows Master Clock

The synchronization of system components or production lines with several EtherCAT Segments can be done with different solutions:

- Synchronization with an external grandmaster clock using IEEE 1588 protocol
- Synchronization by a bridge device.
- Synchronization by Master Clock

Depending on the application the accuracy of the synchronization has different requirements. For the synchronization of two production lines often the accuracy in the range of milliseconds is enough. However if two or more motion control networks should be synchronized it may be necessary to have the Sync-signals exact at the same time in all devices (e.g. for a network wide commutation information) - that means an accuracy better than a microsecond may be needed.

Within one EtherCAT Segment the Distributed Clocks (DC) mechanism can be used, to synchronize the network devices. DC offers accuracy much better than a microsecond. The first device that should be synchronized is used as the reference clock for that segment. The System Time information of this device is distributed cyclically to the other devices to compensate dynamic drift. The static drift and the propagation delay are compensated during startup of the segment.

The synchronization of several EtherCAT Segments with a high accuracy means that the DC reference clocks in the segments must be adjusted. The leading clock is called Grandmaster clock. It can be an external Time information, e.g. GPS or DCF77 receiver (see Figure 27), but it can also be one of the DC reference clocks.

The DC Reference Clocks of the EtherCAT segments need to be adjusted by the EtherCAT Master according to the time information coming from the Grandmaster clock. The EtherCAT Master should cyclically adjust the DC Reference Clock with a unicast write command to the System Time register. The DC slave clocks should be adjusted by a following read-multiple-write command (xRMW).

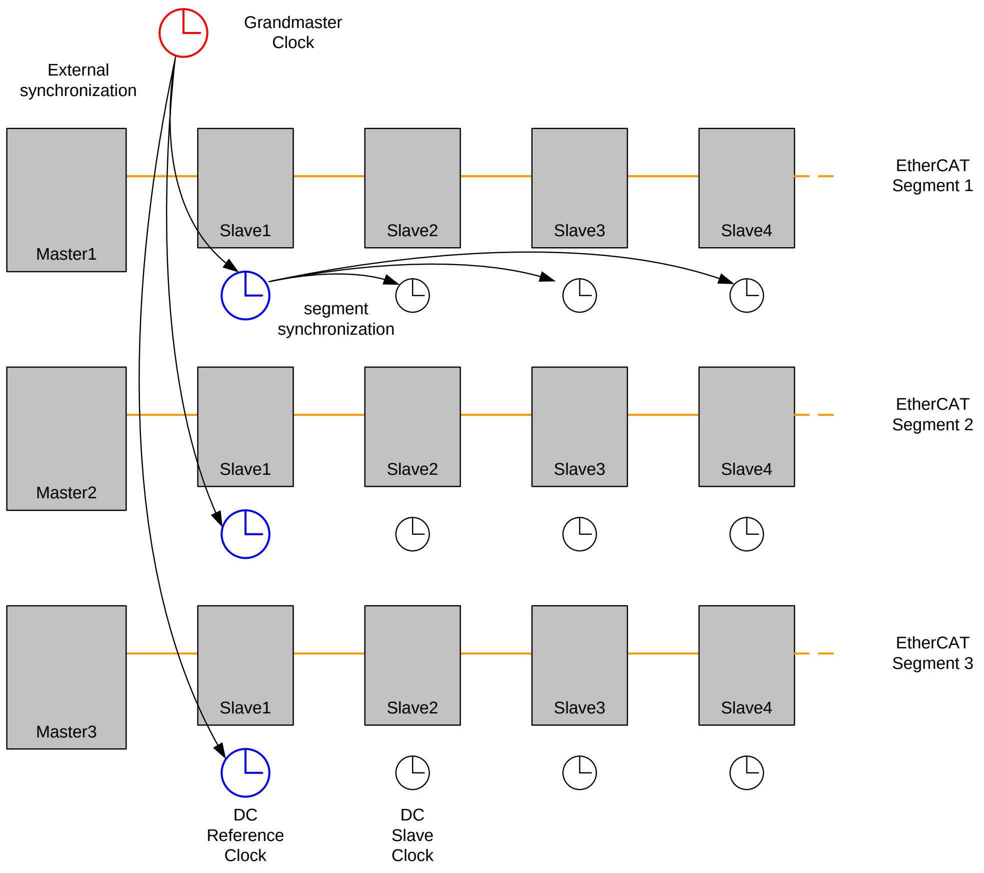


Figure 27: Synchronization of several EtherCAT Segments

21.1.1 Synchronization with an external grandmaster clock using IEEE 1588

The Grandmaster clock is an external clock that supports IEEE 1588 protocol [IEC61588 Precision Clock Synchronization Protocol for Networked Measurement and Control Systems (IEEE Standard)]. This protocol defines mechanisms to synchronize devices in a Standard Ethernet Network.

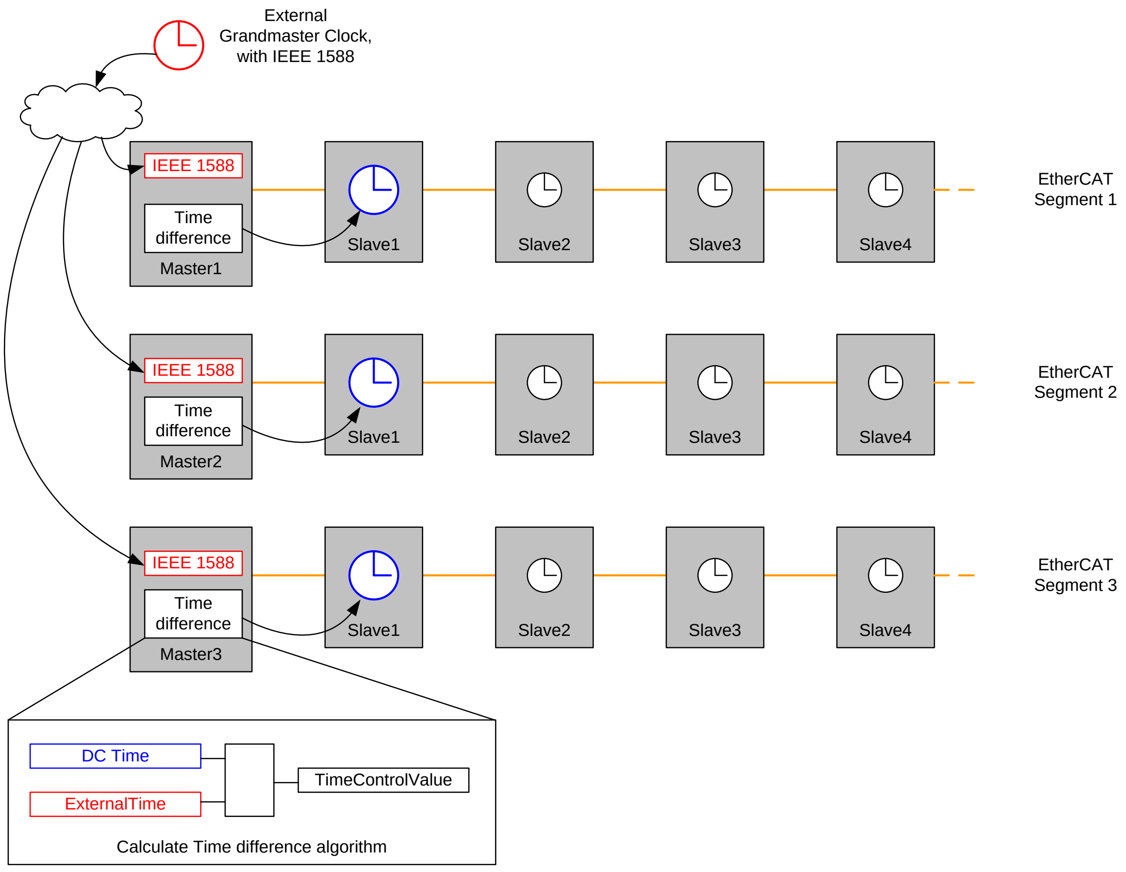


Figure 28: External IEEE 1588 Grandmaster Clock

The control algorithm for the calculation of the time difference between the DC reference clock within an EtherCAT Segment and the IEEE1588 grandmaster clock can be located in the master (Figure 28) or it can be located in one of the slaves (Figure 29) - if the slave offers this algorithm, the master can be kept very simple.

In Figure 28 each Master is connected to the external Grandmaster Clock. The master can calculate the time difference of the 1588-Grandmaster clock to the DC reference clock. The calculated TimeControlValue can be used to adjust the DC Clocks. The accuracy depends on the implementation of the 1588 protocol in the master - when using a standard Ethernet Port in the Master there is no hardware support to capture the timestamps of the incoming telegrams. Therefore the accuracy is not that high.

Figure 29 shows an example were the control algorithm is located in one of the EtherCAT Slaves. This offers the advantage that a special hardware support can be used in the slave and the master can be kept simple. The 1588 telegrams are latched at a legacy Ethernet port with a high precise timestamp in the slave. This external Timestamp can be compared to the high precise DC System Time (1). The result is a TimeControlValue that holds the information if the DC Time runs to fast or to slow. This TimeControlValue can send via PDO to the Master (2) that can adjust the DC Reference Clock in the segment (3).

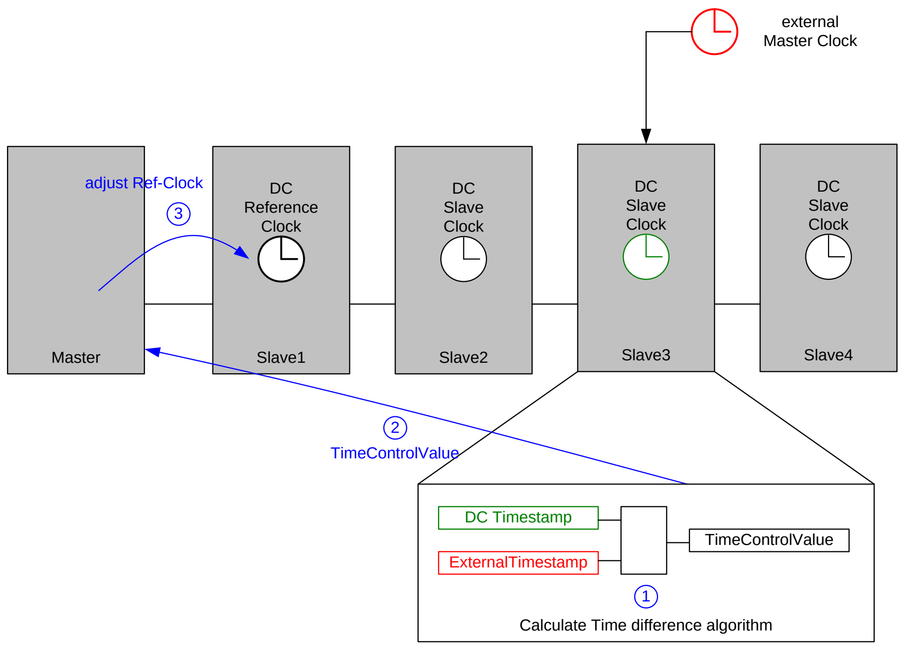


Figure 29: Control Algorithm for Time difference loaded in an EtherCAT Slave

During startup the absolute time information of the Grandmaster Clock must be adjusted to the absolute time of the DC Reference Clock: First the EtherCAT Segment is started and boot-up and the DC Time is set according to the Time information of the master. The Time difference of this segment time to the 1588 Grandmaster time can be very huge so that the adjustment with the DC control algorithm is not sufficient. The setting of the system time offset register would cause a discontinuity in the slaves. Thus an additional system time offset is used within the slave device.

21.1.2 Synchronization by a Bridge device

The synchronization of two or more EtherCAT segments can be done by a Bridge device as shown in Figure 30.

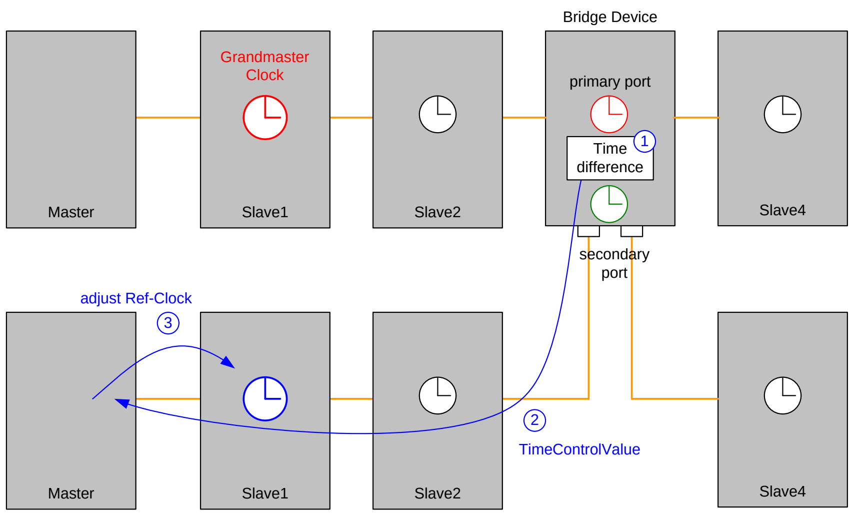


Figure 30: Synchronization of EtherCAT Segments using a Bridge device

The Bridge has two EtherCAT connections (internal two EtherCAT Slave Controllers are used). The primary port is connected to the first segment; the secondary port is connected to the second segment.

One of the two ports can be configured to be the leading time reference (in Figure 30 this is for example the primary port). The Bridge can calculate the time difference and offers the TimeControlValue to the Master. The Master can then adjust the Reference Clock.

During startup the two segments can be powered-on at different times. That means that there will be an absolute time difference between the two segments. This difference has to be taken into account with an additional system time offset as part of the calculation of the time difference algorithm.

21.2 External Synchronization of an EtherCAT Slave

This clause describes the unique handling for an external synchronization in an EtherCAT slave device.

21.2.1 External Synchronization Status

The External Synchronization status 0x10F4 shown in Table 84 and Table 85 contains the status to synchronize an EtherCAT segment with an external network.

Table 84: 0x10F4 External Synchronization Status

Attribute	Value
Index	0x10F4
Name	External synchronization status
Object Code	RECORD
Max SubIndex	19

Table 85: Entries of 0x10F4 External synchronization Status

SubIndex	Description	Data Type	Access	PDO Mapping	Description / Default value
1	Sync Mode	BIT2	R	TxPDO	0: no synchronization, 1: device operates as SYNC master, 2: device operates as SYNC slave
3		0, Bit-Length = 12	R	no	Align
14	Control Value Update Toggle	BOOLEAN	R	TxPDO	toggles every time when the control value was updated
15	Time Stamp Update Toggle	BOOLEAN	R	TxPDO	toggles every time when the time stamps were updated
16	External device not connected	BOOLEAN	R	TxPDO	TRUE: no external synchronization found
17	Internal Time Stamp	UNSIGNED64	R	TxPDO	only for SYNC slave. DC time stamp at the same time as the external time stamp
18	External Time Stamp	UNSIGNED64	R	TxPDO	only for SYNC slave. external time stamp recalculated in DC units (ns)
19	Time Control Value	INT32	R	TxPDO	only for SYNC slave. > 0: this value gives the information how often the System Time register of the DC Master Clock has to be written with a higher value than the actual value of the system Time register for the dynamic drift compensation < 0: this value gives the information how often the System Time register of the DC Master Clock has to be written with a lower value than the actual value of the system Time register for the dynamic drift compensation

21.2.2 External Synchronization Settings

The External Synchronization status 0x10F5 shown in Table 86 and Table 87 contains the settings to synchronize an EtherCAT segment with an external network.

Table 86: 0x10F5 External Synchronization Settings

Attribute	Value
Index	0x10F5
Name	External synchronization settings
Object Code	RECORD
Max SubIndex	18

Table 87: Entries of 0x10F5 External Synchronization Settings

SubIndex	Description	Data Type	Access	PDO Mapping	Description / Default value
1	Sync Master	BOOLEAN	RW	no	TRUE: device acts as SYNC master, FALSE: device acts as SYNC slave
2	32Bit Time Stamps	BOOLEAN	RW	no	TRUE: only 32 bit time stamps are used, FALSE: 64 bit time stamps are used
3		0, Bit-Length=14		no	align
17	Time Control Interval	UNSIGNED16	RW	no	only for SYNC slave: this value defines how often the time stamps will be updated in ms
18	System Time Additional Offset	UNSIGNED64	rw	no	only for SYNC slave: additional offset to the System Time (needed if the external synchronization was disconnected and the System Time differs too much because the System Time Offset of the ESC cannot be changed if the slave is not in INIT)

21.2.3 TxPDO Mapping

Depending on the External synchronization settings the TxPDO differs.

21.2.3.1 SYNC Slave, 64 Bit Time Stamp

Table 88 shows the TxPDO mapping for a SYNC slave with 64 bit time stamp.

Table 88: TxPDO Mapping for SYNC Slave, 64 bit Time Stamp

SubIndex	Description	Data Type	Value
1	Sync Mode	UNSIGNED32	0x10F40102
2	Align	UNSIGNED32	0x0000000C
3	Control Value Update Toggle	UNSIGNED32	0x10F40E01
4	Time Stamp Update Toggle	UNSIGNED32	0x10F40F01
5	External device not connected	UNSIGNED32	0x10F41001
6	Internal Time Stamp	UNSIGNED32	0x10F41140
7	External Time Stamp	UNSIGNED32	0x10F41240
8	Time Control Value	UNSIGNED32	0x10F41310

21.2.3.2 SYNC Slave, 32 Bit Time stamp

Table 89 shows the TxPDO mapping for a SYNC slave with 32 bit time stamp.

Table 89: TxPDO Mapping for SYNC Slave, 32 bit Time Stamp

SubIndex	Description	Data Type	Value
1	Sync Mode	UNSIGNED32	0x10F40102
2	Align	UNSIGNED32	0x0000000E
3	Control Value Update Toggle	UNSIGNED32	0x10F40E01
4	Time Stamp Update Toggle	UNSIGNED32	0x10F40F01
5	Time Stamp invalid	UNSIGNED32	0x10F41001
6	Internal Time Stamp	UNSIGNED32	0x10F41120
7	External Time Stamp	UNSIGNED32	0x10F41220
8	Time Control Value	UNSIGNED32	0x10F41310

21.2.3.3 SYNC Slave, only Status

Table 90 shows the TxPDO mapping for a SYNC slave with only status.

Table 90: TxPDO Mapping for SYNC Slave, Status Only

SubIndex	Description	Data Type	Value
1	Sync Mode	UNSIGNED32	0x10F40102
2	Align	UNSIGNED32	0x0000000E
3	Control Value Update Toggle	UNSIGNED32	0x10F40E01
4	Time Stamp Update Toggle	UNSIGNED32	0x10F40F01
5	Time Stamp invalid	UNSIGNED32	0x10F41001

21.2.3.4 SYNC Master

Table 91 shows the TxPDO mapping for a SYNC Master.

Table 91: TxPDO Mapping for SYNC Master

SubIndex	Description	Data Type	Value
1	Sync Mode	UNSIGNED32	0x10F40102
2	Align	UNSIGNED32	0x0000000E
3	External device not connected	UNSIGNED32	0x10F41001

21.2.4 Master's Control Loop

The master has to adapt the System Time of the DC master clock by writing the System Time register with a higher or lower value than the actual value. This has to be done to start the dynamic drift compensation on the EtherCAT Slave Controller.

There are two possibilities for the master's control loop to calculate how often the system Time register of the DC master clock has to be written to compensate the dynamic drift:

1. Compare the Internal and External Time Stamp which are latched at the same point of time and calculate the number of writing by the master's own algorithm.
2. Use the Time Control Value generated by the device's algorithm.

NOTE 1: With each DC start-up procedure the Control loop Parameter 1 (Address 0x930...0x931) and Control Loop Parameter 3 (Address 0x934...0x935) of all DC capable nodes should be set to following values.
0x0934=4
0x0935=12
0x0930=0x1000 (shall be the last command to reset the DC control unit)

NOTE2: When synchronizing the EtherCAT Network by adjusting the DC reference Clock the following DC Control Loop Parameter in the reference Clock can be used to minimize synchronization jitter:
0x0935: Speed Counter Filter = 0
0x930=0x1000 (shall be the last command to reset the DC control unit)

22 ESM behaviour

The master should transmit process data in state transition PREOP-SAFEOP.

The master shall transmit process data as soon as possible after slaves have confirmed SAFEOP state.

Note: The master shall not wait until slaves have started their Input Update because in some operation modes the slaves may require process data transmission from master to start Input Update.

The slave should transmit process data in state transition PREOP-SAFEOP.

The slave shall update input data as soon as possible after entering state SAFEOP. The slave shall not go to OP state until the inputs are updated in SAFEOP state.

23 Base Data Types

Base Data Types are defined in ETG.1000.6, [5]. This chapter defines additional data types and enhanced descriptions of the existing data types.

Table 92 describes the Base Data Types.

Table 92: Base Data Types

Index	Name	Base Data Type	Bit Size	Description	Range	GUID
0x0001	BOOLEAN	BOOL BIT	1	‘0’: FALSE ‘1’: TRUE		{18071995-0000-0000-0000-000000000030}
0x001E	BYTE	BYTE	8	Unsigned Byte, Octet		{18071995-0000-0000-0000-000000000001}
0x001F	WORD	WORD	16	Two Octet		{18071995-0000-0000-0000-000000000004}
0x0020	DWORD	DWORD	32	Four Octet		{18071995-0000-0000-0000-000000000007}
		Bit String				
0x0030	BIT1	BIT1	1			{18071995-0000-0000-0000-000000000010}
0x0031	BIT2	BIT2	2			{18071995-0000-0000-0000-000000000011}
0x0032	BIT3	BIT3	3			{18071995-0000-0000-0000-000000000012}
0x0033	BIT4	BIT4	4			{18071995-0000-0000-0000-000000000013}
0x0034	BIT5	BIT5	5			{18071995-0000-0000-0000-000000000014}
0x0035	BIT6	BIT6	6			{18071995-0000-0000-0000-000000000015}
0x0036	BIT7	BIT7	7			{18071995-0000-0000-0000-000000000016}
0x0037	BIT8	BIT8	8			{18071995-0000-0000-0000-000000000017}
0x002D	BITARR8	BITARR8	8	8 individual Bits		{18071995-0000-0000-0000-000000000032}
0x002E	BITARR16	BITARR16	16	16 individual Bits		{18071995-0000-0000-0000-000000000033}
0x002F	BITARR32	BITARR32	32	32 individual Bits		{18071995-0000-0000-0000-000000000034}
		Signed Integer				
0x0002	INTEGER8	SINT	8	Short Integer	-128 to 127	{18071995-0000-0000-0000-000000000003}
0x0003	INTEGER16	INT	16	Integer	-32 768 to 32 767	{18071995-0000-0000-0000-000000000006}

Index	Name	Base Data Type	Bit Size	Description	Range	GUID
0x0010	INTEGER24	INT24	24		-2^{23} to 2^{23-1}	{18071995-0000-0000-0000-000000000023}
0x0004	INTEGER32	DINT	32	Double Integer	-2^{31} to 2^{31-1}	{18071995-0000-0000-0000-000000000009}
0x0012	INTEGER40	INT40	40			{18071995-0000-0000-0000-000000000027}
0x0013	INTEGER48	INT48	48			{18071995-0000-0000-0000-000000000029}
0x0014	INTEGER56	INT56	56			{18071995-0000-0000-0000-00000000002B}
0x0015	INTEGER64	LINT	64	Long Integer	-2^{63} to 2^{63-1}	{18071995-0000-0000-0000-00000000000C}
		Unsigned Integer				
0x0005	UNSIGNED8	USINT	8	Unsigned Short Integer	0 to 255	{18071995-0000-0000-0000-000000000002}
0x0006	UNSIGNED16	UINT	16	Unsigned Integer / Word	0 to 65 535	{18071995-0000-0000-0000-000000000005}
0x0016	UNSIGNED24	UINT24	24			{18071995-0000-0000-0000-000000000022}
0x0007	UNSIGNED32	UDINT	32	Unsigned Double Integer	0 to 2^{32-1}	{18071995-0000-0000-0000-000000000008}
0x0018	UNSIGNED40	UINT40	40			{18071995-0000-0000-0000-000000000026}
0x0019	UNSIGNED48	UINT48	48			{18071995-0000-0000-0000-000000000028}
0x001A	UNSIGNED56	UINT56	56			{18071995-0000-0000-0000-00000000002A}
0x001B	UNSIGNED64	ULINT	64	Unsigned Long Integer	0 to 2^{64-1}	{18071995-0000-0000-0000-00000000000B}
		Floating				
0x0008	REAL32	REAL	32	Floating point		{18071995-0000-0000-0000-00000000000D}
0x0011	REAL64	LREAL	64	Long float		{18071995-0000-0000-0000-00000000000E}
		GUID				
0x001D	GUID	GUID	128	Globally unique identifier with a length of 128 Bit		{18071995-0000-0000-0000-000000000021}

Table 93 describes the Base Data Types with variable length.

Table 93: Base Data Types with variable length

Index	Name	Base Data Type	Bit Size	Description	Range	GUID
		Strings				
0x0009	VISIBLE_STRING	STRING(n)	8*n	VisibleString (1 octet per character)		{18071995-0000-0000-0000-00010000xxxx} with xxxx equals n (in HEX)
		Octet field				
0x000A	OCTET_STRING	ARRAY [0..n] OF BYTE	8*(n+1)	Sequence of octets (data type BYTE)		{18071995-0000-0000-0000-00030000xxxx} with xxxx equals n+1 (in HEX)
0x000B	UNICODE_STRING	ARRAY [0..n] OF UINT	16*(n+1)	Sequence of UINT		{18071995-0000-0000-0000-000B0000xxxx} with xxxx equals n+1 (in HEX)
0x0260	ARRAY_OF_INT	ARRAY [0..n] OF INT	16*(n+1)	Sequence of INT		{18071995-0000-0000-0000-000A0000xxxx} with xxxx equals n+1 (in HEX)
0x0261	ARRAY_OF_SINT	ARRAY [0..n] OF SINT	8*(n+1)	Sequence of SINT		{18071995-0000-0000-0000-00080000xxxx} with xxxx equals n+1 (in HEX)
0x0262	ARRAY_OF_DINT	ARRAY [0..n] OF DINT	32*(n+1)	Sequence of DINT		{18071995-0000-0000-0000-000C0000xxxx} with xxxx equals n+1 (in HEX)
0x0263	ARRAY_OF_UDINT	ARRAY [0..n] OF UDINT	32*(n+1)	Sequence of UDINT		{18071995-0000-0000-0000-000D0000xxxx} with xxxx equals n+1 (in HEX)

Table 94 describes pre-defined records.

Table 94: Pre-defined records

Index	Name	Base Data Type	Bit Size	Description	Range	GUID
0x0021	PDO_MAPPING	--		For PDO_MAPPING definition see ETG.1000		{7FF678B5-BA48-42E6-9A0A-CBFCBECC8412}
0x0022	*Reserved*			*Reserved for compatibility reasons*		
0x0023	IDENTITY	--		For IDENTITY definition see ETG.1000		{7E3EE6E2-BBAB-4468-9F5A-68E5EBBF4412}
0x0024	*Reserved*			*Reserved for compatibility reasons*		

Index	Name	Base Data Type	Bit Size	Description	Range	GUID
0x0025	COMMAND_PAR	--		For COMMAND_PAR definition see ETG.1000		{0F324177-734C-4D75-89B0-468BB8E8F222}
0x0026	*Reserved*			*Reserved for compatibility reasons*		
0x0027	PDO_PARAMETER	--		For PDO_PARAMETER definition see ETG.1020		
0x0028	ENUM	--		For DEFTYPE_ENUM definition see ETG.1020		{18071995-0000-0000-0000-00000000001e}
0x0029	SM_SYNCHRONISATION	--		For SYNC_PAR definition see ETG.1000		{36BEED8B-6733-4853-A49C-3E5EA40DF14E}
0x002A	RECORD			No pre-defined Record structure. Can be used as a general data type for Records.		{FB01EF56-963F-48E7-9B0A-09691C15038B}
0x002B	BACKUP_PARAMETER	--		For BACKUP_PARAMETER definition see ETG.1020, Object 0x10F0		{D354C782-6ECB-492E-BAAD-96BA69834B4E}
0x002C	MODULAR_DEVICE_PROFILE	--		For MODULAR_DEVICE_PROFILE definition see ETG.5001, Object 0xF000		{BE8233A1-BFBD-4A98-9220-B58C21910B27}
0x0281	ERROR_SETTING	--		For DEFTYPE_ERRORHANDLING definition see ETG.1020, Object 0x10F1		{A7AE60A4-C53D-4802-B5AB-E3A1BD83A6F1}
0x0282	DIAGNOSIS_HISTORY	--		For DEFTYPE_DIAGHISTORY definition see ETG.1020, Object 0x10F3		{DF7DD567-FF9E-4CAB-BD30-0DA54F9898D0}
0x0283	EXTERNAL_SYNC_STATUS	--		For DEFTYPE_SYNCSTATUS definition see ETG.1020, Object 0x10F4		{1265D0D5-7626-4469-A2F4-67993EB72708}
0x0284	EXTERNAL_SYNC_SETTINGS	--		For DEFTYPE_SYNCSETTINGS definition see ETG.1020, Object 0x10F5		{9E0628D3-D369-41C9-AA18-0556FCA2D9D6}
0x0285	DEFTYPE_FSOEFRAME	--		For DEFTYPE_FSOEFRAME definition see ETG.5120		{CD5B2900-7A5E-41DC-972A-85B642FDCCA1}
0x0286	DEFTYPE_FSOECOMMPAR	--		For DEFTYPE_FSOECOMMPAR definition see ETG.5120		{2DB92E24-DA89-482F-85E4-3AFB6B534F3D}

Index	Name	Base Data Type	Bit Size	Description	Range	GUID
0x0287 – 0x07FF	*Reserved*			*Reserved for future use*		