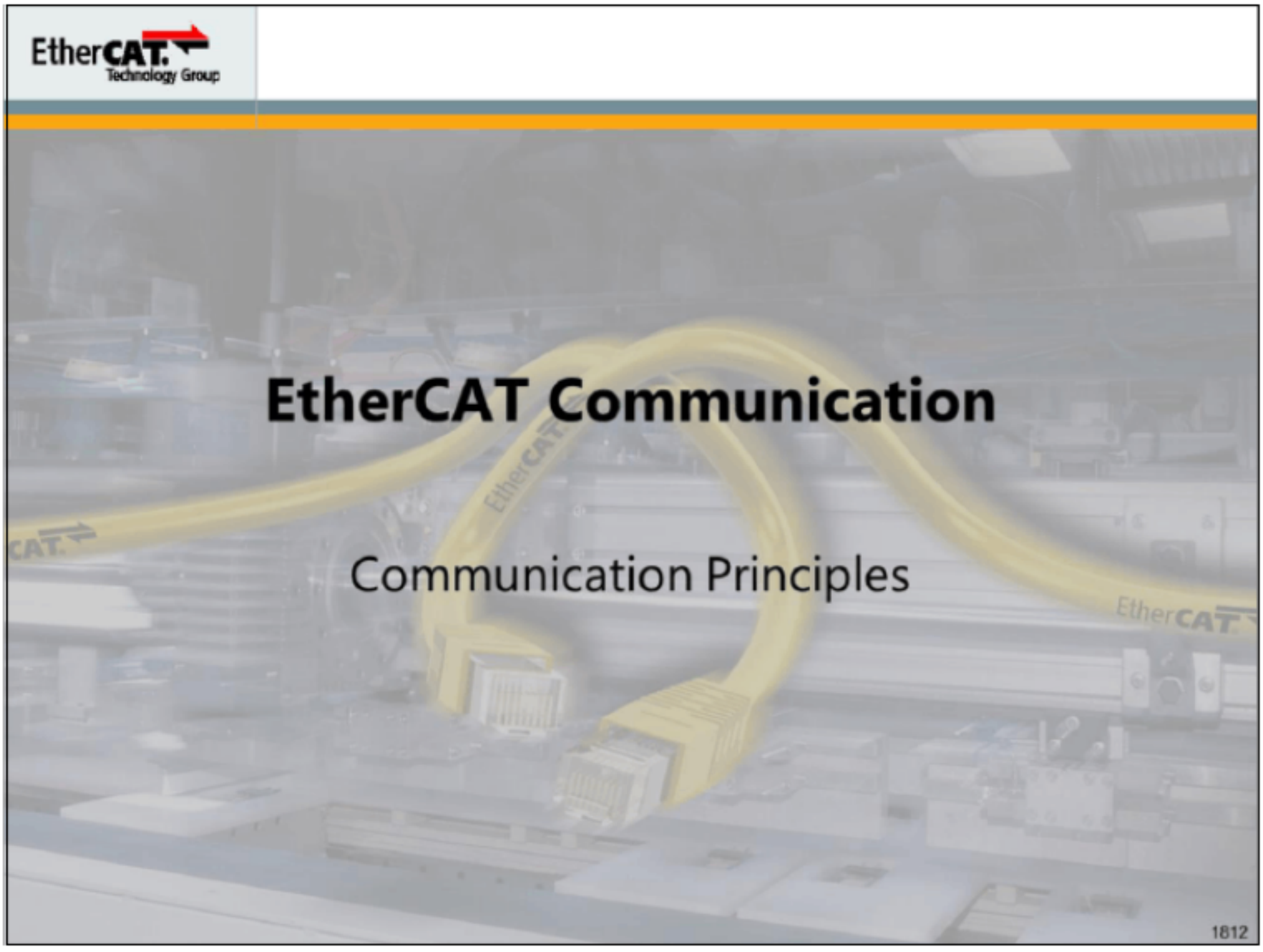

EtherCAT Technology Group
EtherCAT Communication
Communication Principles
1812

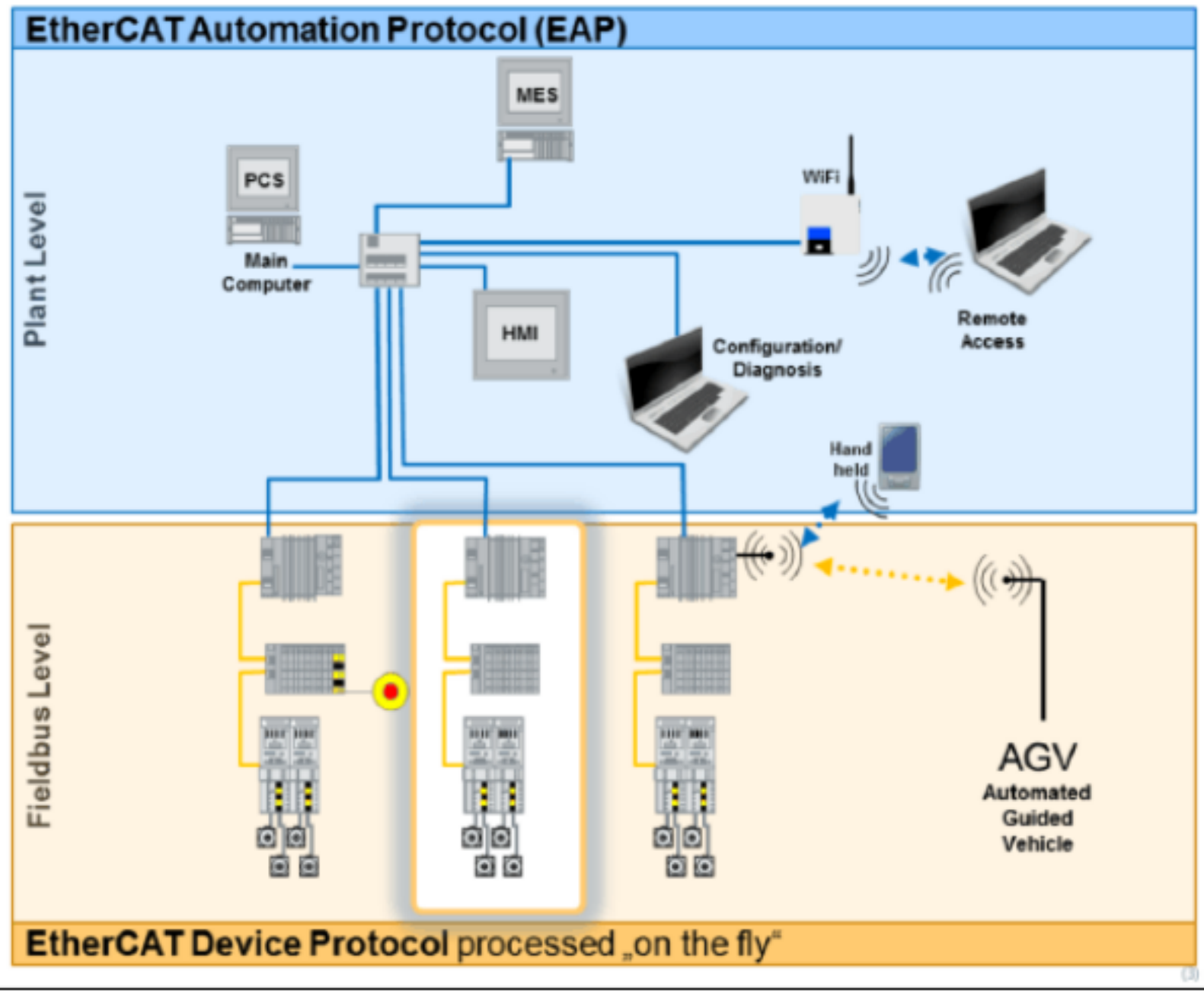
Agenda

- Introduction
- ESI: Device Description
- Configuration Tool
- EtherCAT Master
- Physical Layer
- Data Link Layer
  - EtherCAT Slave Controller
  - Distributed Clocks
- Application Layer
  - State Machine
  - Mailbox Protocols
  - Process Data
- Device Profiles
- Synchronization Modes
- SII: Slave Information Interface
- Diagnosis
- Standards & Implementation

© EtherCAT Technology Group

What is EtherCAT?

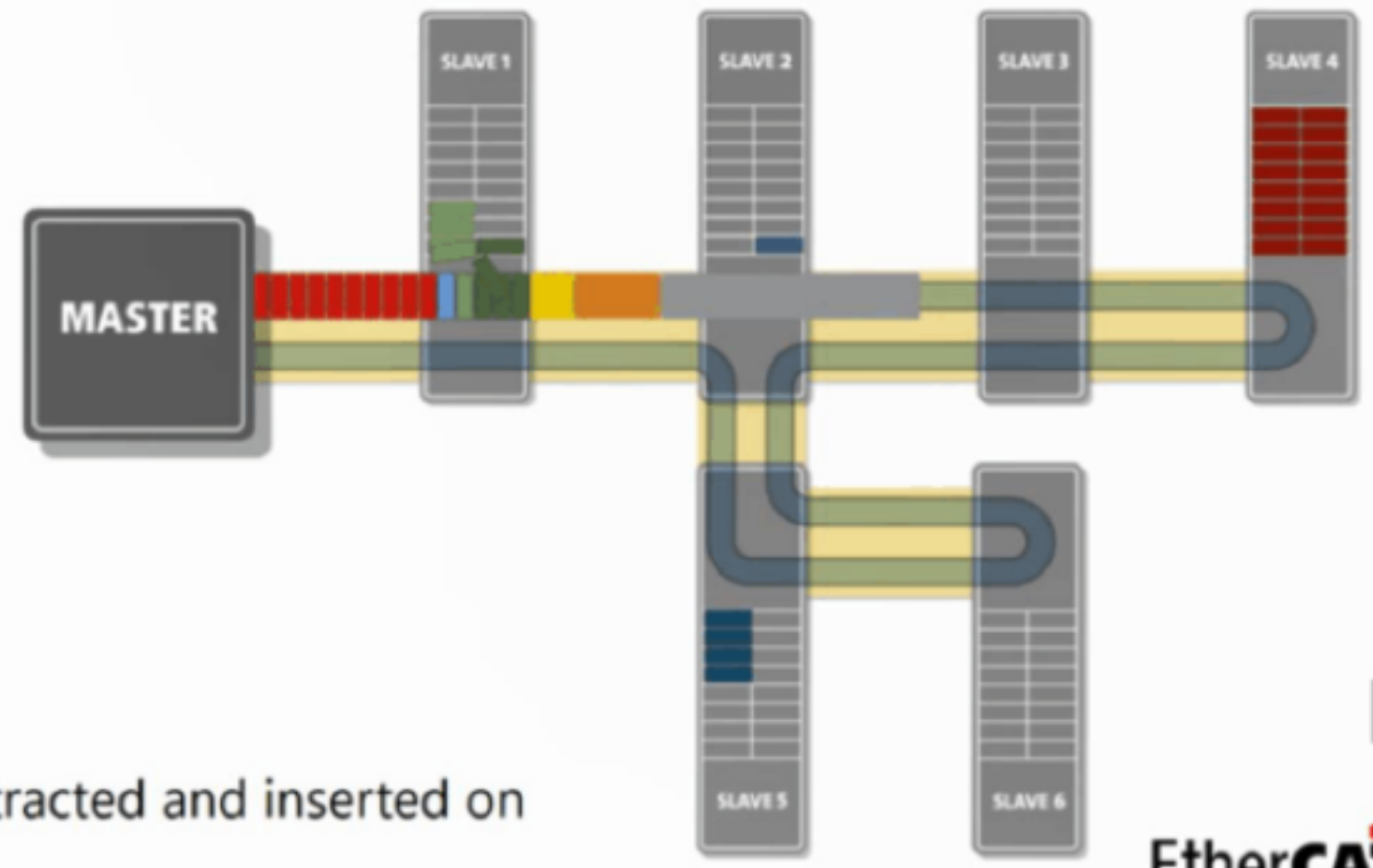
© EtherCAT Technology Group



EtherCAT Device Protocol (EtherCAT fieldbus): master-slave communication between real-time controllers and field devices (I/Os, servodrives, sensors, actuators, ⋯).

EtherCAT Automation Protocol (EAP): communication between controllers, or between controllers and MES/ERP systems, on standard factory networks.

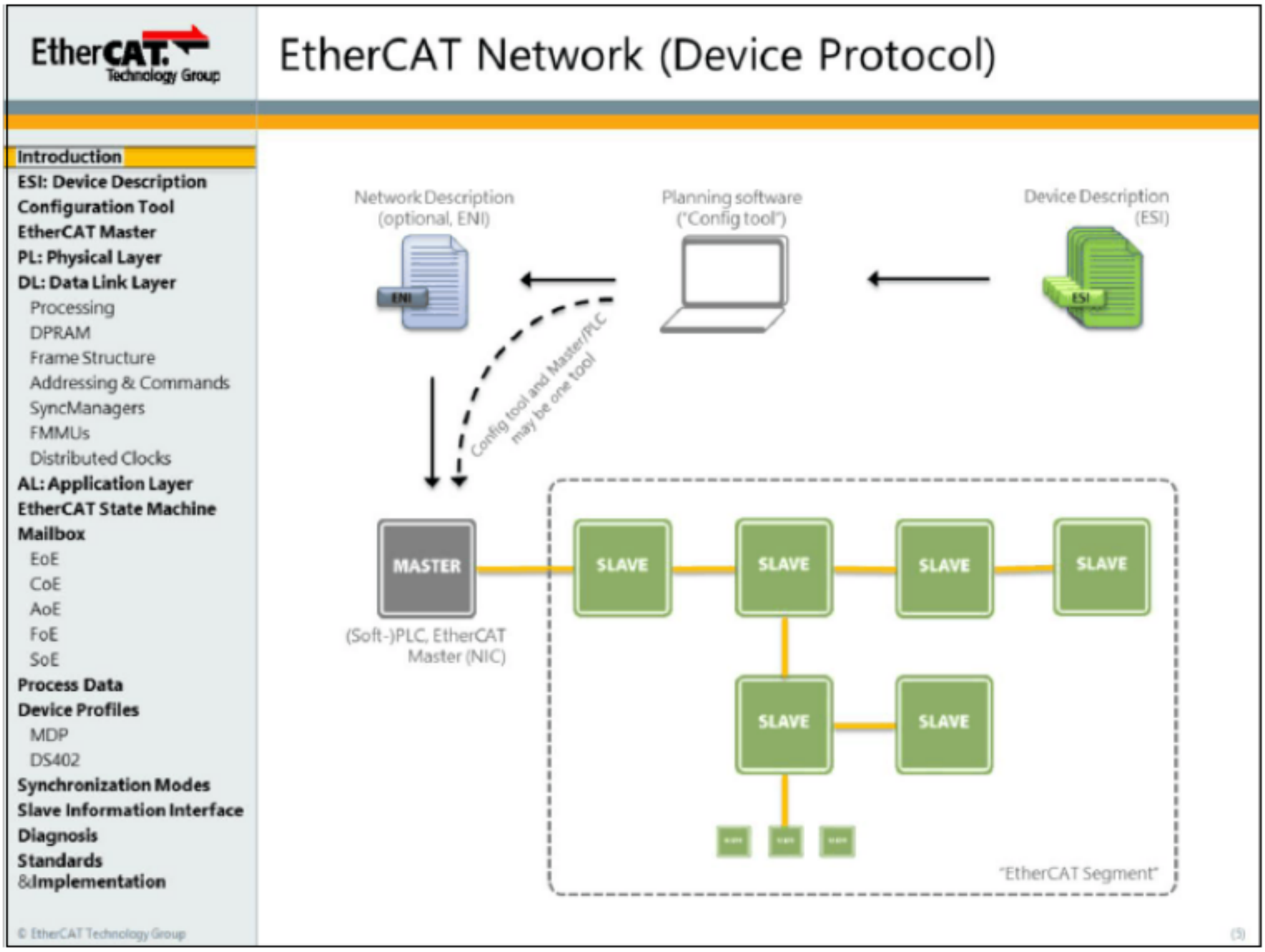
Functional Principle: Ethernet “on the fly”



- Data extracted and inserted on the fly
- Cyclic and acyclic communication

© EtherCAT Technology Group

Cyclic data size can range from 1 Bit to 60 kByte (by using several frames if needed)



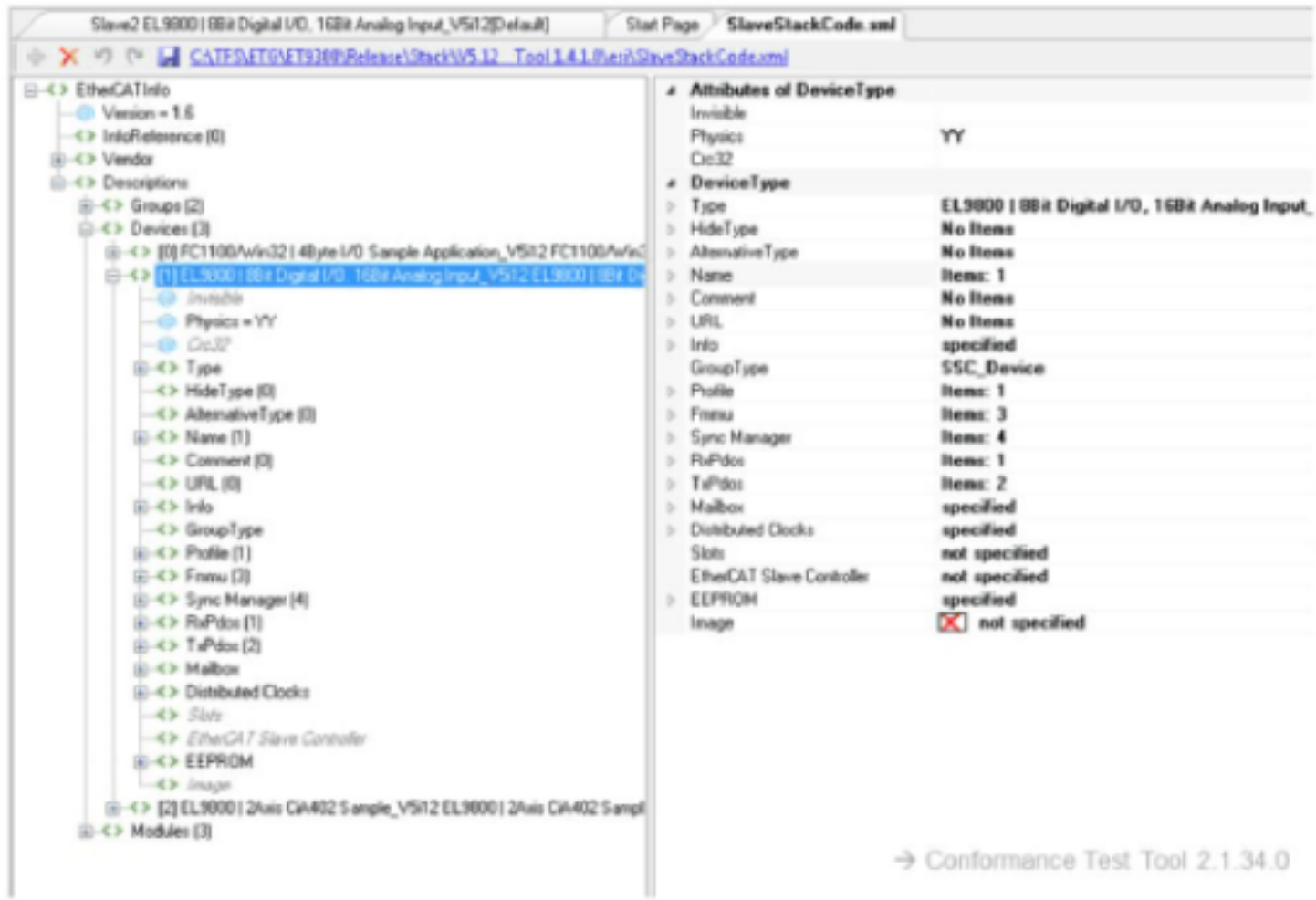
ESI: EtherCAT Slave Information

ENI: EtherCAT Network Information

EtherCAT Slave Information (ESI)

ESI file contains information about:

- Vendor
- Device (several devices can be described in one file)



Reference: ETG.2000 ,EtherCAT Slave Information “

ESI file can be edited in Conformance Test Tool, as well as with any XML editor.

EtherCAT Configuration Tool

- Reads Device Descriptions (ESI)
- Enables the configuration of the network
  - Topology, parameters, process data, ...
- Can be a standalone tool or part of a software suite
- Export network description (ENI)

© EtherCAT Technology Group

Reference: ETG.2100 “ EtherCAT Network Information ”

Configuration Tool: EtherCAT Knowledge Base

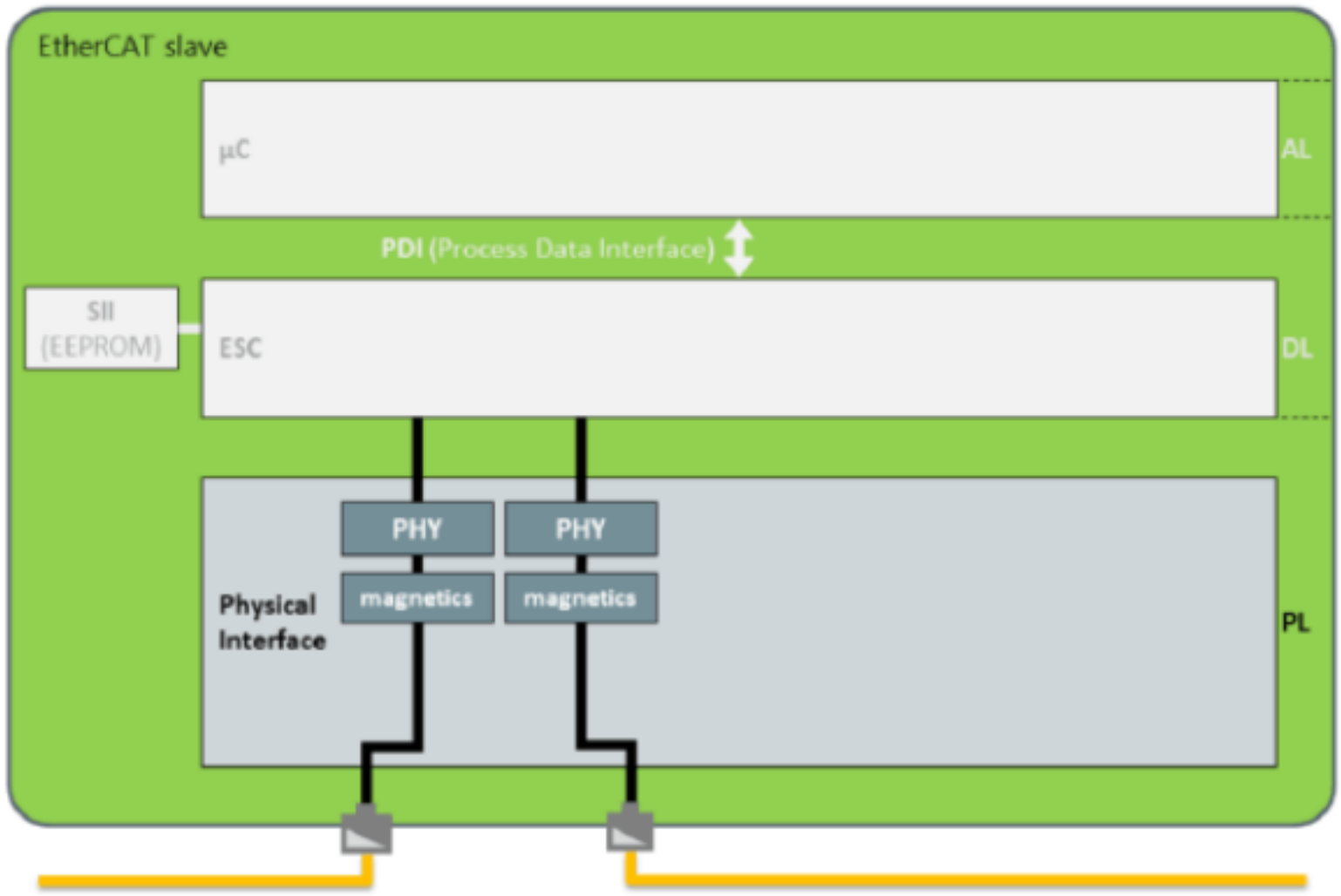
EtherCAT Master

- Reads ENI file or uses own Config Tool
- Handles network interface (NIC)
- Operates the network
  - Send cyclic and acyclic telegrams
  - State Machine
- Provides interface to PLC

© EtherCAT Technology Group

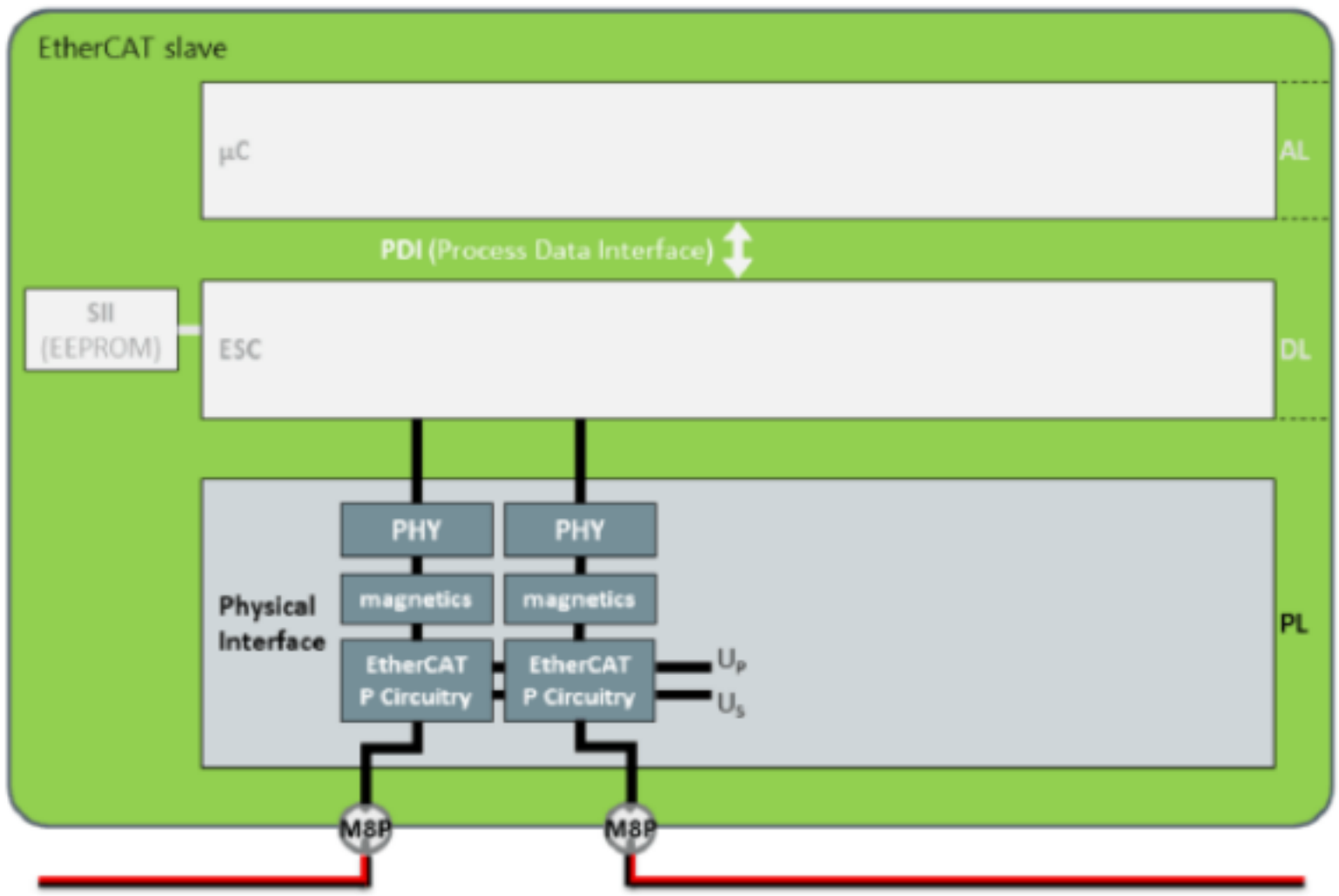
NIC : Network Interface Controller

PL | EtherCAT



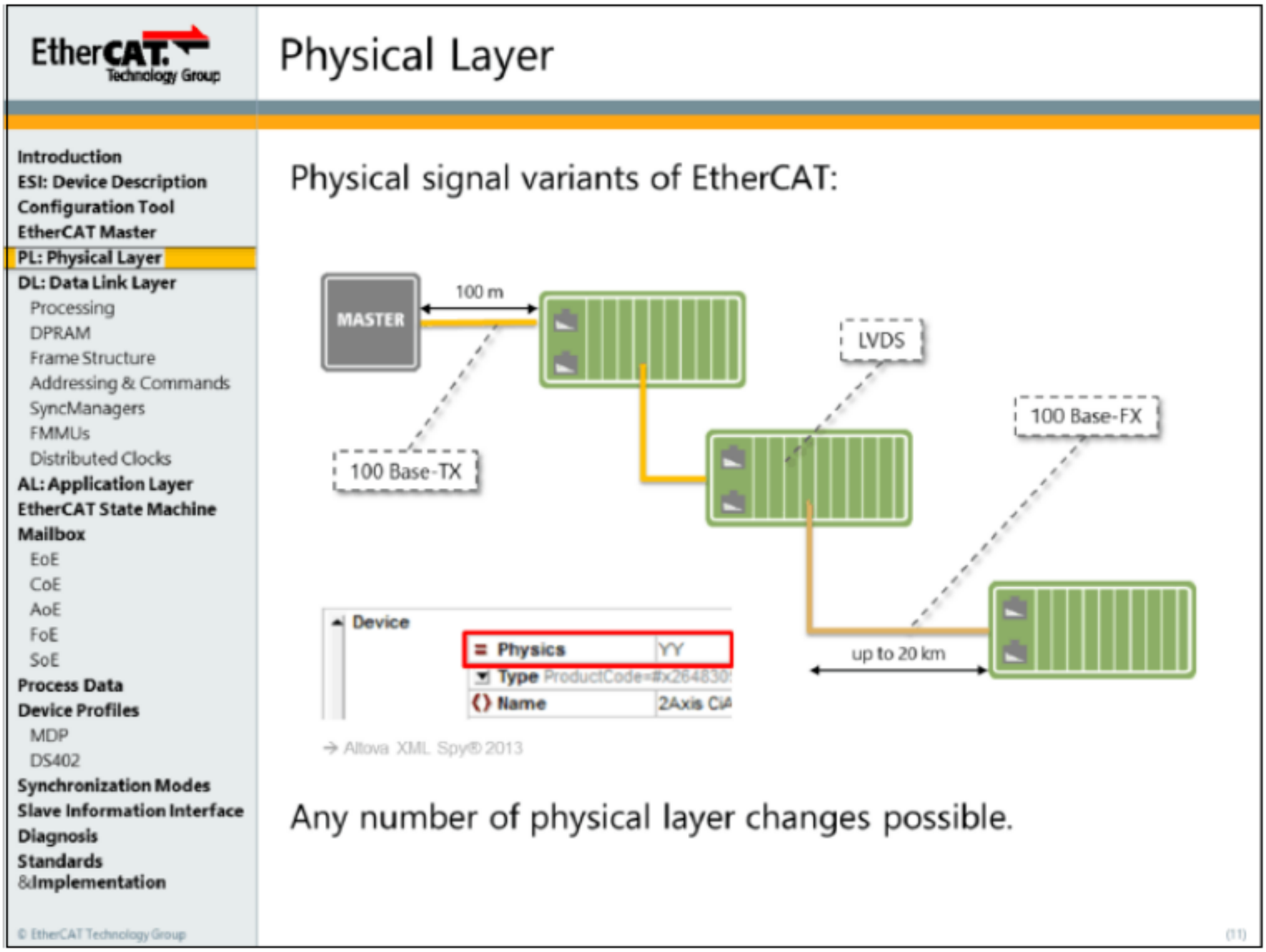
© EtherCAT Technology Group

PL | EtherCAT P



© EtherCAT Technology Group

Further information: https://www.ethercat.org/p



100 BASE-TX (copper cable up to 100 m between 2 nodes)

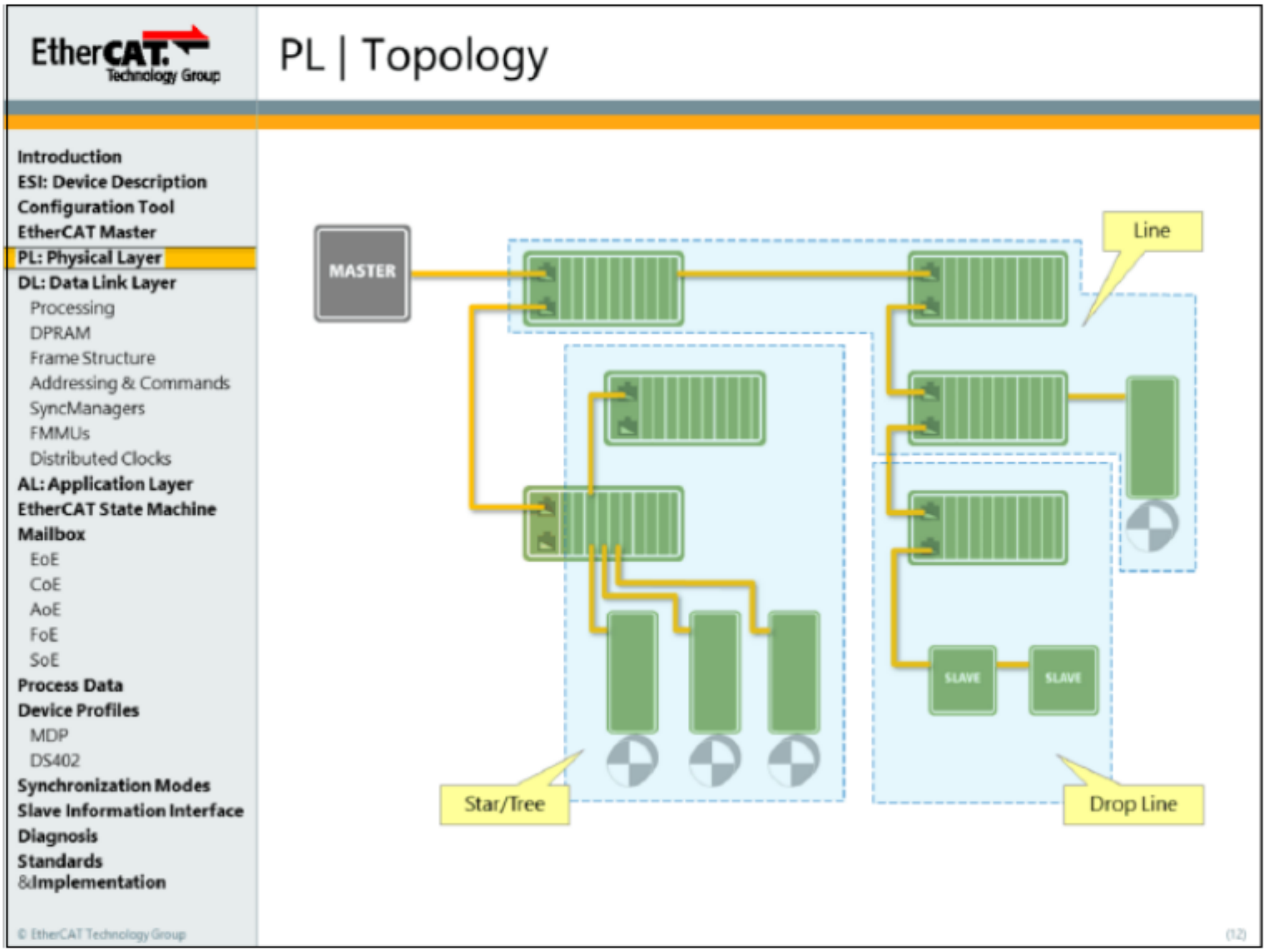
- ? Most popular physical layer for Fast Ethernet
- ? Shielded twisted pair (STP) with 2 pairs of wires
- ? Cable categories CAT5, 6, 7 can be used
- ? RJ45 connector standard, M12 connector for IP67

100 BASE-FX (fiber optic up to 20 km between 2 nodes)

- ? All media options possible
- ? Requirements for TX-to-FX converter, e.g.
  - Link Lost Forwarding
  - No store-and-forward

LVDS (backplane connection for modular devices)

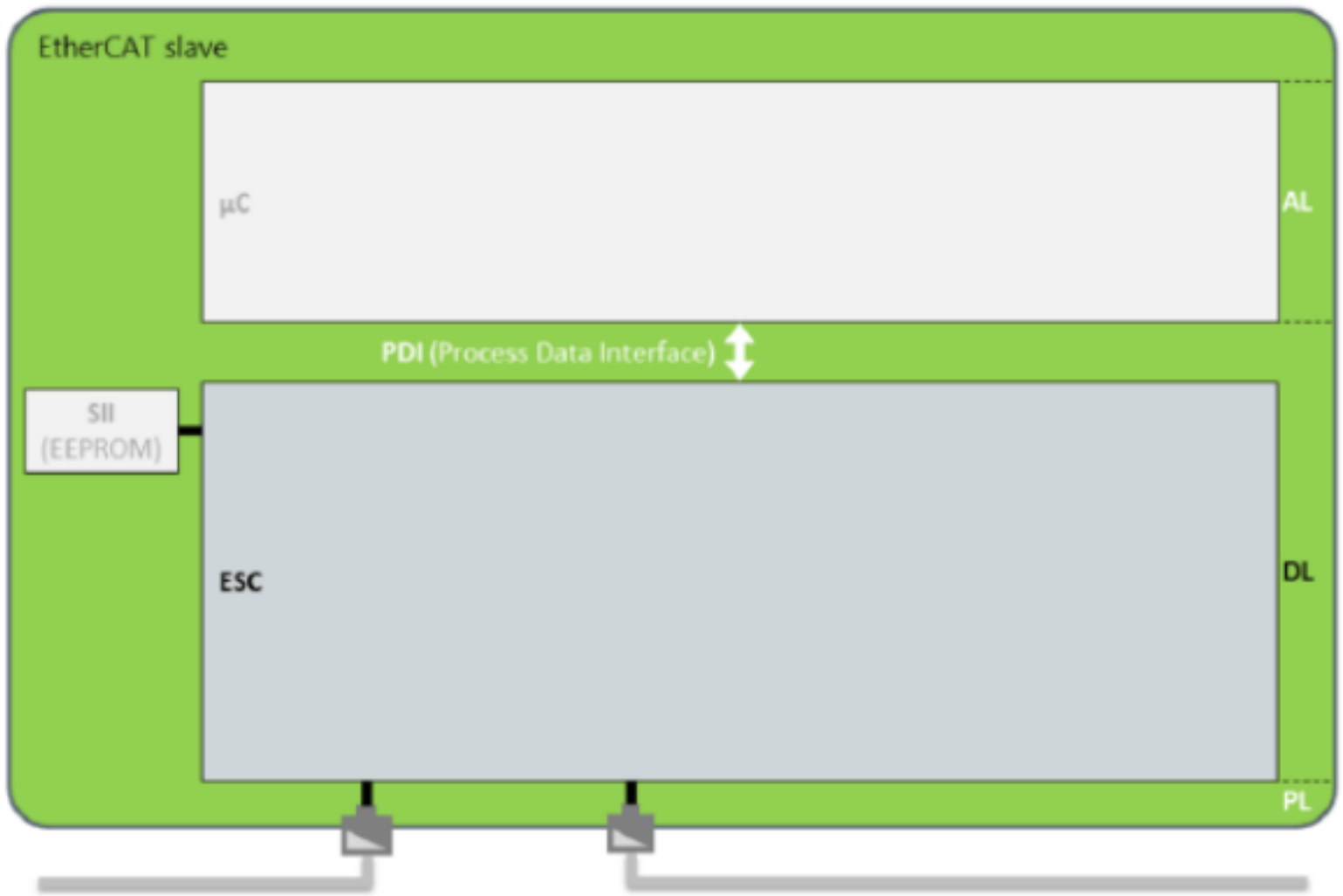
- ? Interface for low cost backplane applications
- ? Not intended for connections over cable
- ? according to ANSI/TIA/EIA-644



Flexible Topology

Up to 65.535 devices within one EtherCAT network

DL | ESC



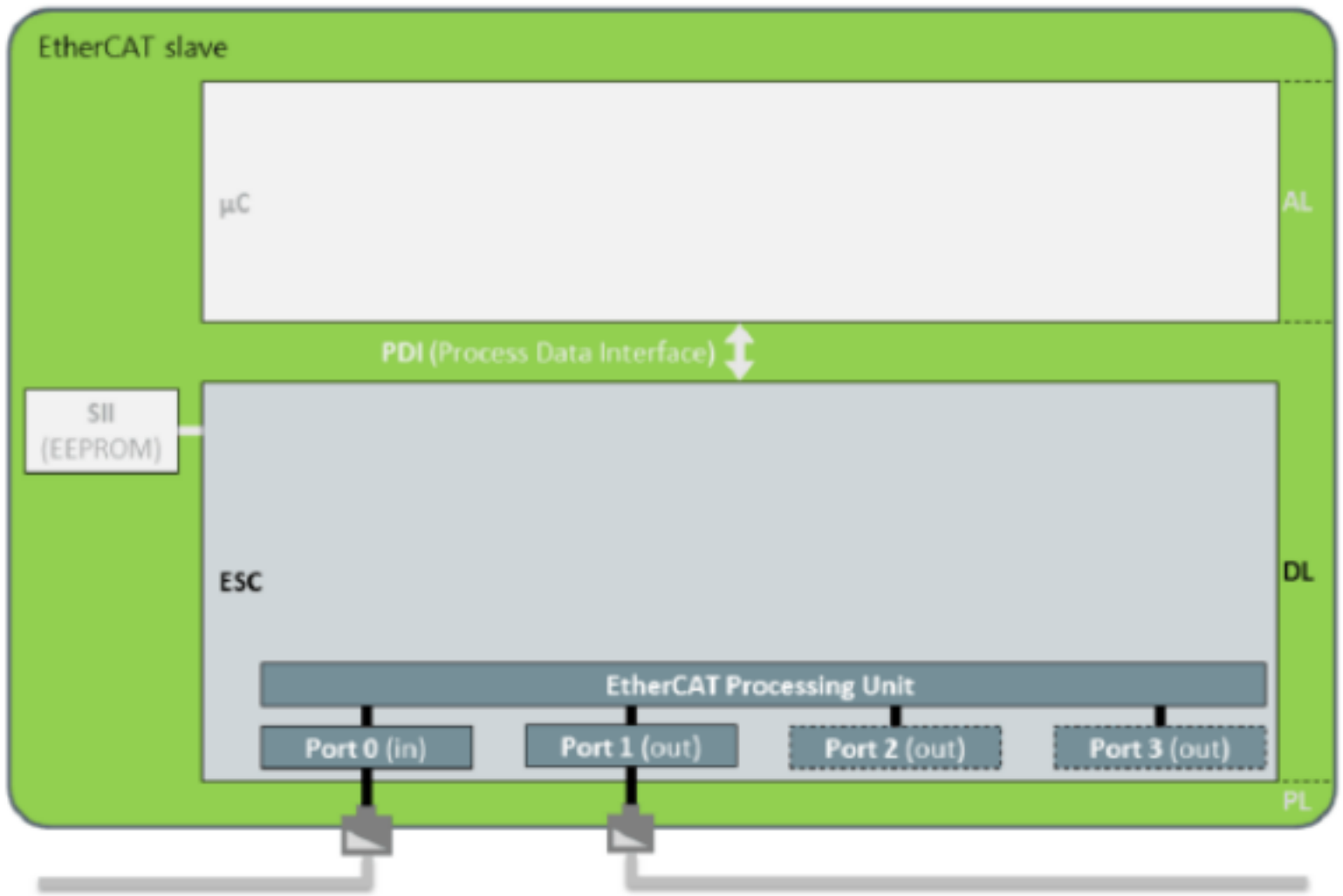
© EtherCAT Technology Group

Purpose of EtherCAT Slave Controller (ESC)

ESC overtakes the low-level, hard real-time tasks in an EtherCAT slave device

- Access to Physical Medium & Link Handling
- Frame Routing
- Addressing & On-the-fly processing
- Configuration of ESC building blocks
- Hardware support to Distributed Clocks synchronization
- Set up AL State Machine interactions

© EtherCAT Technology Group

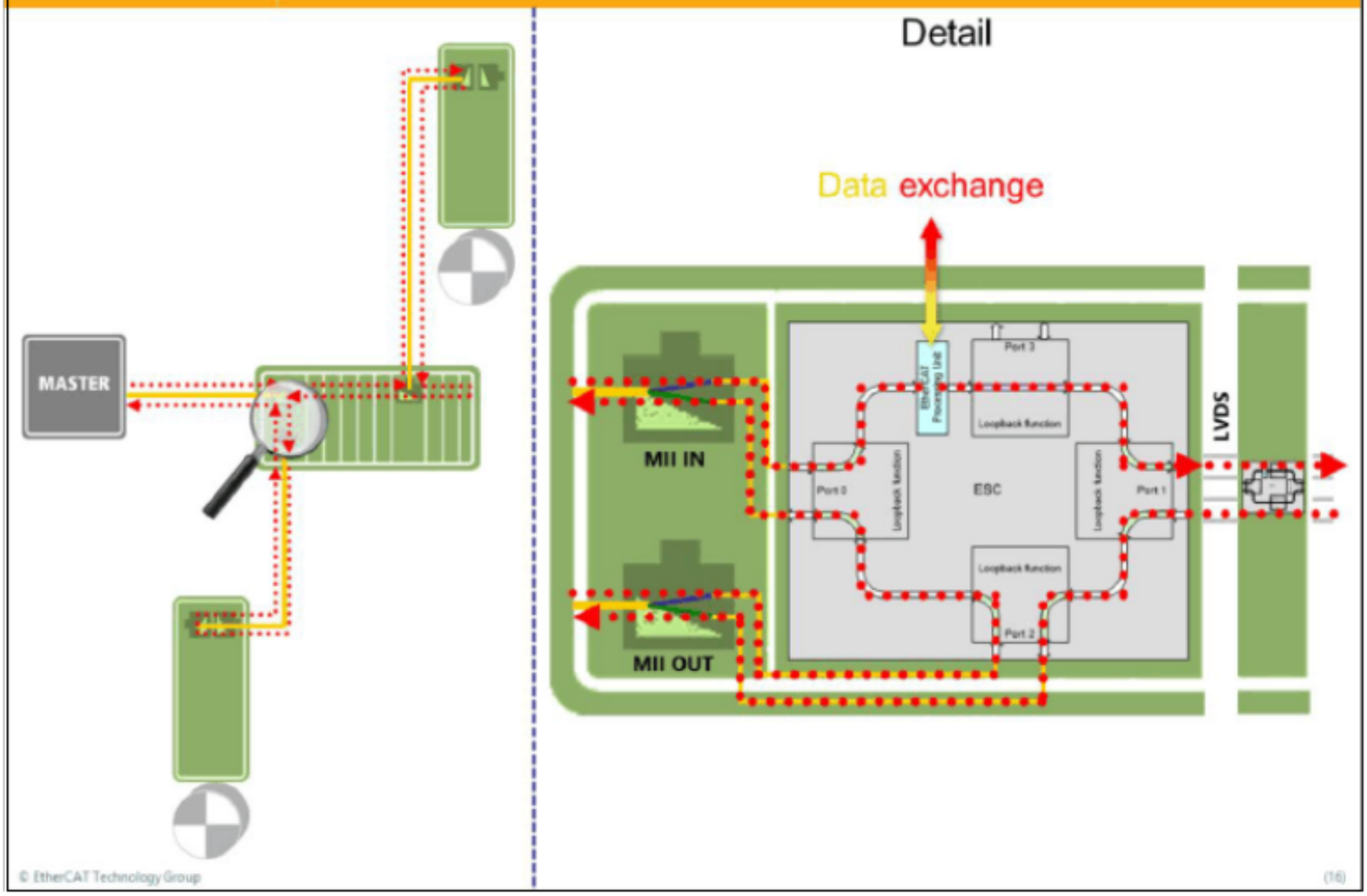


Port 0 is the upstream port and always leads to the master (IN port). This port supports special functions:

- Port 0 automatically opens when all ports are closed, acting as recovery port for the slave
- Port 0 enables to detect and drop frames circulating on the network

© EtherCAT Technology Group

DL | Frame Processing

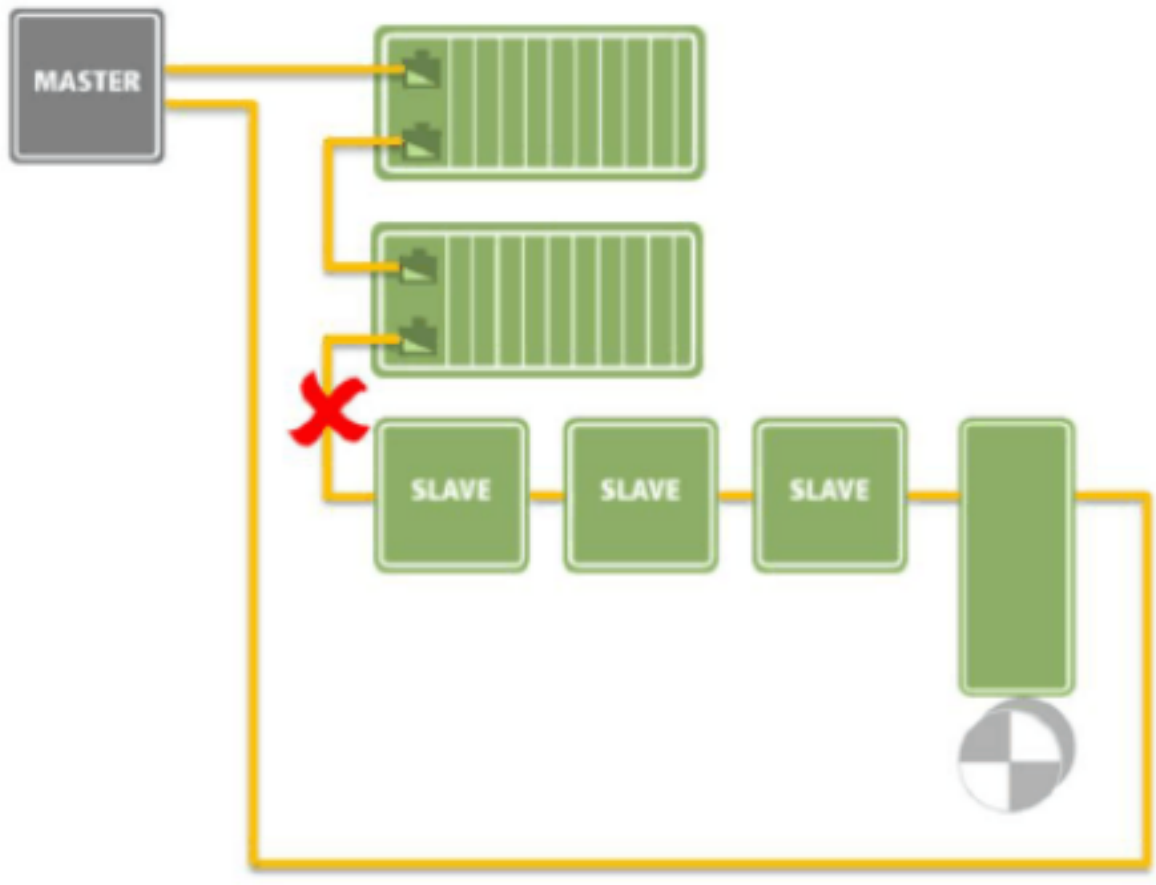


© EtherCAT Technology Group

PL | Cable Redundancy

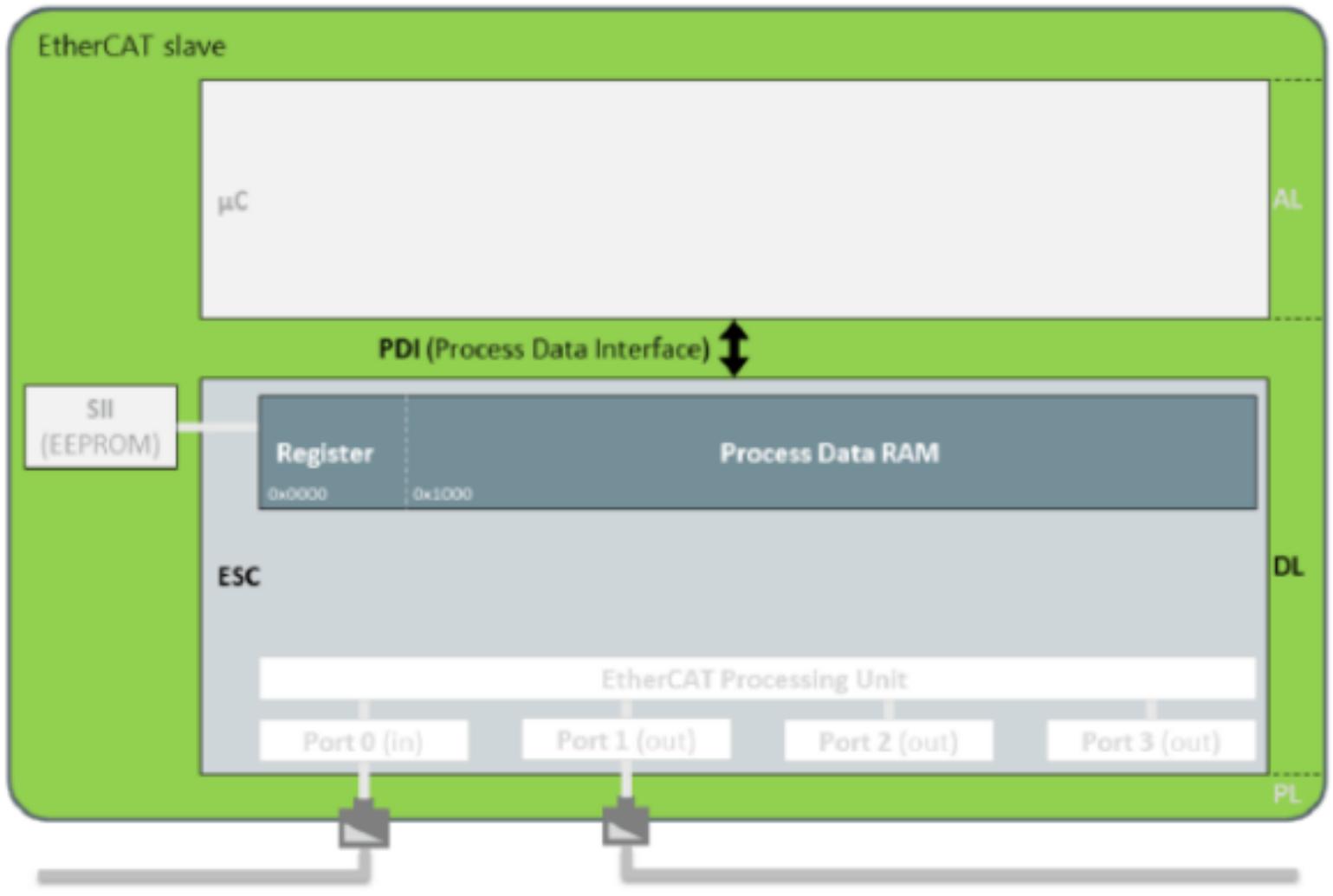
© EtherCAT Technology Group

Only a second Ethernet port is needed on the master – possible with all EtherCAT Slave devices



On slave side, cable redundancy is automatically supported by ESCs: no additional features needed.

DL | DPRAM

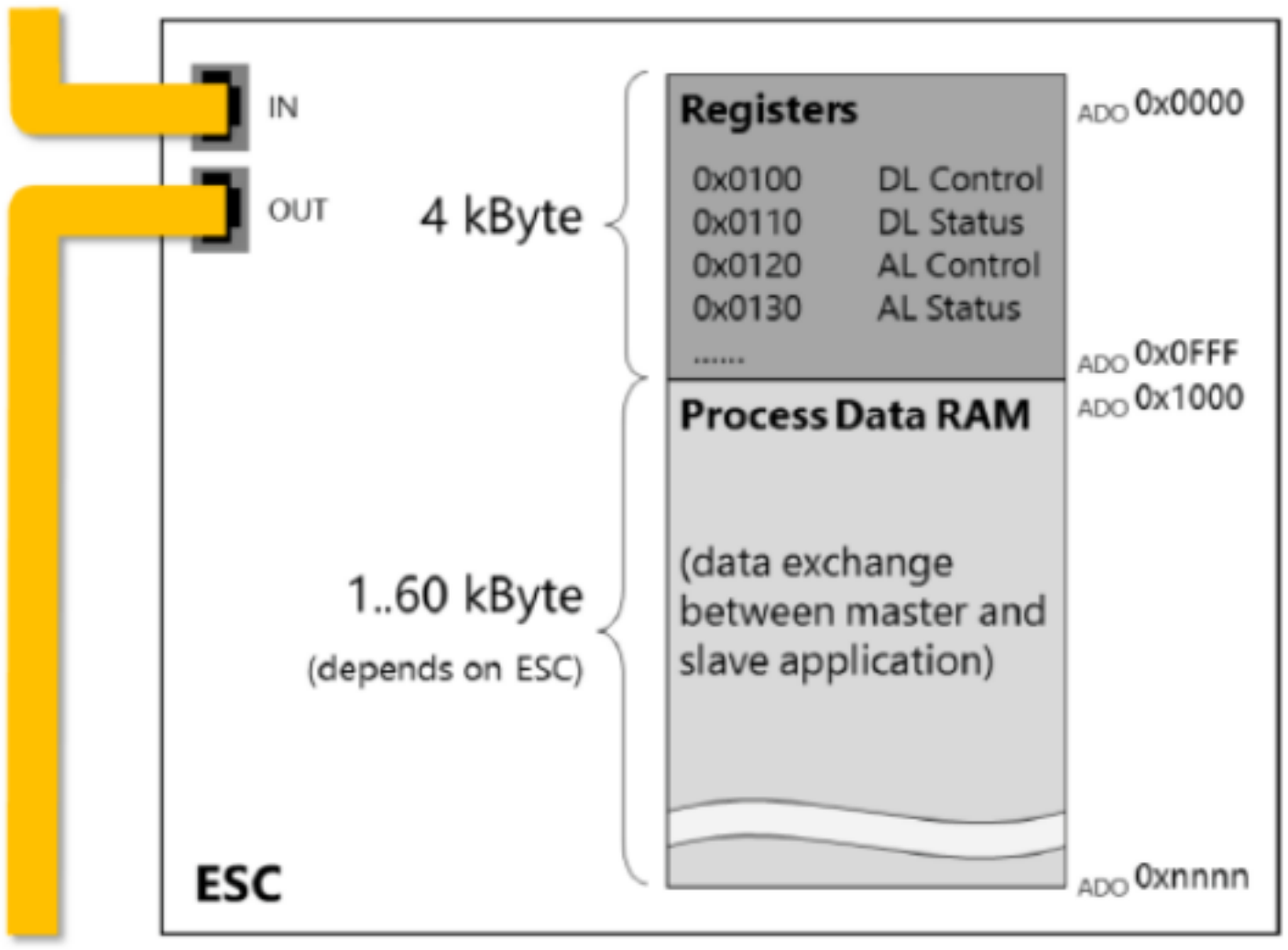


© EtherCAT Technology Group

© EtherCAT Technology Group

DL | DPRAM

The DPRAM consists of two ranges: Registers and Process Data RAM.



Registers contain low-level ESC settings and diagnostic information.

Process Data RAM is a free memory available for master and slave application to exchange data.

Write Access:

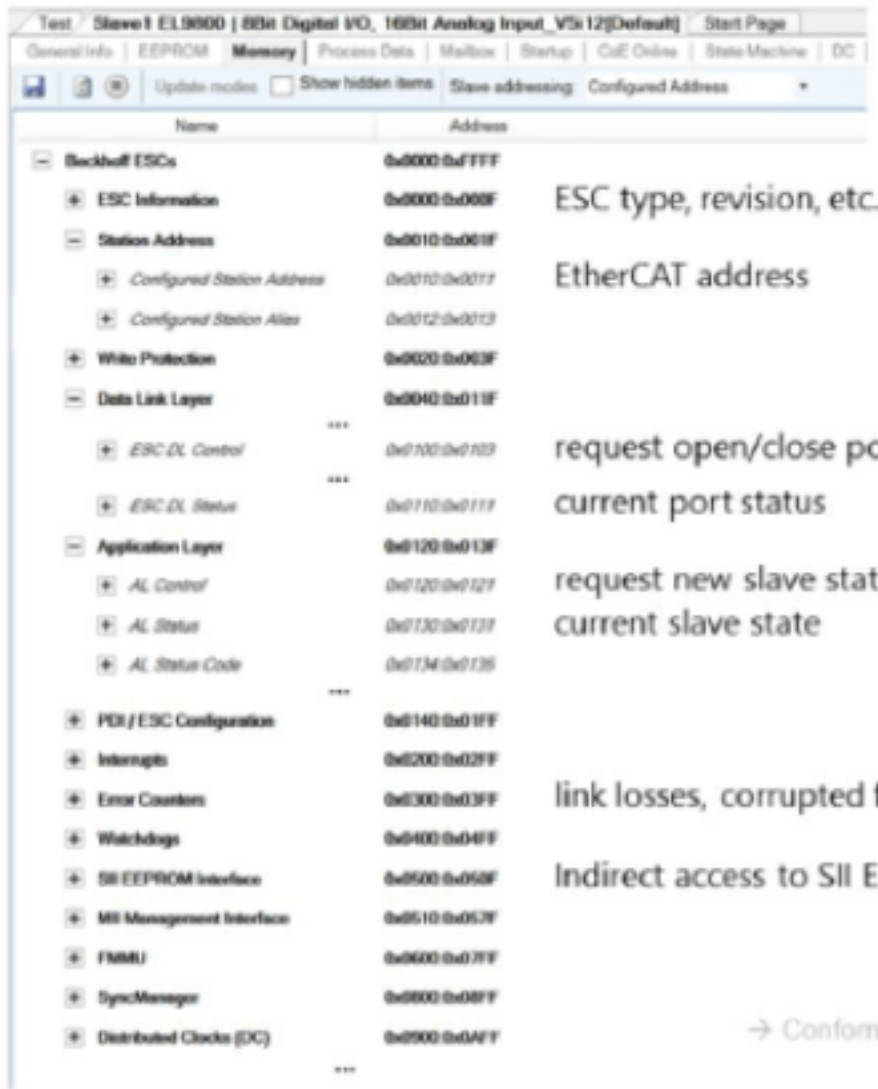
? Shadow registers for all registers integrated

? DPRAM write shall be controlled by SyncManagers

© EtherCAT Technology Group

Registers

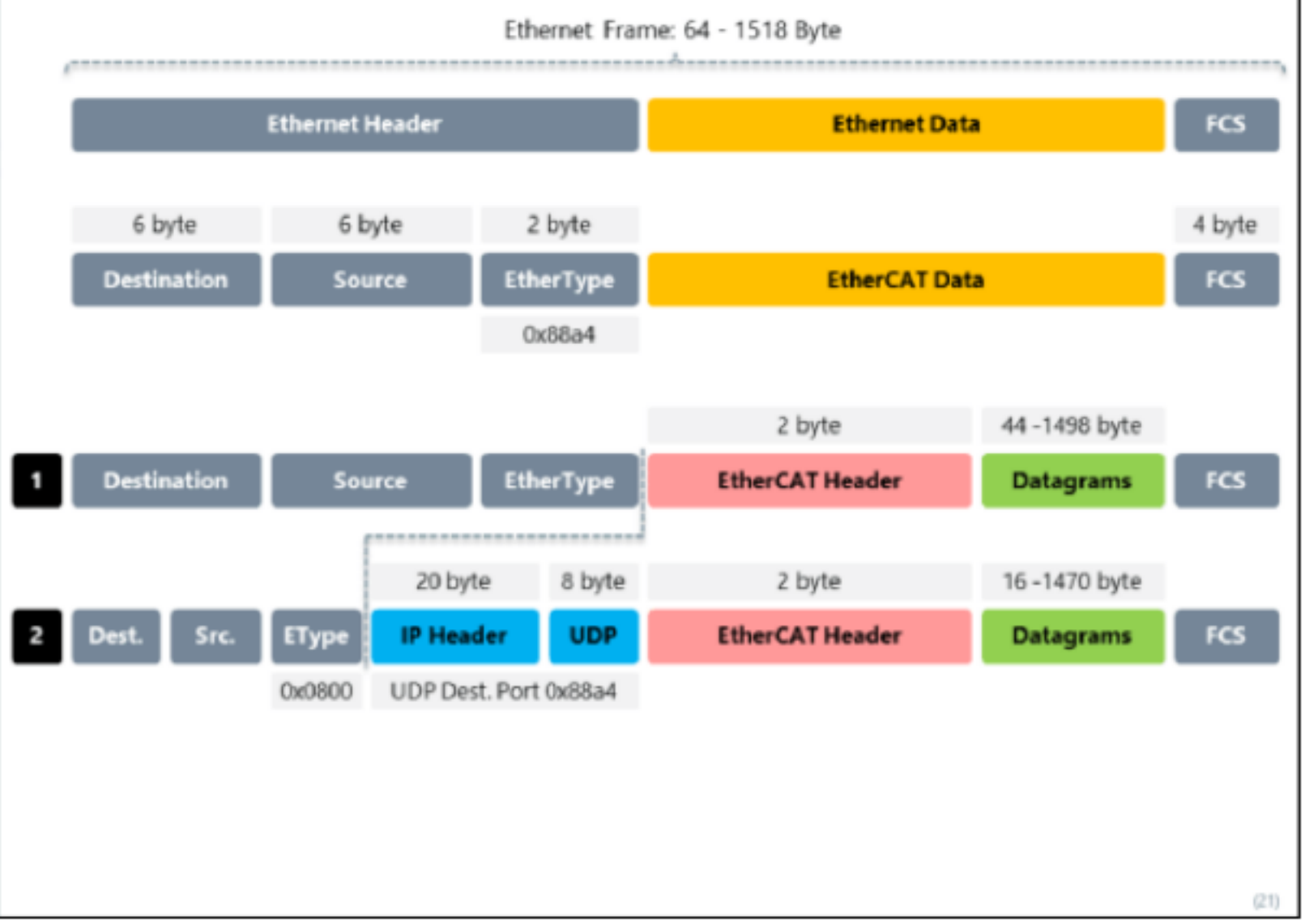
- Contain basic ESC settings/information
- Structure is identical for all ESCs



Reference: ESC Hardware Data Sheet

Ethernet / EtherCAT Frame Structure

EtherCAT uses standard Ethernet IEEE 802.3 frames:



© EtherCAT Technology Group

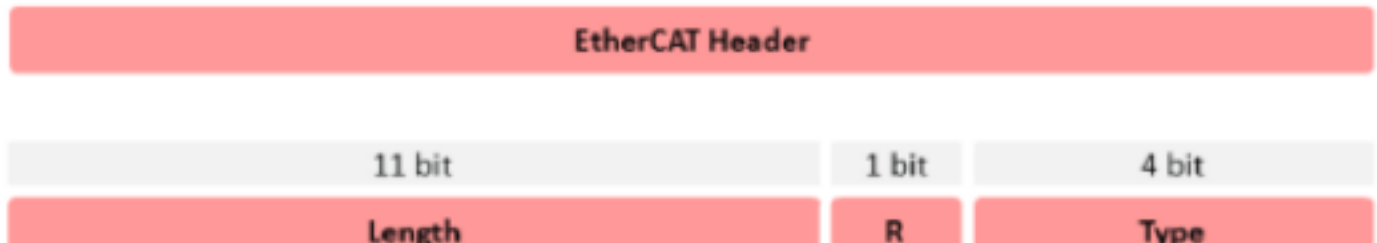
1. EtherCAT Datagrams are carried by raw Ethernet frames
   - ? Default case, always possible when the EtherCAT segment is connected to the master directly (EtherCAT Direct Mode)

2. EtherCAT Datagrams are encapsulated into UDP/IP protocol
   - ? Mandatory when the EtherCAT segment is connected to the master through a standard switch (EtherCAT Open Mode)
   - ? Also possible in Direct Mode, e.g. if the master uses the standard OS driver to send frames

© EtherCAT Technology Group

EtherCAT Frame Header

The EtherCAT header contain the information about the EtherCAT type carried by the frame:



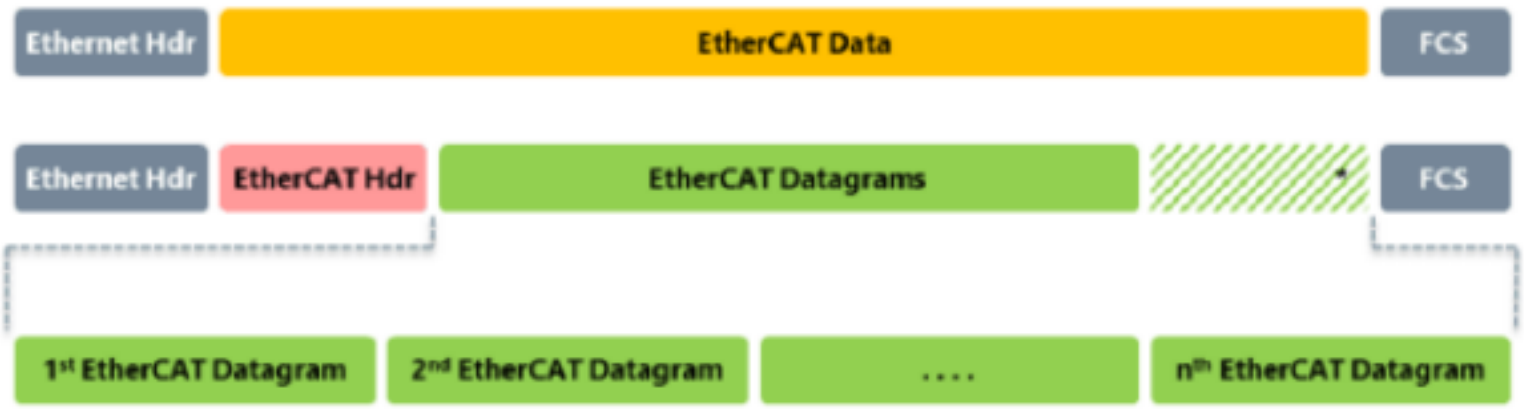
Type	Meaning
0	Reserved
1	EtherCAT Device Communication The only type that is evaluated by the ESC!
2,3	Reserved
4	EAP Process Data Communication
5	EAP Mailbox Communication
6-15	Reserved

EAP: EtherCAT Automation Protocol

Types 4 and 5: Master-Master Communication

EtherCAT Datagrams

The payload of an EtherCAT frame (with Type = 1) consists of a variable number of EtherCAT Datagrams.

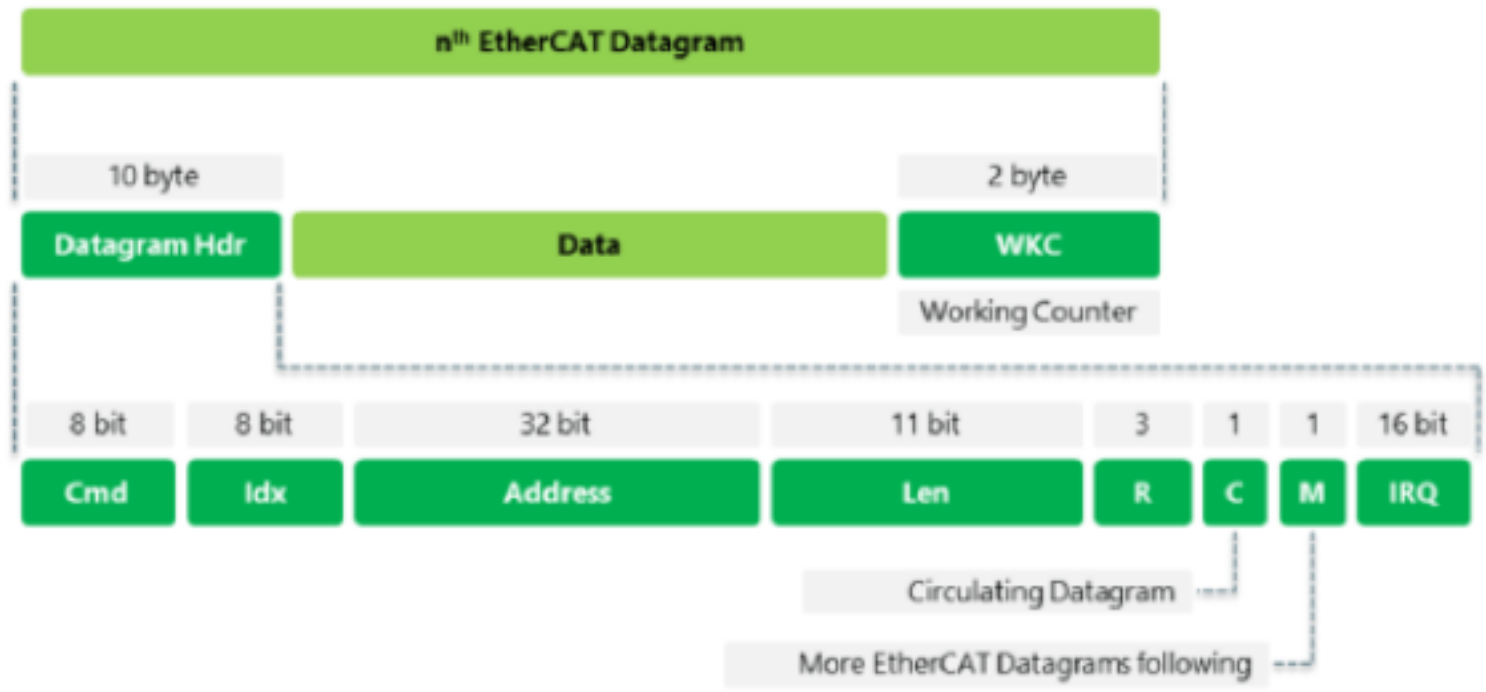


*add 1-32 padding bytes if Ethernet frame is less than 64 bytes

© EtherCAT Technology Group

EtherCAT Datagram Header

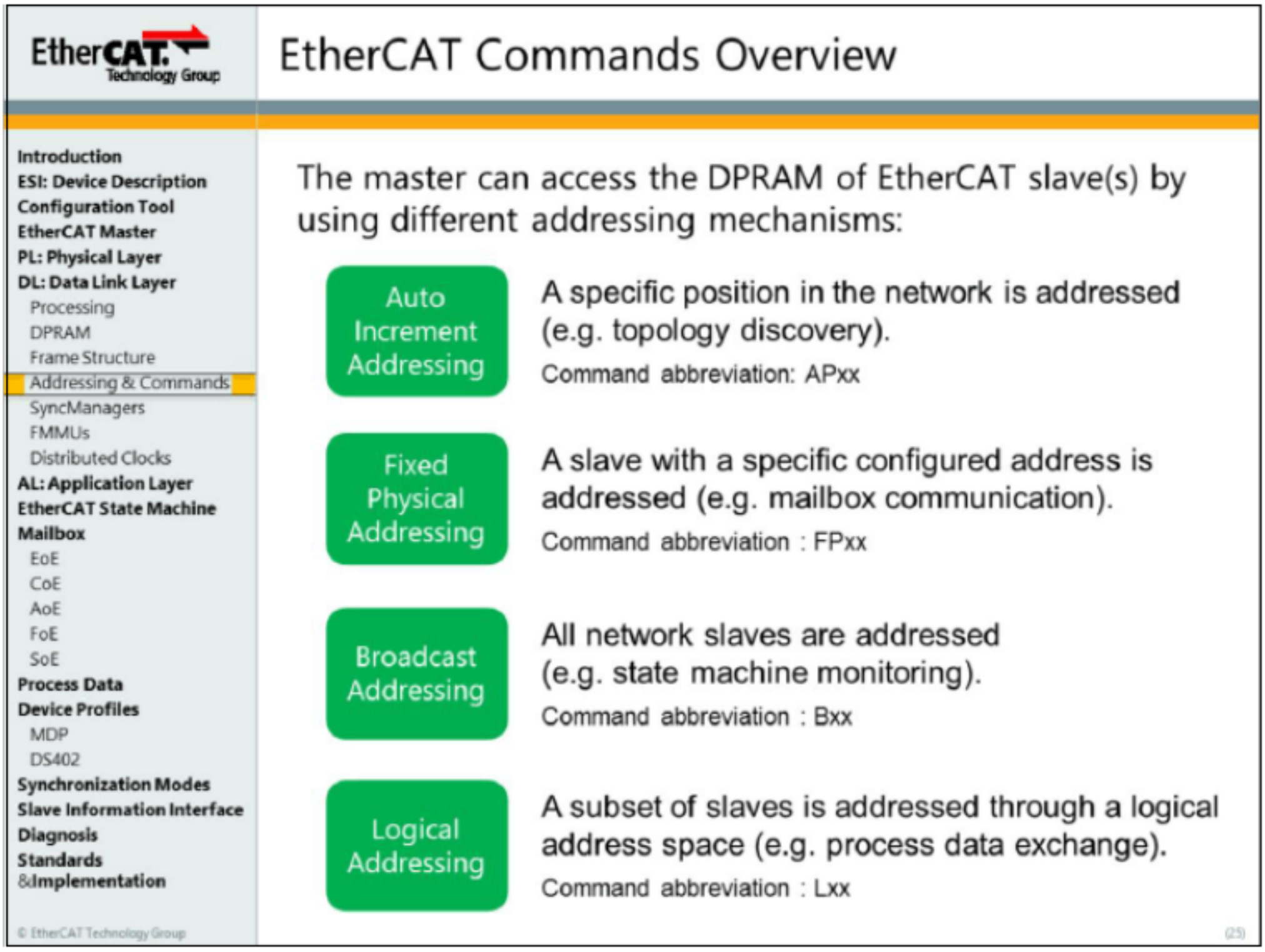
Each EtherCAT Datagram consists of a Header, a Data field and a Working Counter (WKC).



© EtherCAT Technology Group

Cmd: datagram type

Idx : set by master to identify lost or duplicated datagrams



Each addressing mode is combined with the command (Cmd):

xxRD (Read) → addressed slave(s) write content into the datagram

xxWR (Write) → addressed slave(s) read content out of the datagram

xxRW (ReadWrite) → addressed slave(s) read, and thereafter write content into the same datagram

xxRMW = (ReadMultipleWrite) → the addressed slave writes content into the datagram, all other slaves read it out of the same datagram

Address Field

Address field in each datagram is 32-bit

- Higher 16 bits address the device (65.535 devices possible) and lower 16 bits specify an offset in DPRAM (max. 64 kByte) of addressed device

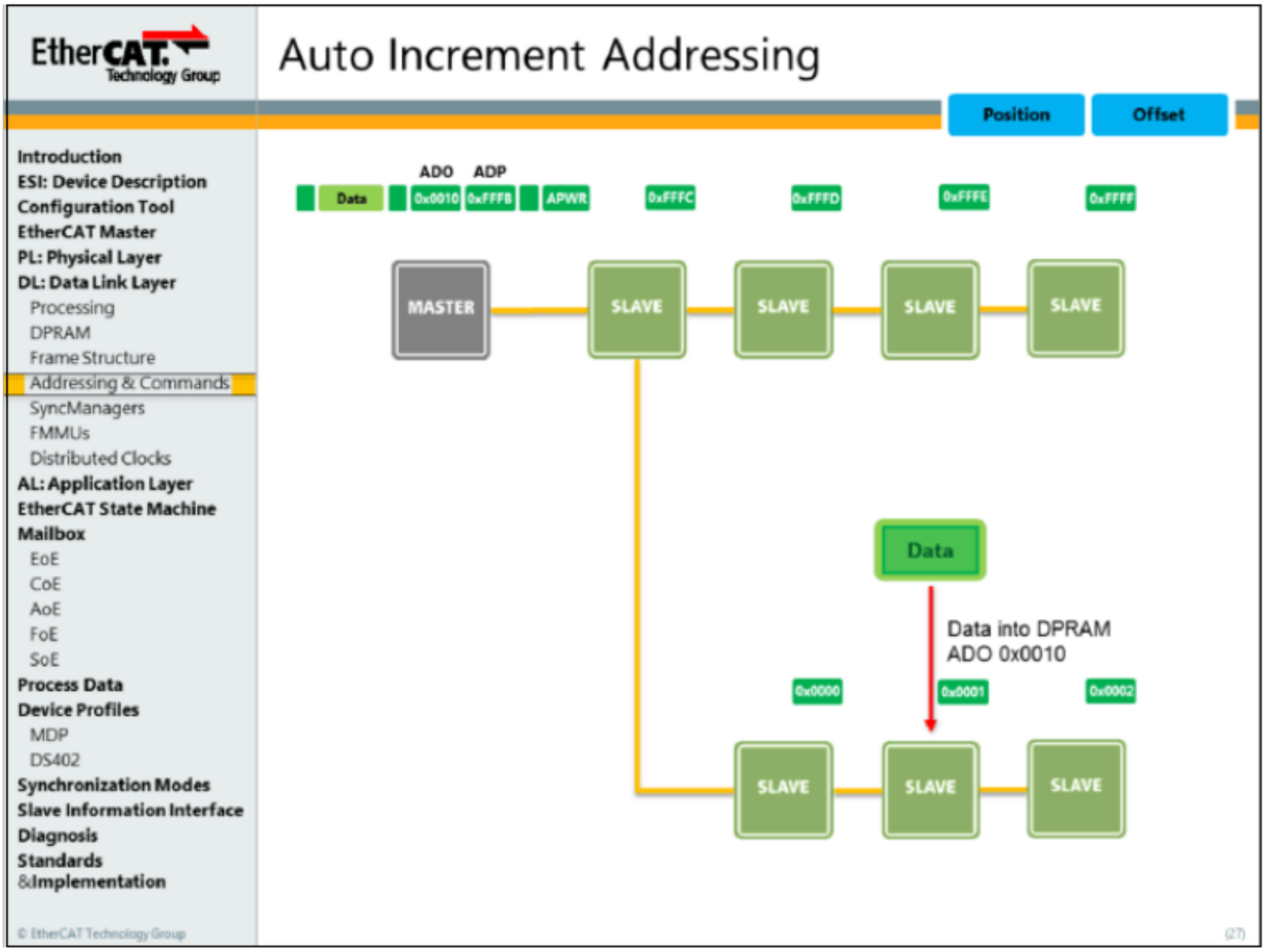
or

- 32 bits used for logical addressing

© EtherCAT Technology Group

ADP = Address Position → which slave is addressed

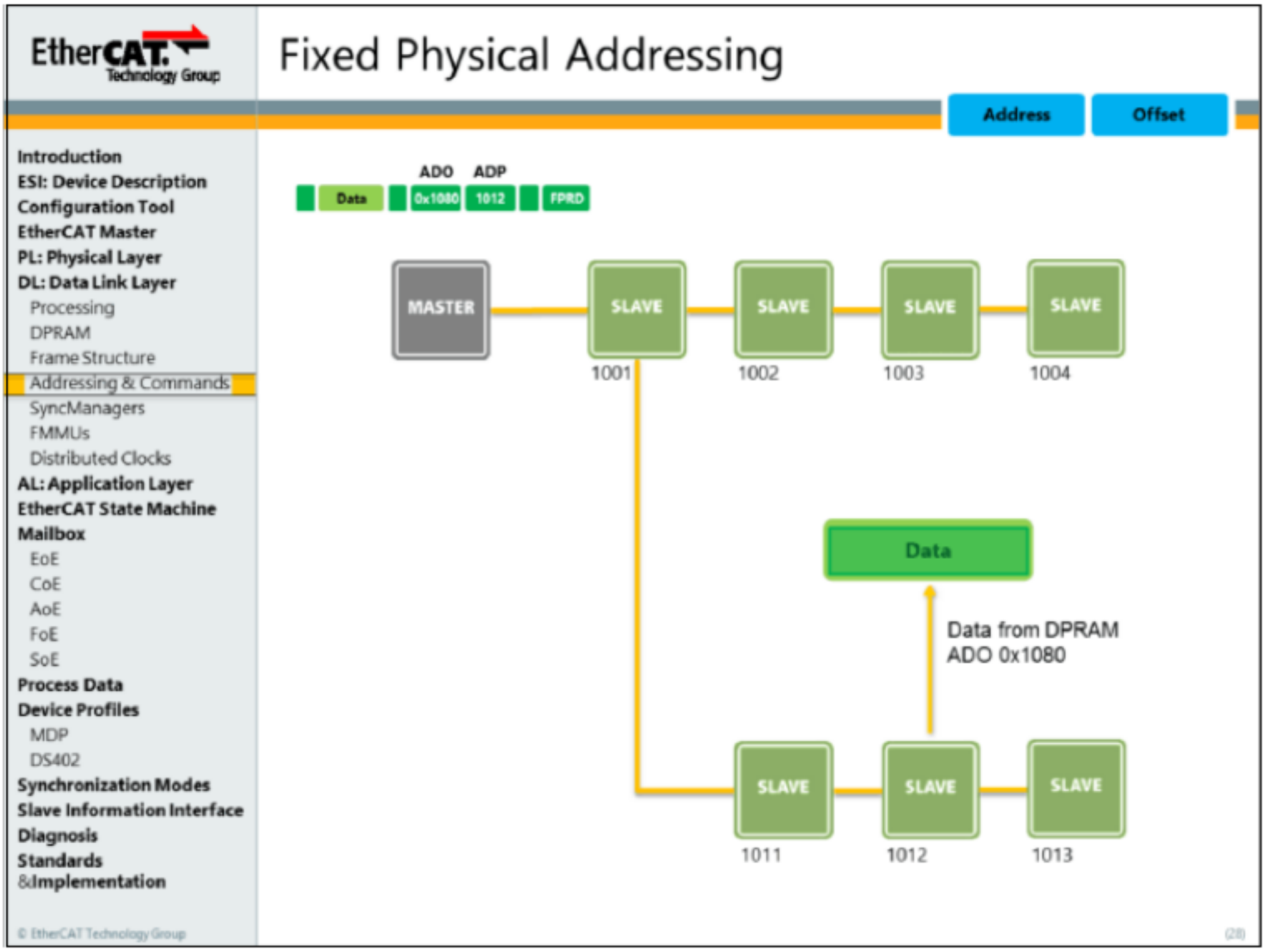
ADO = Address Offset → which DPRAM location is accessed



Master writes a negative Auto Increment address (16 bit) into ADP depending on position

- ? Slave which receives ADP == 0x0000 is addressed
- ? Each slave increments ADP by 1

Offset addresses local Address Space of device

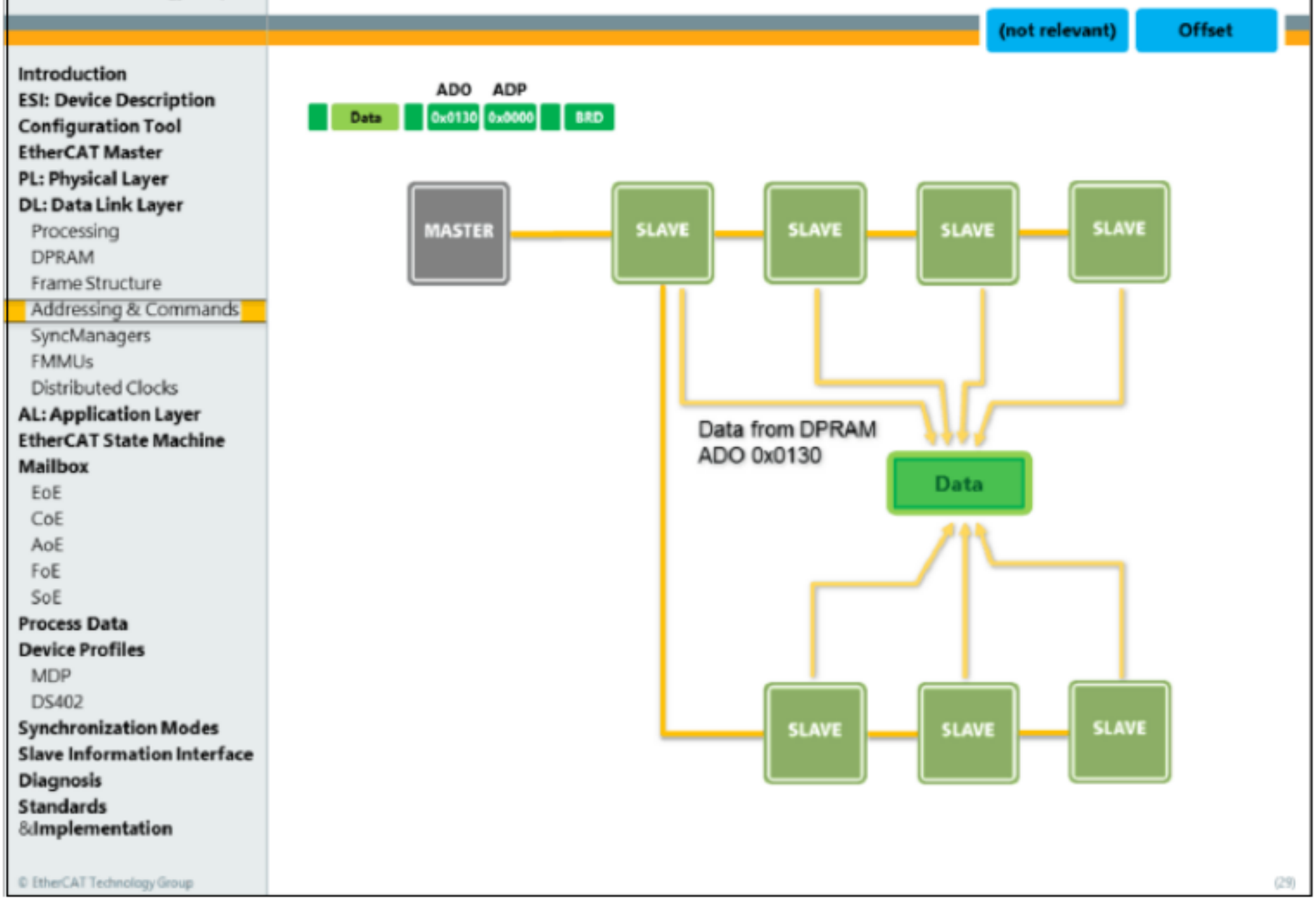


A fixed 16-bit address is assigned to each slave (written by master into $_{ADO}$ 0x0010)

- ? Independent from slave position
- ? Written by master during start-up
- ? Fixed address lost after power loss

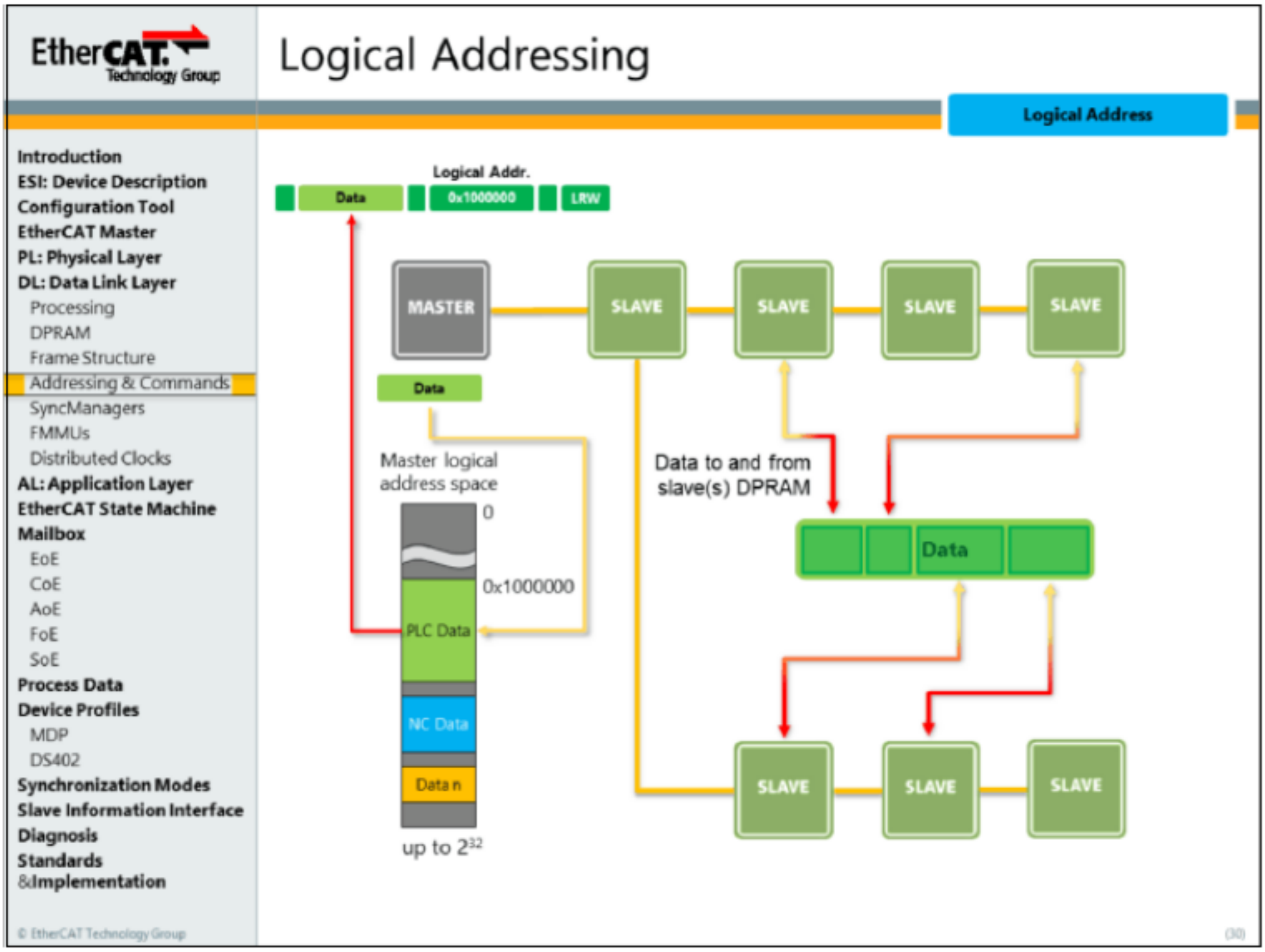
Offset addresses local Address Space of device

Broadcast Addressing



© EtherCAT Technology Group

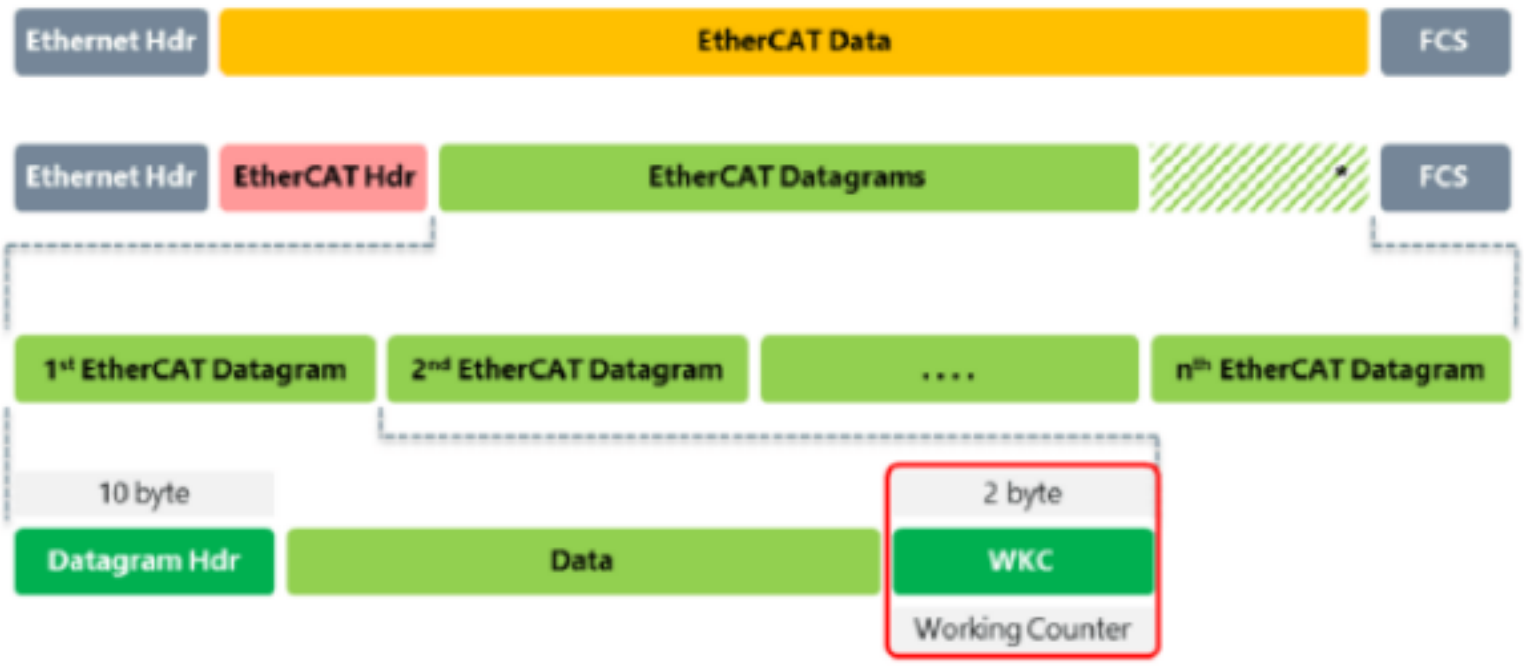
Read data are logic OR of single register values



Slave reads/writes the logical process image of master

More slaves can share the same memory interval

EtherCAT Datagram Working Counter



*add 1-32 padding bytes if Ethernet frame is less than 64 bytes

© EtherCAT Technology Group

Working Counter (WKC)

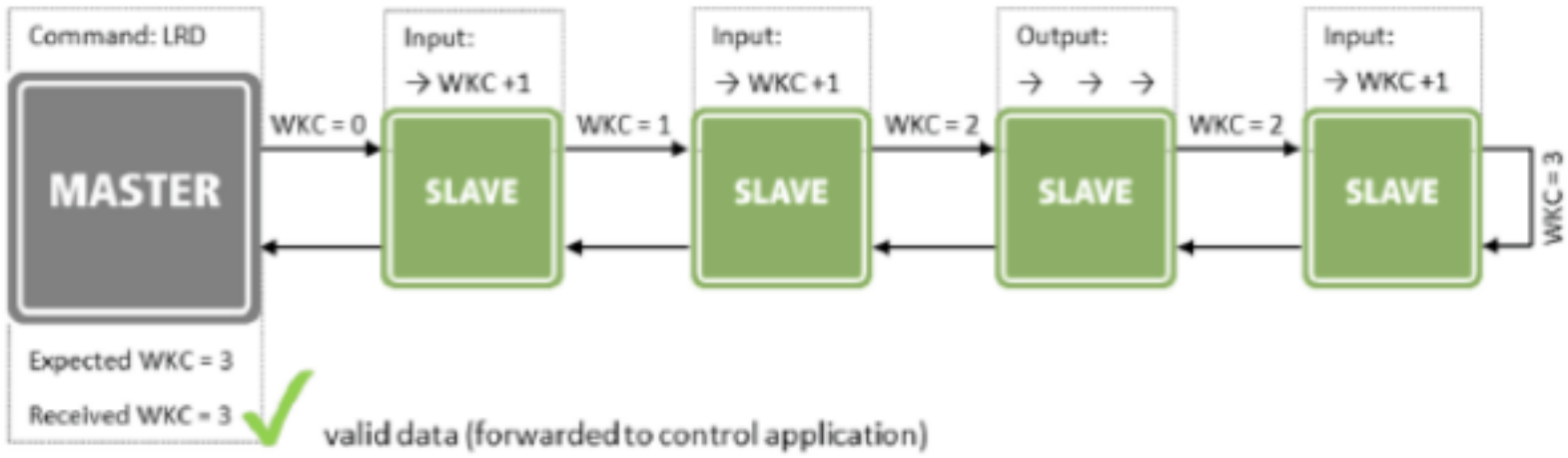
- WKC is incremented after each successful access
- Increment depends on the command:

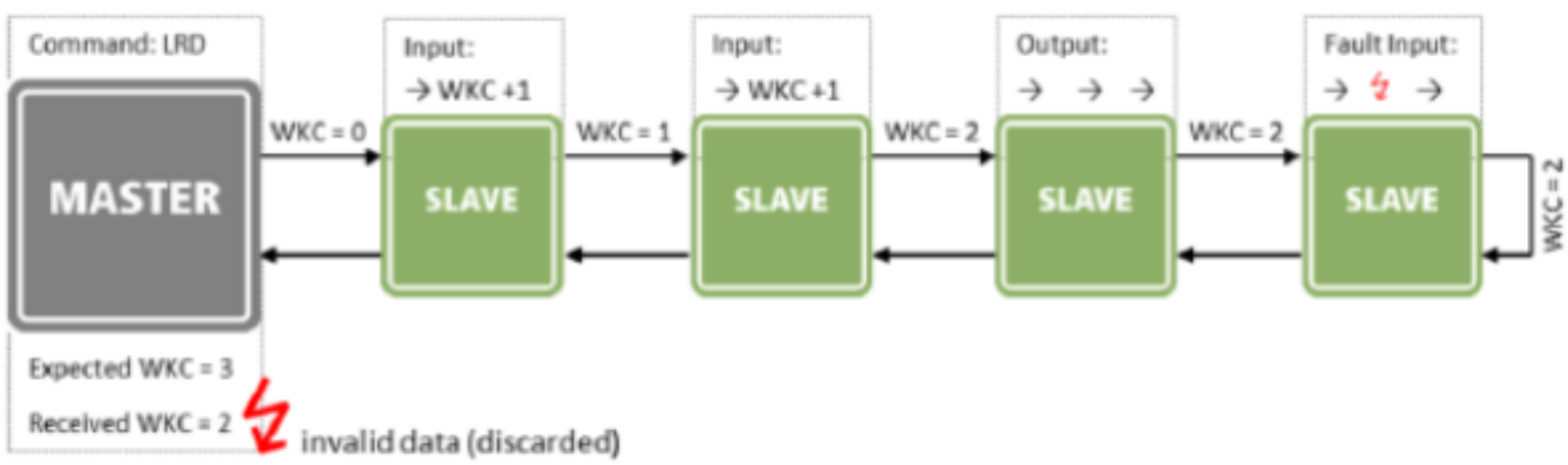
Command		Increment
Read	xxRD	+1
Write	xxWR	+1
Read Write → Read → Write	xxRW	 +1 +2

- The returning WKC value is compared by master with the expected value
- In case of invalid WKC, input data carried by the corresponding datagram are discarded

© EtherCAT Technology Group

Working Counter Example (LRD)

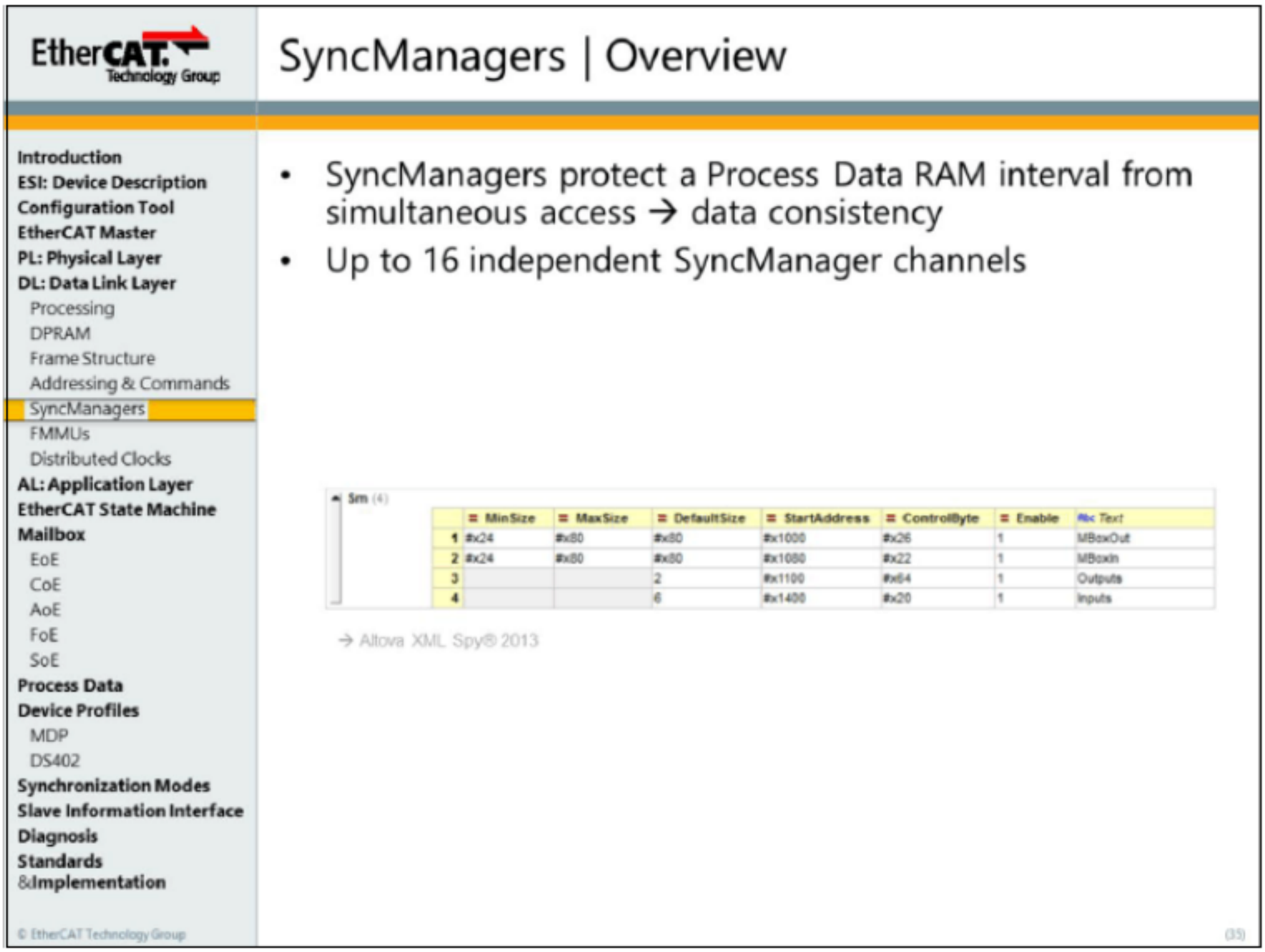




© EtherCAT Technology Group

DL | SyncManager

© EtherCAT Technology Group



The SyncManager configuration registers start at $_{ADO}$ 0x0800

SM0:

0x800:0x801 : Physical Start Address

0x802:0x803 : Length (in Byte)

0x804 : Control Register (including type and direction)

0x805 : Status

0x806 : Activate

0x807 : PDI Control

SM1:

0x808:0x809 : Physical Start Address

...

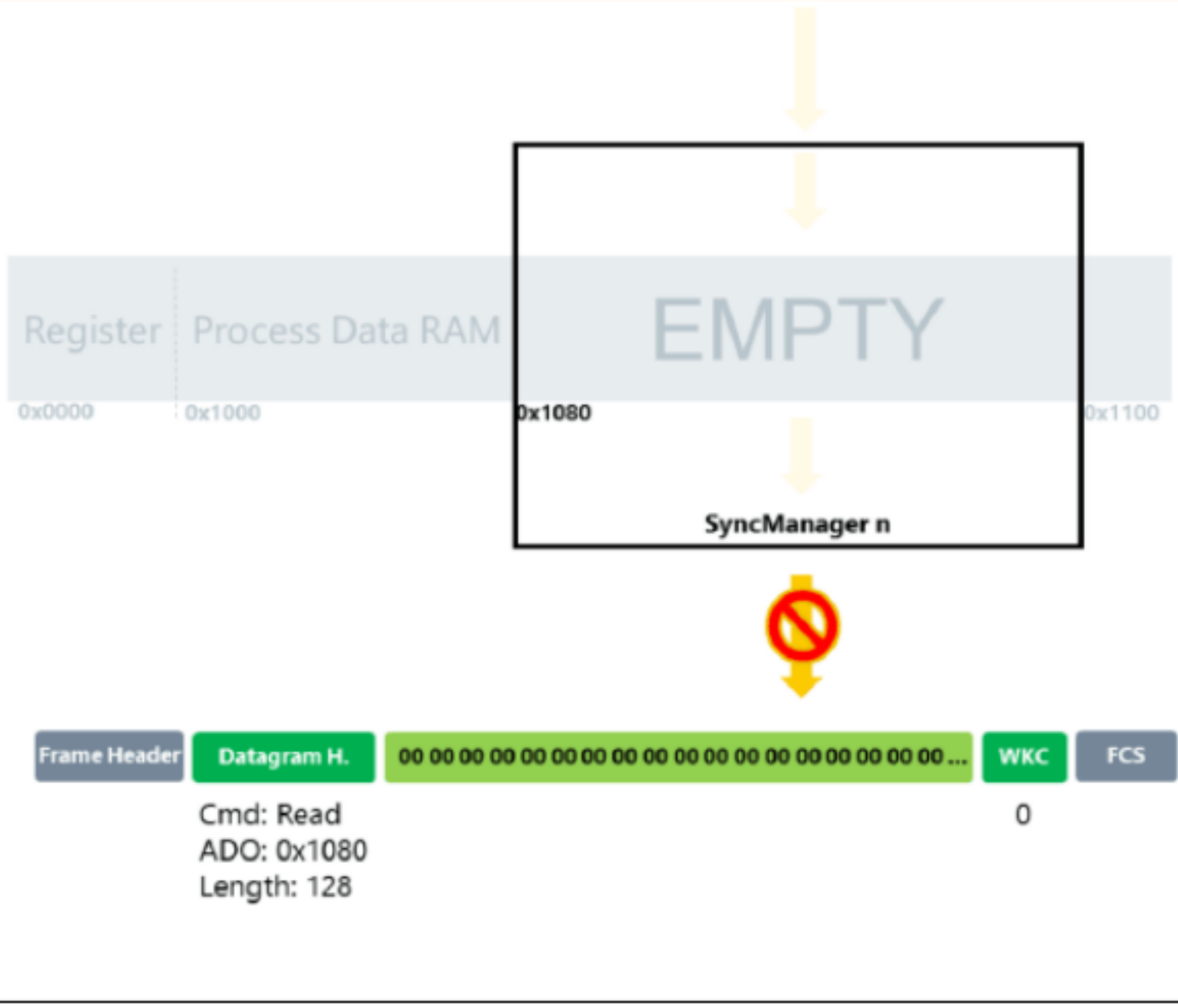
SyncManagers | Types

- **Mailbox (1-buffer) Type**
  - 1 buffer SyncManager supports handshake
  - Data overflow protection
  - Writing side must write before reading side can read
  - Reading side must read before writing side can write again
  - Used for non-process data communication

- **Buffered (3-buffer) Type**
  - 3 buffer SyncManager guarantees consistent data delivery and access to the newest data any time
  - Always a free buffer to write
  - Always a consistent buffer to read (except before the first writing)
  - Used for process data communication

© EtherCAT Technology Group

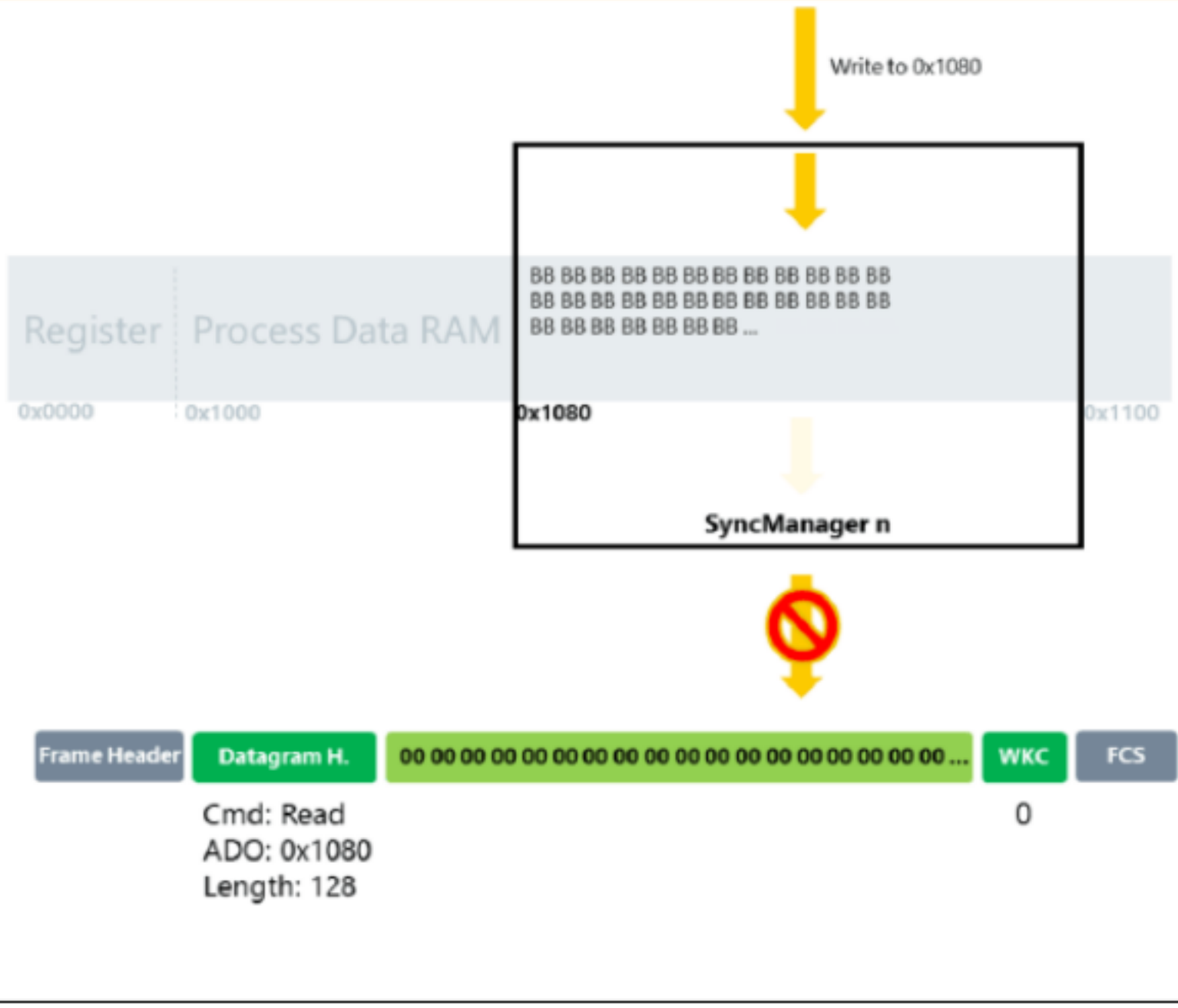
Mailbox Type (1-buffer) Read Example



© EtherCAT Technology Group

Read access to an empty SyncManager failes

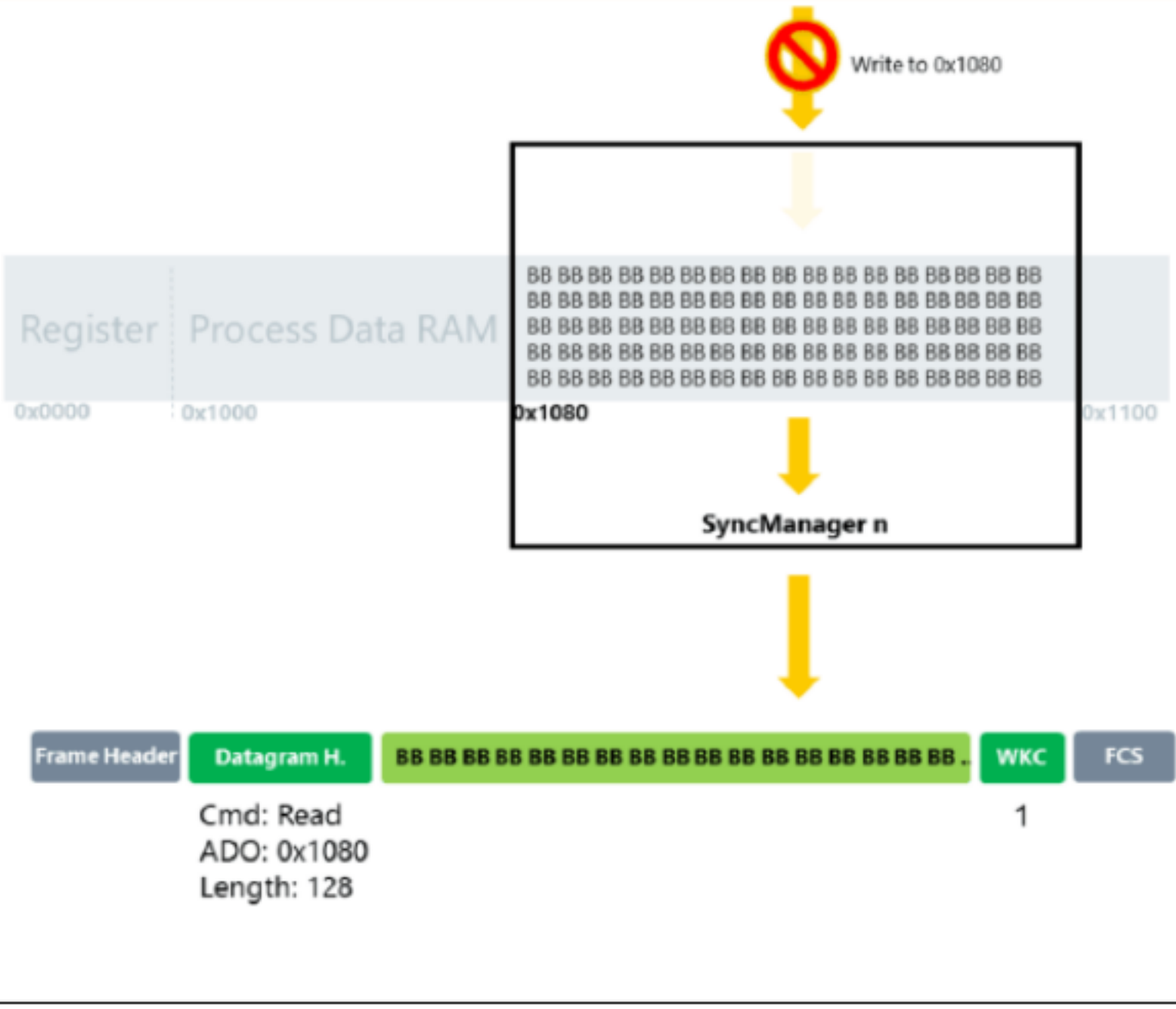
Mailbox Type (1-buffer) Read Example



© EtherCAT Technology Group

Read access to a not complete written SyncManager failes

Mailbox Type (1-buffer) Read Example

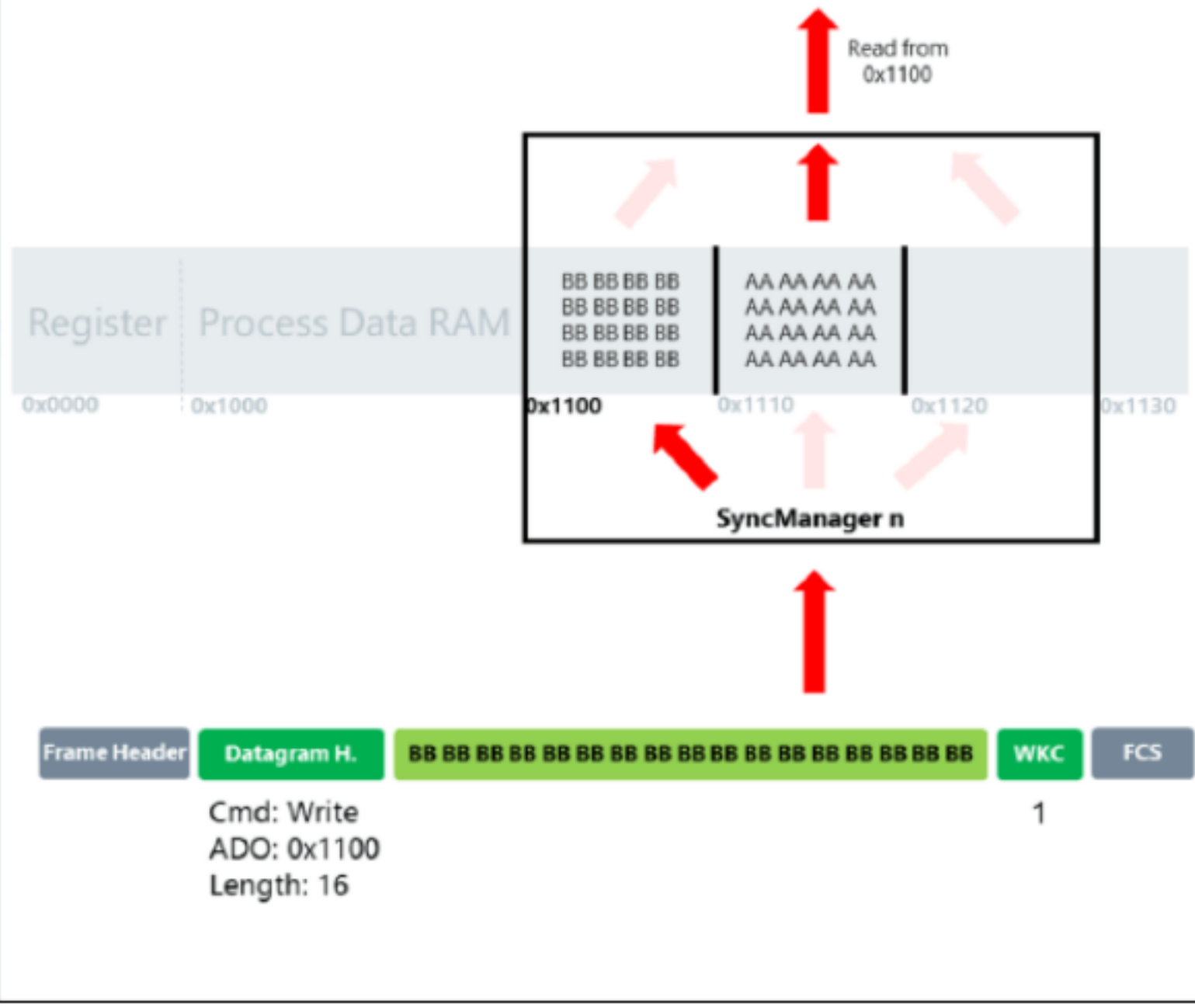


© EtherCAT Technology Group

Read access to a full SyncManager is valid

Write acceess to a full or not complete read SyncManager fails

Buffered Type (3-buffer) Write Example

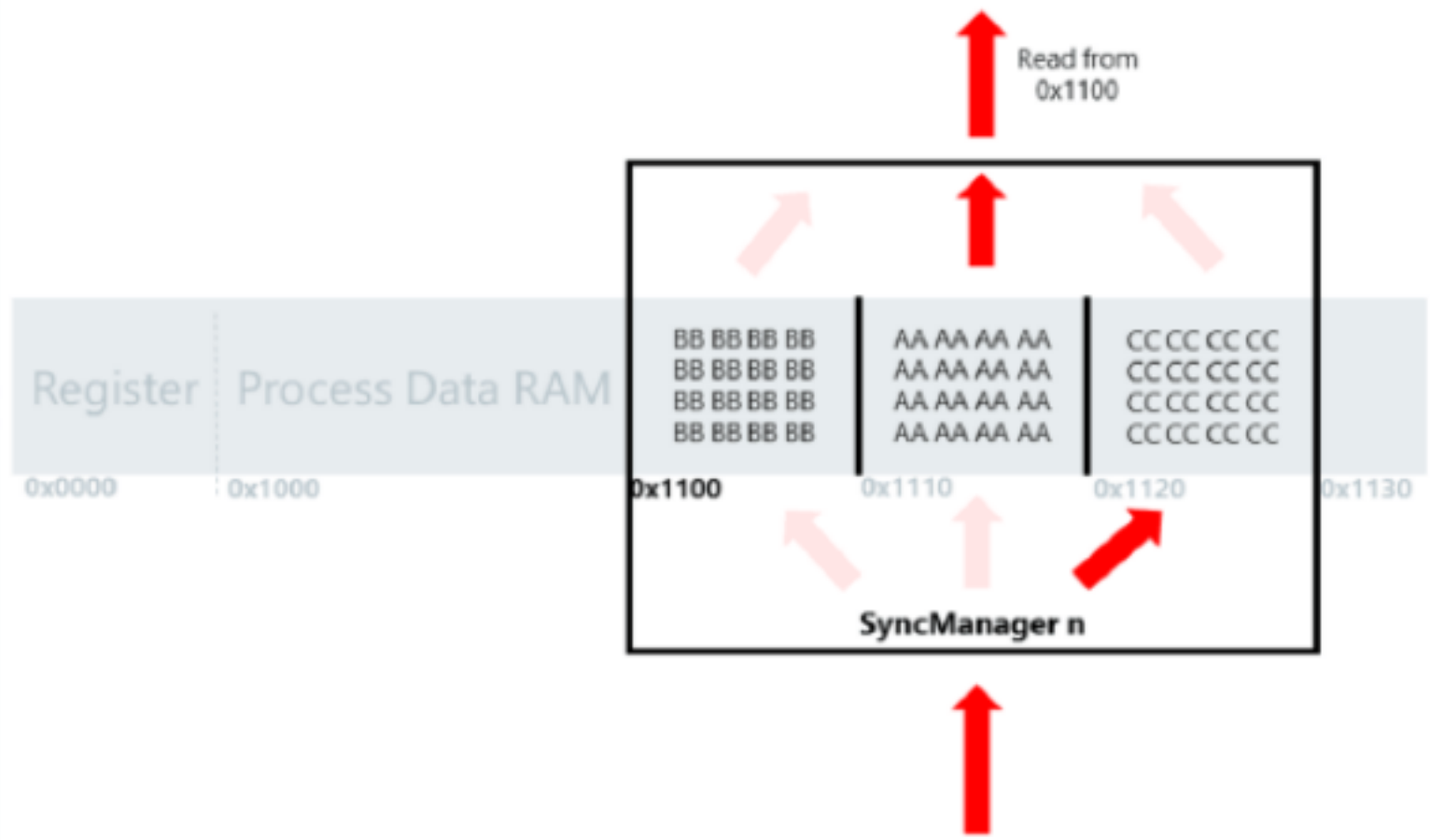


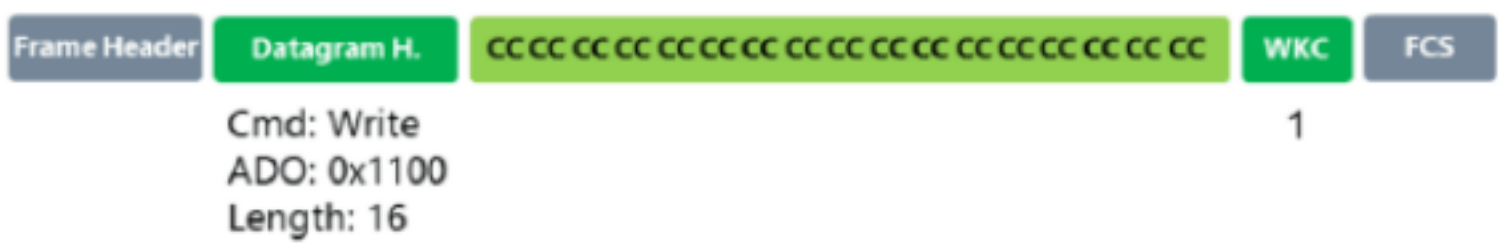
© EtherCAT Technology Group

To exchange n data bytes, 3*n memory bytes required

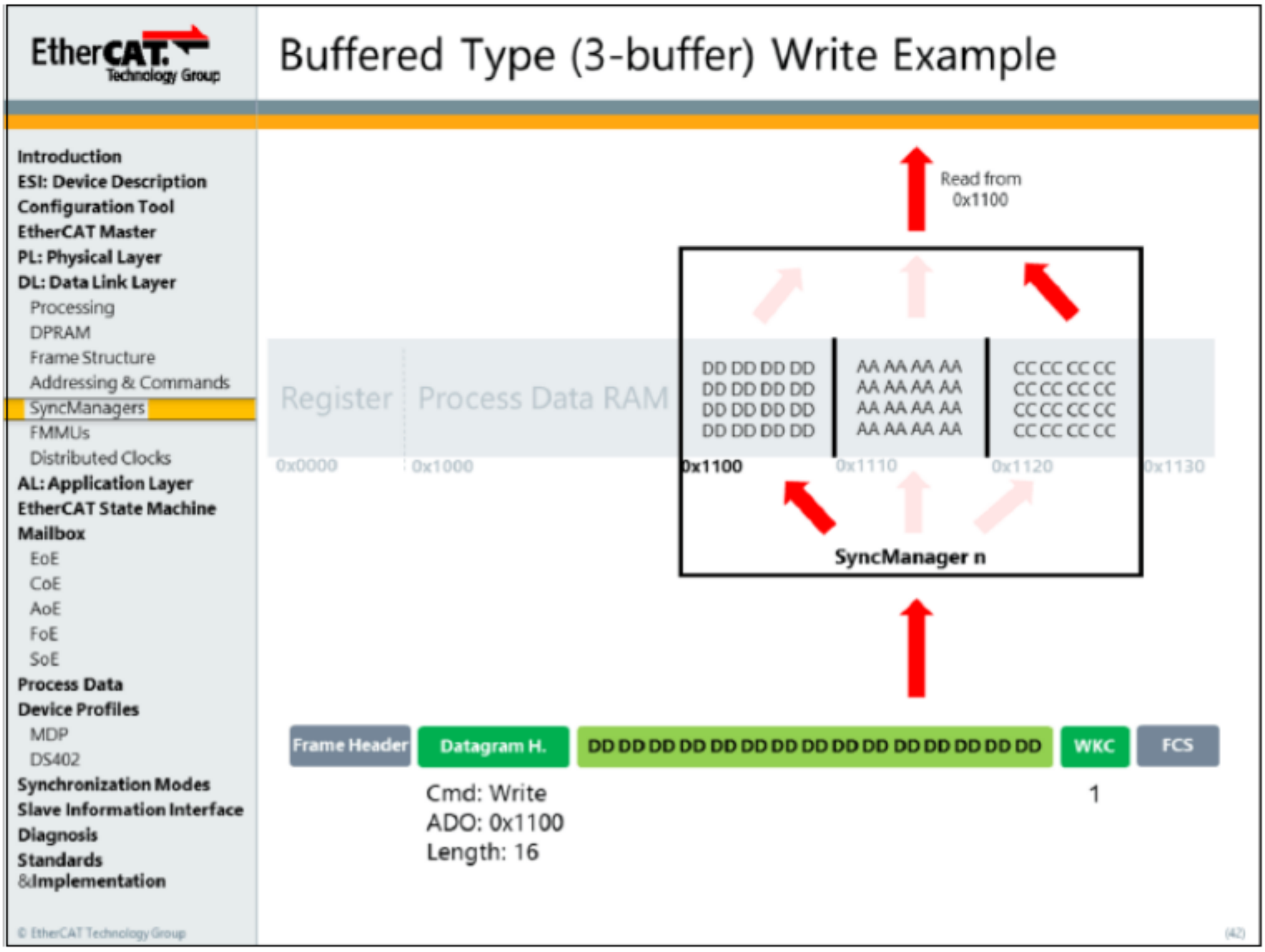
Mechanism handled by SyncManager (ECAT and PDI always use start address), shall be taken into account when determining Start Address of next SyncManager.

Buffered Type (3-buffer) Write Example





© EtherCAT Technology Group

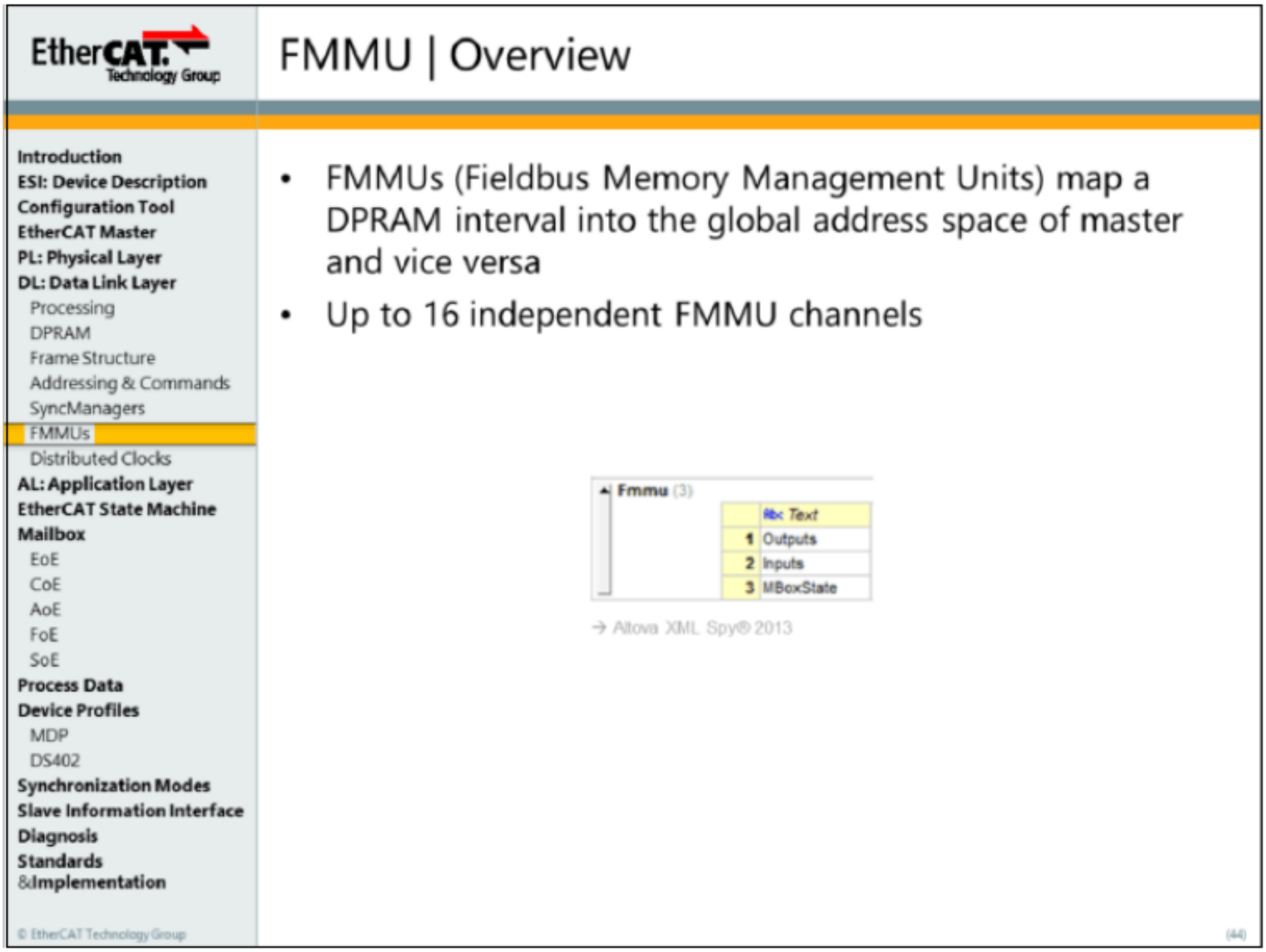


Data ,C “ is provided for read access because it is the latest complete available data

Data ,B “ is overwritten by data ,D “

DL | FMMU

© EtherCAT Technology Group



Bit-wise configuration of the memory section possible

FMMU configuration registers start at $_{ADO}$ 0x0600

FMMU0:

0x600:0x603 : Logical Start Address

0x604 : 0x605 : Length (number of all concerned Bytes)

0x606 : Logical Start bit (within the first byte)

0x607 : Logical Stop bit (within the last byte)

0x608:0x609 : Physical Start Address

0x60A : Physical Start bit

0x60B : Type (including direction)

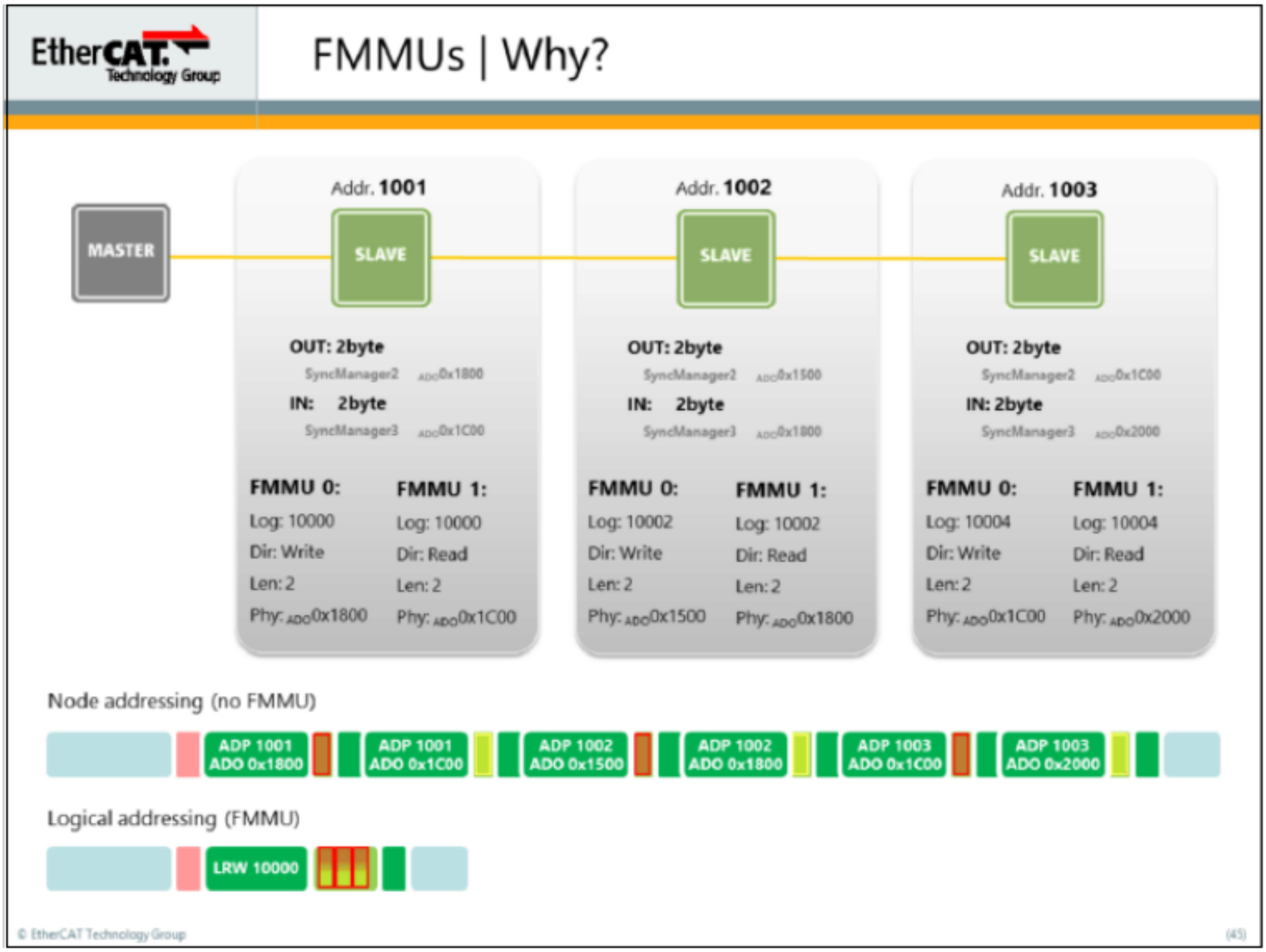
0x60C : Activate

0x60D:0x60F : Reserved

FMMU1:

0x610:0x613 : Logical Start Address

… .



Without FMMUs (APxx, FPxx): each slave needs to be addressed with a single Datagram (10 bytes header, 2 bytes WKC). Each communication direction needs a seperate Datagram (x2).

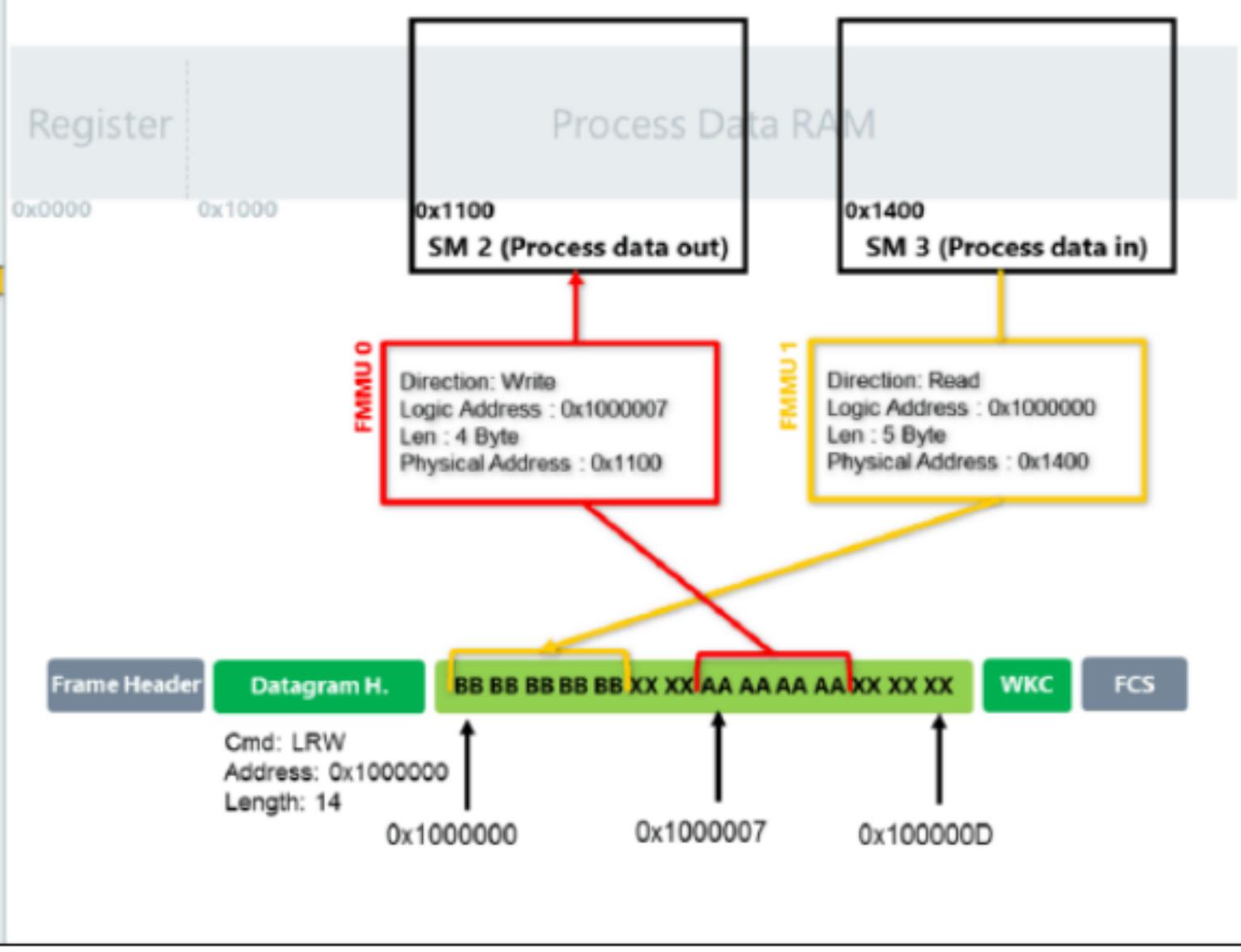
With FMMUs (Lxx): each slave reads and writes its data in the same position. Multiple slaves can share the same Datagram.

SyncManager & FMMU | Assignment

- Standard assignment
  - With mailbox support
    - SM0: Mailbox output
    - SM1: Mailbox input
    - SM2: Process data outputs → FMMU0
    - SM3: Process data inputs → FMMU1
  - Without mailbox support
    - SM0: Process data outputs → FMMU0 (or inputs if no outputs available)
    - SM1: Process data inputs → FMMU1
- Process data always fits exactly into a SyncManager

© EtherCAT Technology Group

SyncManager & FMMU | Operation



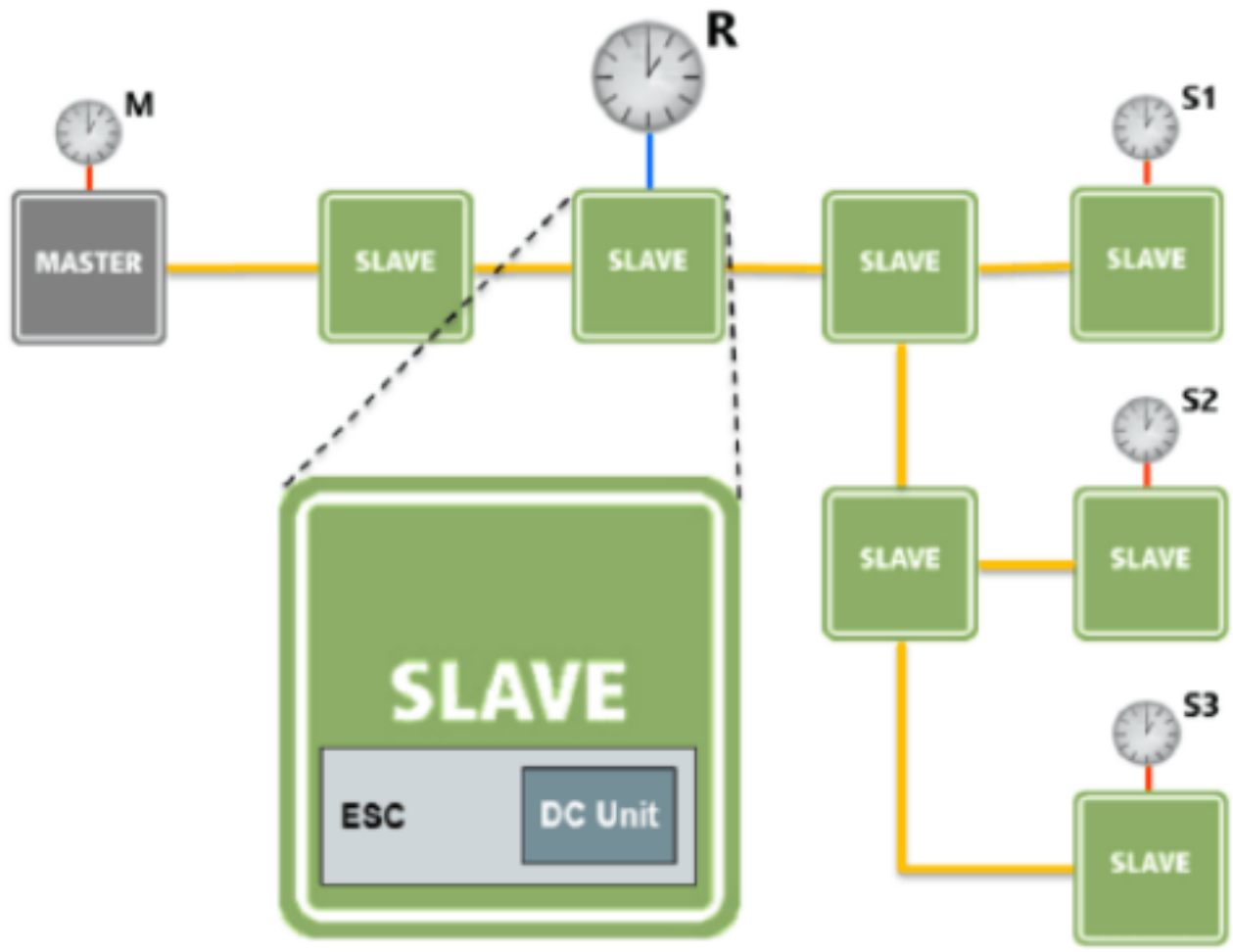
© EtherCAT Technology Group

Addresses and Range for the SyncManager are defined by the ESI file

© EtherCAT Technology Group

Distributed Clocks

Enable precise synchronization (<< 1 µs) of EtherCAT devices.



System Time : time shared by all DC-synchronous devices.

- Beginning: January, 1 st 2000 at 0:00h
- Base unit: 1 ns
- Format: 64-bit value

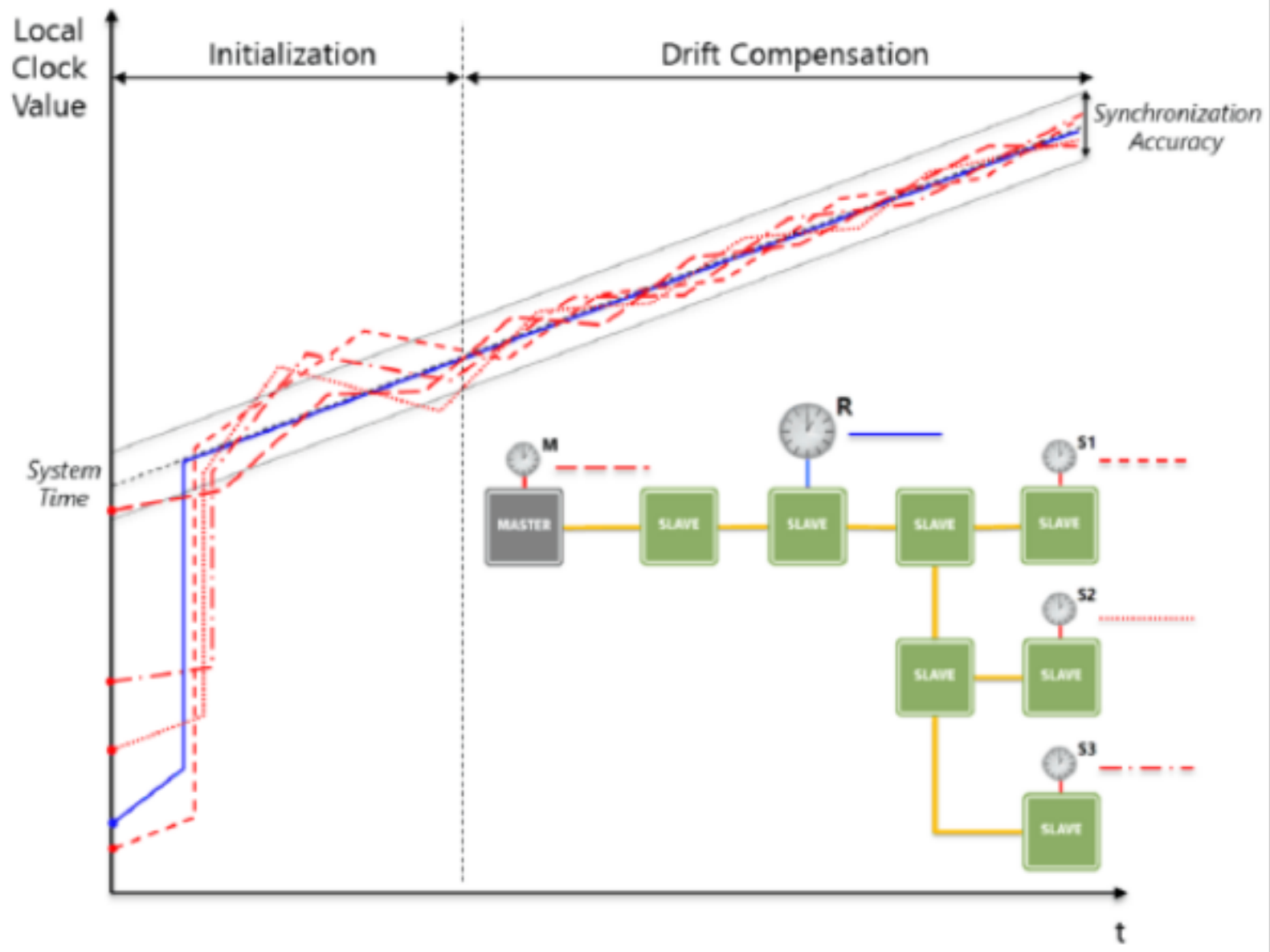
Reference Clock : EtherCAT device holding System Time.

First DC-synchronized slave encountered by frames

DC synchronization relies on ESC hardware features:

- Latch of local time value on each port when triggered by master
- Adjustment of local clock speed based on an external set-point

DC | Operation



© EtherCAT Technology Group

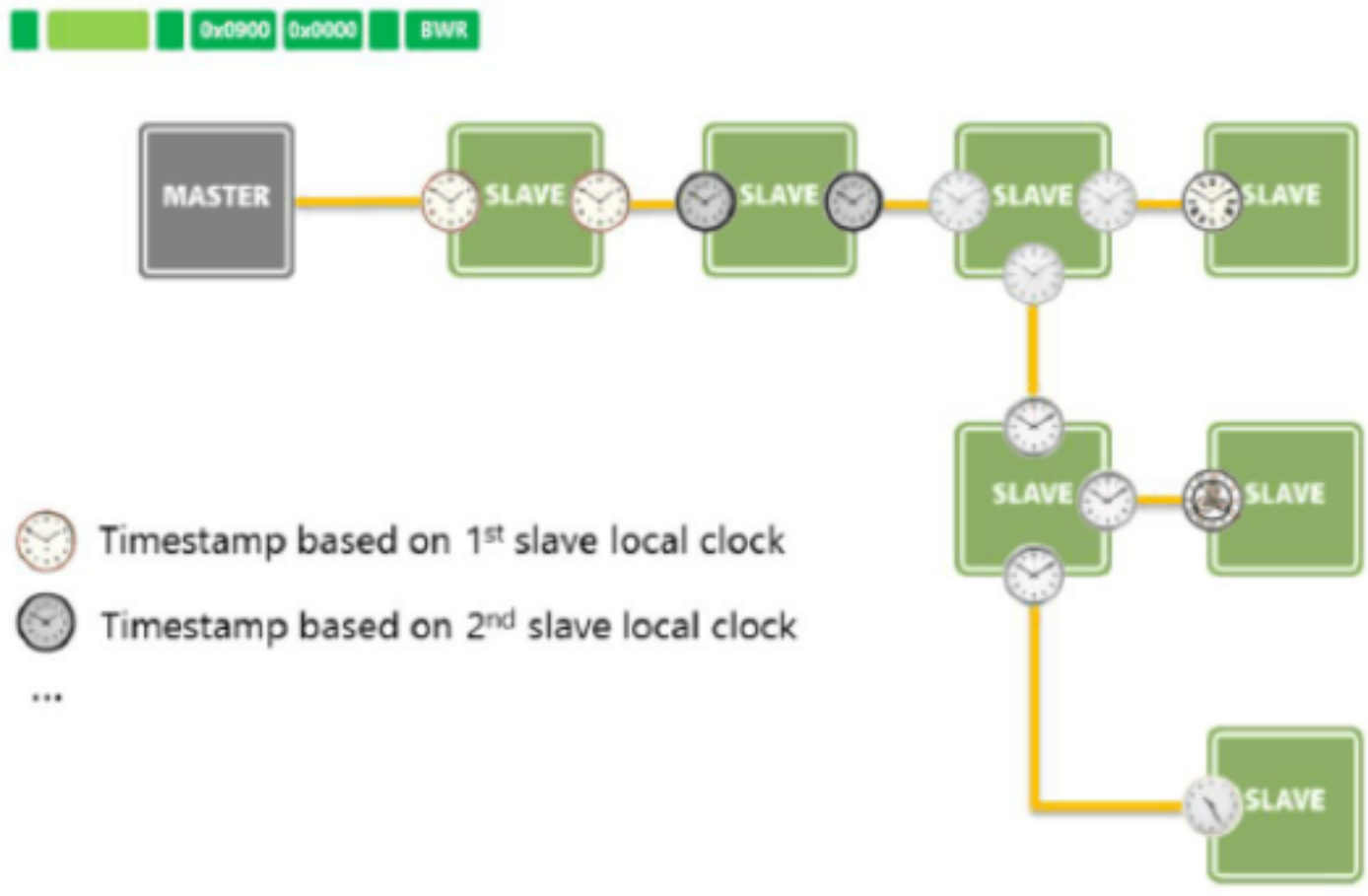
DC operation consists of 2 phases:

Initialization : initially independent clocks are adjusted to the System Time definition

Drift Compensation : following deviations are corrected in order to keep synchronization

DC | Initialization & Drift Compensation

- The master triggers slaves to latch their local time values when a standard command is received on each port.



Latch is activated by write access to $_{ADO}$0x0900

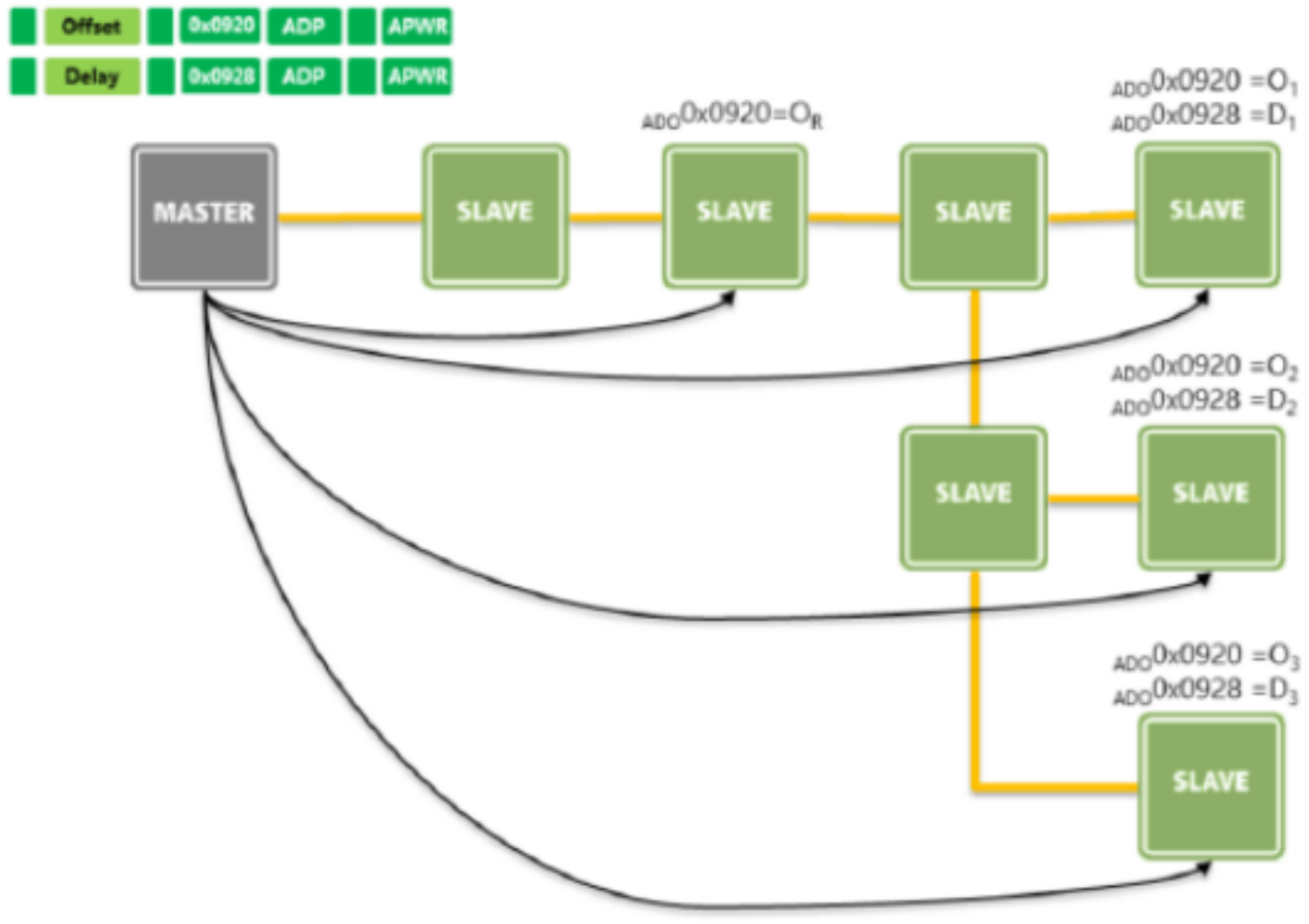
? Local time at port X is latched at SOF (Start of Frame)

? At EOF (End of Frame) latched time at port X is copied into $_{ADO}$(0x0900+4*X)

$_{ADO}$(0x0900+4*X) is in local clock units (at this stage, the local clock in each slave is still uncorrelated from others)

DC | Initialization & Drift Compensation

- The master reads time values latched from each slave, and calculates an offset and a delay for each DC device.



© EtherCAT Technology Group

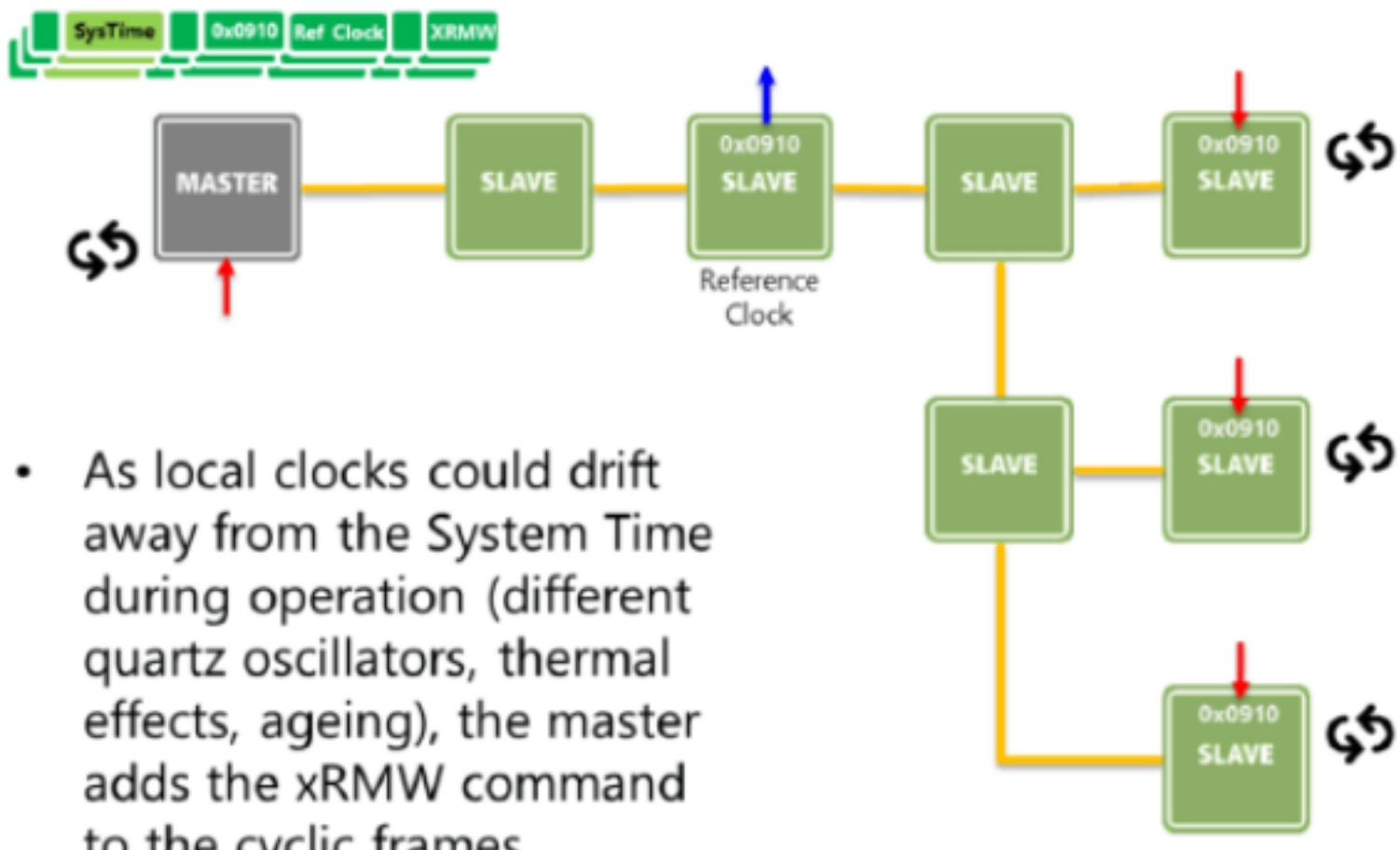
Offset ：absolute difference between the local clock of slave N and the standard definition of System Time

Delay：propagation time of frames between the reference clock and slave N

© EtherCAT Technology Group

DC | Initialization & Drift Compensation

- Residual misalignments between local clocks are by triggering slaves to adjust clock speed multiple times.



- As local clocks could drift away from the System Time during operation (different quartz oscillators, thermal effects, ageing), the master adds the xRMW command to the cyclic frames.

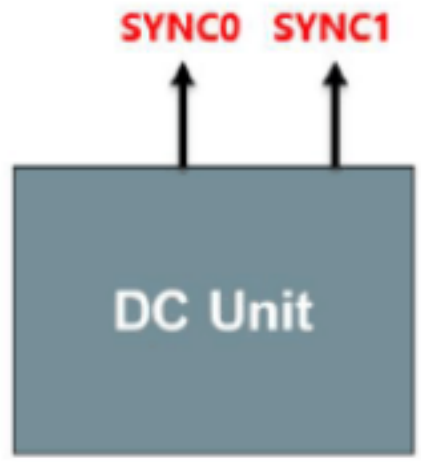
Whenever $_{ADO}$ 0x0910 is written, slave compares received System Time with its local time (corrected with Delay and Offset:

(Local Clock + Offset) − (Received System Time + Delay) > 0 → Local Clock decelerates

(Local Clock + Offset) − (Received System Time + Delay) < 0 → Local Clock accelerates

Sync Events

- Digital outputs for the ESC: are set when the System Time reaches predefined values.

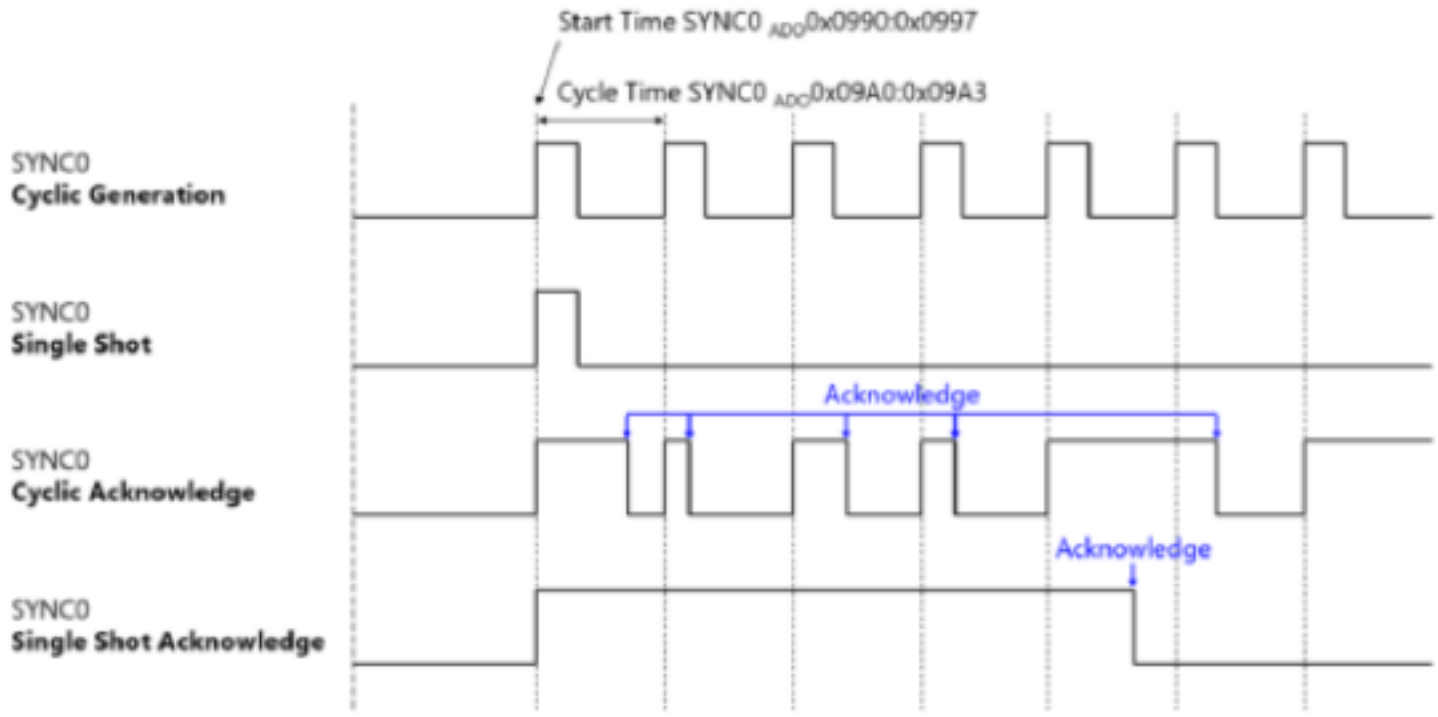


- Can be used as:
  - Direct digital outputs (simple devices)
  - Interrupt sources for µC (complex devices)

© EtherCAT Technology Group

Sync Signal Generation Modes

- Four operation modes are supported for SYNC signals (configured via dedicated registers).

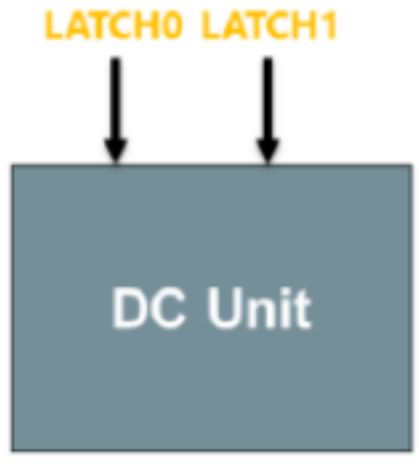


- Second sync event (SYNC1) has a multiple integer period and optionally an offset with respect to SYNC0 event.

© EtherCAT Technology Group

Latch Events

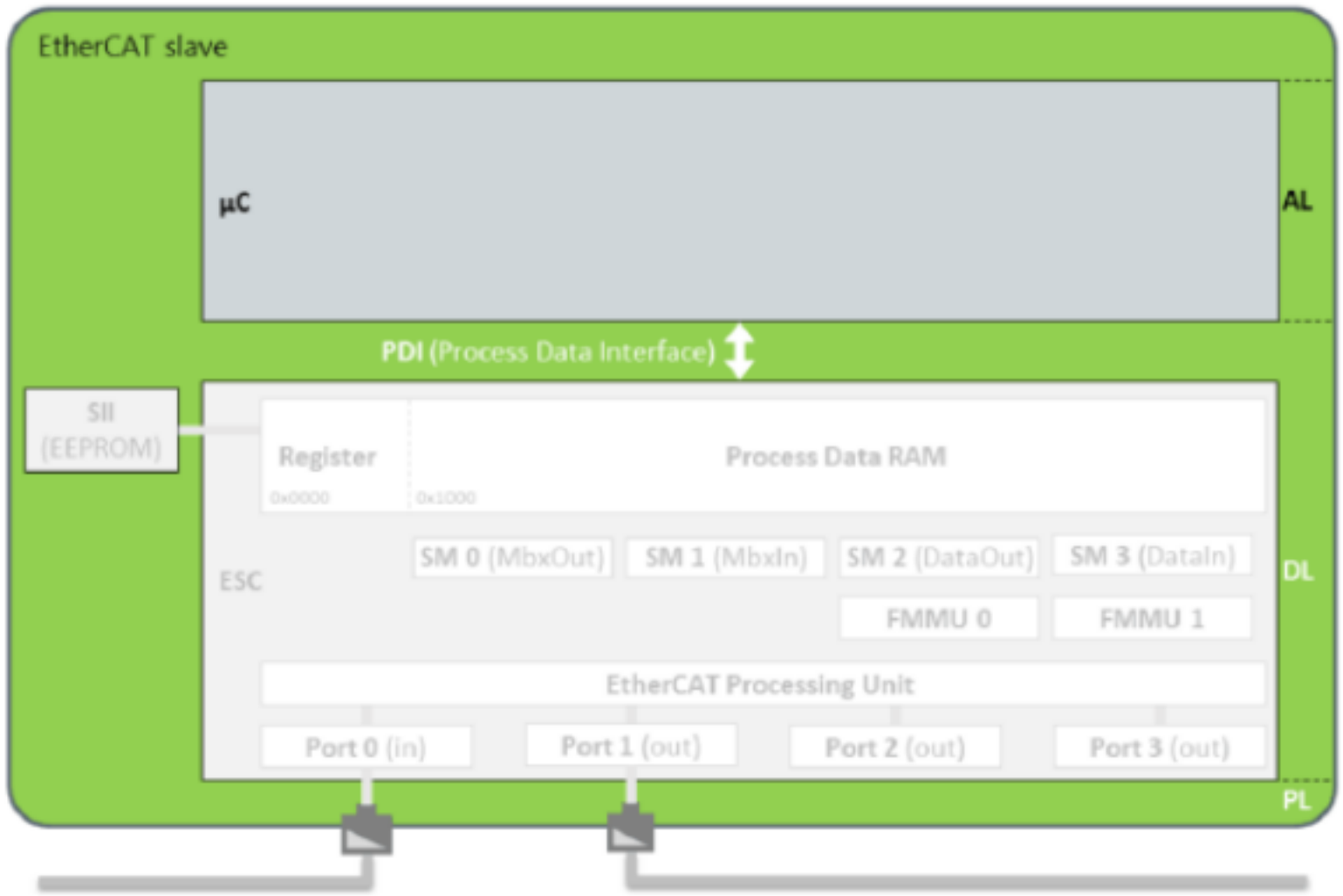
- Digital inputs for the ESC: when a positive and/or negative edge is detected, the corresponding System Time value is stored.



- The *Latch Time* registers ($_{ADO}$0x09B0:0x09CF) contain the stored time stamps
  - Can be cyclically read by the master as process data
- Single or continuous latch configurable

© EtherCAT Technology Group

AL | µC



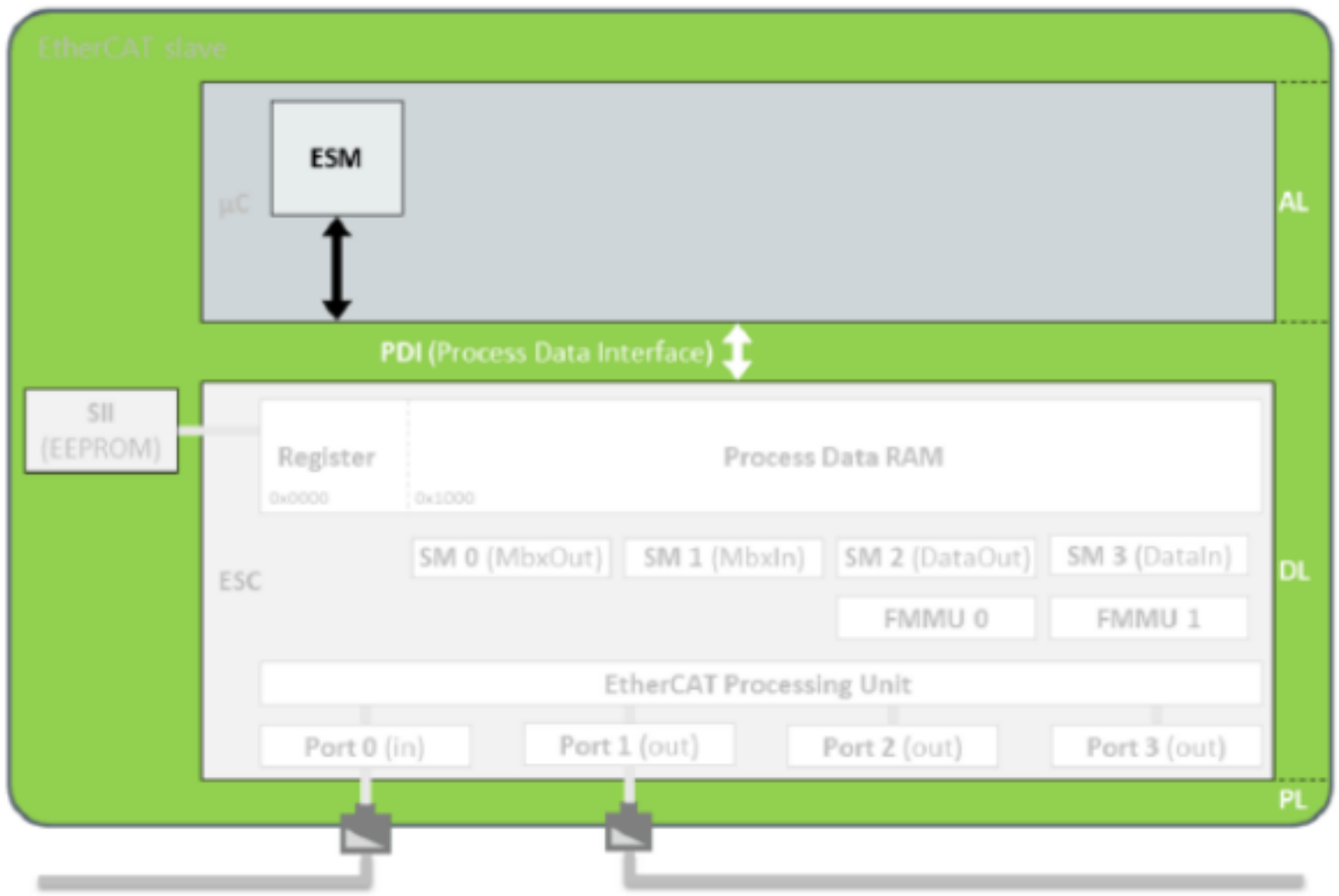
© EtherCAT Technology Group

Purpose of Application Layer (AL)

- EtherCAT State Machine
  - Boot-up of device
  - Error handling
- Mailbox communication
  - Access parameters of an EtherCAT slave
  - Asynchronous transfer
  - Mailbox protocols
    - Ethernet over EtherCAT (EoE)
    - CAN application protocol over EtherCAT (CoE)
    - File transfer over EtherCAT (FoE)
    - Servo Drive over EtherCAT (SoE)
    - ADS over EtherCAT (AoE)
- Process data handling

© EtherCAT Technology Group

AL | EtherCAT State Machine (ESM)

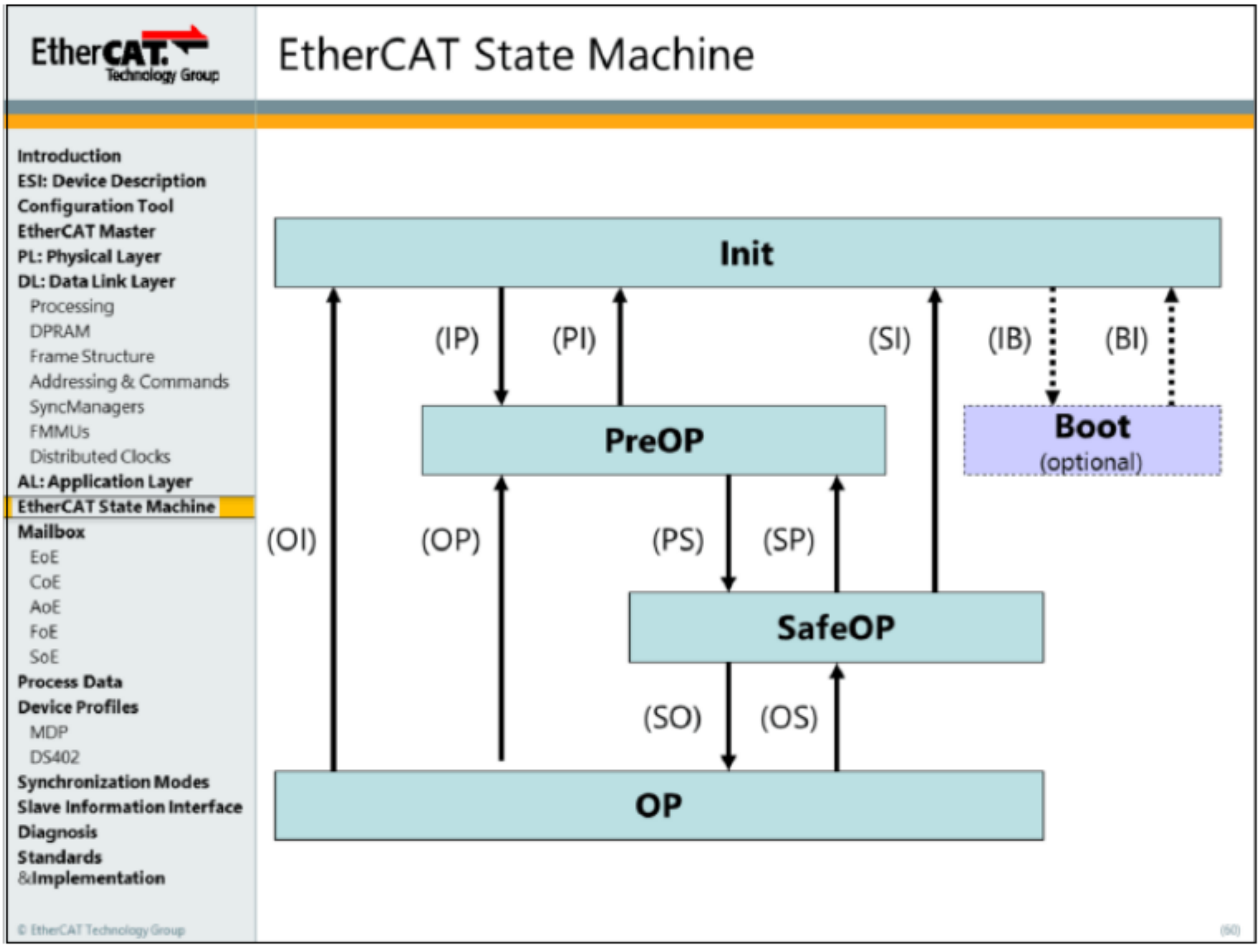


© EtherCAT Technology Group

Purpose of EtherCAT State Machine

- Defines general communication states of EtherCAT slave devices
- Specifies the initialization and error handling of EtherCAT slave devices → Boot-up of the network
- States correspond to the communication relationship between master and slave
- Requested and current state of a slave device are reflected in the AL Control and AL Status registers

© EtherCAT Technology Group



Init = Initialization

PreOP = Pre-Operational

SafeOP = Safe-Operational

OP = Operational

Boot = Bootstrap

State: INIT

- **Init State**
  - No communication on the Application Layer
  - Master has access to the DL-Information registers

- **Transition to PreOP**
  - Master configures register, at least:
    - EtherCAT Fixed Address ($_{ADO}$0x0010)
    - Mailbox SyncManagers
  - Master requested PreOperational state
    - sets AL Control register
    - wait for AL Status register confirmation

© EtherCAT Technology Group

State: PREOP

- **PreOP State**
  - Mailbox communication on the Application Layer
  - No Process Data communication

- **Transition to SafeOP**
  - Master configures application parameters using the Mailbox
    - e.g.: process data mapping, application-specific settings
  - Master configures DL Register
    - Process Data SyncManagers
    - FMMUs
  - Master requested SafeOperational state
    - sets AL Control register
    - wait for AL Status register confirmation

© EtherCAT Technology Group

State: SAFE-OP

- **SafeOP State**
  - Mailbox communication on the Application Layer
  - Process Data communication, but only Inputs are evaluated – Outputs in 'Safe' state

- **Transition to OP**
  - Master sends valid Outputs
  - Master requests Operational state
    - sets AL Control register
    - wait for AL Status register confirmation

© EtherCAT Technology Group

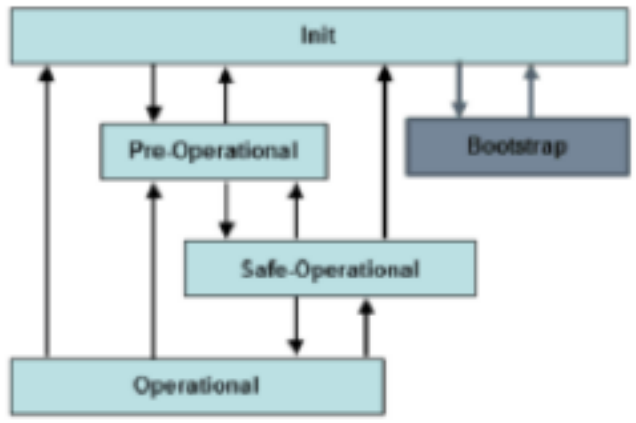
State: OP

- **OP State**
  - Inputs and Outputs are valid

© EtherCAT Technology Group

State: BOOT

- **Boot State**
  - Boot State is optional – but recommended if firmware updates necessary
  - State changes only from and to 'Init'
  - No Process Data communication
  - Communication via Mailbox on Application Layer
  - Special mailbox configuration possible, e.g. larger mailbox size
  - Only FoE protocol (file transfer) available

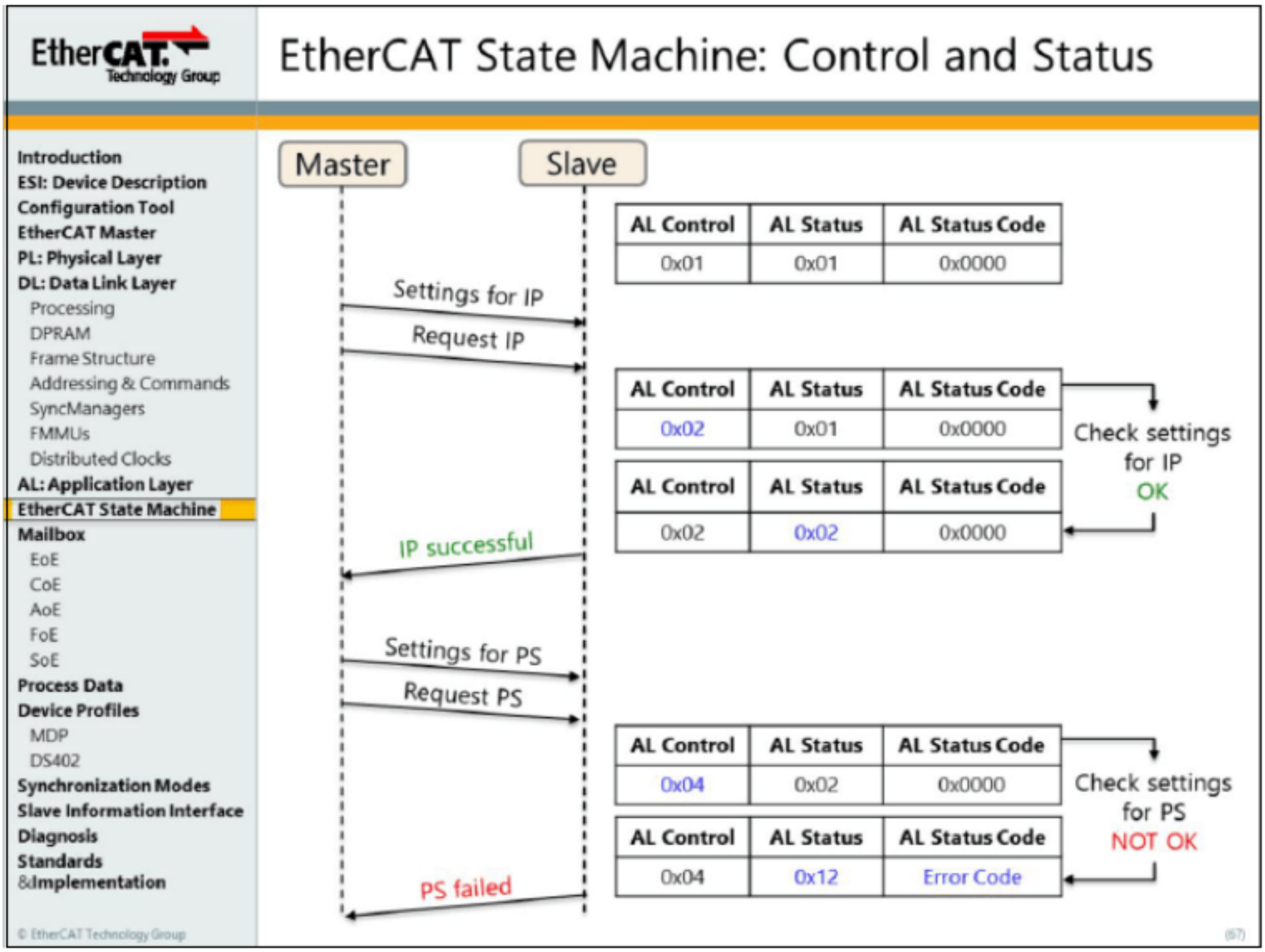


© EtherCAT Technology Group

EtherCAT State Machine: Control and Status

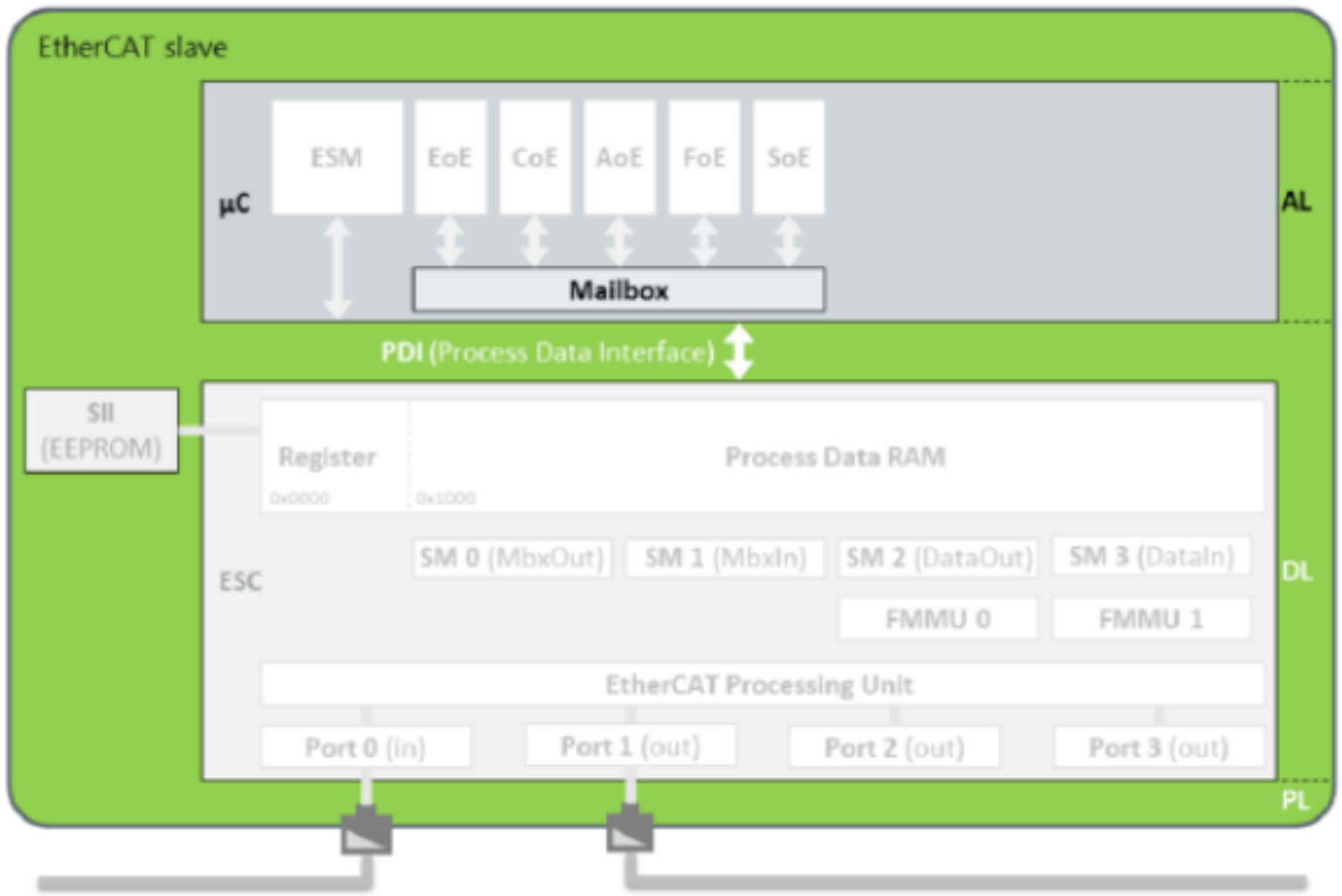
- Registers for EtherCAT State Machine (ESM)
  - **AL Control** ($_{ADO}$0x0120)
    - Master requests a new state to the slave
    - Master acknowledges state machine errors
  - **AL Status** ($_{ADO}$0x0130)
    - Slave reports its current state
    - Slave indicates state machine errors
  - **AL Status Code** ($_{ADO}$0x0134)
    - In case of error (e.g. rejecting) an error code is set

© EtherCAT Technology Group



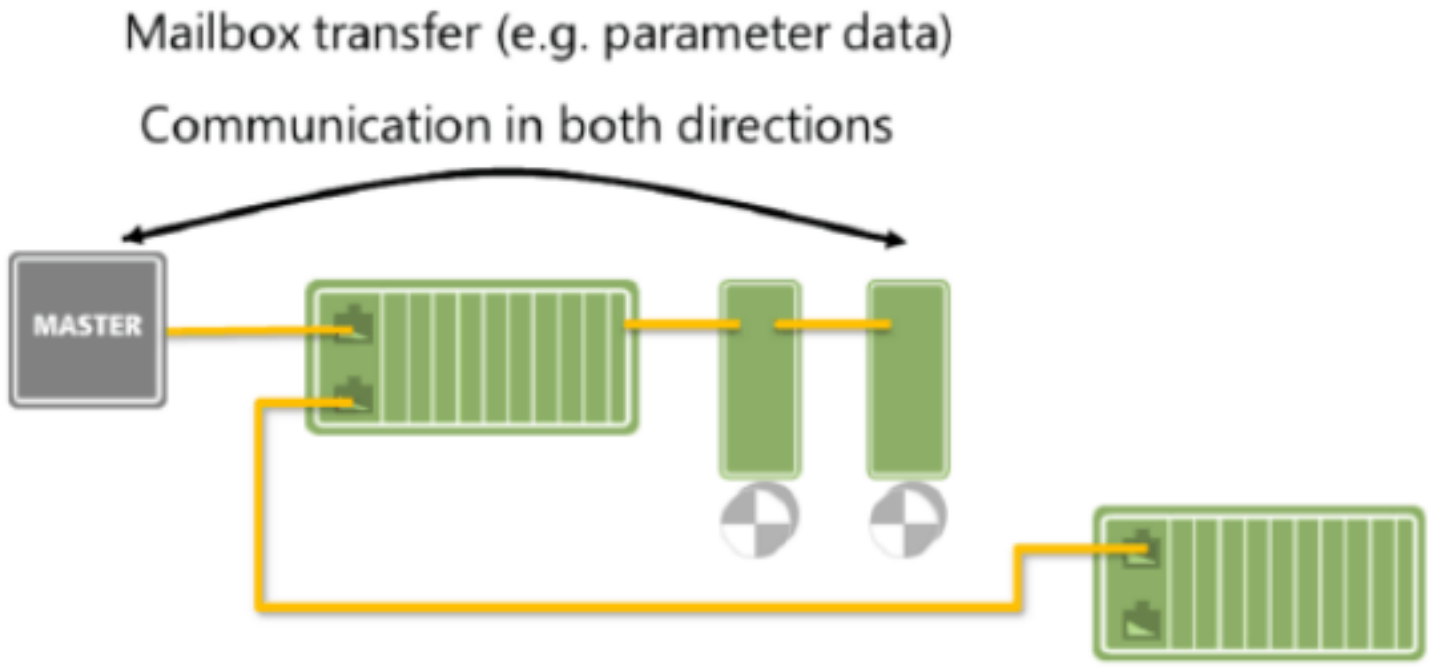
AL Status Codes are specified in ETG.1020.

AL | Mailbox



© EtherCAT Technology Group

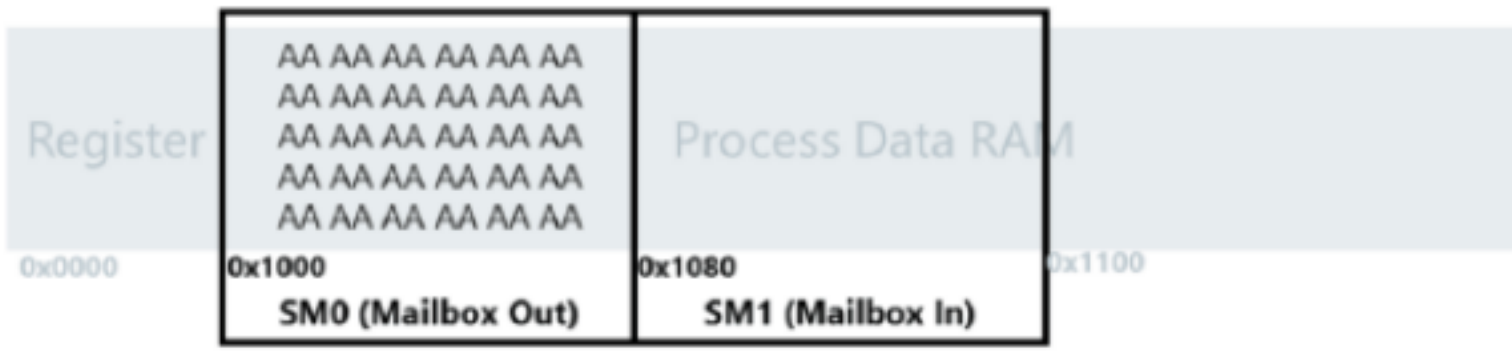
Mailbox Transfer (acyclic communication)

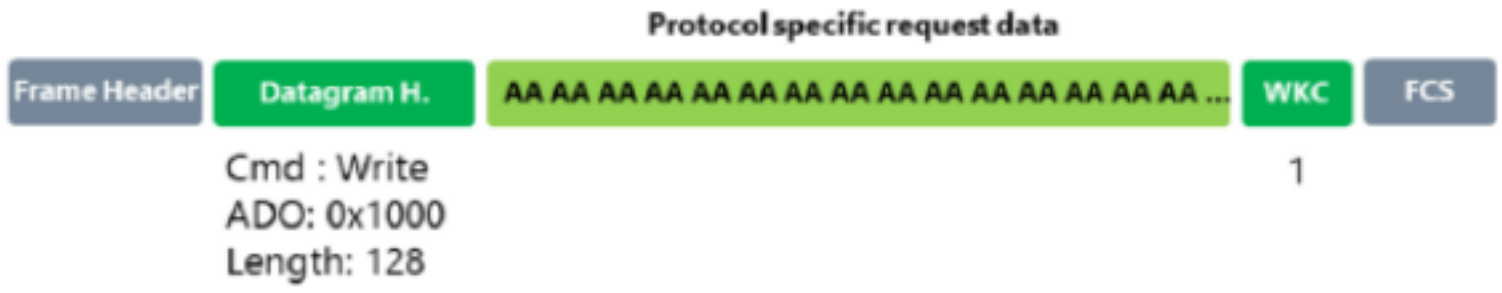


- Available from state PreOperational
- 2 dedicated SyncManagers
  - SM0 (“MBoxOut”): Master to Slave
  - SM1 (“MBoxIn”): Slave to Master
- Multiple protocols defined
- Simple IO-Device (no parameter) → no Mailbox necessary

© EtherCAT Technology Group

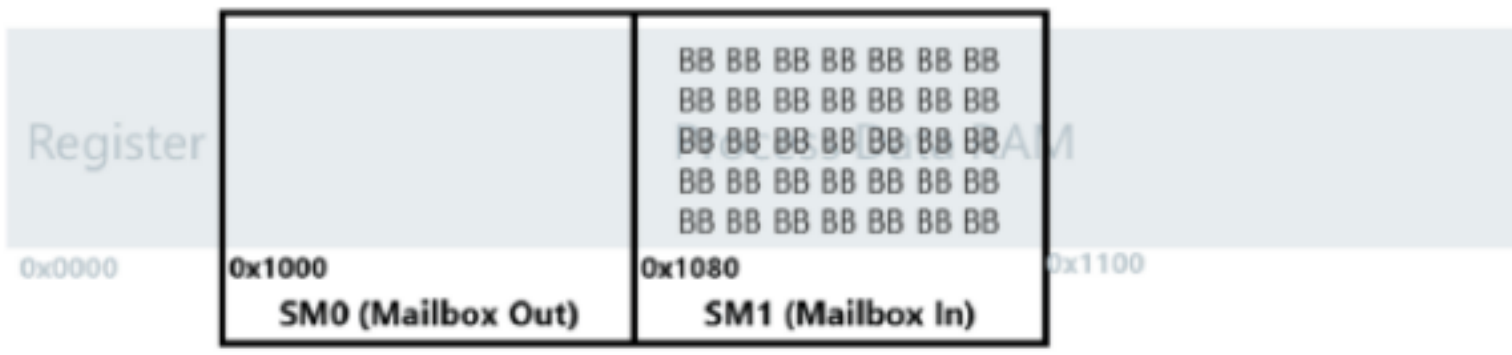
Mailbox Transfer

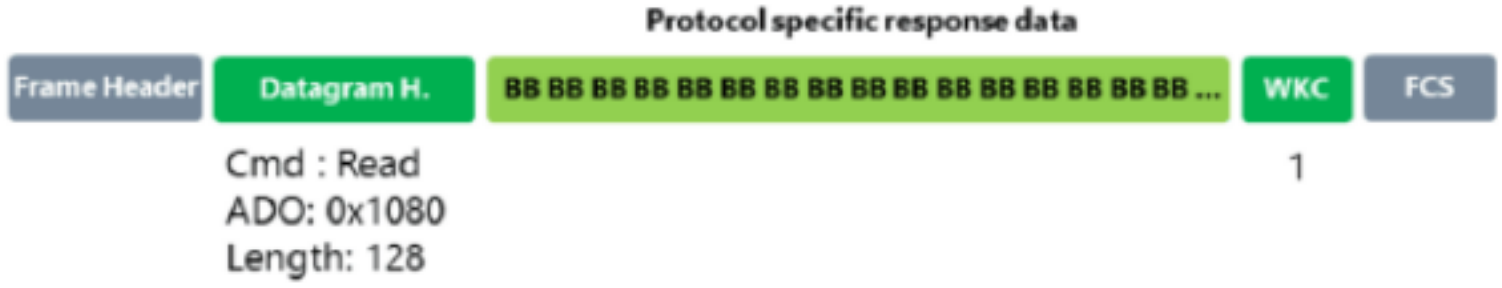




© EtherCAT Technology Group

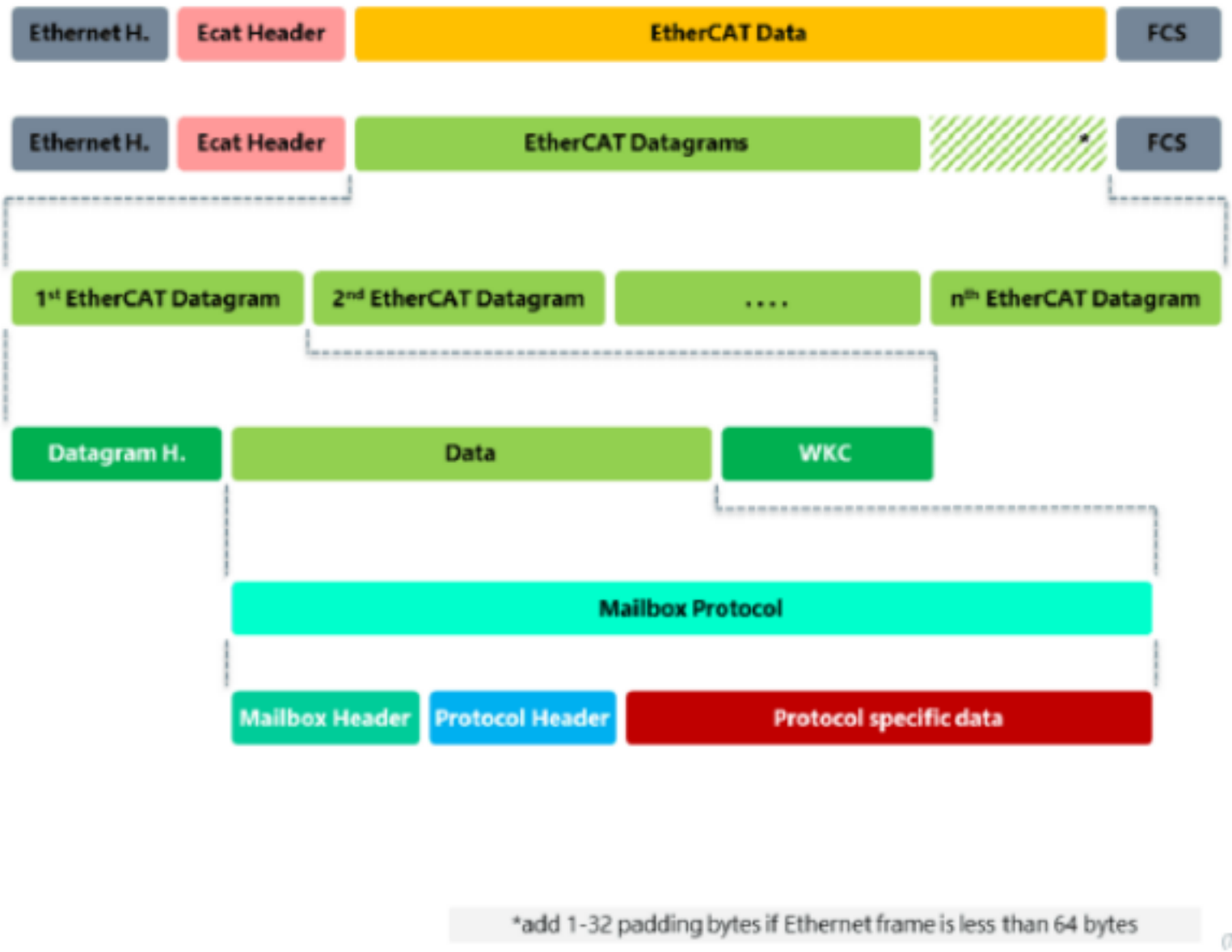
Mailbox Transfer





© EtherCAT Technology Group

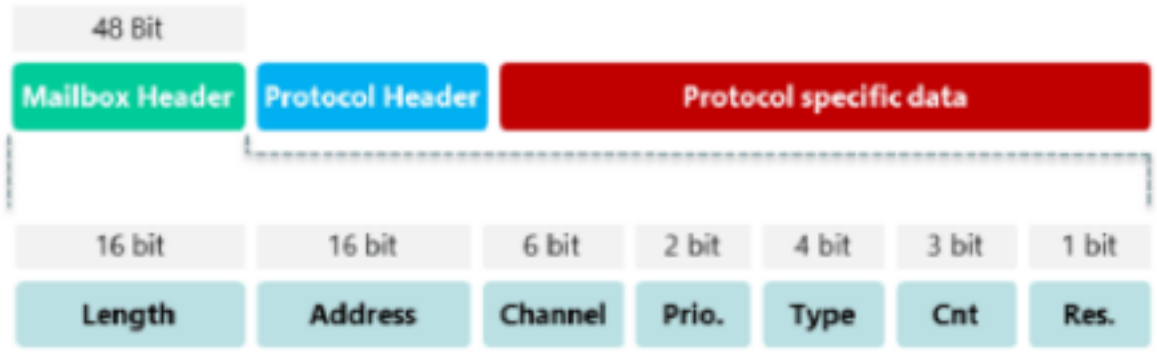
Mailbox Interface Datagram



*add 1-32 padding bytes if Ethernet frame is less than 64 bytes

© EtherCAT Technology Group

Mailbox Header



Length	Length of following data in byte
Address	Fixed address of source slave (S -> M) Fixed address of the destination slave (M->S)
Channel	reserved for future use
Priority	reserved for future use
Type	Mailbox Type, Protocol identifier for following data 0 Mailbox Error 1 AoE (ADS over EtherCAT) 2 EoE (Ethernet over EtherCAT) 3 CoE (CAN application protocol over EtherCAT) 4 FoE (File Access over EtherCAT) 5 SoE (Servo Drive over EtherCAT) 15 VoE (Vendor specific profile over EtherCAT)
Cnt	Sequence number Iterates from 1-7 with every new mailbox service

© EtherCAT Technology Group

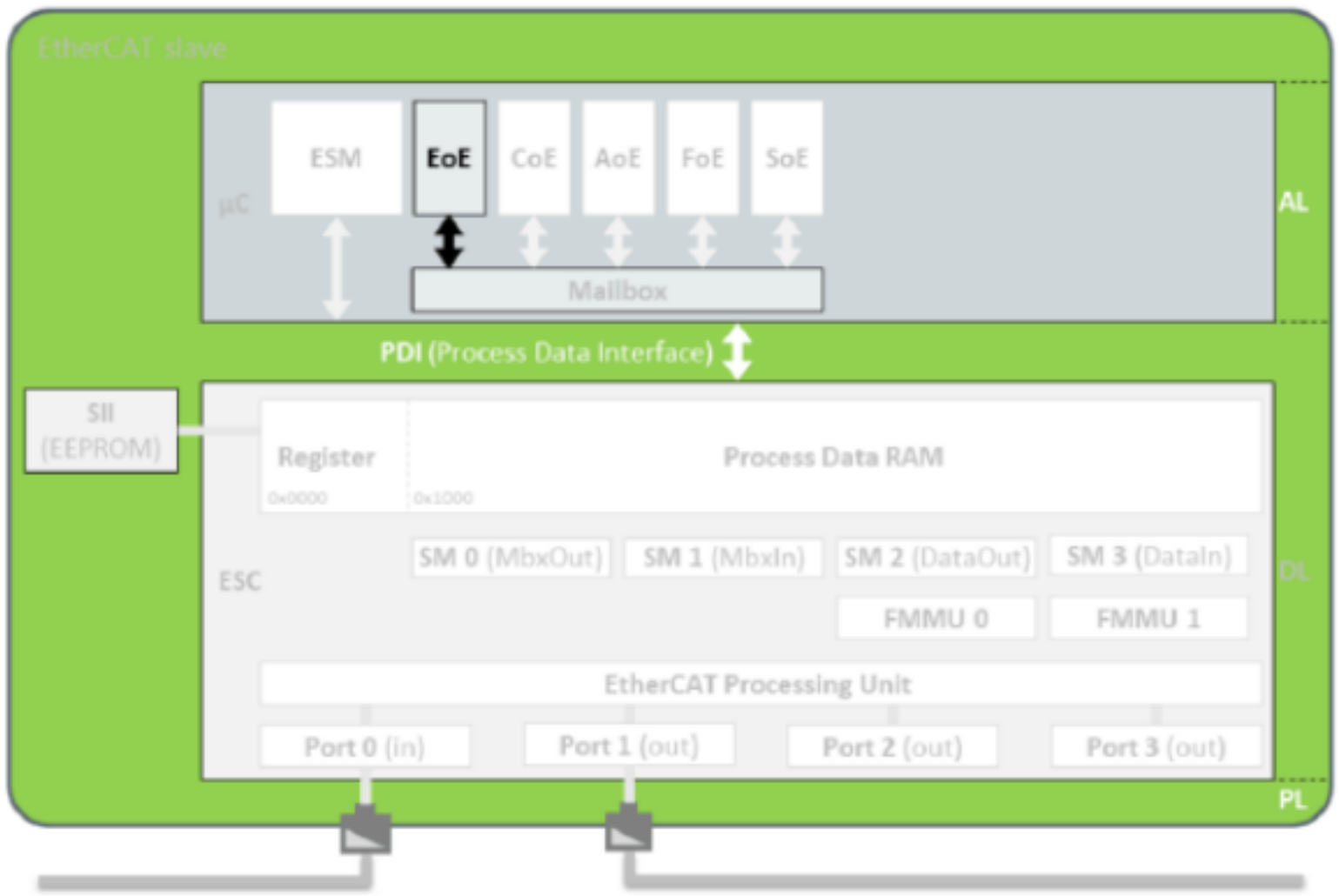
Mailbox Protocol Types

- Ethernet over EtherCAT (**EoE**)
  - Tunnels standard Ethernet communication (e.g. TCP/IP) over EtherCAT
- CAN application protocol over EtherCAT (**CoE**)
  - Access of a CANopen® object dictionary
- ADS over EtherCAT (**AoE**)
  - Routes data to subordinated or cascaded systems
- File Access over EtherCAT (**FoE**)
  - Download and upload files (e.g. firmware download)
- Servo Drive over EtherCAT (**SoE**)
  - Access the Servo Profile Identifier (IDN)
- Vendor specific Profile over EtherCAT (**VoE**)
  - Vendor specific protocol tunneled over EtherCAT

© EtherCAT Technology Group

* CANopen is a trademark of the CAN in Automation e.V.

AL | EoE



© EtherCAT Technology Group

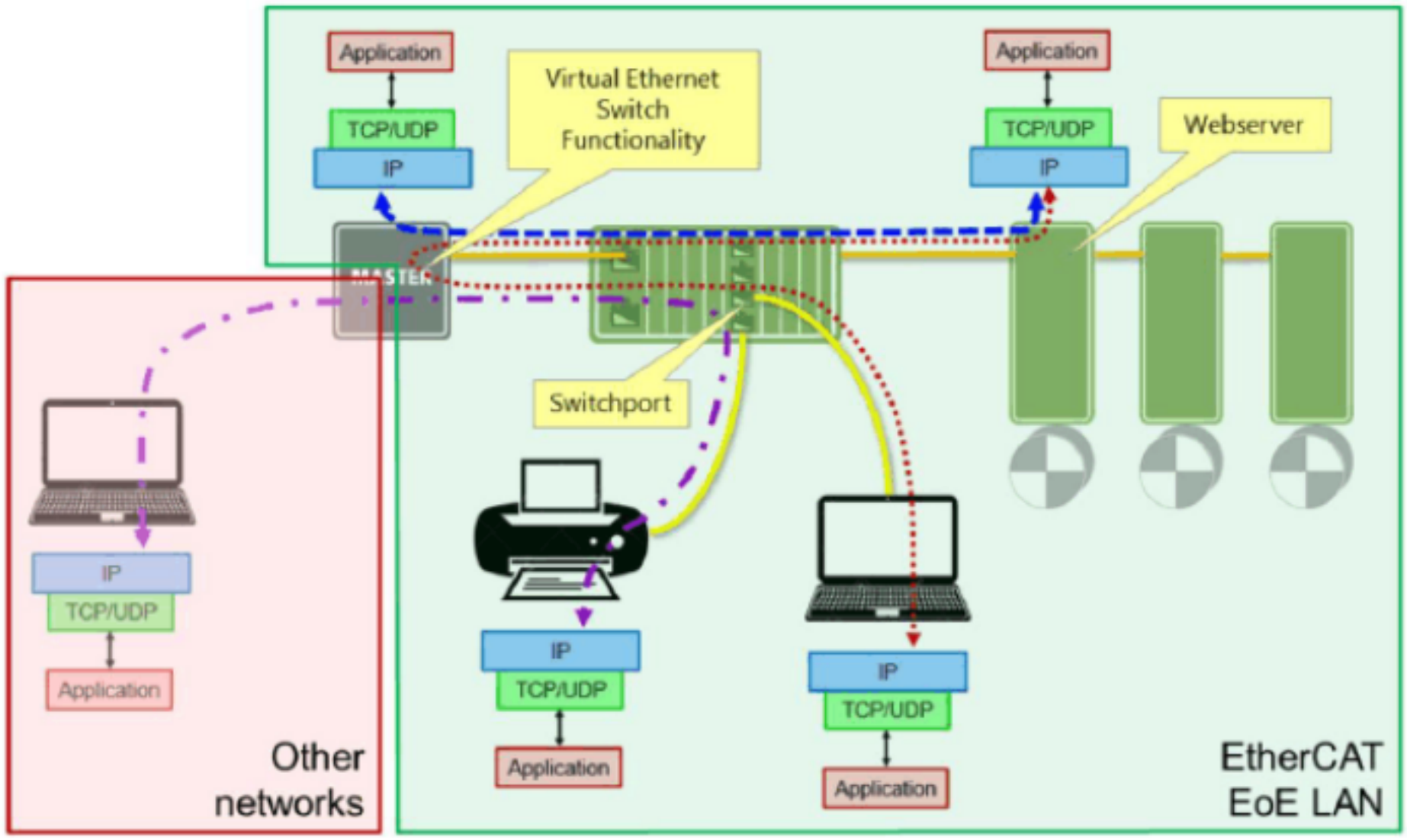
Ethernet over EtherCAT (EoE)

- Tunnels Ethernet communication over EtherCAT
  - Tunneling allows the master to optimize Ethernet communication without affecting the process data exchange.
- Two types of devices supporting EoE
  - Devices with locally running TCP/IP-based applications (e.g. Web Server)
  - Infrastructure devices (Switchports) providing connection for external TCP/IP-based devices.
- EtherCAT master supporting EoE behaves as 'Virtual Ethernet Switch' (Layer 2)
  - If communication with other LAN networks shall be possible, IP routing in OS/Master needed.

© EtherCAT Technology Group

EoE Architecture

- TCP/IP end points are connected transparently

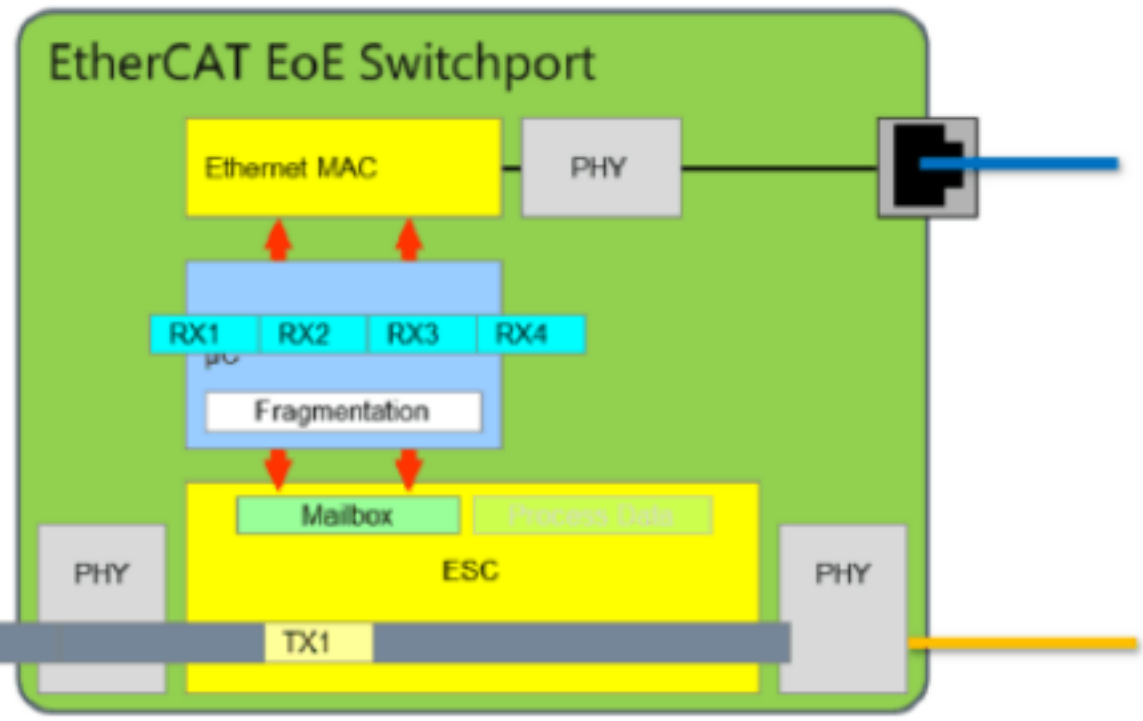


© EtherCAT Technology Group

TCP/IP traffic can be routed both within the EtherCAT EoE LAN and to/from other networks (in this case, IP Routing in master device is needed).

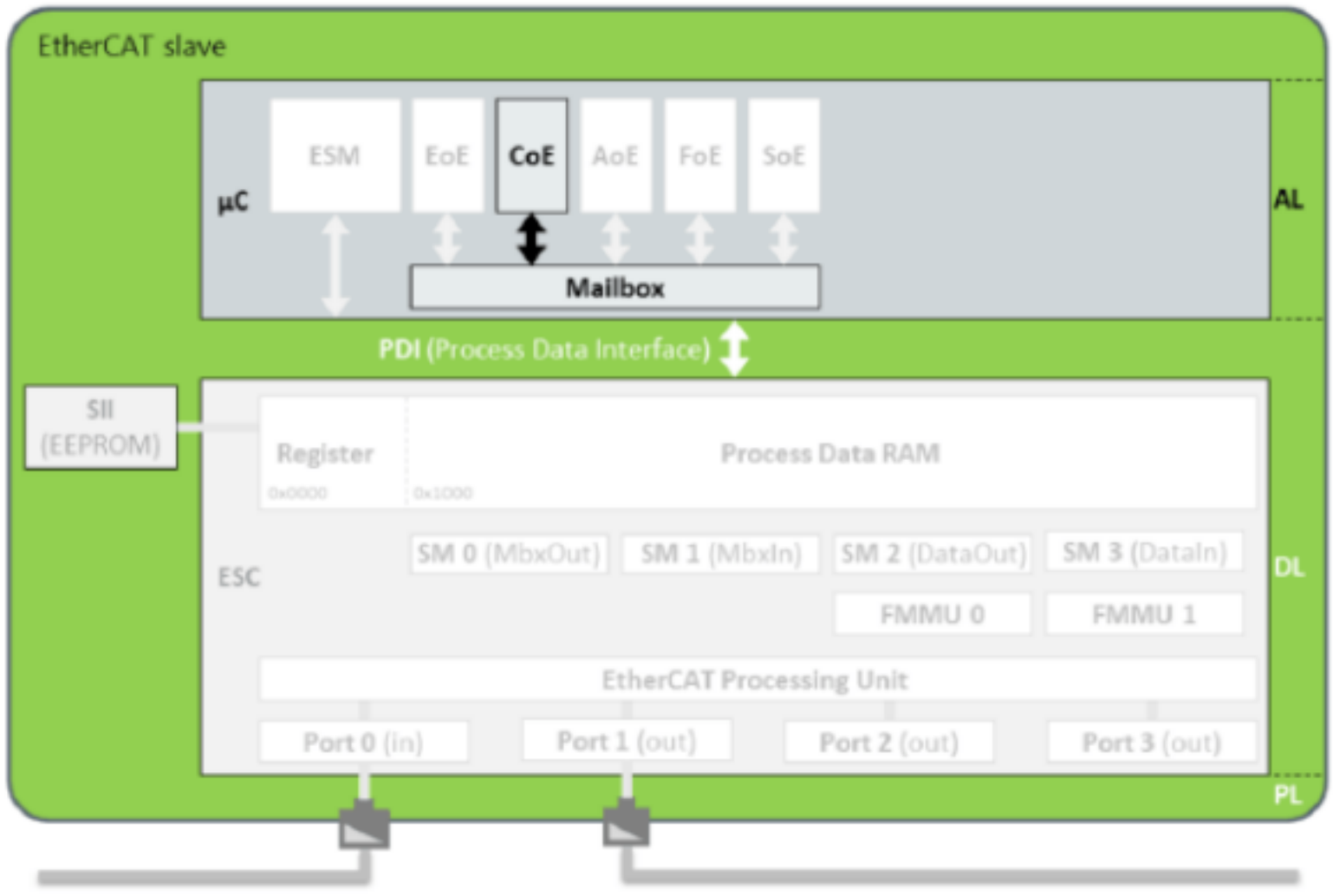
EoE Switchport: Any Ethernet Protocol

- Interface to any Ethernet Device or Network
- Ethernet Frames are inserted into EtherCAT Protocol:
  - 'Ethernet over EtherCAT'



© EtherCAT Technology Group

AL | CoE



© EtherCAT Technology Group

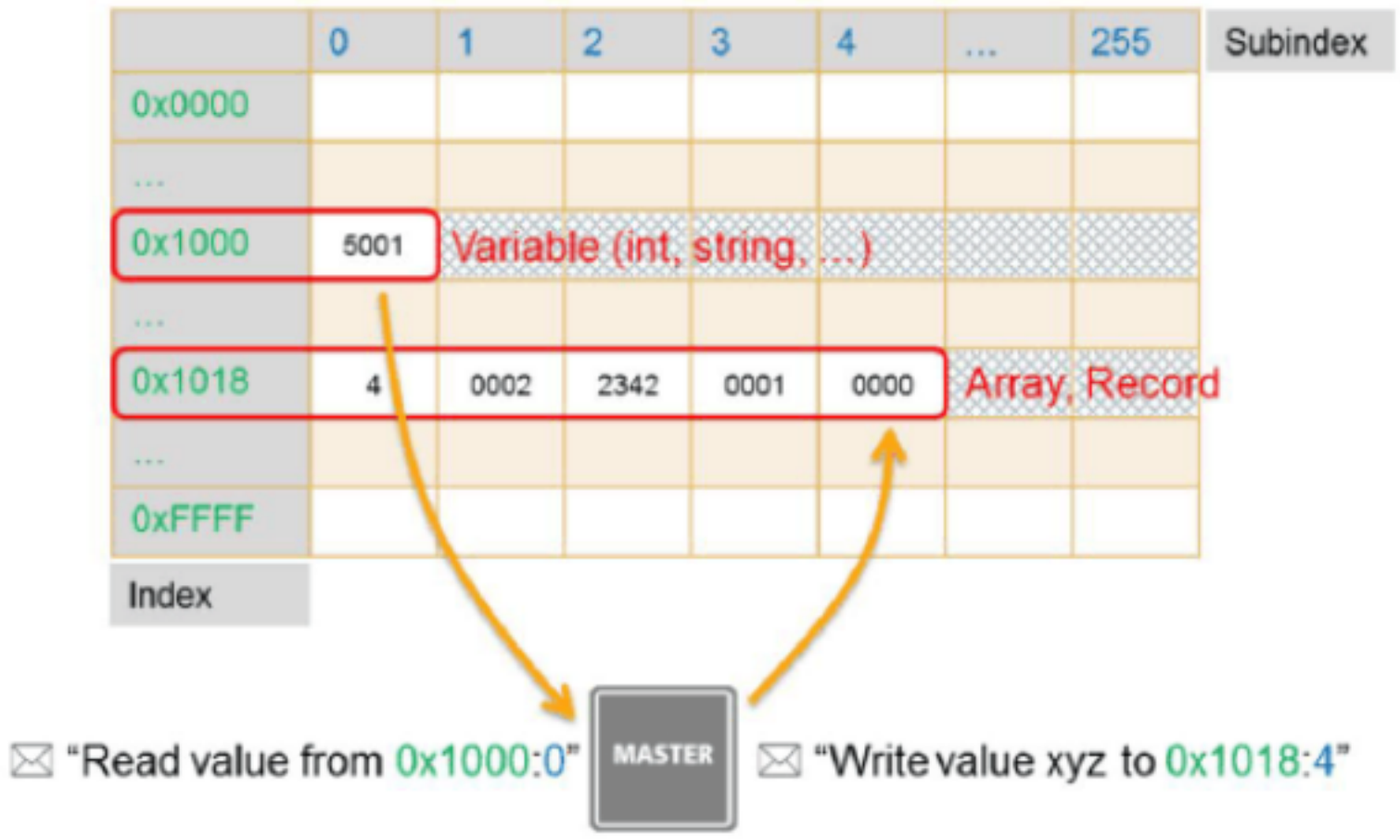
Recommended protocol for service data access

- ? Configuration of communication parameter
- ? Configuration of device specific parameter

Easy migration path from CANopen? Devices to EtherCAT device supporting CoE protocol Stacks can be re-used

Object Dictionary (OD)

- Table of variables, addressed by *Index* and *Subindex*
- Part of the Application Layer, i.e. part of stack



© EtherCAT Technology Group

Object Dictionary (OD)

	0	1	2	3	4	…	255	Subindex
0x0000								
…								
0x1008	'Slave'							
…								
0x1018	4	0002	2342	0001	0000			
…								
0xFFFF								
Index								

5 value

Does not exist

"Object"

"Object entry"

© EtherCAT Technology Group

Object Dictionary Structure

Specified areas within the object dictionary

Index	Use
0x0000 – 0x0FFF	Data type area Examples: 0x0001 – Boolean 0x0008 – Real32 …
0x1000 – 0x1FFF	Communication area Examples: 0x1008 – Device Name 0x1018 – Identity Object (Vendor ID, S/N, …) …
0x2000 – 0x5FFF	Manufacturer specific area Examples: Parameters and process data of the device …
0x6000 – 0xFFFF	Profile specific area Examples: Parameters and process data of the device, specified in a profile (e.g. servo drive). …

© EtherCAT Technology Group

Object Dictionary in ESI

OD is described in element Profile of ESI file.

Profile
ChannelInfo (3)
Dictionary
DataTypes
Objects
Object (23)

	Index	Name	Type	BitSize	Info	Flags
1	#x1000	Device type	UDINT	32	Info	Flags
2	#x1001	Error register	USINT	8	Info	Flags
3	#x1008	Device name	STRING(51)	408	Info	Flags
4	#x1009	Hardware version	STRING(4)	32	Info	Flags
5	#x100A	Software version	STRING(4)	32	Info	Flags
6	#x1018	Identity	DT1018	144	Info	
7	#x10F1	Error Settings	DT10F1	64	Info	
8	#x10F8	Timestamp Object	ULINT	64		Flags
9	#x1601	DO RxPDO-Map	DT1601	304	Info	
10	#x1802	TxPDO Parameter	DT1802	24	Info	
11	#x1A00	DI TxPDO-Map	DT1A00	304	Info	
12	#x1A02	AI TxPDO-Map	DT1A02	272	Info	
13	#x1C00	Sync manager type	DT1C00	48	Info	
14	#x1C12	RxPDO assign	DT1C12	32	Info	
15	#x1C13	TxPDO assign	DT1C13	48	Info	
16	#x1C32	SM output parameter	DT1C32	488	Info	
17	#x1C33	SM input parameter	DT1C33	488	Info	
18	#x6000	DI Inputs	DT6000	24	Info	
19	#x6020	AI Inputs	DT6020	48	Info	
20	#x7010	DO Outputs	DT7010	24	Info	
21	#x8020	AI Settings	DT8020	112	Info	
22	#xF000	Modular Device Profile	DTF000	48	Info	
23	#xF010	Module Profile List	DTF010	112	Info	

→ Altova XMLSpy® 2013

© EtherCAT Technology Group

SDO & PDO

Service **D**ata **O**bject

- Access to an Object acyclically using Mailbox services
- Used for parameters, diagnostics, …

Process **D**ata **O**bject

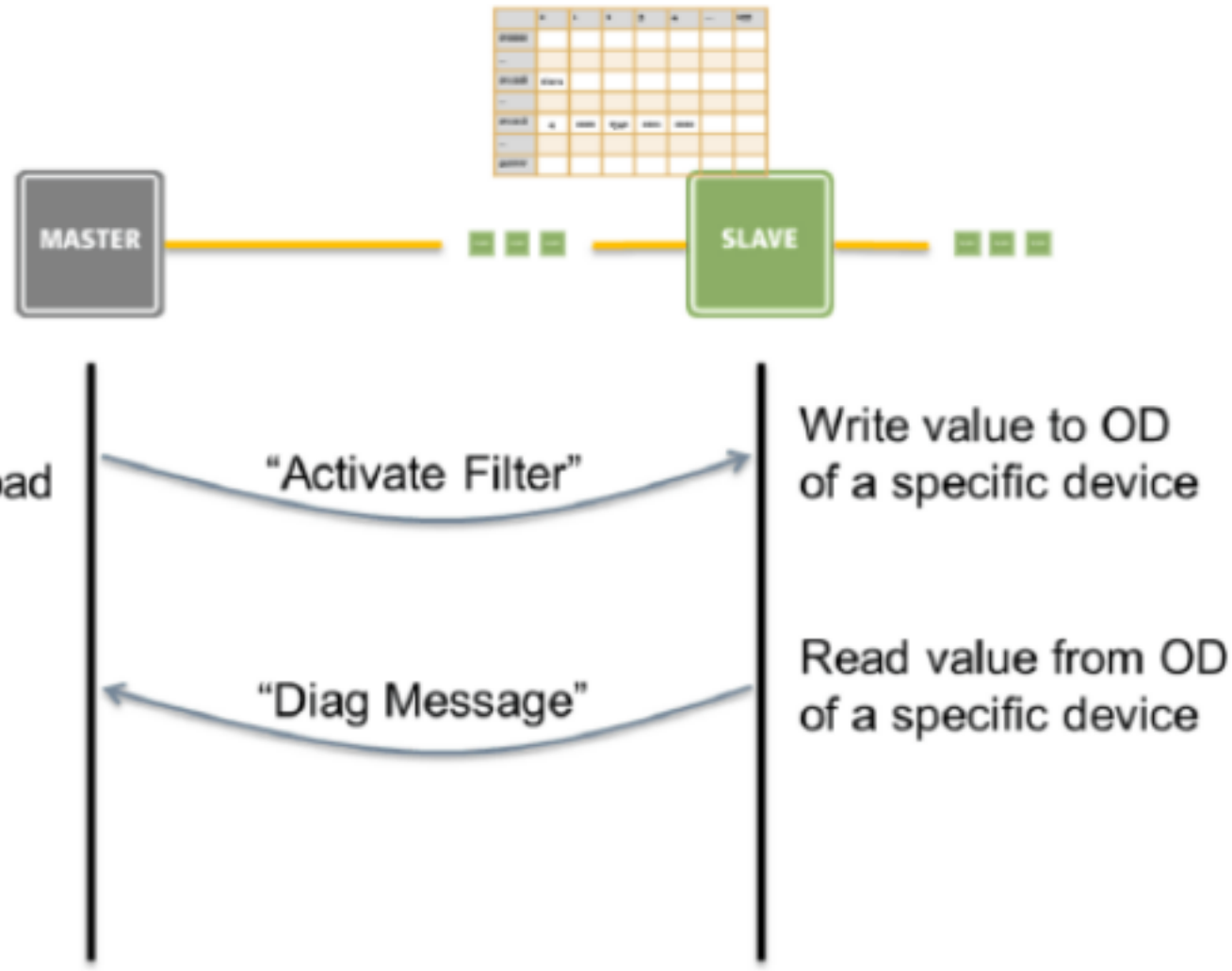
- CANopen®: Access to objects cyclically as process data, but EtherCAT does not use Mailbox communication for process data

→ PDO: "application data package, intended to be transmitted cyclically as process data"

© EtherCAT Technology Group

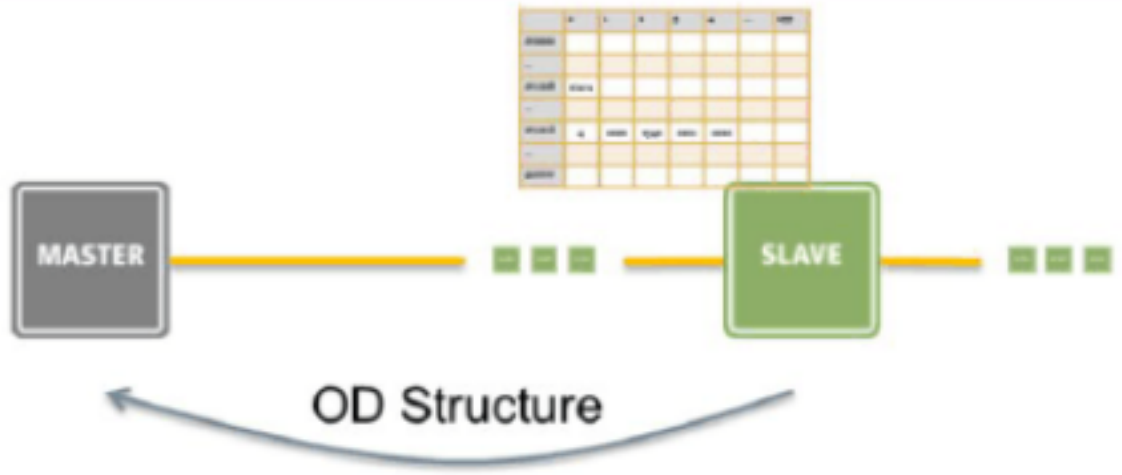
*) CANopen® is a registered trademark of the CAN in Automation e.V., Erlangen, Germany

CoE Services: SDO Down-/Upload



© EtherCAT Technology Group

CoE Services: SDO Information



Information about OD
- Available objects (list of indexes)

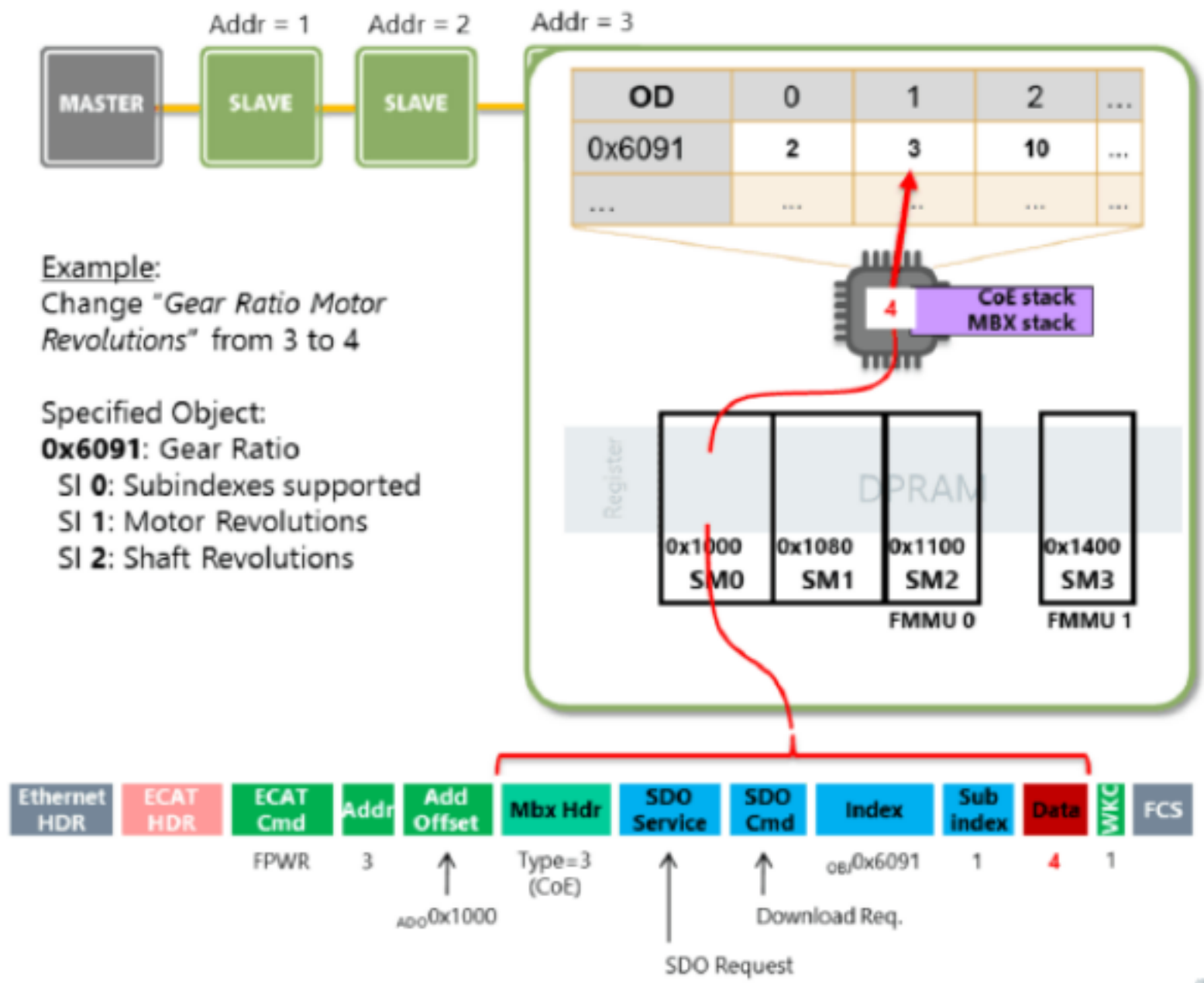
Description of an Object
- Name
- Code (Variable, Array, Record)

Description of an Object Entry
- Name
- Access rights
- …

© EtherCAT Technology Group

Example: SDO Download (Request)

© EtherCAT Technology Group

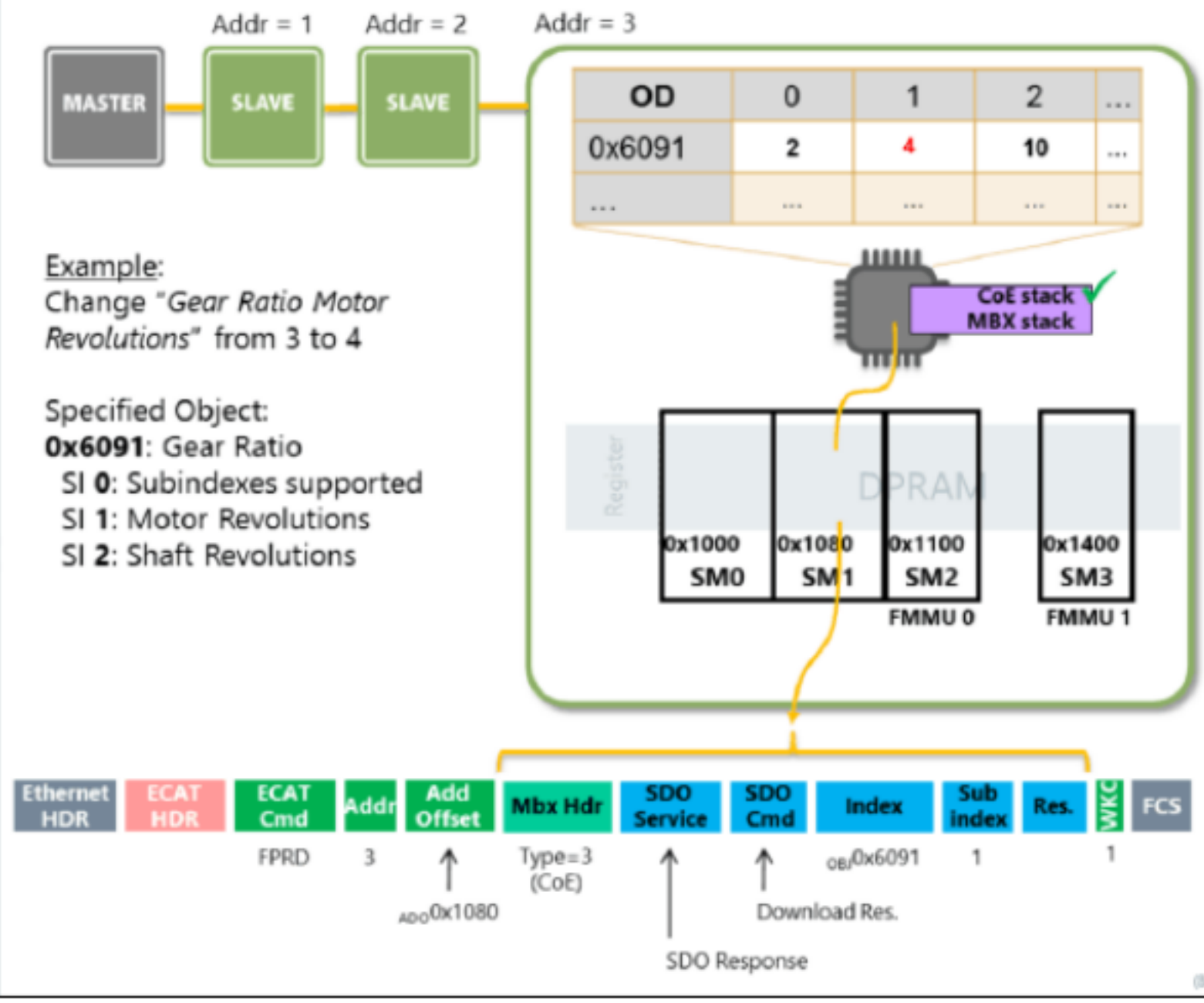


Example:
Change *"Gear Ratio Motor Revolutions"* from 3 to **4**

Specified Object:
0x6091: Gear Ratio
SI **0**: Subindexes supported
SI **1**: Motor Revolutions
SI **2**: Shaft Revolutions

Example: SDO Download (Response)

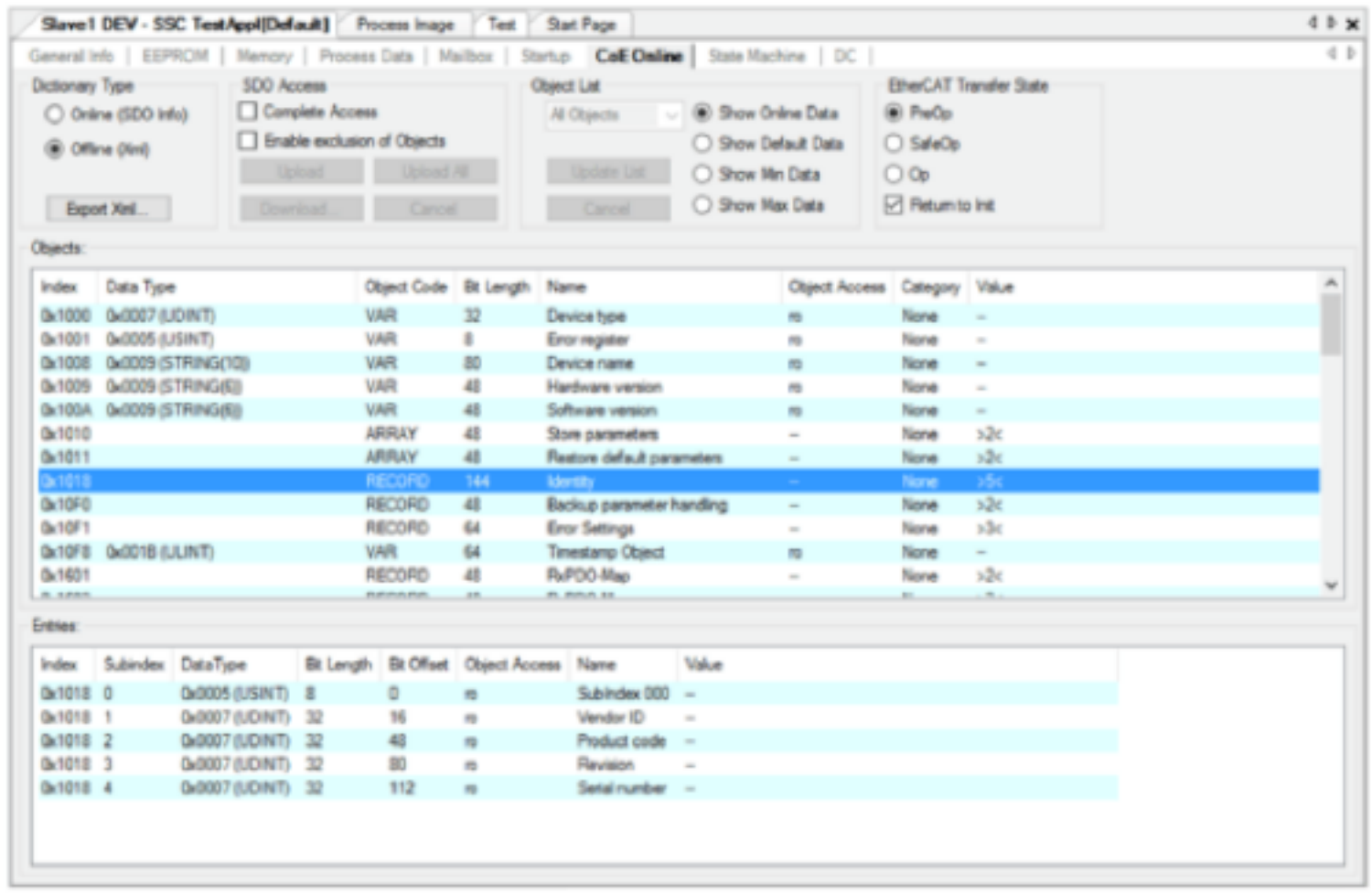
© EtherCAT Technology Group



Example:
Change *"Gear Ratio Motor Revolutions"* from 3 to 4

Specified Object:
0x6091: Gear Ratio
SI **0**: Subindexes supported
SI **1**: Motor Revolutions
SI **2**: Shaft Revolutions

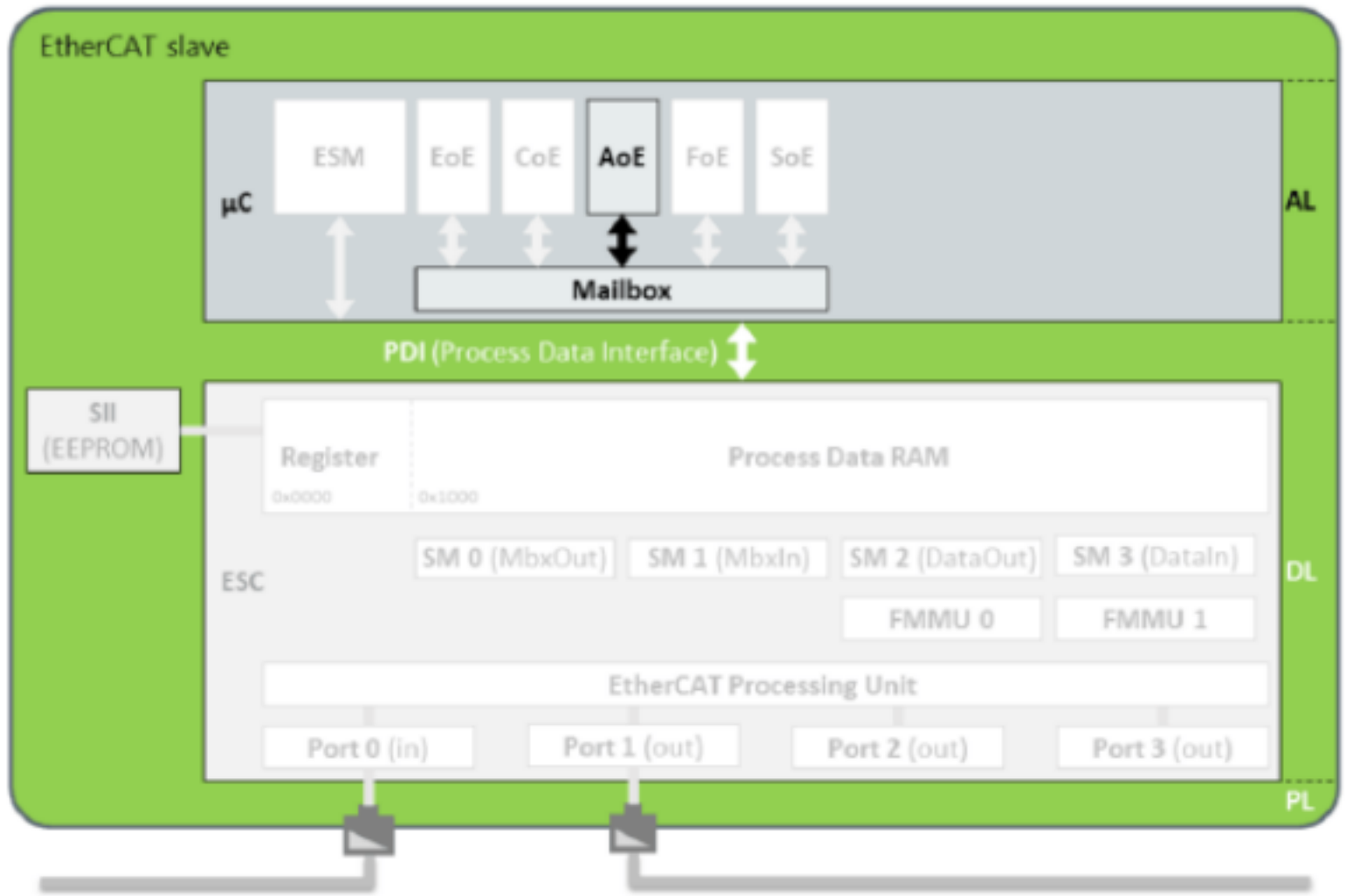
CoE | Object Dictionary (example)



→ Conformance Test Tool 2.1.34.0

© EtherCAT Technology Group

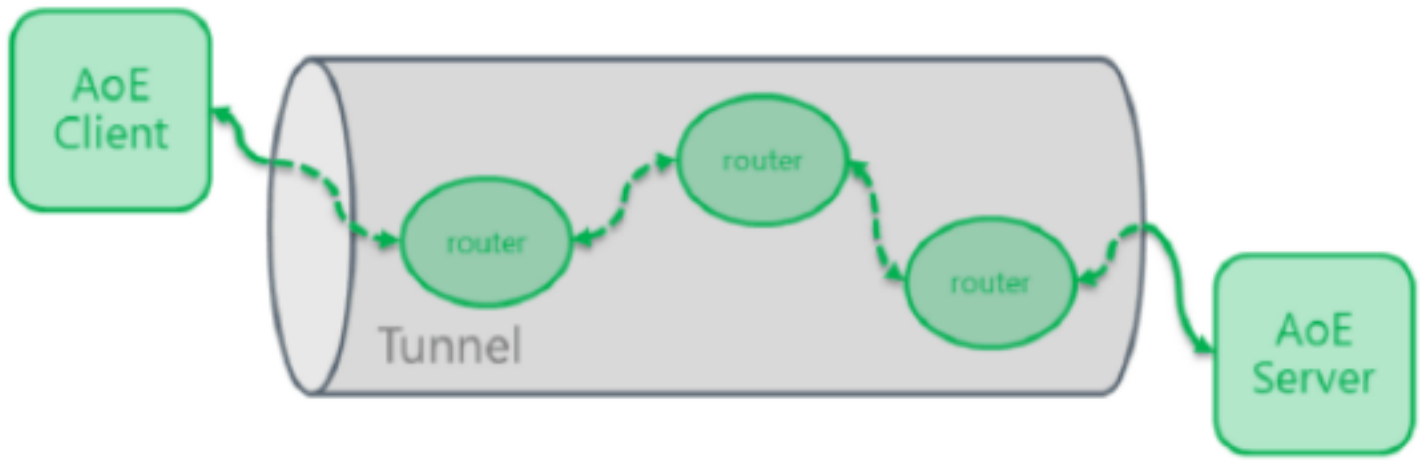
AL | AoE



© EtherCAT Technology Group

AoE | Client-Server Relationship

AoE (ADS over EtherCAT) is a client-server protocol:



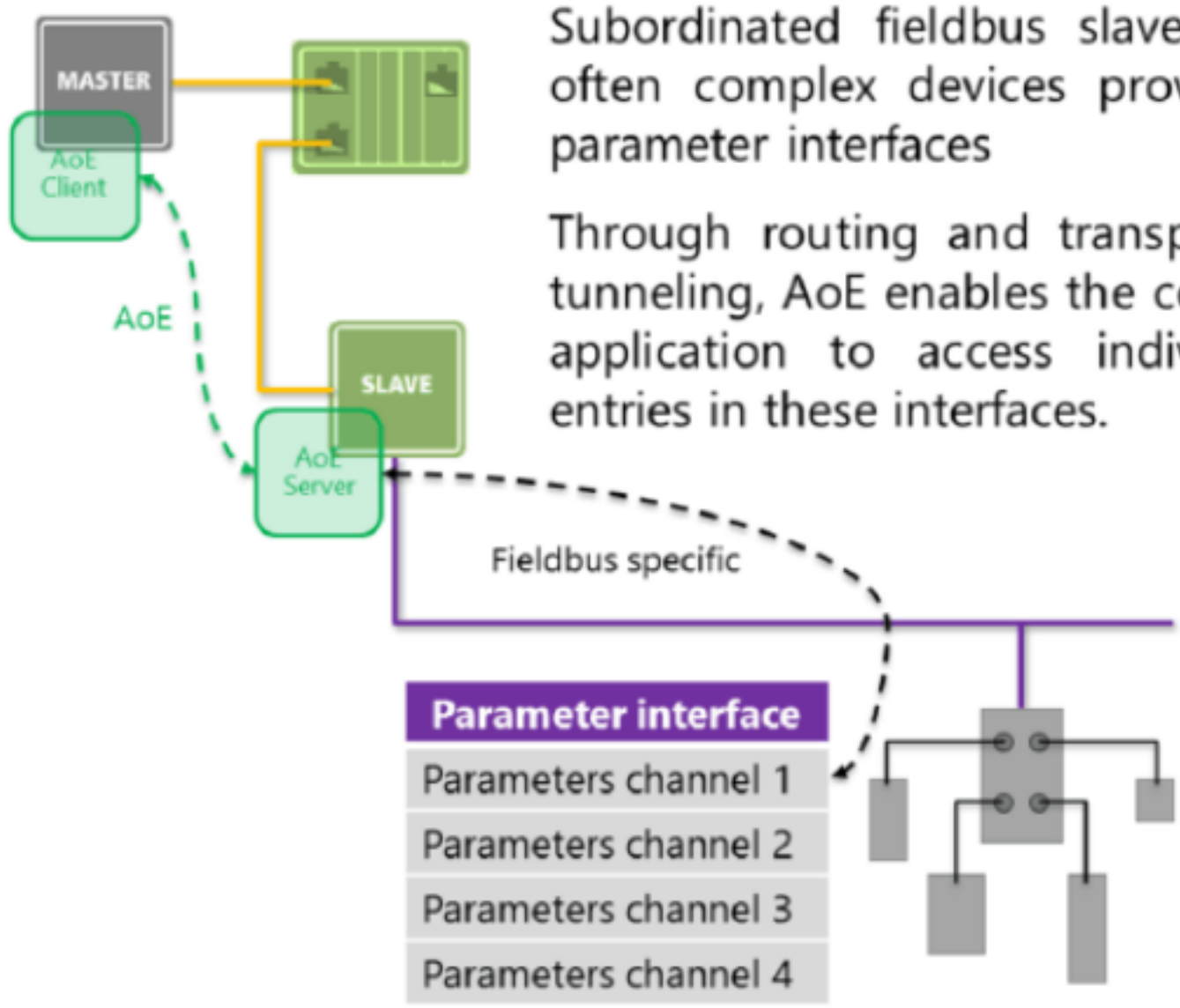
- Provides tunneling and routing mechanisms through EtherCAT, yet requiring no TCP/IP stack (≠ EoE)
- Identifies messages uniquely, enabling transfer of parallel services (≠ CoE)
- Can map other Mailbox protocols as payload
- Standard for acyclic data transfer on EAP

© EtherCAT Technology Group

5 services defined:

Read :	Read data from AoE server
Write :	Write data to AoE server
ReadWrite :	Write data to, and read data from AoE server
WriteControl :	Change state of AoE server or its sub-device
Fragmentation :	Fragmentation of one of previous AoE services

AoE | User Case: Gateways

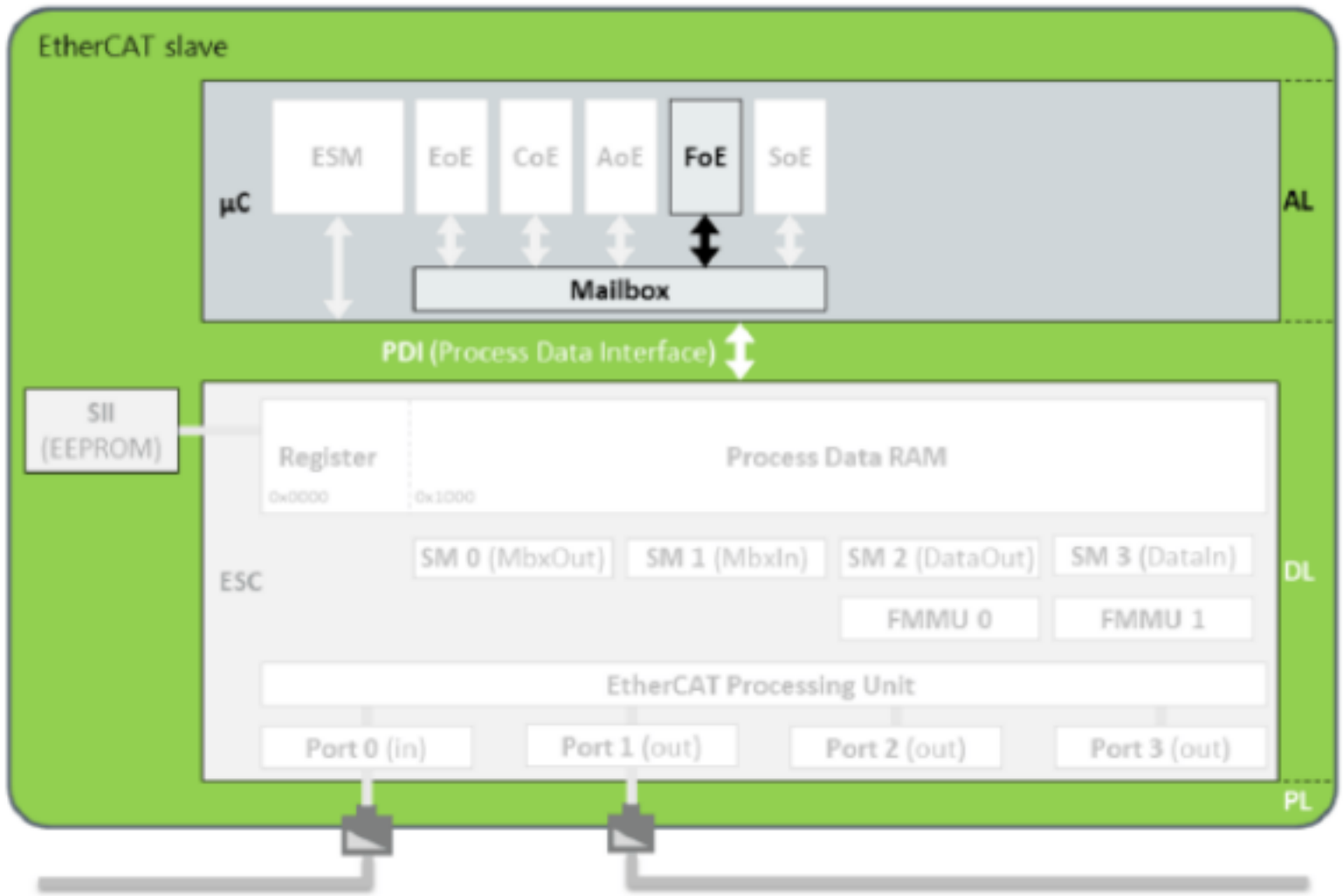


Subordinated fieldbus slaves are often complex devices providing parameter interfaces

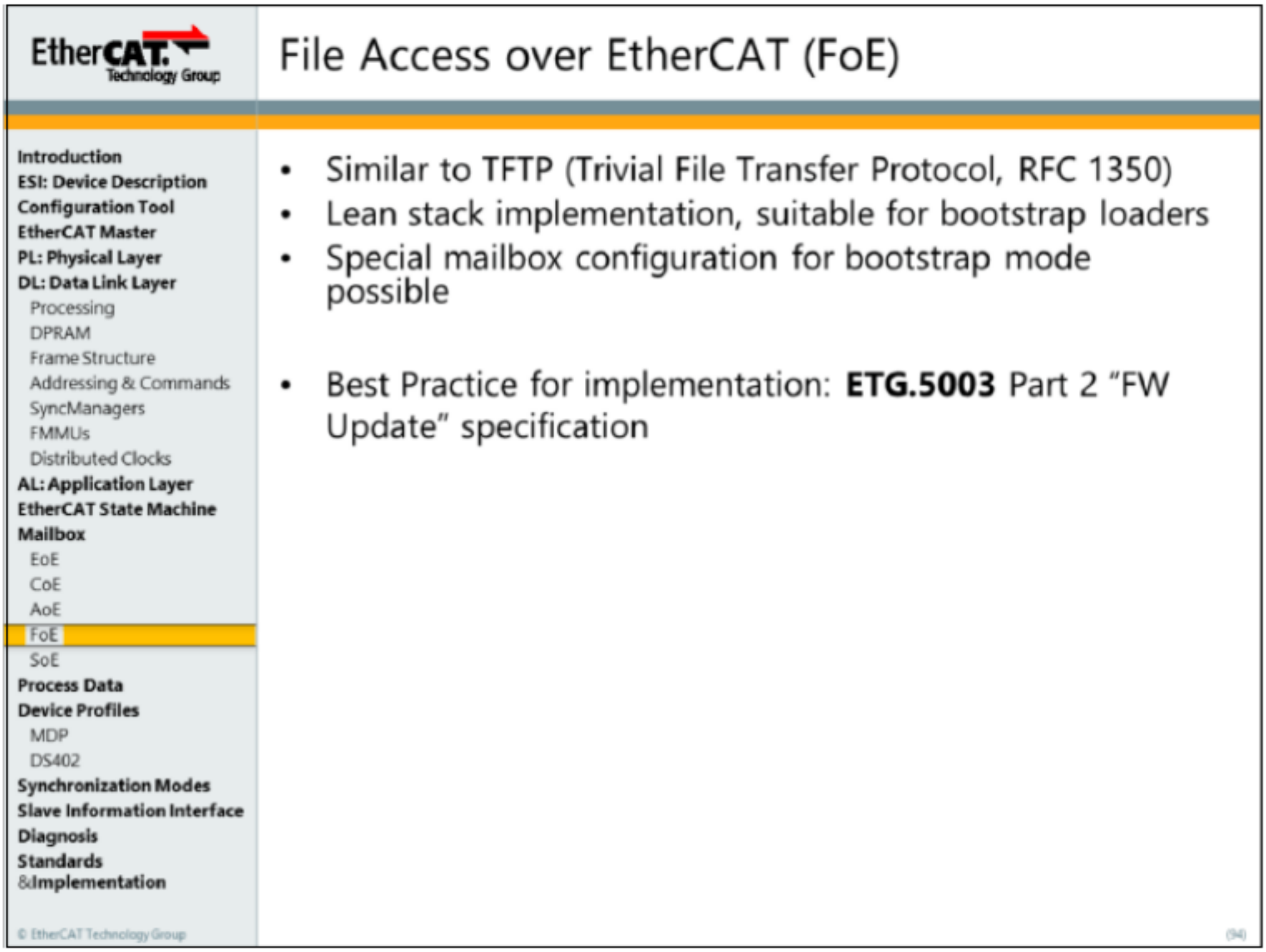
Through routing and transparent tunneling, AoE enables the control application to access individual entries in these interfaces.

© EtherCAT Technology Group

AL | FoE



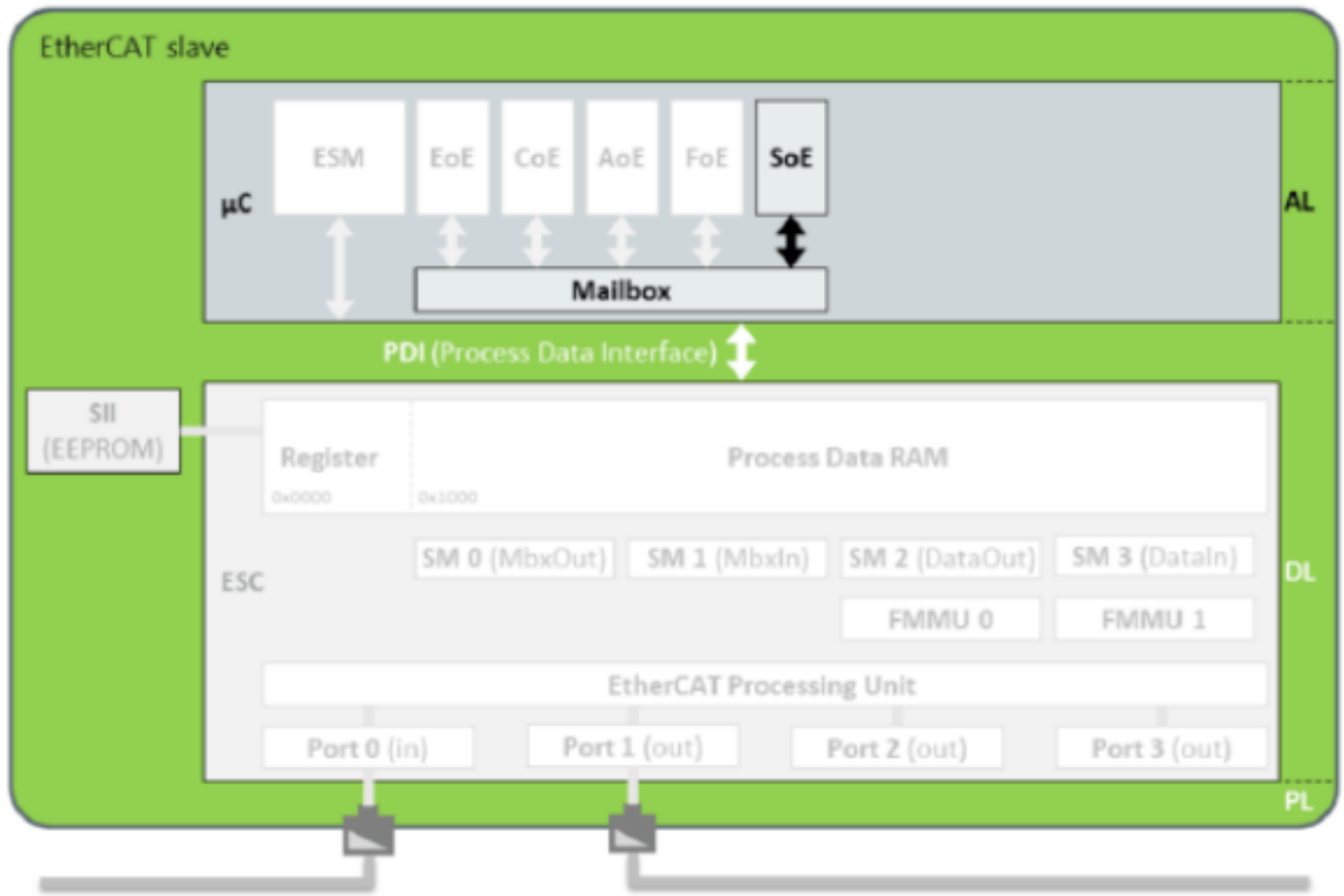
© EtherCAT Technology Group



6 services defined:

WRQ: Write request with "file name"
RRQ: Read request with "file name"
DATA: Data block (full mailbox size used)
ACK: Acknowledgment of DATA and WRQ requests
ERR: Error notification with predefined error codes
BUSY: Busy notification in case of longer procedures, extension to TFTP (e.g. erasing of flash modules)

AL | SoE



© EtherCAT Technology Group

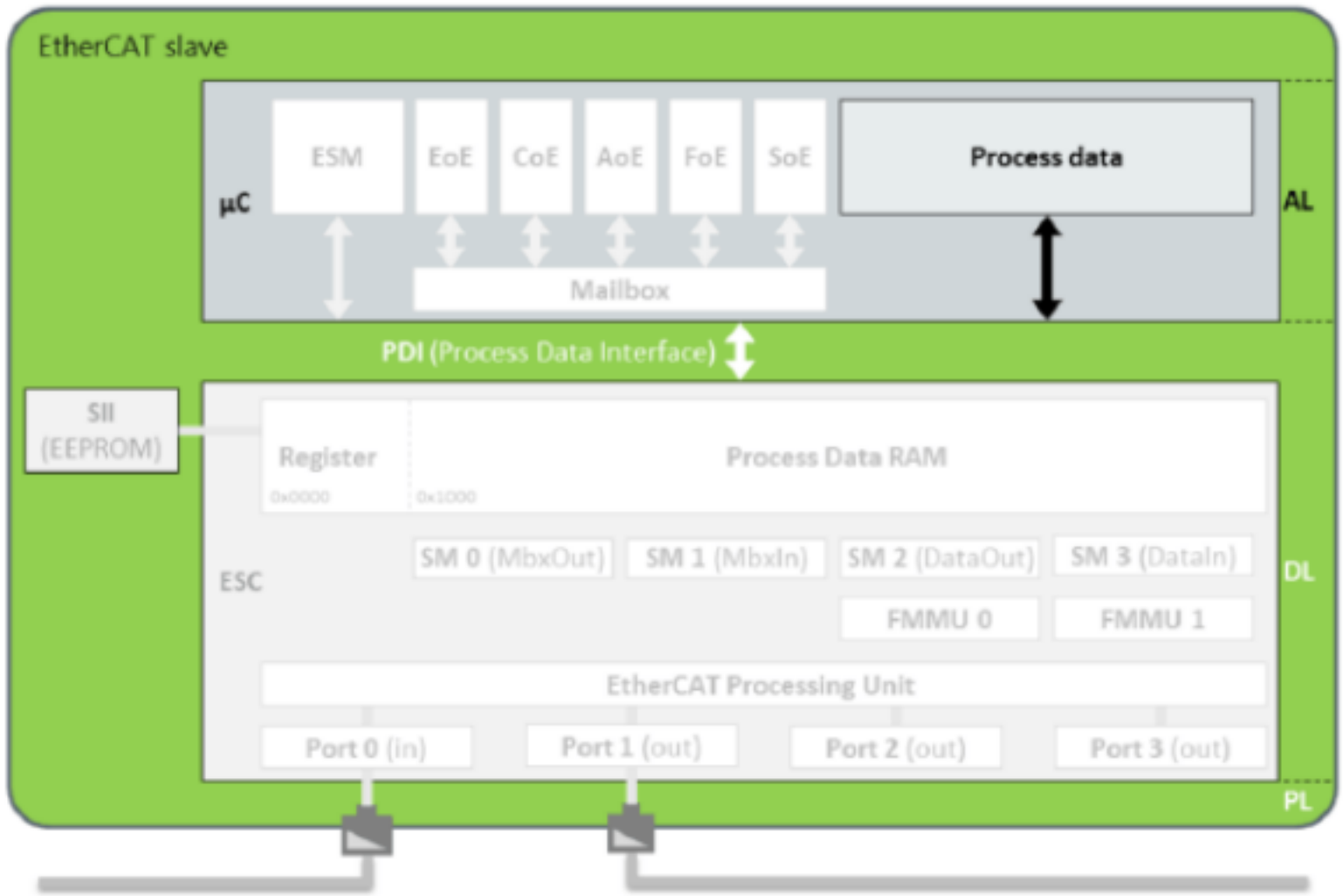
Servo Drive over EtherCAT (SoE)

- Implements Service Channel
  - Read / Write to several elements of an IDN
  - Support of Procedure Commands
  - Slave Info
- The mapping of the IEC 61800-7-1 Annex D (SERCOS™) on EtherCAT is described in IEC 61800-7-3 Annex D

© EtherCAT Technology Group

IDN = Ident Number

Process Data (cyclically, no Mailbox)



© EtherCAT Technology Group

Process Data Handling | Example (EVA Kit)

µC Application:

8 LEDs (digital outputs)

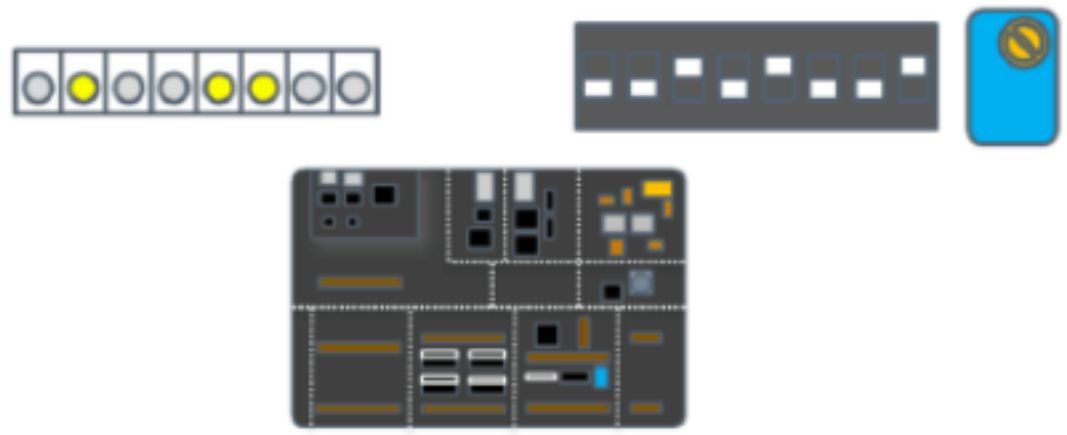
8 Switches (digital inputs)

1 potentiometer (analog input)

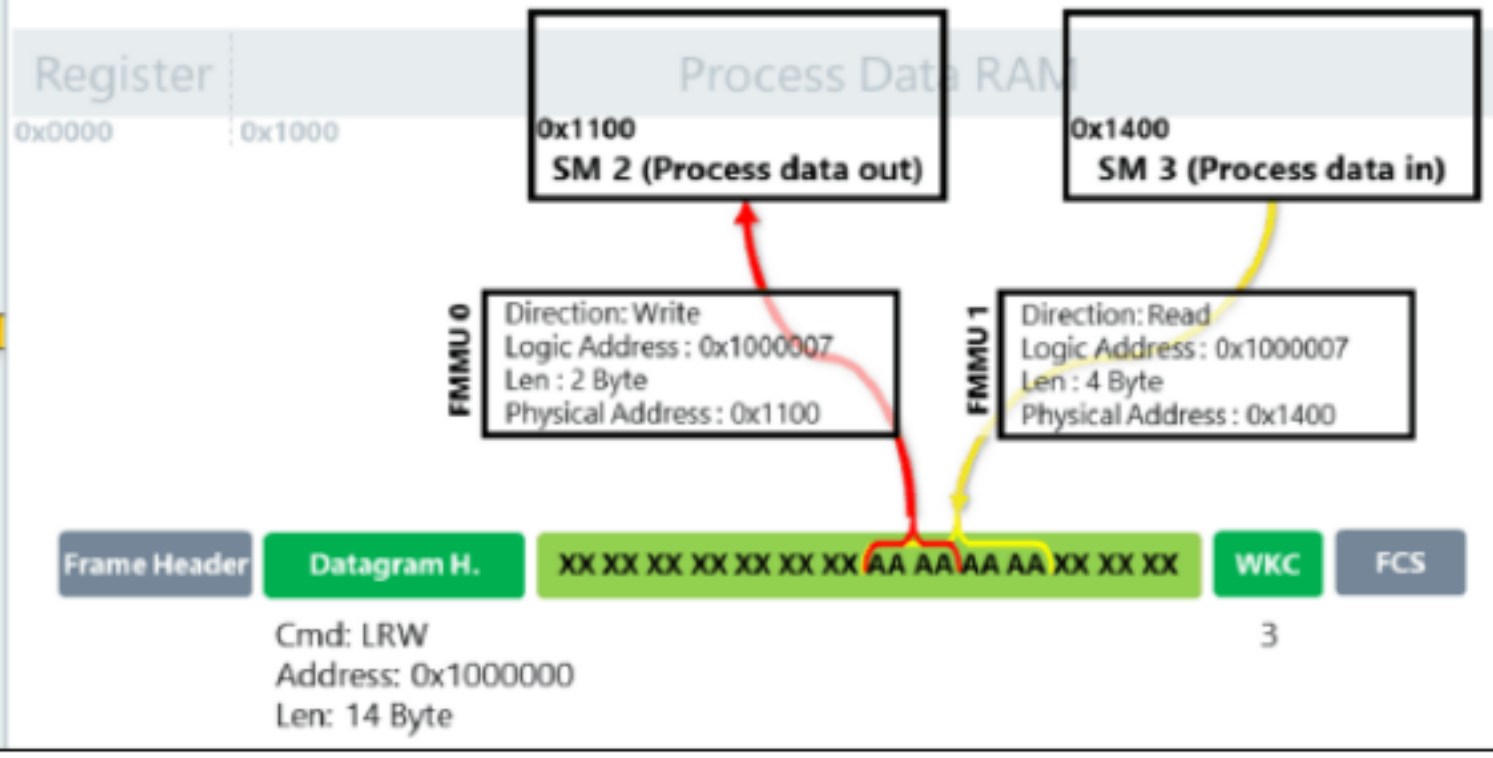
© EtherCAT Technology Group

Example based on Profile 5001 (Modular Device Profile MDP) refer to Device Profiles

Process Data Handling | Data Exchange



Mapping?



© EtherCAT Technology Group

PDO Mapping | Application Structure

© EtherCAT Technology Group

Channel 1

BOOL	*Switch 1*
BOOL	*Switch 2*
BOOL	*Switch 3*
BOOL	*Switch 4*
BOOL	*Switch 5*
BOOL	*Switch 6*
BOOL	*Switch 7*
BOOL	*Switch 8*
8 bit Padding	

1st TxPDO
Digital Inputs
2 Byte

Channel 2

BOOL	*LED 1*
BOOL	*LED 2*
BOOL	*LED 3*
BOOL	*LED 4*
BOOL	*LED 5*
BOOL	*LED 6*
BOOL	*LED 7*
BOOL	*LED 8*
8 bit Padding	

1st RxPDO
Digital Outputs
2 Byte

Channel 3

BOOL	*Underrange*
BOOL	*Overrange*
BIT2	*Limit 1*
BIT2	*Limit 2*
8 bit Padding	
BOOL	*TxPdoState*
BOOL	*TxPdoToggle*
INT	*Analog Input*

2nd TxPDO
Analog Inputs
4 Byte

PDO Mapping | Object Structure

OD	0	1	2	3	4	...	255
0x0000							
...							
0x1600 - 0x17FF	RxPDO Mapping Objects: Area in OD for Output "Packages" offset = 0x01 (one PDO for each channel)						
...							
0x1A00 - 0x1BFF	TxPDO Mapping Objects: Area in OD for Input "Packages" offset = 0x01 (one PDO for each channel)						
...							
0x6000 - 0x6FFF	Area for Input variables offset = 0x10 (16 objects for each channel)						
0x7000 - 0x7FFF	Area for Output variables offset = 0x10 (16 objects for each channel)						
...							
0xFFFF							

© EtherCAT Technology Group

PDO Mapping | Process Data Objects

BOOL *Switch 1*
BOOL *Switch 2*
BOOL *Switch 3*
BOOL *Switch 4*
BOOL *Switch 5*
BOOL *Switch 6*
BOOL *Switch 7*
BOOL *Switch 8*

Channel 1:

→ Describe variables at 0x6000-0x600F

OD	0	1	2	3	4	…	8
0x0000							
…							
0x6000	8 "Highest Subindex"	0 "Switch 1"	0 "Switch 2"	1 "Switch 3"	0 "Switch 4"	…	1 "Switch 8"
…							
0xFFFF							

© EtherCAT Technology Group

PDO Mapping | Process Data Objects

BOOL *LED 1*
BOOL *LED 2*
BOOL *LED 3*
BOOL *LED 4*
BOOL *LED 5*
BOOL *LED 6*
BOOL *LED 7*
BOOL *LED 8*

Channel 2:

→ Describe variables at 0x7010-0x701F

OD	0	1	2	3	4	...	8
0x0000							
...							
0x7010	8 "Highest Subindex"	0 "LED 1"	1 "LED 2"	0 "LED 3"	0 "LED 4"	...	0 "LED 8"
...							
0xFFFF							

© EtherCAT Technology Group

PDO Mapping | Process Data Objects

BOOL	*Underrange*
BOOL	*Overrange*
BIT2	*Limit 1*
BIT2	*Limit 2*
BOOL	*TxPdoState*
BOOL	*TxPdoToggle*
INT	*Analog Input*

Channel 3:

→ Describe variables at 0x6020-0x602F

OD	0	1	2	3	4	…	8
0x0000							
…							
0x6020	8 "Highest Subindex"	0 *"Under range"*	0 "Over range"	01 "Limit 1"	00 "Limit 2"	…	27135 "Analog Input"
…							
0xFFFF							

© EtherCAT Technology Group

PDO Mapping | Rx/TxPDOs in OD

PDO Mapping Object defines the "Container" (PDO)

- SI0: Highest available Subindex
- SI1-255: reference to process data written as XXXXYYZZ
  - XXXX: Object of referenced variable
  - YY: Subindex of referenced variable
  - ZZ: Bitsize of referenced variable

OD	0	1	2	3	4	...	9
...							
0x1601	9 "Highest Subindex"	70100101 "1st Entry"	70100201 "2nd Entry"	70100301 "3rd Entry"	70100401 "4th Entry"	...	00000008 "8 bit padding"
...							
0x1A00	9 "Highest Subindex"	60000101 "1st Entry"	60000201 "2nd Entry"	60000301 "3rd Entry"	60000401 "4th Entry"	...	00000008 "8 bit padding"
0x1A01		Second channel has no Inputs → PDO mapping object does not exist					
0x1A02	9 "Highest Subindex"	60200101 "1st Entry"	60200201 "2nd Entry"	60200302 "3rd Entry"	60200402 "4th Entry"	... (incl. padding)	60201116 "8th Entry"
...							

© EtherCAT Technology Group

SM PDO Assign | Assign Objects

Done:

- ✓ Define structure of variables of application
- ✓ Place the variables on the OD
- ✓ Define PDOs in the PDO Mapping Objects

Todo:

- Assign PDOs to SyncManagers (cyclic process data)

→ *SyncManager PDO Assign Object*

OD	0	1	2	…
…				
0x1C1x	"Highest Subindex"	xxxx "1st PDO"	xxxx "2nd PDO"	…
…				

Example: SM**2** (outputs) → SM PDO Assign Object = 0x1C1**2**
SM**3** (inputs) → SM PDO Assign Object = 0x1C1**3**

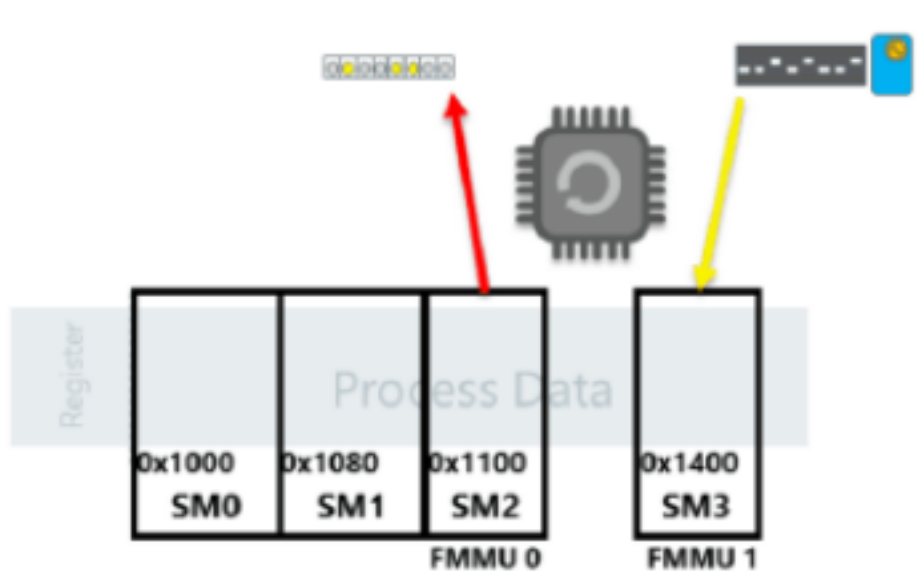
© EtherCAT Technology Group

SM PDO Assign | Example (EVA Kit)

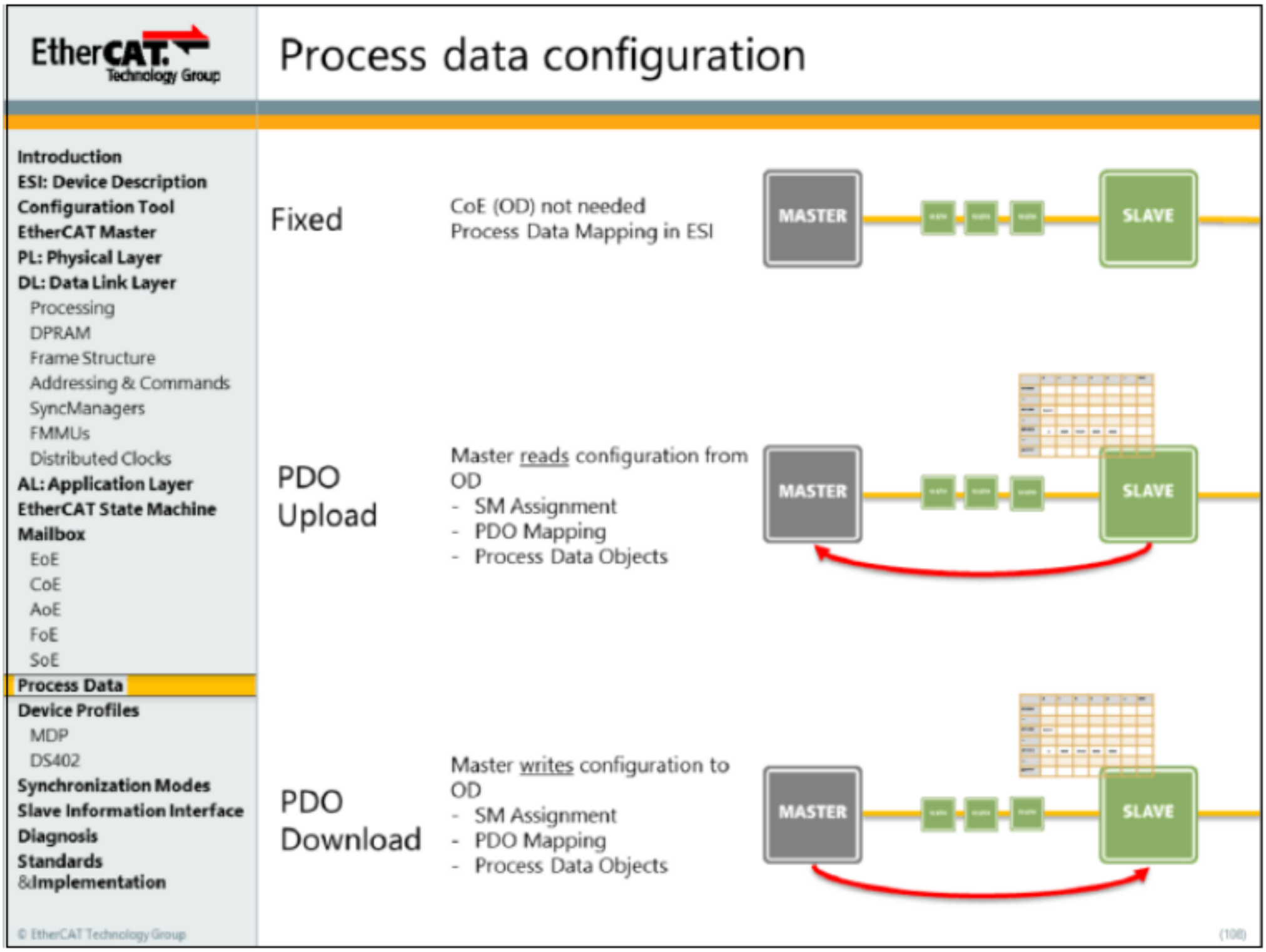
OD	0	1	2
…			
0x1C12	1 "Highest Subindex"	1601 1st RxPDO	
0x1C13	2 "Highest Subindex"	1A00 1st TxPDO	1A02 2nd TxPDO
…			

Assign Object for SM2 (outputs)

Assign Object for SM3 (inputs)



© EtherCAT Technology Group



Fixed process data

? No upload/download of process data configuration needed

? Defined in device description (ESI/SII)

PDO upload (PdoUpload)

? Process data configuration defined in the slave
(e.g. via serial cable and vendor specific software, gateway to another fieldbus,

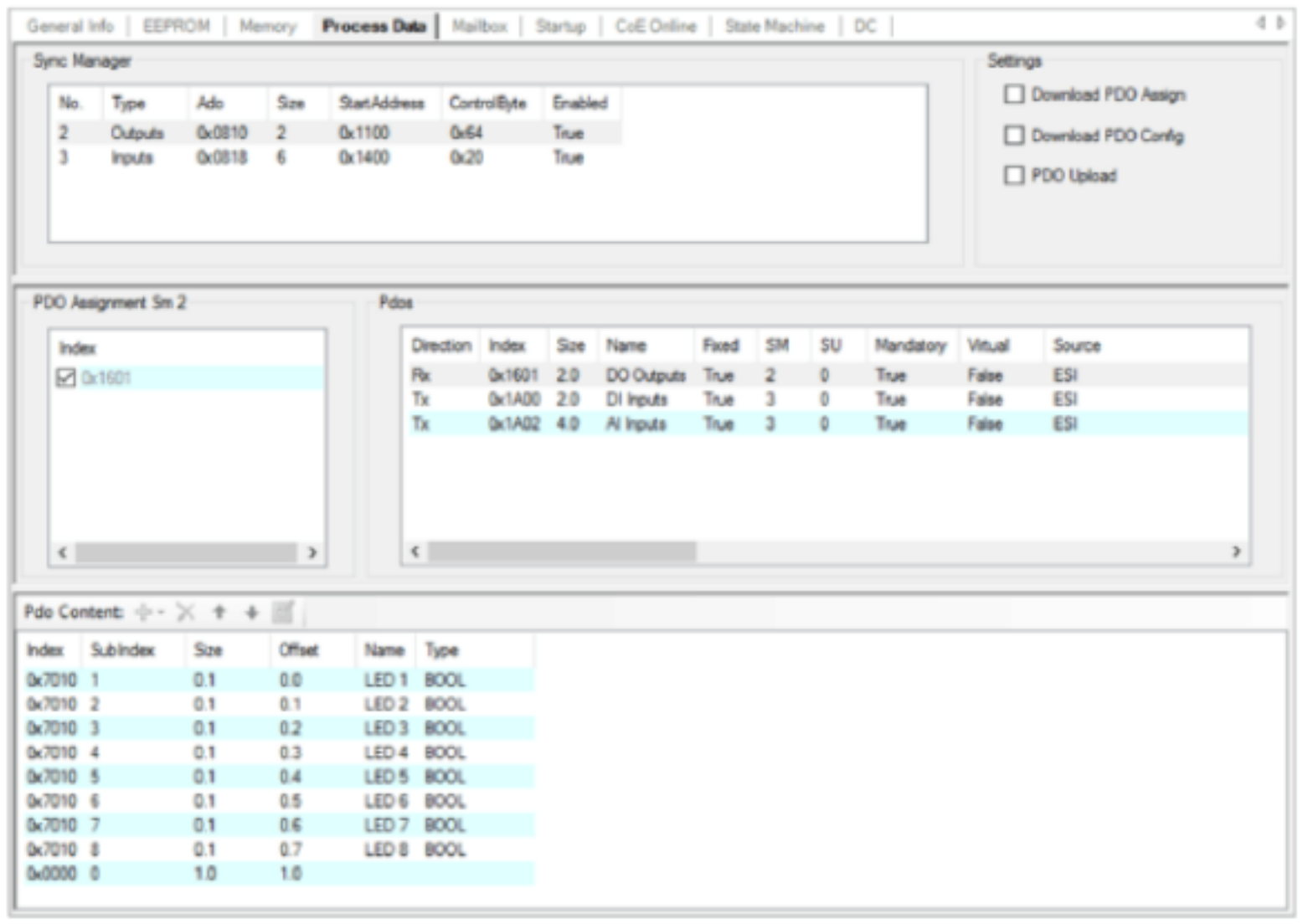
? EtherCAT configuration tool reads the process data configuration from the slave via SDO upload service

PDO download (PdoAssign/PdoConfig)

? Process data configuration defined by user in the EtherCAT configuration tool

? Configuration is written to the slave via SDO download

Process data configuration (example)



Screenshot (partial): Conformance Test Tool 2.1.34.0

© EtherCAT Technology Group

Configuration Tools shall provide interface to change process data configuration

Process Data Structure in ESI

ESI uses elements TxPDO and TxPDO similar to OD structure.

RxPdo

Fixed	true
Mandatory	true
Sm	2
Index	#x1601
Name	DO RxPDO-Map

Entry (9)

	Index	SubIndex	BitLen	Name	DataType
1	#x7010	1	1	LED 1	BOOL
2	#x7010	2	1	LED 2	BOOL
3	#x7010	3	1	LED 3	BOOL
4	#x7010	4	1	LED 4	BOOL
5	#x7010	5	1	LED 5	BOOL
6	#x7010	6	1	LED 6	BOOL
7	#x7010	7	1	LED 7	BOOL
8	#x7010	8	1	LED 8	BOOL
9	#x0	0	8		

TxPdo

Fixed	true
Mandatory	true
Sm	3
Index	#x1A00
Name	DI TxPDO-Map

Entry (9)

	Index	SubIndex	BitLen	Name	DataType
1	#x6000	1	1	Switch 1	BOOL
2	#x6000	2	1	Switch 2	BOOL
3	#x6000	3	1	Switch 3	BOOL
4	#x6000	4	1	Switch 4	BOOL
5	#x6000	5	1	Switch 5	BOOL
6	#x6000	6	1	Switch 6	BOOL
7	#x6000	7	1	Switch 7	BOOL
8	#x6000	8	1	Switch 8	BOOL
9	#x0	0	8		

TxPdo

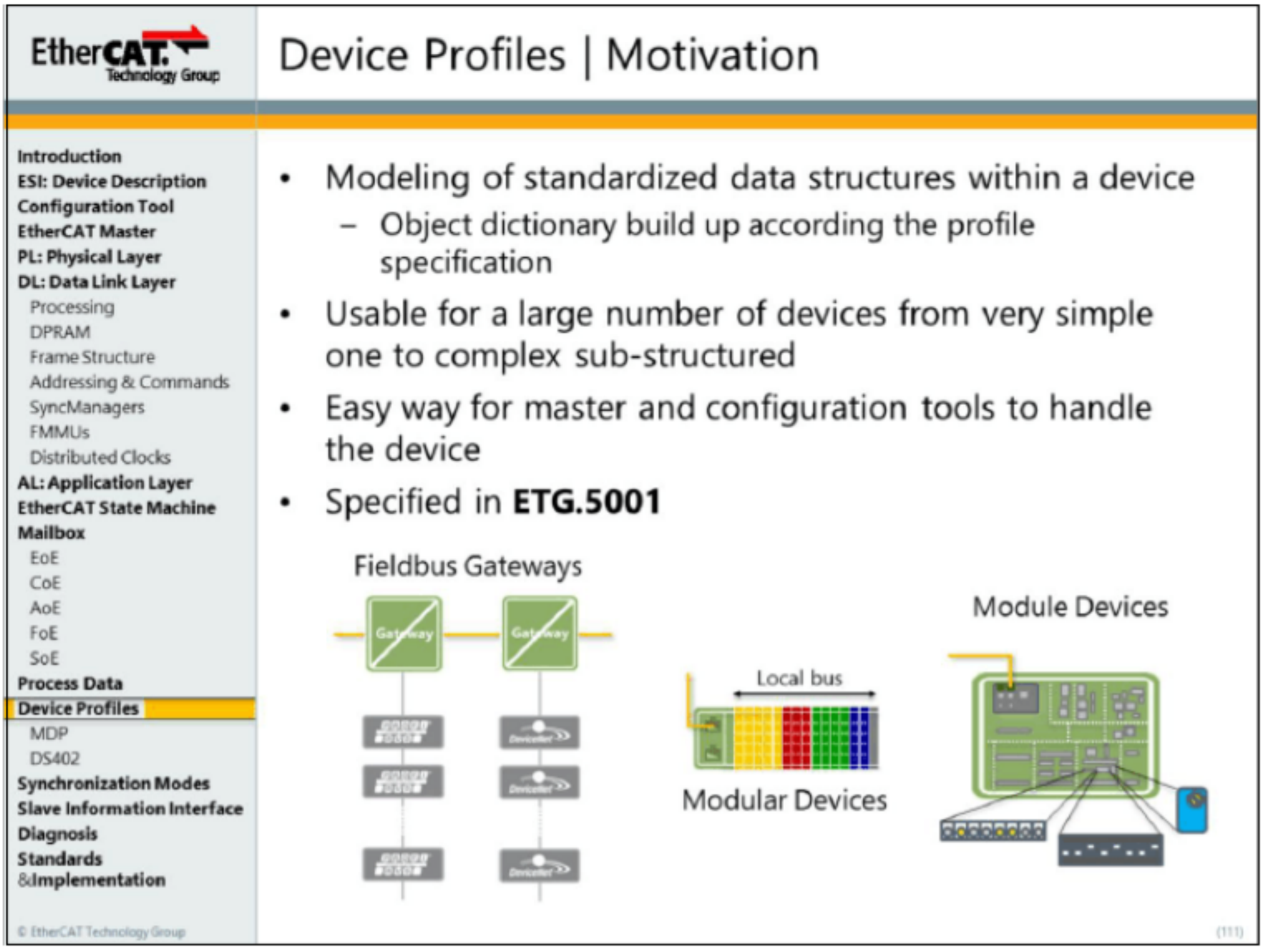
Fixed	true
Mandatory	true
Sm	3
Index	#x1A02
Name	AI TxPDO-Map

Entry (8)

	Index	SubIndex	BitLen	Name	DataType
1	#x6020	1	1	Underrange	BOOL
2	#x6020	2	1	Overrange	BOOL
3	#x6020	3	2	Limit 1	BIT2
4	#x6020	5	2	Limit 2	BIT2
5	#x0	0	8		
6	#x1802	7	1	TxPDOState	BOOL
7	#x1802	9	1	TxPDO Toggle	BOOL
8	#x6020	17	16	Analog input	INT

→ Altova XML Spy® 2013

© EtherCAT Technology Group



Fieldbus Gateways

? Gateways to other fieldbus networks

? Each connected fieldbus device represents “ module ”

Modular Devices

? with physical connectable modules and/or functional modules

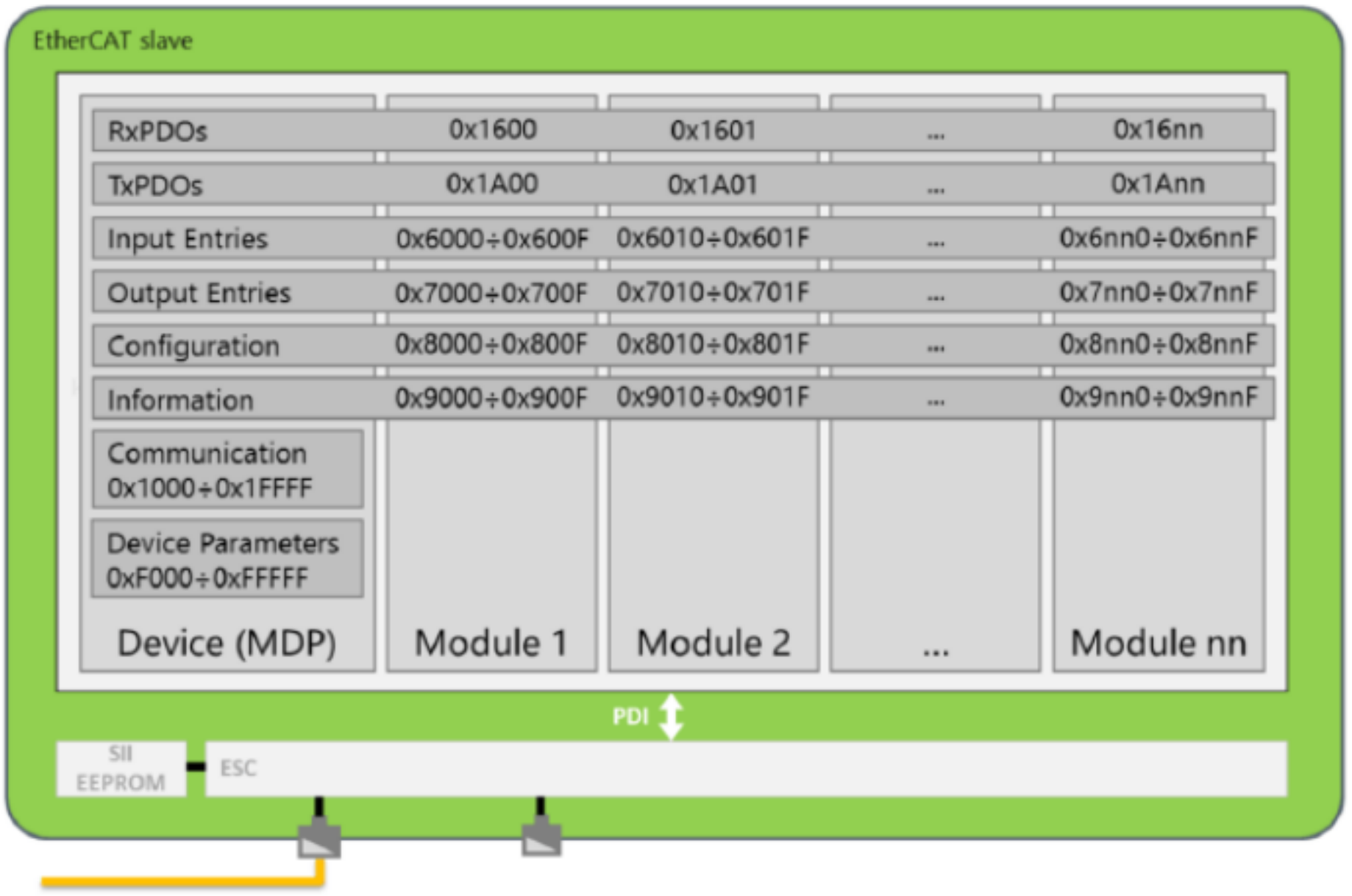
? Each connected module represents a “ module ”

Module Devices

? EtherCAT device which consists of several channels (e.g. analog input, analog output, ⋯)

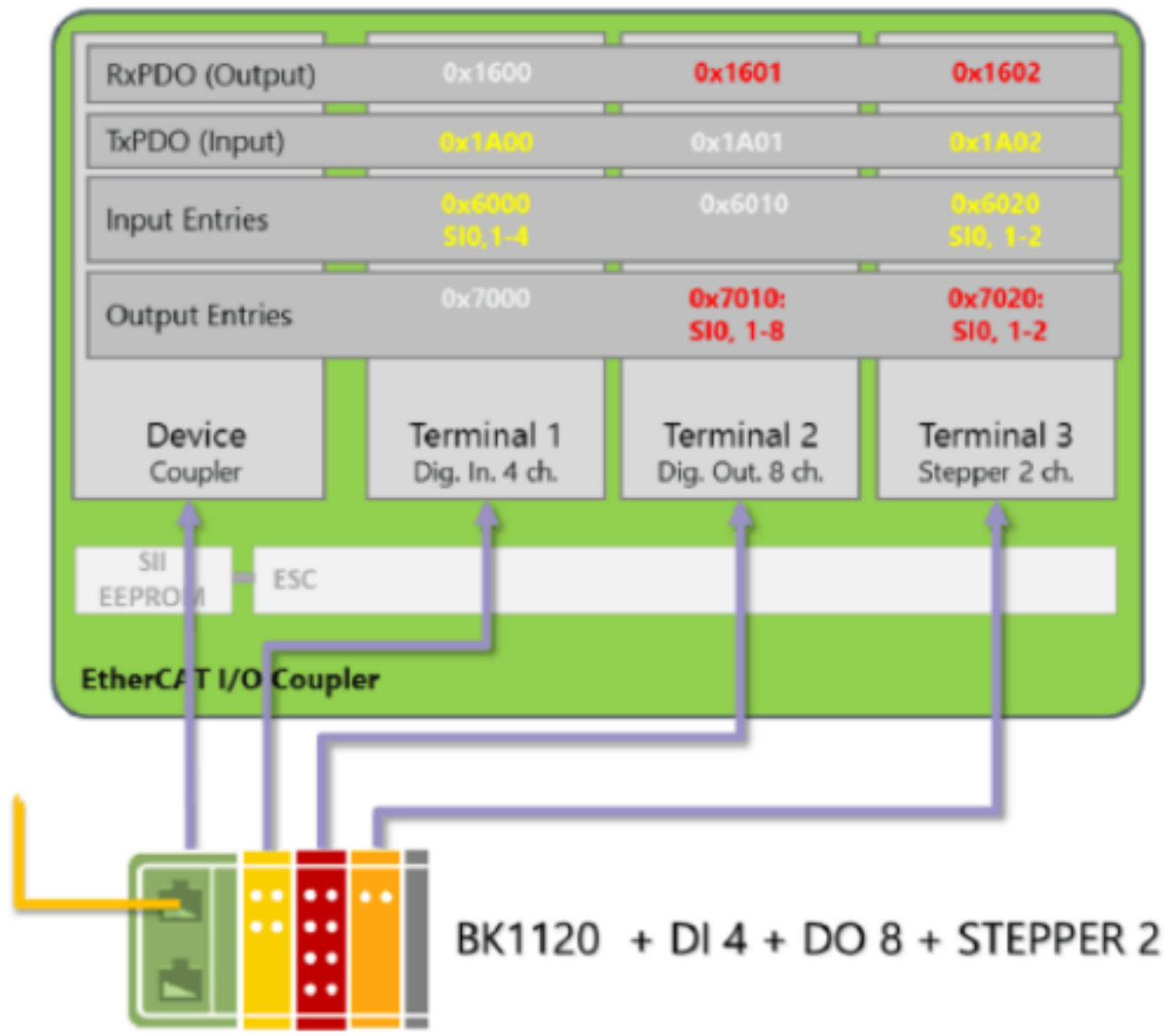
? Each channel represents a “ module ”

MDP | OD Structure



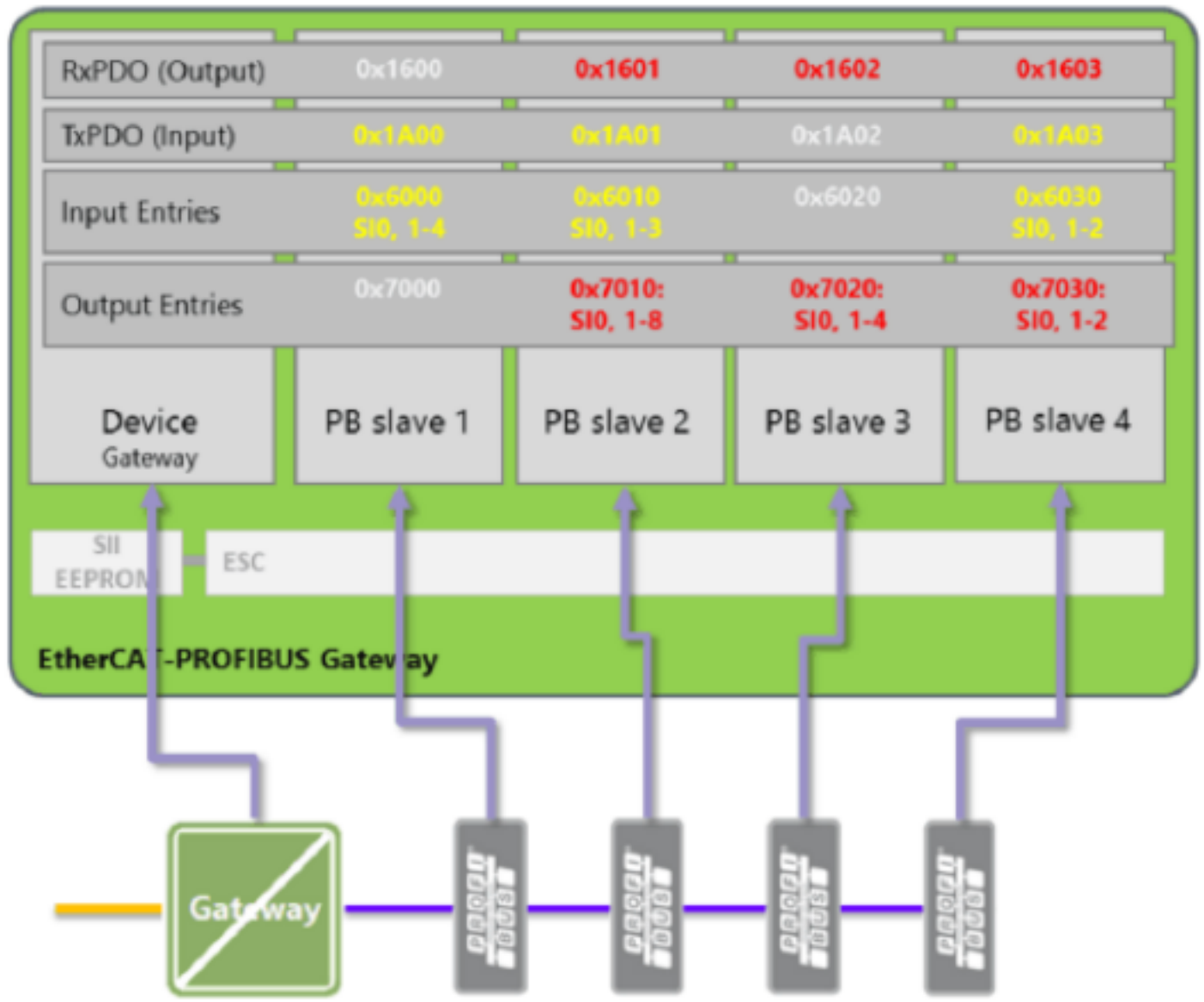
© EtherCAT Technology Group

MDP | modular device (example)



© EtherCAT Technology Group

MDP | fieldbus gateway (example)



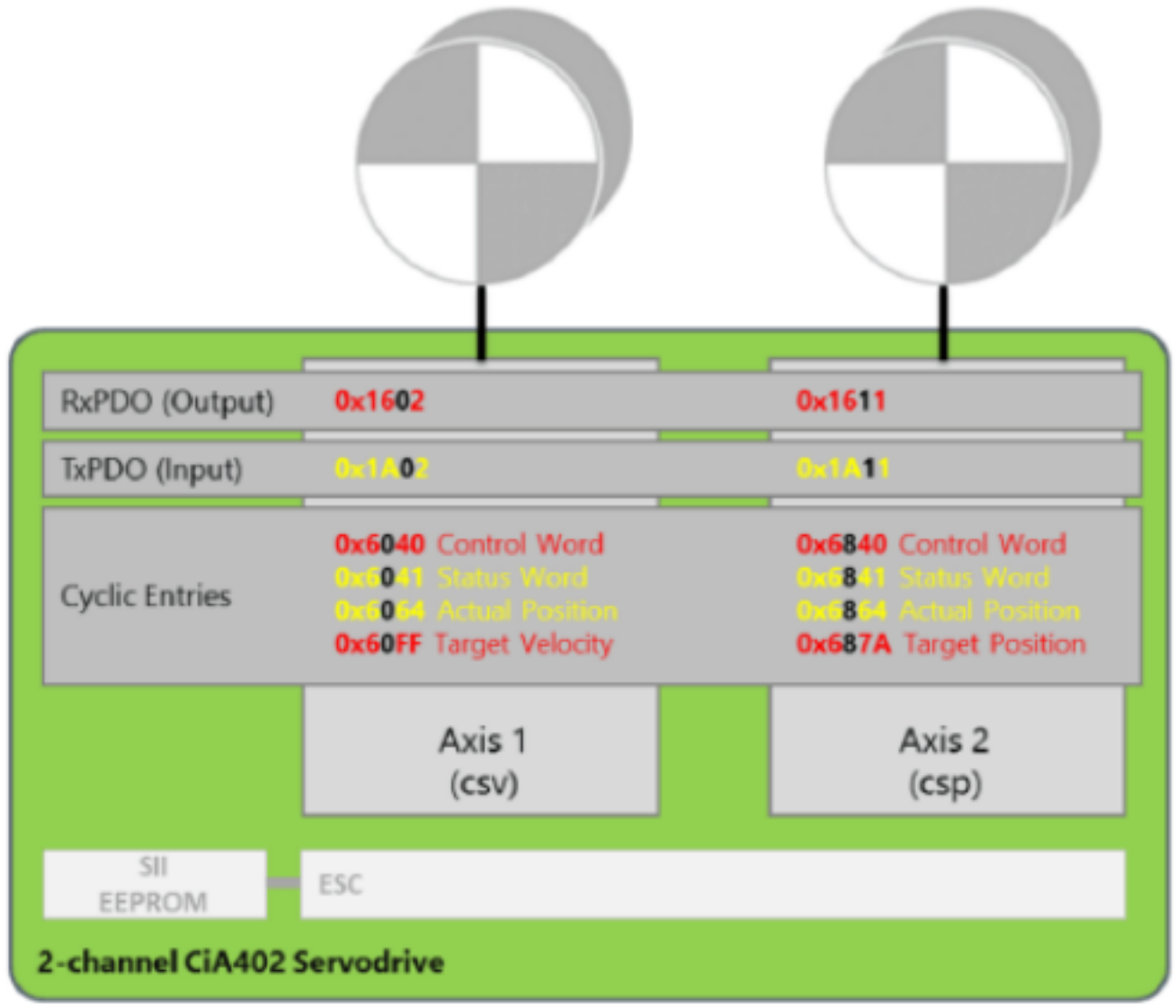
© EtherCAT Technology Group

CiA402 Drive Profile

- Implementation Guideline **ETG.6010** for the CiA402 Drive Profile
  - Specify a common behavior of EtherCAT CiA402 servo drives according to IEC 61800-7
- Scope
  - EtherCAT CiA402 Servo Drives
  - No frequency converter
- Contents
  - Clarifications of the state machine
  - Modes of operation
  - Function Groups (FG)
    - FG Position, FG Velocity, FG Torque
    - FG Torque Limiting, FG Homing, FG Touch Probe
  - Endless Positioning

© EtherCAT Technology Group

CiA402 | servodrive (example)



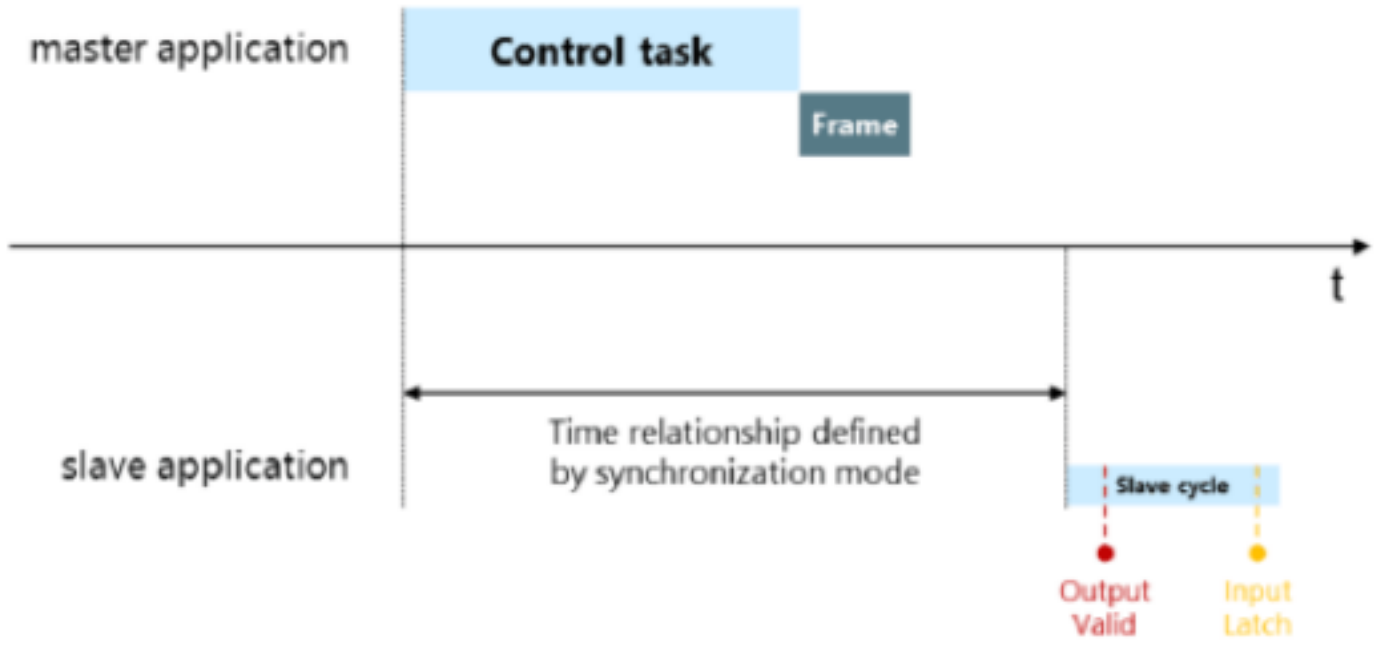
© EtherCAT Technology Group

csp = Cyclic Synchronous Position

csv = Cyclic Synchronous Velocity

Synchronization Modes

- Define a time relationship between the cyclic process data update on master and on slave side.

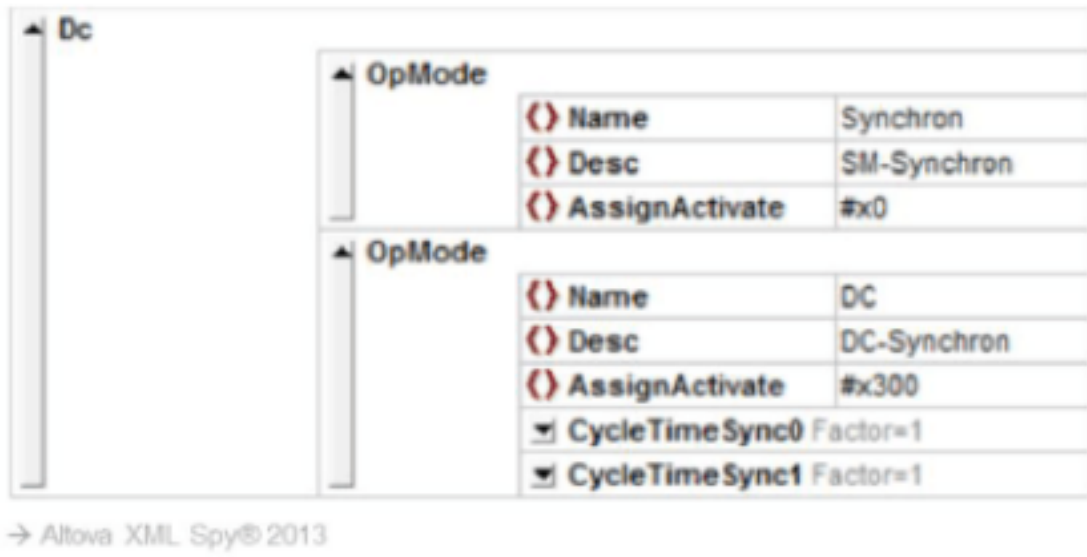


- Available synchronization modes for a slave:
  - Free Run
  - SM-Synchronous (synchronized to cyclic frames)
  - DC-Synchronous (synchronized to SYNC interrupts)

© EtherCAT Technology Group

Synchronization Modes

- Synchronization modes supported by a slave and corresponding parameters are described in ESI file:

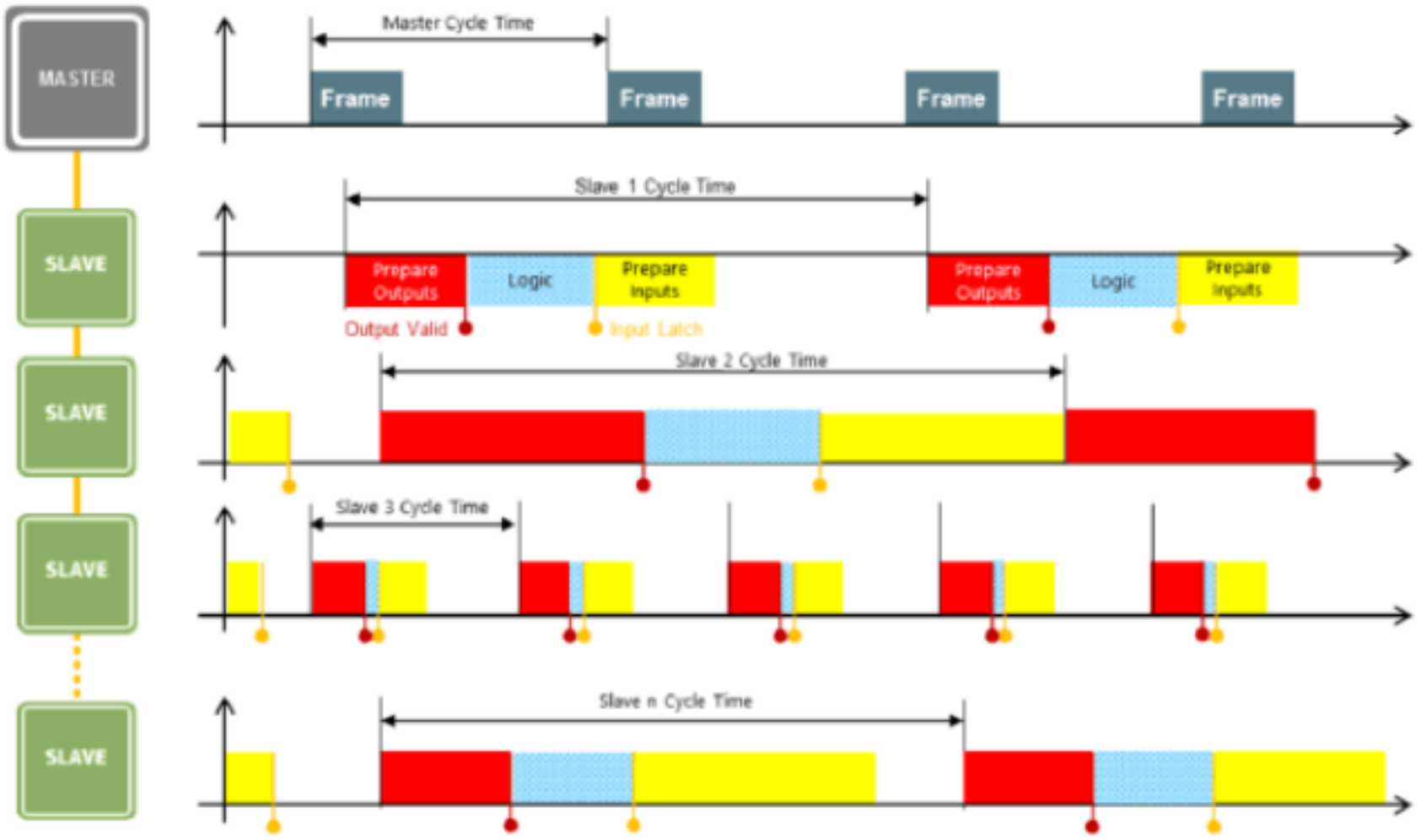


- Synchronization relevant information is described in CoE objects 0x1C3**2** (outputs) and 0x1C3**3** (inputs).

© EtherCAT Technology Group

Synchronization modes, together with CoE objects 0x1C32/33 are specified in ETG.1020.

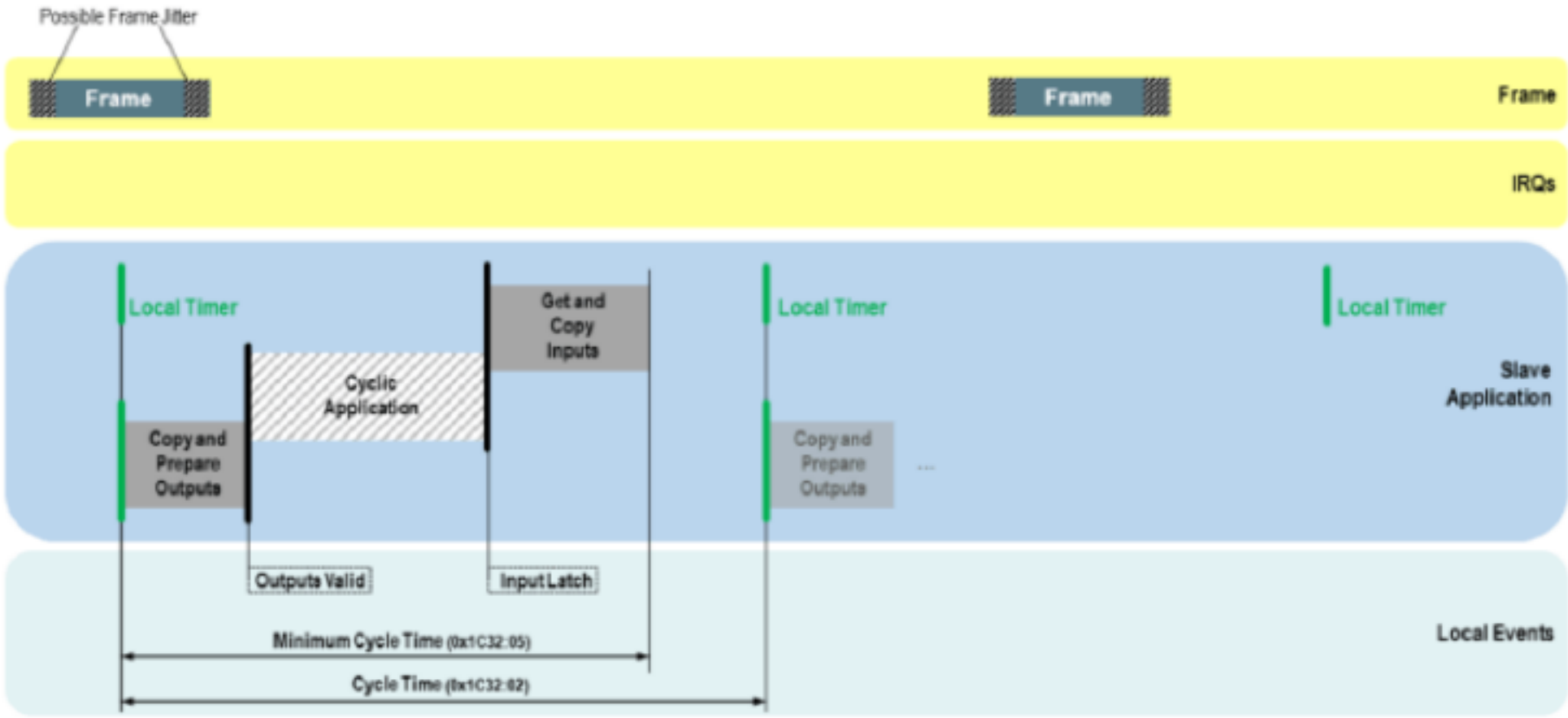
Free Run



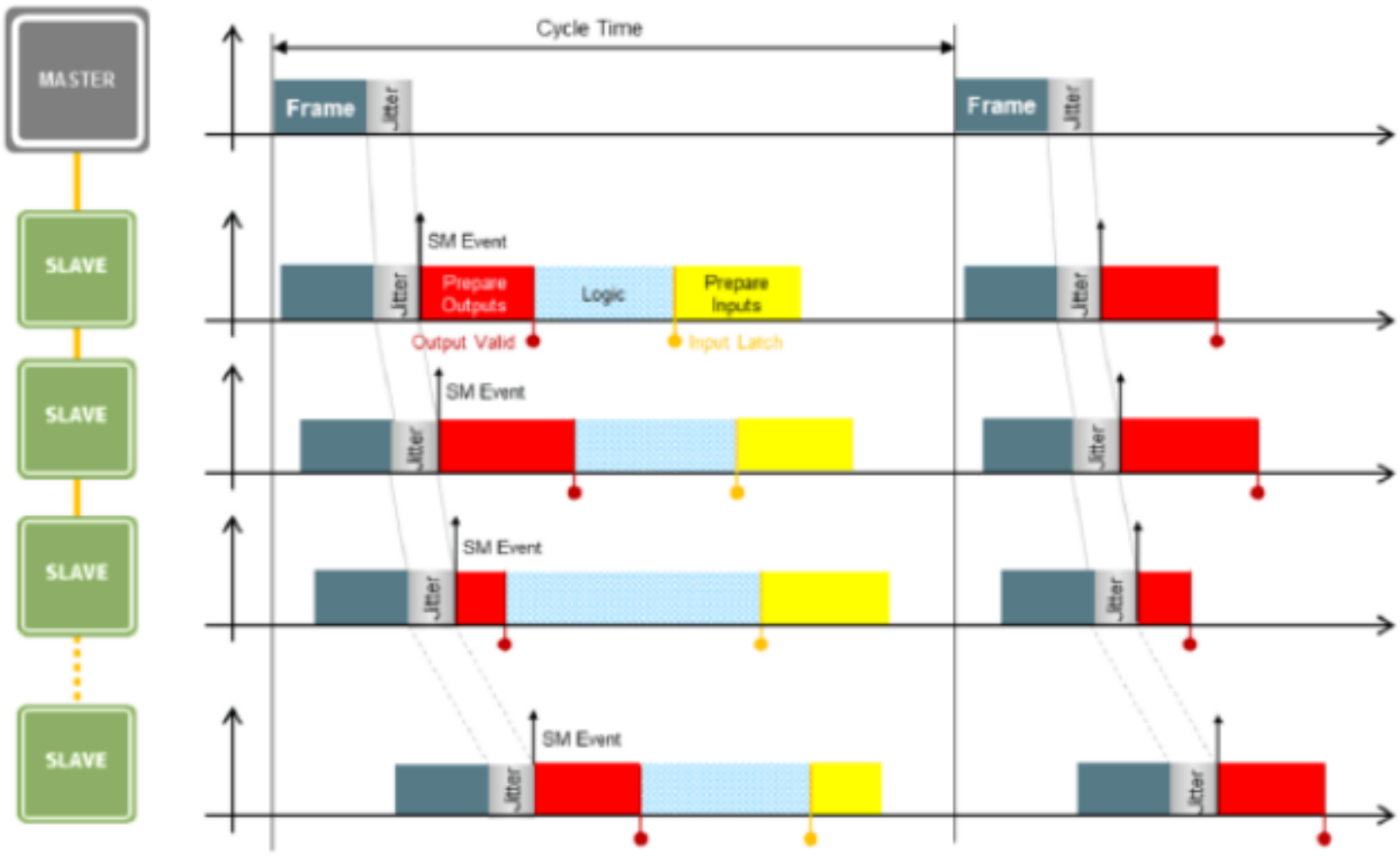
© EtherCAT Technology Group

Free Run

Local Timer



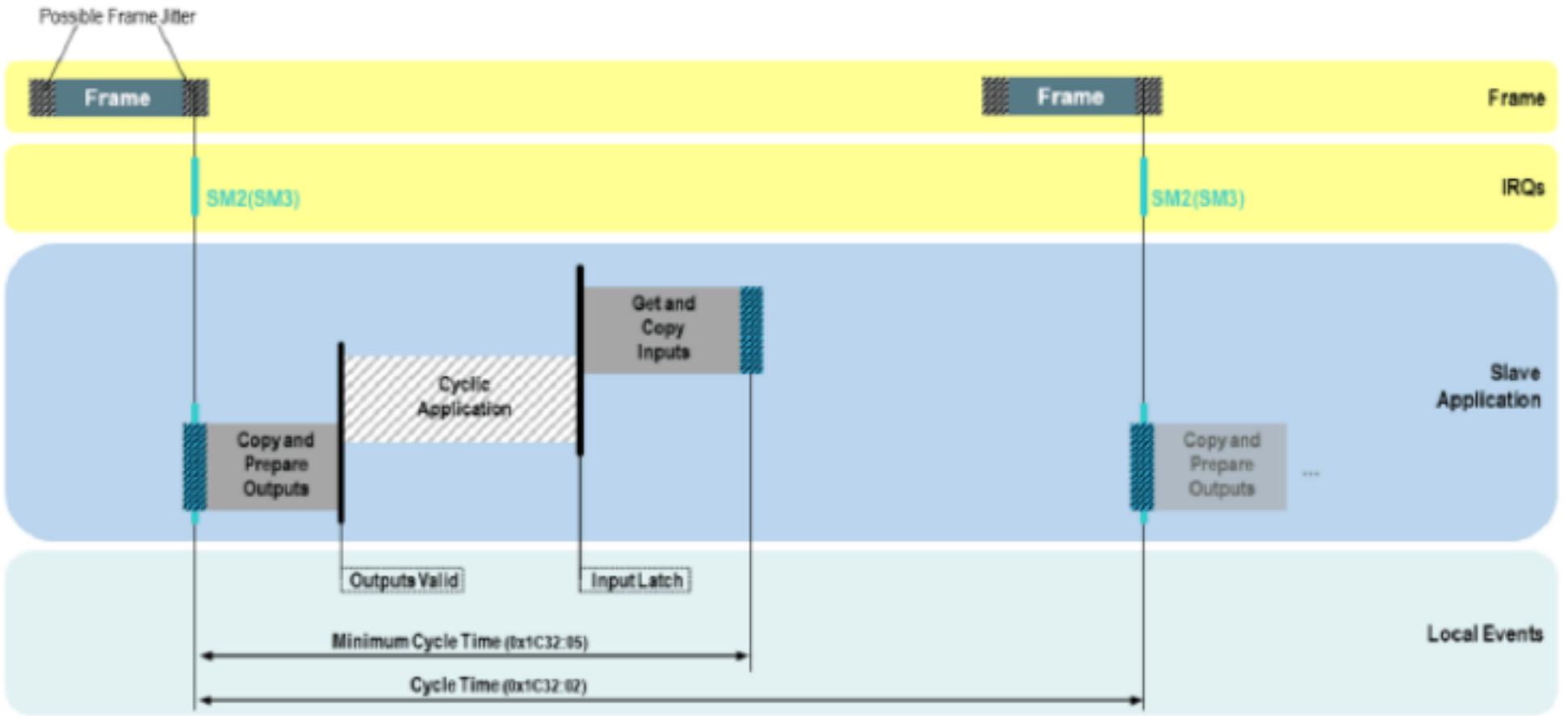
SM-Synchronous



© EtherCAT Technology Group

SM-Synchronous

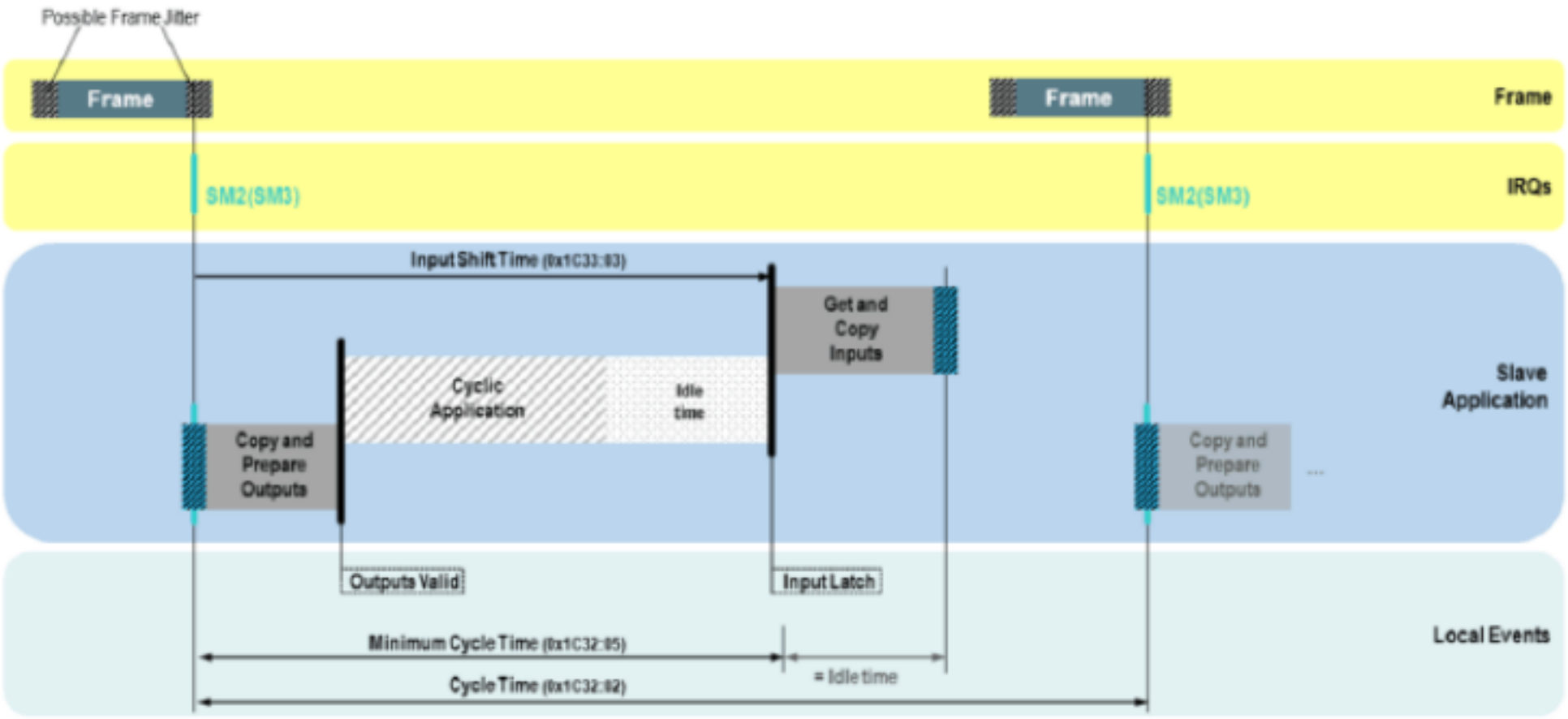
SM Event



© EtherCAT Technology Group

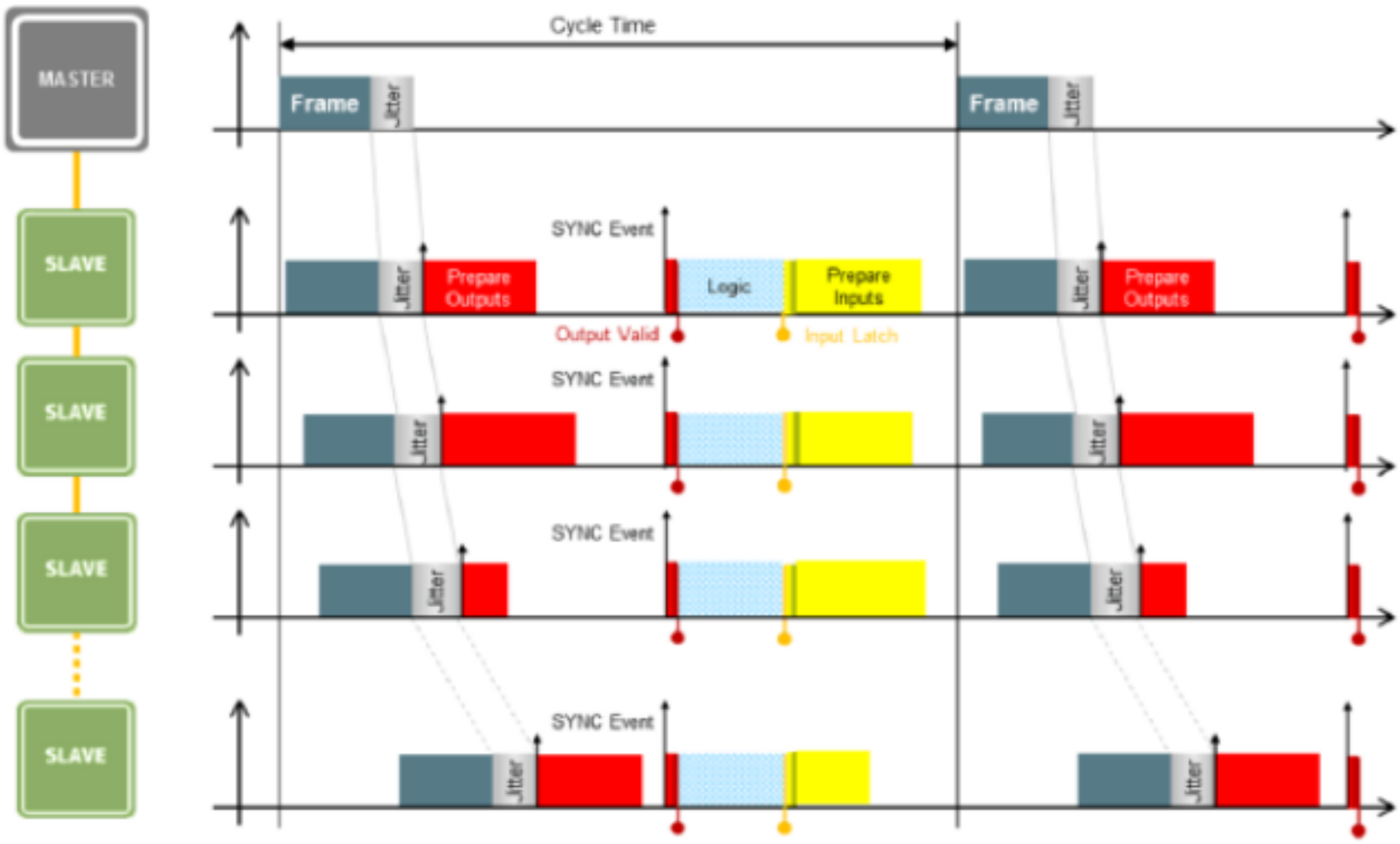
SM-Synchronous

SM Event (Shift of Input Latch via 0x1C33:03)



© EtherCAT Technology Group

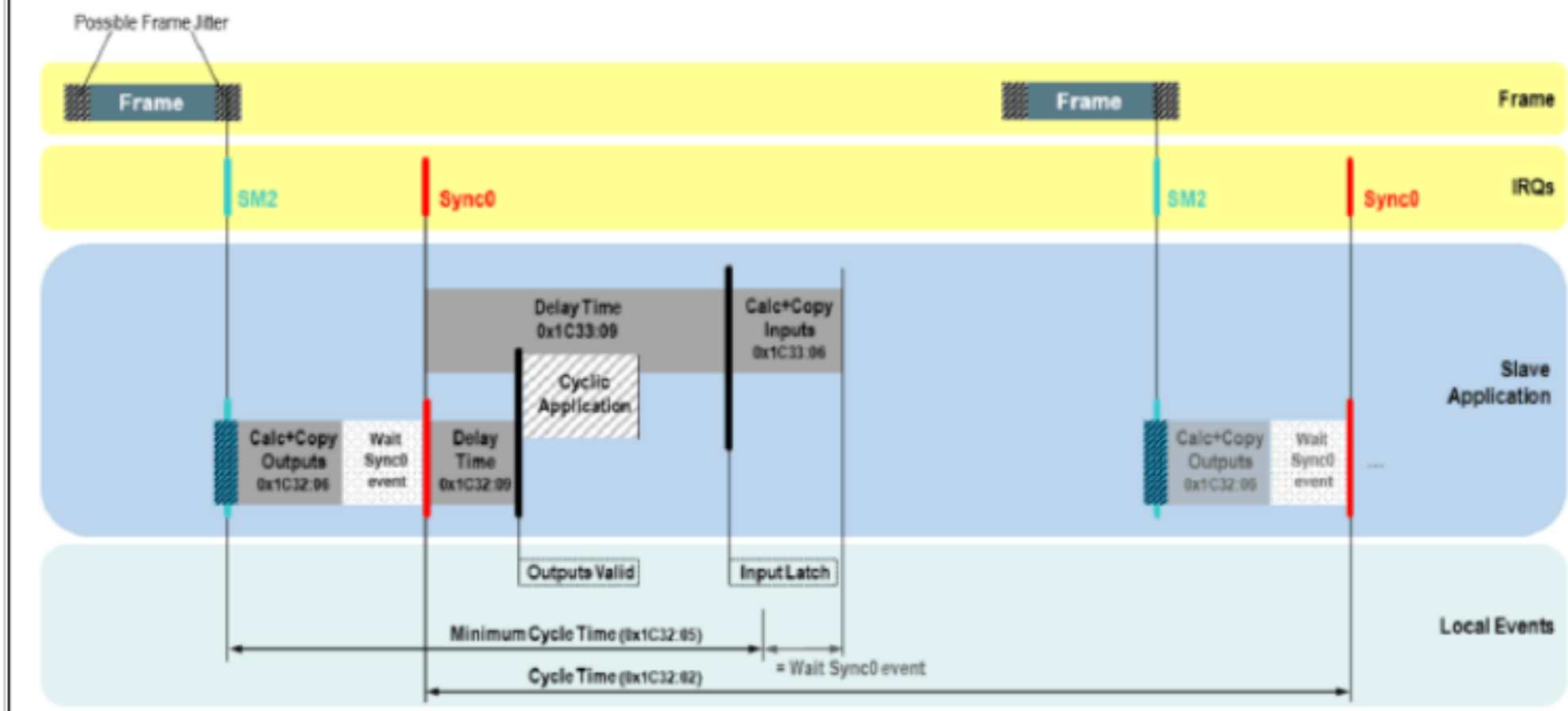
DC-Synchronous



© EtherCAT Technology Group

DC-Synchronous

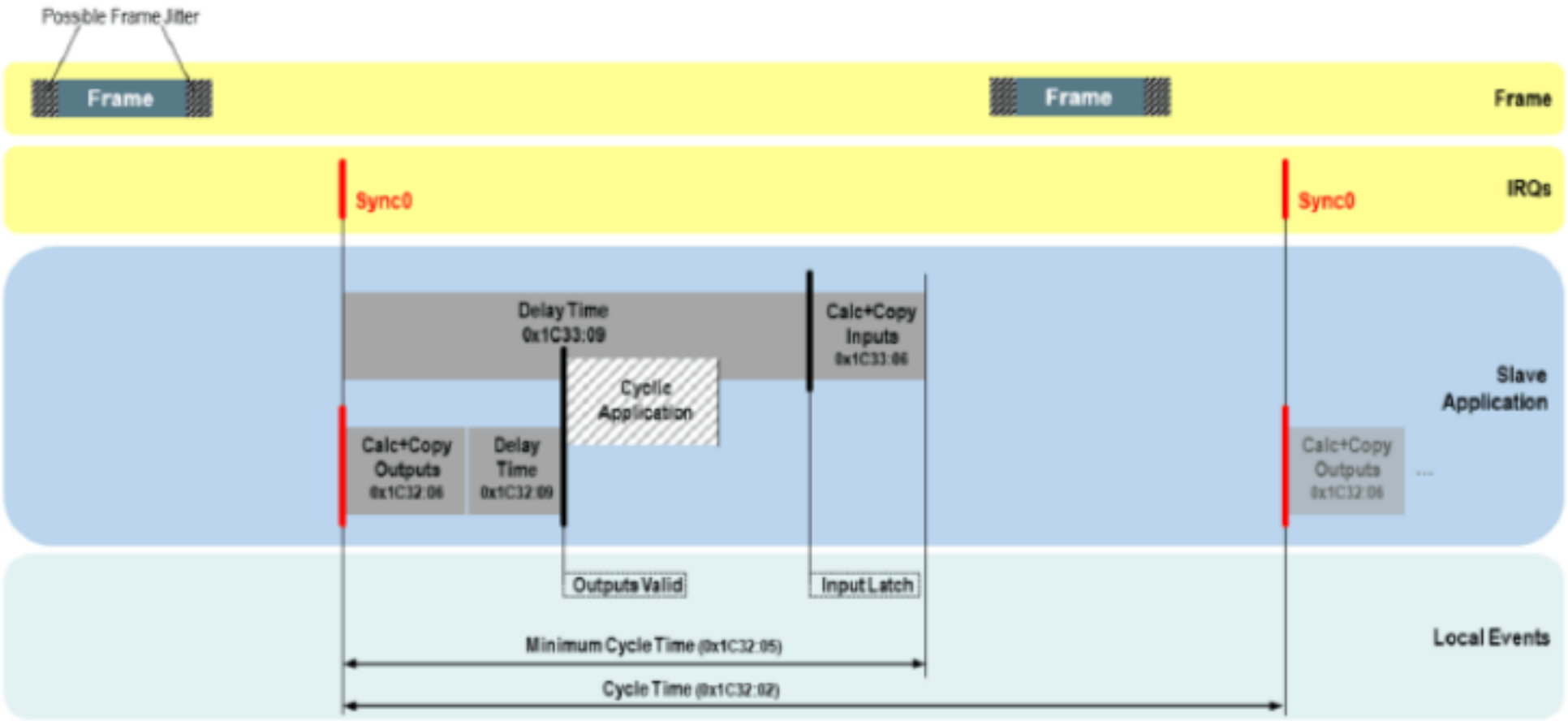
SM Event, Sync0 Event



© EtherCAT Technology Group

DC Mode 1

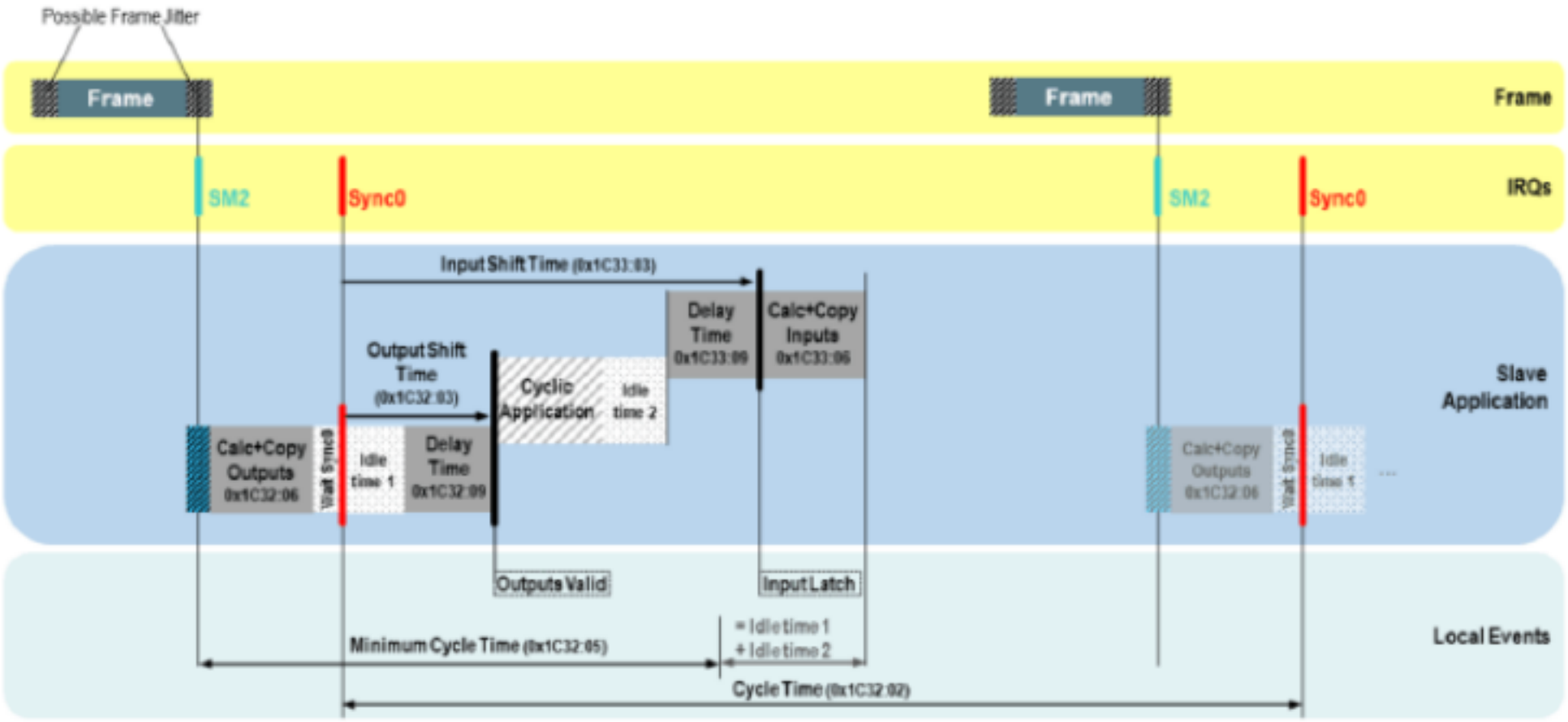
Sync0 Event



© EtherCAT Technology Group

DC-Synchronous

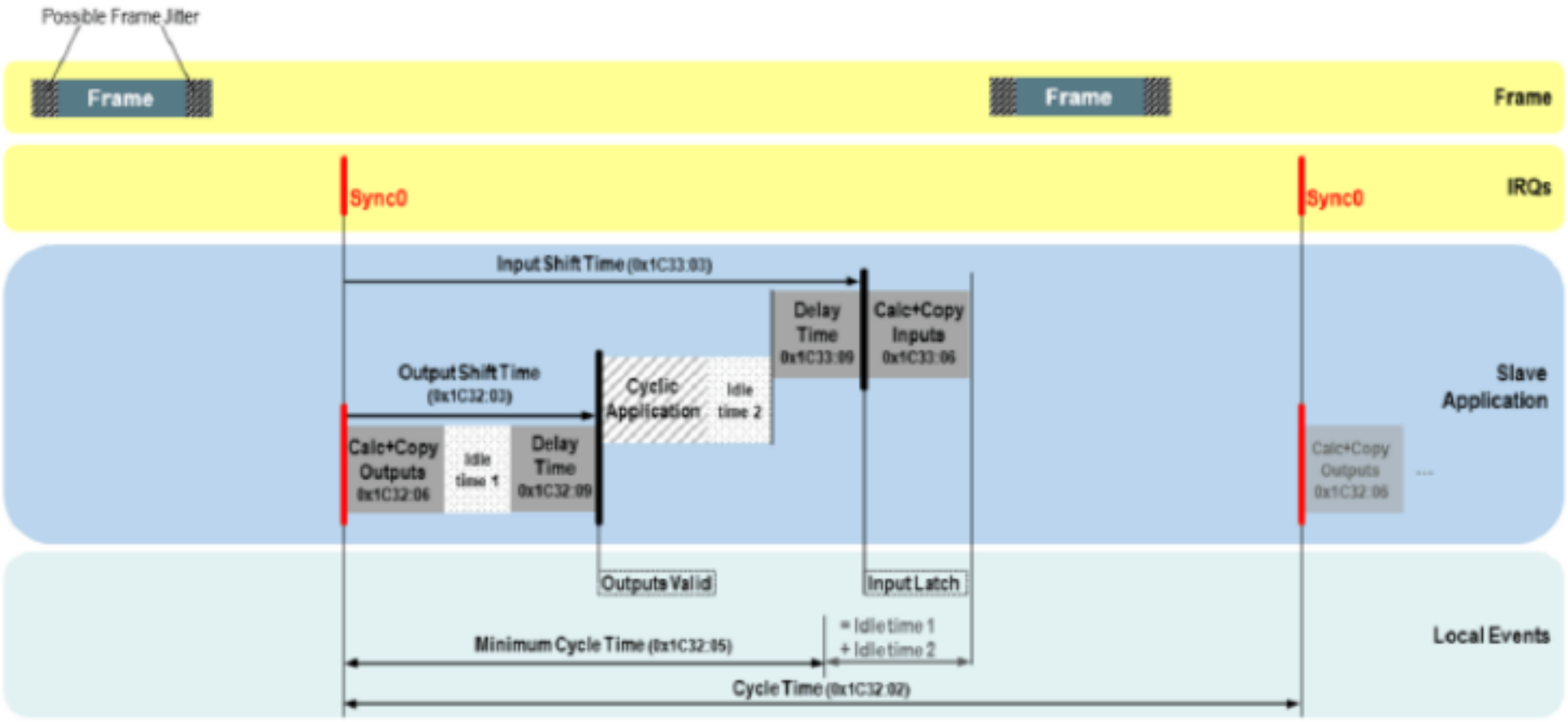
Sync0 Event (Shift of Outputs Valid and/or Input Latch via 0x1C3x:03)



© EtherCAT Technology Group

DC-Synchronous

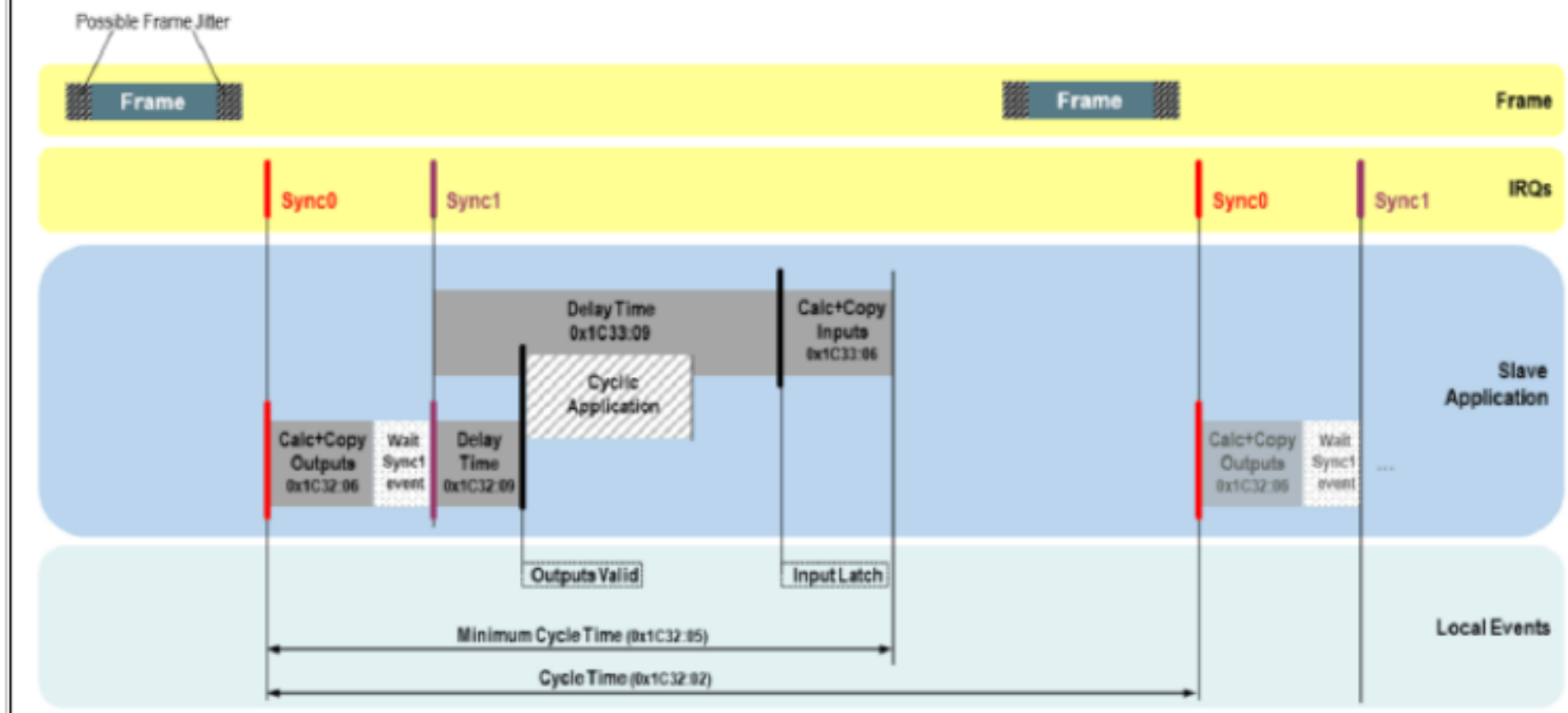
Sync0 Event (Shift of Outputs Valid and/or Input Latch via 0x1C3x:03)



© EtherCAT Technology Group

DC-Synchronous

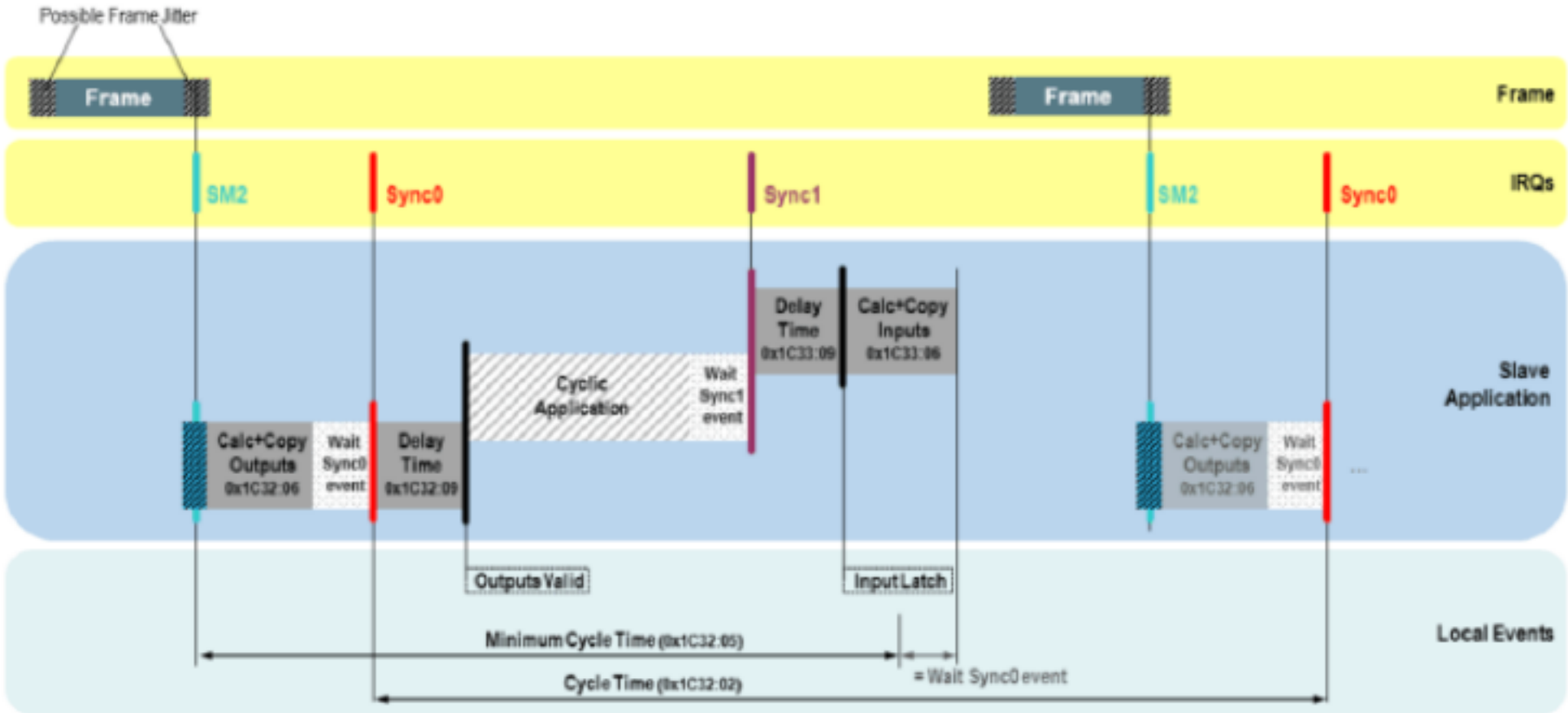
Sync0 Event, Sync1 Event



© EtherCAT Technology Group

DC-Synchronous

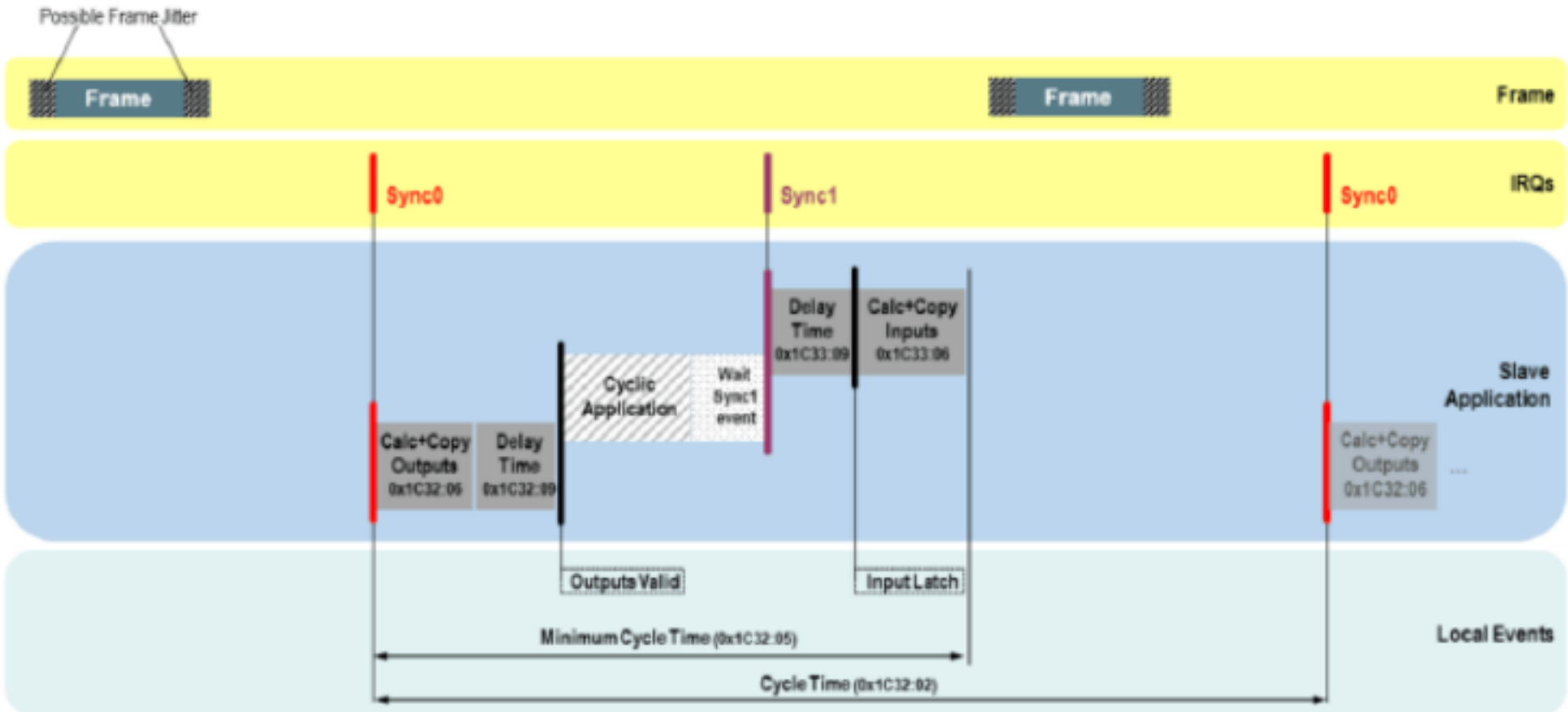
SM Event, Sync0 Event (Shift of Input Latch via Sync1 Event)



© EtherCAT Technology Group

DC-Synchronous

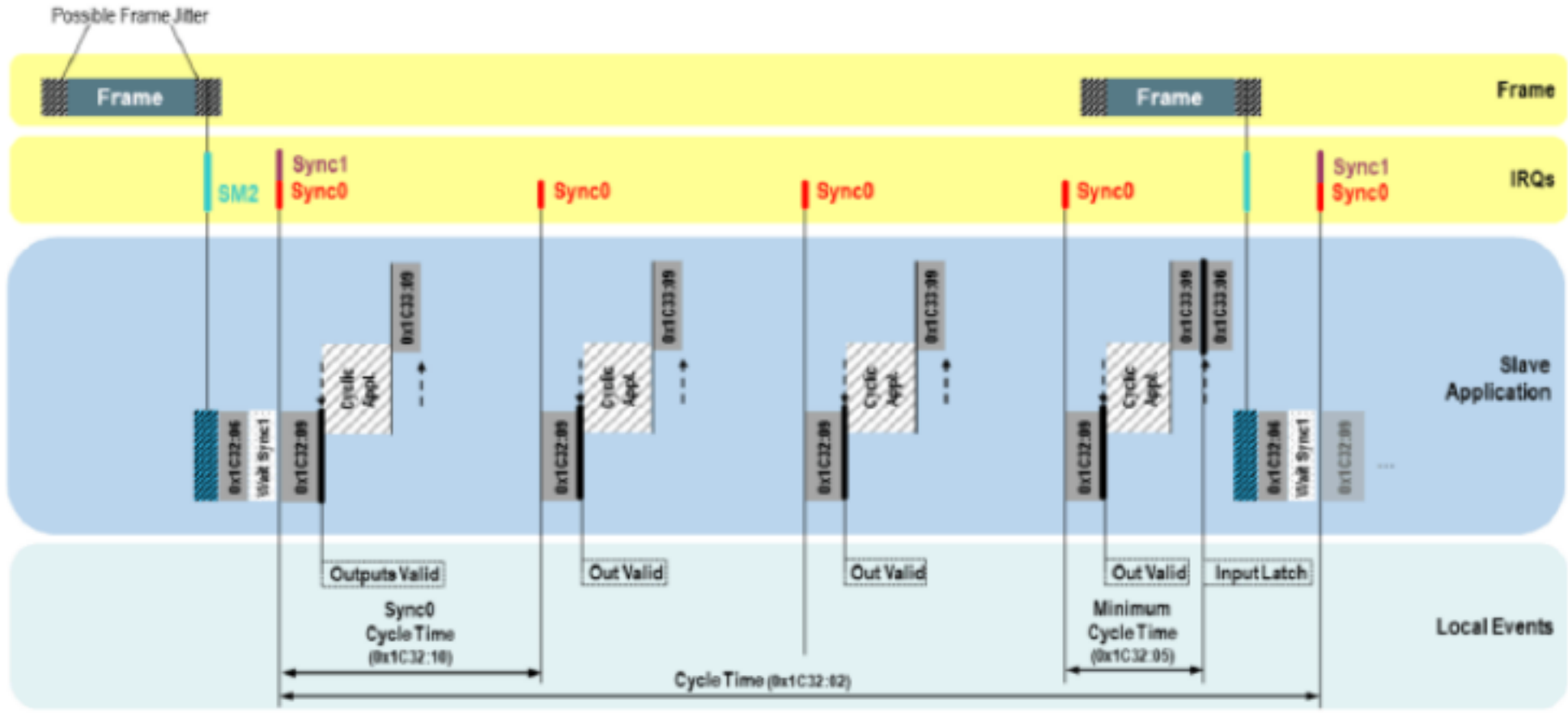
Sync0 Event (Shift of Input Latch via Sync1 Event)



© EtherCAT Technology Group

DC-Synchronous

SM Event, Sync1 Event (subordinated cycles via Sync 0)

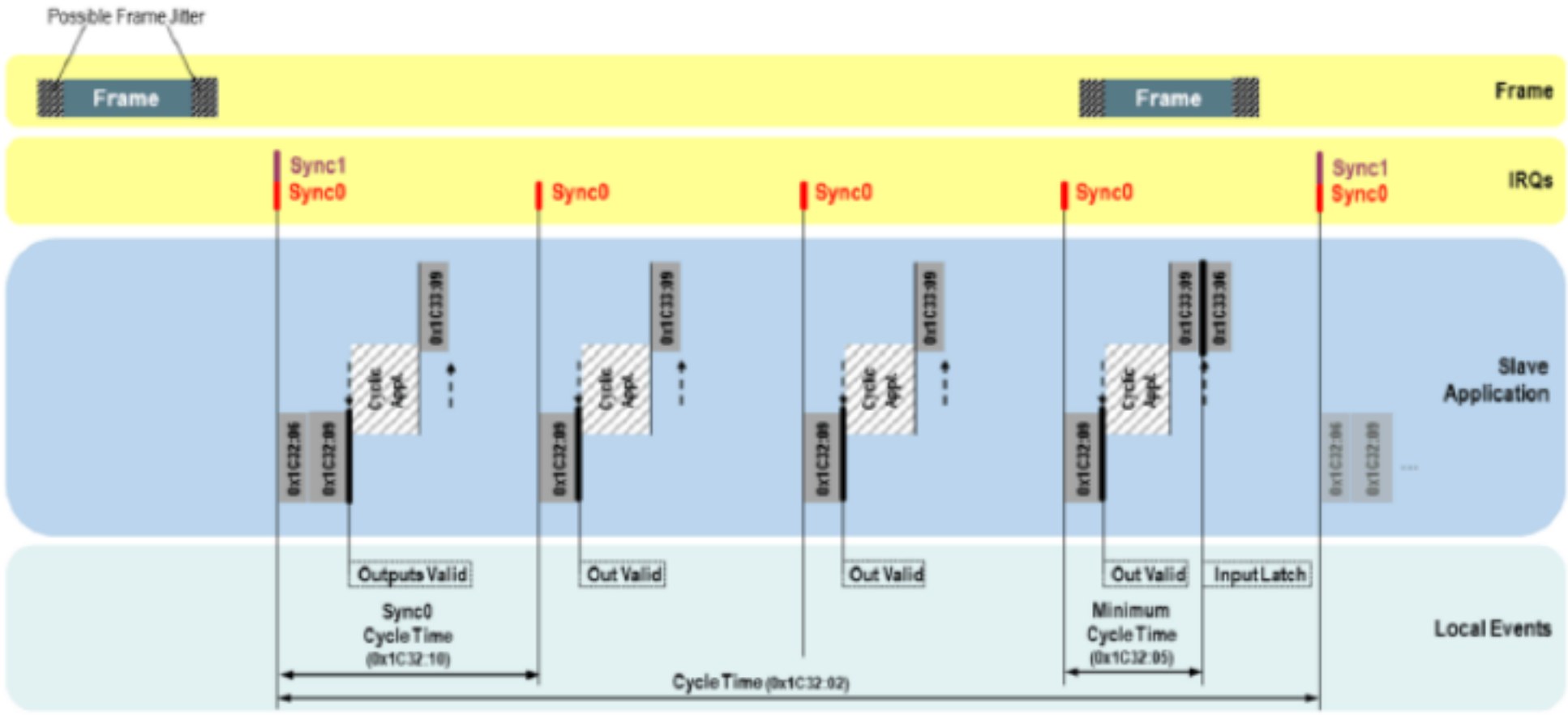


Possible Output Valid /
Input Latch

© EtherCAT Technology Group

DC-Synchronous

Sync1 Event (subordinated cycles via Sync 0)



Possible Output Valid / Input Latch

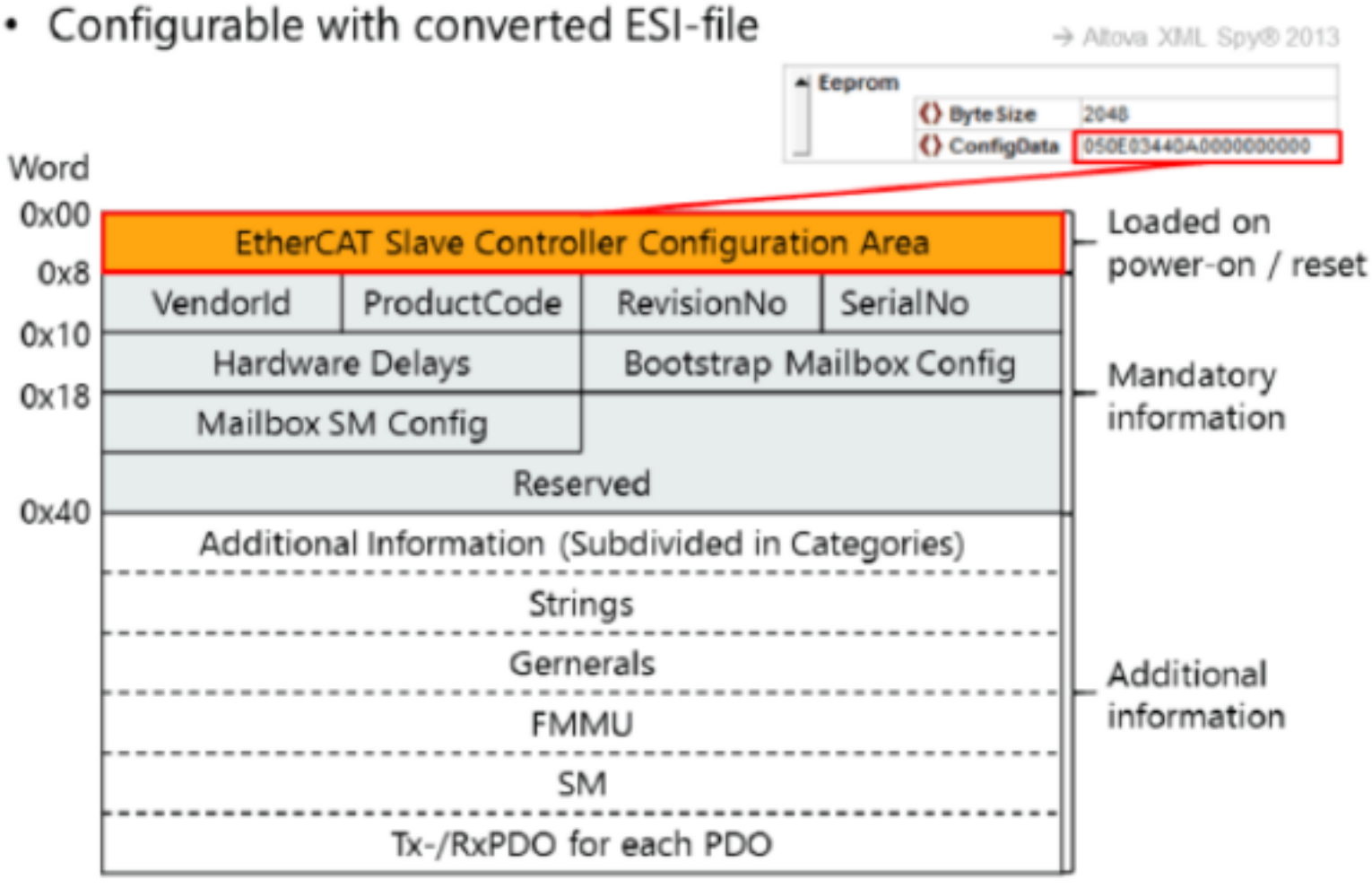
© EtherCAT Technology Group

SII | Slave Information Interface

© EtherCAT Technology Group

SII | Slave Information Interface

- Contains basic configuration information (mandatory)
  - Physical: EEPROM up to 4 Mbit
  - Emulated: ESC related
- Configurable with converted ESI-file

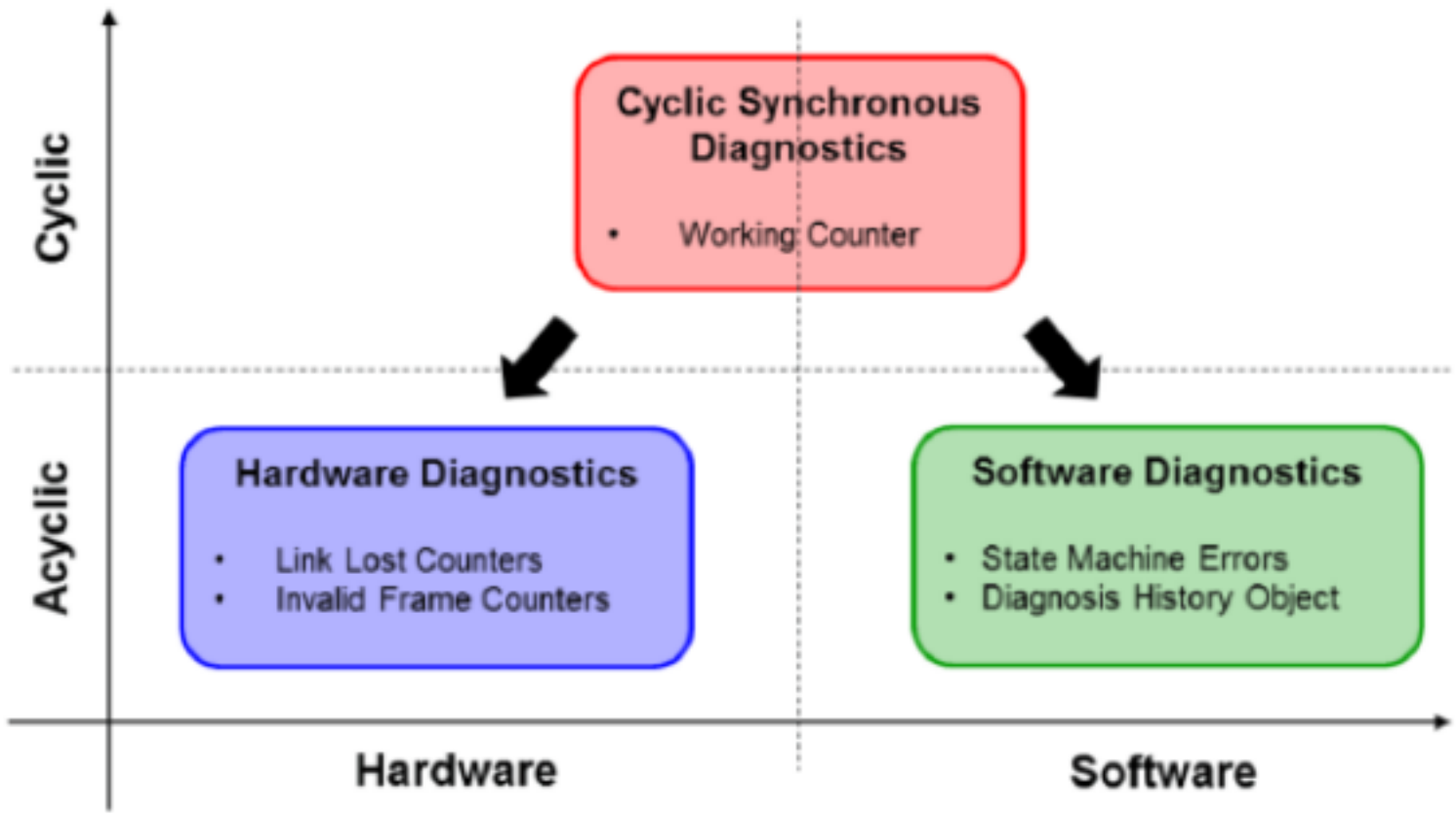


© EtherCAT Technology Group

Specified in ETG.2010

EtherCAT Diagnosis Features

EtherCAT provides extensive **diagnostic information** both at hardware and at software level.



© EtherCAT Technology Group

EtherCAT Diagnosis Features

Hardware Diagnostics (Physical + Datalink Layer)

- Link Lost Counters
  → incremented every time a port loses the physical link
- Invalid Frame Counters
  → incremented every time a port detects a corrupted frame

Software Diagnostics (Application Layer)

- State Machine Errors
  → are triggered in case of EtherCAT generic software errors
- Diagnosis History Object
  → enables slave to report application specific errors

© EtherCAT Technology Group

EtherCAT Diagnosis Features

Additional information about EtherCAT diagnostic features is available in the "EtherCAT Diagnosis for Users" document, available in different languages on the ETG website:

EtherCAT Diagnosis for Users

This slide set intends to provide an overview over the diagnostic capabilities provided by EtherCAT. It contains a description of the basic diagnosis functionalities and the most typical error scenarios within an EtherCAT network.

It is primarily intended for end users and plant operators, as well as for machine builders and system integrators. For additional information about EtherCAT diagnostics -including more detailed error scenarios –which could be of interest for EtherCAT master and slave manufacturers, please refer to slide set "EtherCAT Diagnosis For Developers".

For comments regarding the slides please contact info@ethercat.org.

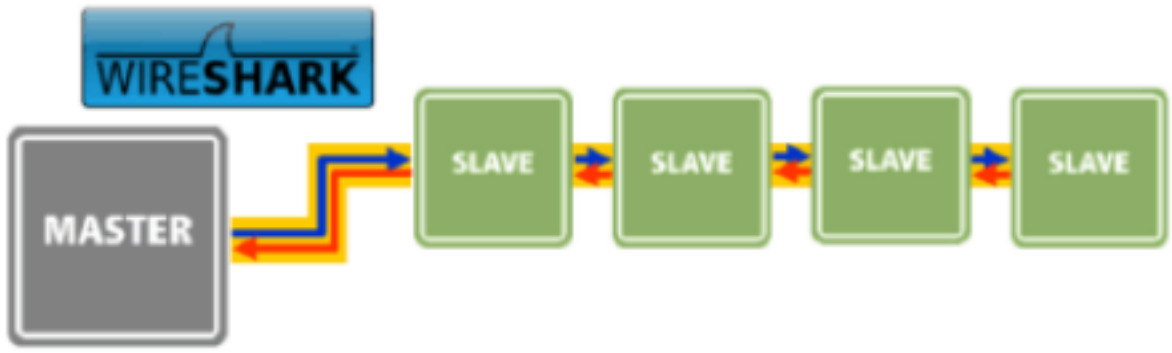
	Description	Language	Type	Date	Size	Ver.	Status
	EtherCAT Diagnosis for Users	EN	PDF	Sep 21, 2017	1,59 MB		
	ユーザ向けEtherCAT診断方法	JP	PDF	Jan 12, 2018	1,55 MB		
	Diagnóstico EtherCAT para usuarios	ES	PDF	Sep 21, 2017	1,58 MB		
	针对用户的EtherCAT诊断	CN	PDF	Jun 30, 2017	1,86 MB		
	EtherCAT-Diagnose für Anwender	DE	PDF	Sep 21, 2017	1,58 MB		
	Diagnostica EtherCAT per Utilizzatori	IT	PDF	Sep 21, 2017	1,60 MB		

© EtherCAT Technology Group

Monitoring of EtherCAT Communication

EtherCAT traffic can be traced via freely available software tools like Wireshark.

Network Monitor*

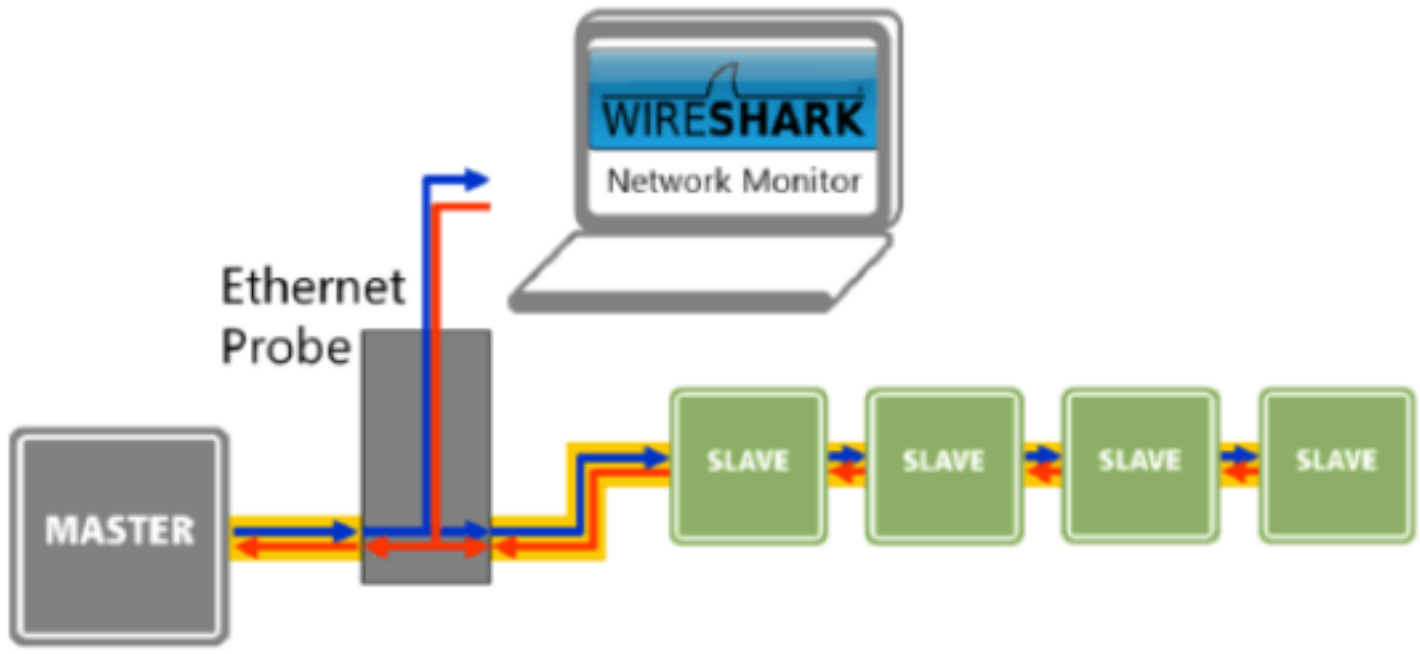


- Masters sends an EtherCAT Frame (broadcast)
  → Monitor gets the 1st copy (unprocessed)
- Frame returns from EtherCAT Slave Devices
  → Monitor gets the 2nd copy (processed)

*Attention: At low cycle times order of frames might be mixed because of timing restrictions within NDIS protocol driver.

© EtherCAT Technology Group

Monitoring of EtherCAT Communication

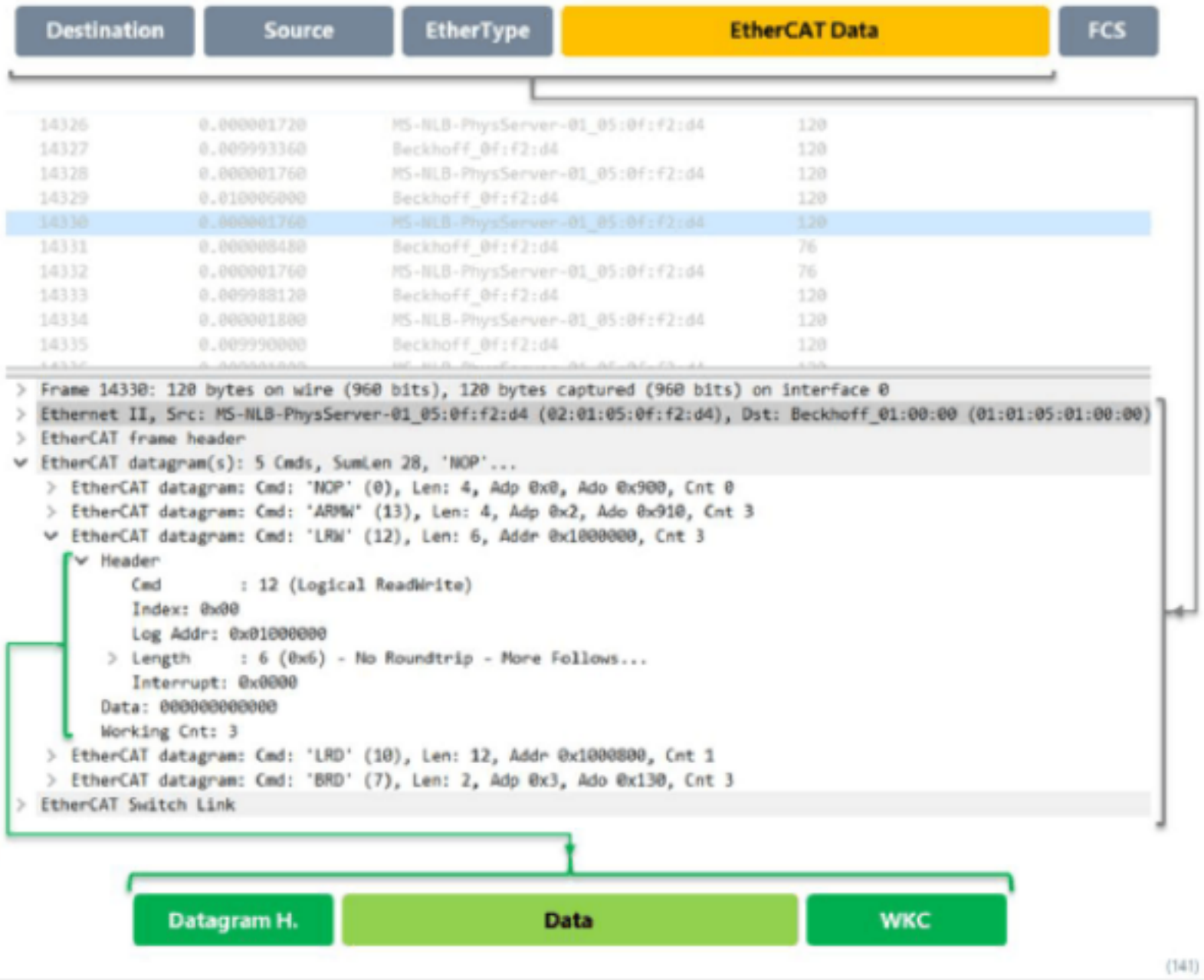


- Masters sends an EtherCAT Frame (broadcast)
  → Monitor gets the 1st copy (unprocessed) w/ Timestamp
- Frame returns from EtherCAT Slave Devices
  → Monitor gets the 2nd copy (processed) w/ Timestamp

☑ Real-time performance is not affected (no jitter, no delay)

© EtherCAT Technology Group

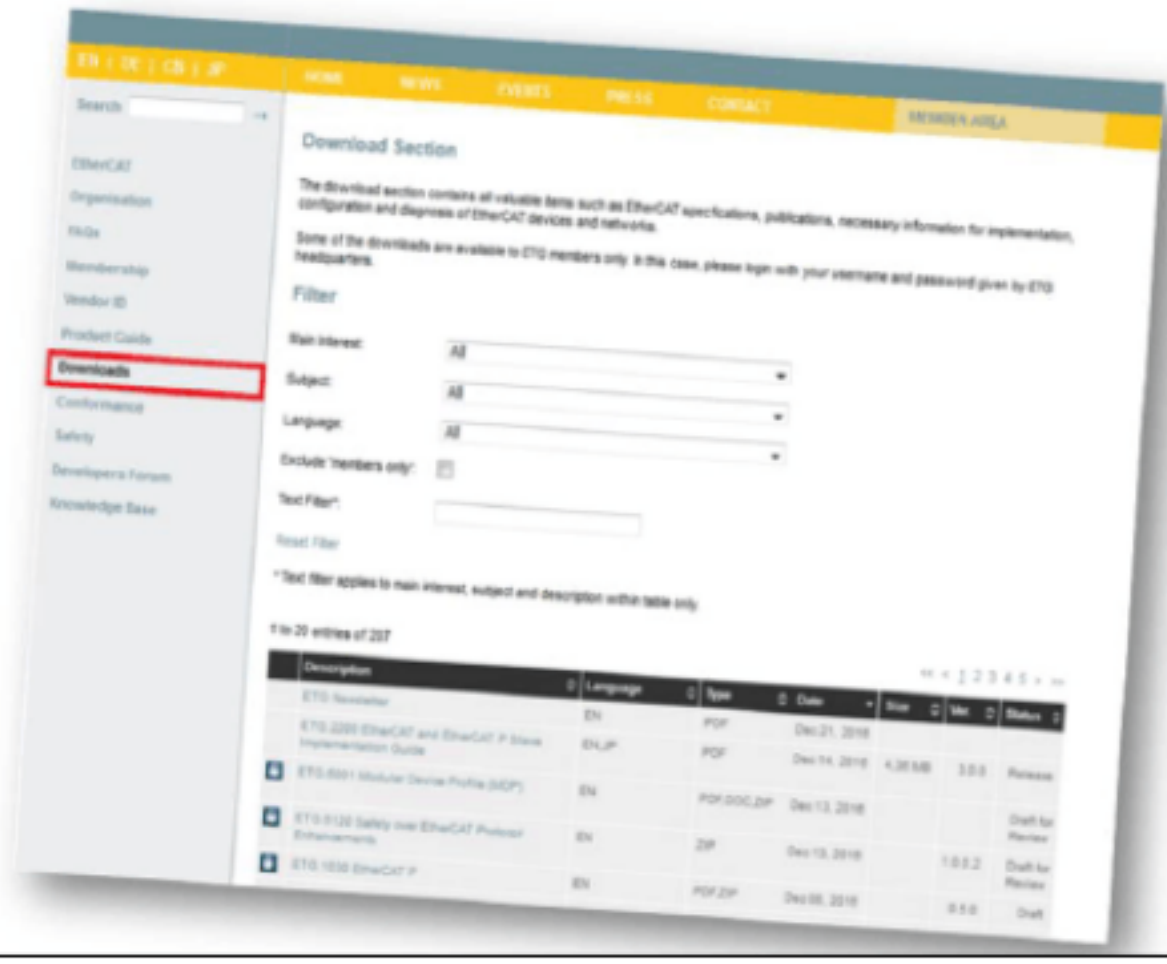
Monitoring of EtherCAT Communication



© EtherCAT Technology Group

Standards and References

- Specification of EtherCAT is done in the EtherCAT Technology Group (ETG)
- Specifications available at www.ethercat.org/download
- International standardization



© EtherCAT Technology Group

Standards and References

- EtherCAT Knowledge Base
  www.ethercat.org/kb
  - Glossary to find documentation
  - Technical descriptions which explain or supplement the technical specification
  - How-To's and answers

- EtherCAT Developers Forum
  www.ethercat.org/forum
  - Crossroad of EtherCAT experts and beginners
  - Multiple topics covering all EtherCAT aspects

- Please help us to keep response times short – check our online information first

© EtherCAT Technology Group

IEC 61158 / ETG.1000

- Digital data communication for measurement and control – Fieldbus for use in industrial control systems
- *The* communication standard
- EtherCAT is named Type 12 in IEC 61158 (no brand names allowed)
- Transformation of the communication protocol to a common model
- ETG document with same content: **ETG.1000**

© EtherCAT Technology Group

IEC 61158 – DL/AL services and protocols

Users	Implementers
AL Services **Part 5 in IEC 61158** • Model and Concepts • Data type definitions • Application Objects • Service description • Communication Management	**AL Protocol** **Part 6 in IEC 61158** • Syntax definition and Coding • Application Relationship Procedures • State Machines
DL Services **Part 3 in IEC 61158** • Model and Concepts • Service description • Register Description (DL objects)	**DL Protocol** **Part 4 in IEC 61158** • Coding • Medium Access • State Machines

© EtherCAT Technology Group

Further EtherCAT standards

Additional important specifications are:

- **ETG.1300** Indicator and Labeling specification
  - defines the implementation of indicators signaling the EtherCAT communication state, errors and the link status.
  - the location, labeling and blink codes of the indicators are defined
  - defines the labeling of the EtherCAT Ports
- **ETG.9001** Marking Rules
  - specifies the marking rules for products and the corresponding documentation using the EtherCAT technology
  - Use of trademarks and logo

© EtherCAT Technology Group

These specifications can be found on the ETG Website
http://www.ethercat.org/en/publications.html#members_area

EtherCAT P | Related documentation - ETG

Specifications:

ETG.1030
EtherCAT P

ETG.1000.2P
Physical Layer

ETG.1030.1
EtherCAT P Connector

ETG.2000
EtherCAT Slave Information
- ESI Schema

ETG.9001
Marking Rules

- Landing site: www.ethercat.org/p

Overview in:

EtherCAT and EtherCAT P Implementation Guide (ETG.2200)

© EtherCAT Technology Group

EtherCAT Vendor ID

© EtherCAT Technology Group

Vendor ID in IEC 61158-6-12 (EtherCAT Standard)

5.6.7.4.6 Identity Object

The Identity Object dictionary entry (index 0x1018) is specified in Table 72.

Table 72 – Identity Object

Sub-Index	Description	Data type	M/O/C	Access	PDO Mapping	Value
0	Number of entries	UNSIGNED8	M	R	No	4
1	Vendor ID	UNSIGNED32	M	R	No	Assigned uniquely by ETG
2	Product Code	UNSIGNED32	M	R	No	Assigned uniquely by Vendor
3	Revision Number	UNSIGNED32	M	R	No	Assigned uniquely by Vendor Bit 0-15: Minor Revision Number of the device Bit 16-31: Major Revision Number of the device
4	Serial Number	UNSIGNED32	M	R	No	Assigned uniquely for this device by Vendor 0 if there is no serial number given

© EtherCAT Technology Group

Vendor ID Principles

- End users do not have to be ETG member and do not need an ETG vendor ID – even though they are welcome to join ETG and also may get a vendor ID.
- Each vendor of an EtherCAT Device shall be an ETG member and shall obtain and maintain a Vendor ID from the EtherCAT Technology Group.
- The Vendor ID is free of charge.
- The vendor shall implement the Vendor ID in each EtherCAT Device prior to making it available on the market.
- Manufacturers of Communication Devices shall distribute those Communication Devices using their Secondary Vendor ID (range 0xE0000000:0xEFFFFFFF). Optionally they may also use their Vendor ID, e.g. for conformance testing. Use of a Secondary Vendor ID in an Automation Device is prohibited.

© EtherCAT Technology Group

How to obtain an EtherCAT Vendor ID?

https://www.ethercat.org/memberarea/vendorid/

EtherCAT Technology Group

EN | CN | JP — HOME NEWS EVENTS PRESS CONTACT

Search →

EtherCAT
Organisation
FAQs
Membership
Vendor ID
Product Guide
Downloads
Conformance
EtherCAT P
Safety
Developers Forum
Knowledge Base
Login/Logout

Vendor ID Assignment Form

Each EtherCAT compliant device has to implement the worldwide unique Vendor ID assigned by ETG. The EtherCAT Vendor ID Usage is governed by the ETG Vendor ID Policy and the corresponding Vendor ID agreement, which asks for conformance with the EtherCAT specifications. In order to clarify this policy, the general rules are repeated below:

- Each vendor of an EtherCAT device has to be an ETG member and has to obtain and maintain a valid Vendor ID from the EtherCAT Technology Group.
- The Vendor ID is free of charge.
- The vendor has to implement the Vendor ID in each EtherCAT device prior to making it available on the market.
- Manufacturers of machines, which integrate and use EtherCAT devices in combination with or in such machines, are not required to apply for and use a Vendor ID.

Vendor ID assignment is free of charge.

(→) required field

Company: Beckhoff Automation GmbH & Co. KG

☐ Please check this box and fill out „Department/Branch" field only, if you need several Vendor IDs for your company. In this case please pick a specific name, i.e. "Germany", "Medical", etc. Names like "R&D", "Development" or "Manufacturing" may not be accepted due to their broad range within the company. Note that bit 24 ..27 will be assigned automatically by ETG when "Department/Branch" field is filled out.

31	28 27	24 23	0
Reserved	Department	Company	
MSB			LSB

Department/Branch:
Address: →
ZIP: →
City: →
Country: →
Contact Person: →
Phone: →
Fax:
Email: →

© EtherCAT Technology Group

EtherCAT Vendor ID Agreement

- Applicants for an EtherCAT Vendor ID have to accept the Vendor ID Agreement

- **The Vendor ID Agreement:**
- Demands Conformance for EtherCAT products
- Governs the use of the EtherCAT Trademarks (including reference to the EtherCAT Marking Rules)
- Contains a Disclaimer (“Technology provided “as is”, with no warranty implied…)


EtherCAT Technology Group Vendor ID Usage Agreement

© EtherCAT Technology Group

Conformance Testing
+
Product Certification

© EtherCAT Technology Group

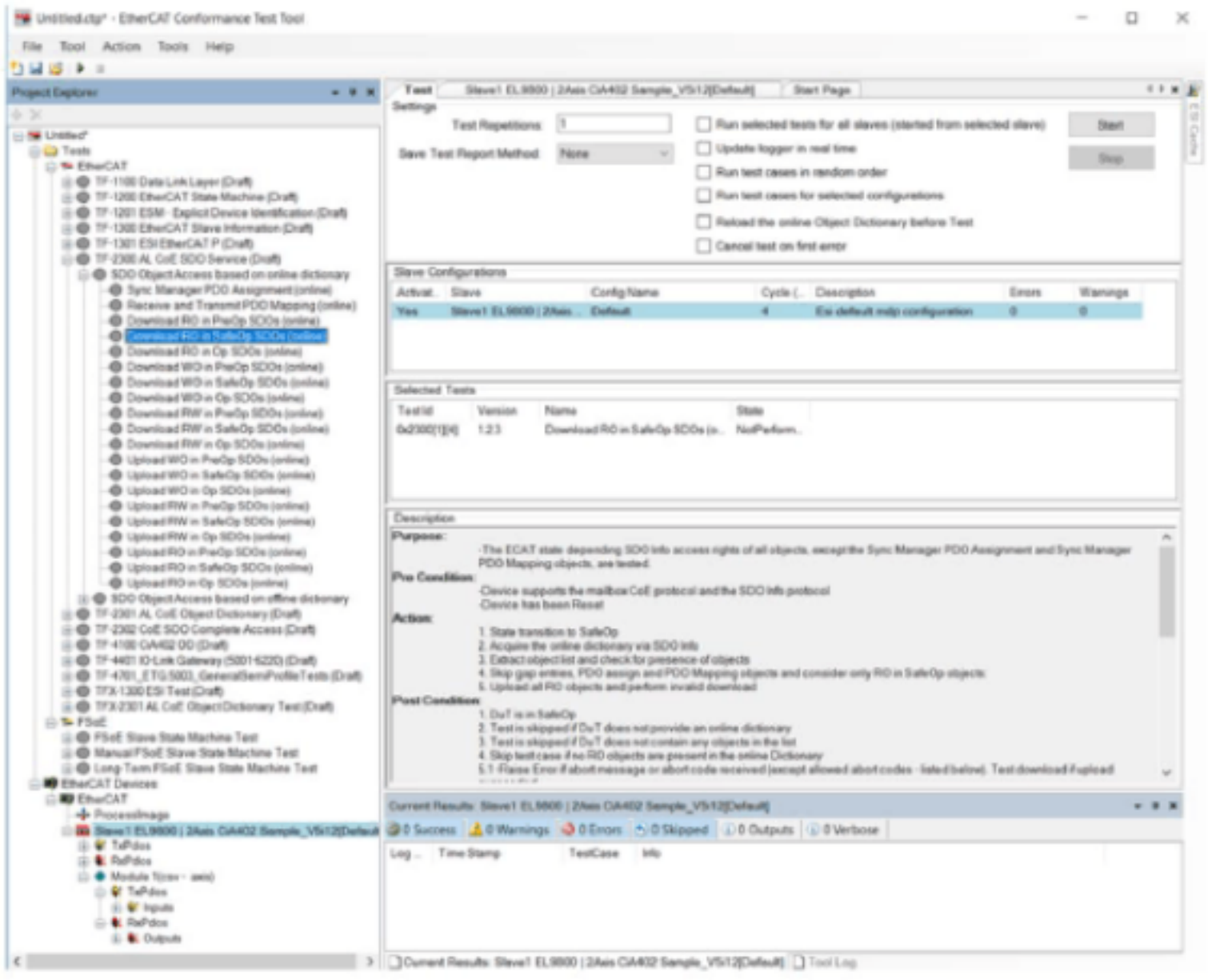
Conformance Test Tool (CTT)

- Test Tool
  - Validates conformity for protocol layer
  - Helps to find errors during development due to detailed error description
  - Helps to improve support: detailed information saved with CTT project file

- I am an EtherCAT device vendor. Do I have to license the conformance test tool?
  - Yes. The ETG takes conformance very seriously, and the availability of the conformance test tool at each and every device vendors R&D lab is an important cornerstone in this process

© EtherCAT Technology Group

Conformance Test Tool (CTT)



© EtherCAT Technology Group

Frequently Asked Questions (I)

- Do I have to submit my EtherCAT device to the EtherCAT Test Center for testing?
  - No. Conformance Testing with the Test Tool „at home" is sufficient to meet the minimum requirements of the Vendor ID agreement.

- Can I get a Conformance Certificate based on the test results obtained in my R&D lab?
  - No. The Conformance Certificate can only be issued after successfully passing the test at an accredited EtherCAT Test Center.

- Does the test in the EtherCAT Test Center exceed the test done with the Conformance Test Tool (at home)?
  - Yes. The test in the EtherCAT Test Center also includes an interoperability test, checking for conformance regarding the indicator and labeling spec, the marking rules etc.

© EtherCAT Technology Group

Frequently Asked Questions (II)

- Does the Certificate expire?
  - No. The Certificate confirms that a device of a certain release has passed the current test version in the EtherCAT Test Center. Of course the Certificate can neither confirm that all future releases of the device will also pass, nor that the current device release will pass all future enhancements of the EtherCAT Conformance Test.
- Do I have to submit my device again once I released a newer version?
  - No. However, according to the Vendor ID agreement, you will have to test future releases of your product against the conformance test tool in your R&D lab. Of course you may also submit your device again to the EtherCAT Test Center and obtain a new certificate!

© EtherCAT Technology Group

Conformance Testing Procedure

- Vendor contacts ETG office if he wants EtherCAT CT
- ETG checks Vendor ID and provides Test Contract – which also allows one to select the EtherCAT Conformance Test Center (ETC).
- Based on choice of vendor, ETG office forwards request to ETC
- ETC provides formal offer to vendor (ETG is not involved in any financial transaction)
- ETC provides checklist to vendor (how to prepare, what to send, etc.)
- Vendor sends device to ETC (or brings it there) .
- ETC tests device.
- ETC sends Test Report to Vendor and to ETG office.
- If test was passed successfully, ETG provides Certificate.

© EtherCAT Technology Group

Conformance Test Mark

- Devices that have passed the ETC Conformance Test may carry the official conformance test mark
- End users are encouraged to include the availability of the conformance test mark in their vendor and device selection process.

EtherCAT Conformance tested

© EtherCAT Technology Group

Official Conformance Test Centers

Savage, MN

EtherCAT Test Center
Beckhoff Automation LLC
13130 Dakota Ave
Savage, MN 55378, USA
Tel.: +1 952-428-7399
etc-na@beckhoff.com

Beijing

EtherCAT Test Center
Beihang University
Laboratory of Numerical Control and Automation (LNC)
37 Xueyuan Road Haidian District
100191 Beijing, China
Tel.: +86 10 82 33 97 17
etc-china@buaa.edu.cn

Nuremberg

EtherCAT Test Center
Beckhoff Automation GmbH
Ostendstraße 196
90482 Nuremberg, Germany
Tel.: +49 (911) 5 40 56 20
Fax: +49 (911) 5 40 56 29
etc@beckhoff.com

Kyoto

EtherCAT Test Center
ASTEM RI
134 Chudoji Minami-machi, Shimogyo,
Kyoto 600-8813 Japan
Tel.: +81 (75) 366 0143
Fax: +81 (75) 325 3024
etc@testlab.astem.or.jp

© EtherCAT Technology Group

Thank you for your attention!
Questions?

EtherCAT Technology Group
Headquarters
Ostendstraße 196
90482 Nuremberg, Germany

Phone: +49 911 54056 20
Fax: +49 911 54056 29
Email: info@ethercat.org
conformance@ethercat.org
(for conformance testing)
Web: http://www.ethercat.org