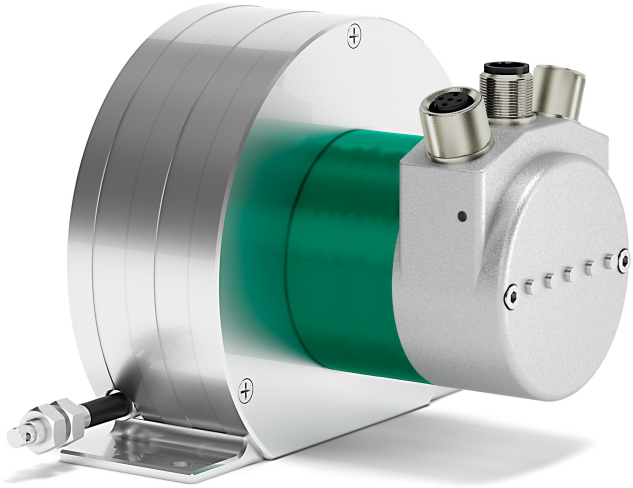
Smart encoders & actuators

User's guide

SFA-5000-EC
SFA-10000-EC



EtherCAT®

in compliance with ETG.1000

- 5000 mm & 10000 mm draw-wire encoder
- Integrated 27 bit absolute multiturn encoder
- Programmable resolution down to 24 µm
- M12 connectors
- Implements the CoE protocol and the EtherCAT State Machine

Suitable for the following models:

- SFA-5000-EC
- SFA-10000-EC

General Contents

This publication was produced by Lika Electronic s.r.l. 2018. All rights reserved. Tutti i diritti riservati. Alle Rechte vorbehalten. Todos los derechos reservados. Tous droits réservés.

This document and information contained herein are the property of Lika Electronic s.r.l. and shall not be reproduced in whole or in part without prior written approval of Lika Electronic s.r.l. Translation, reproduction and total or partial modification (photostat copies, film and microfilm included and any other means) are forbidden without written authorisation of Lika Electronic s.r.l.

The information herein is subject to change without notice and should not be construed as a commitment by Lika Electronic s.r.l. Lika Electronic s.r.l. reserves the right to make all modifications at any moments and without forewarning.

This manual is periodically reviewed and revised. As required we suggest checking if a new or updated edition of this document is available at Lika Electronic s.r.l.'s website. Lika Electronic s.r.l. assumes no responsibility for any errors or omissions in this document. Critical evaluation of this manual by the user is welcomed. Your comments assist us in preparation of future documentation, in order to make it as clear and complete as possible. Please send an e-mail to the following address info@lika.it for submitting your comments, suggestions and criticisms.

lika

General contents

Subject Index

Typographic and iconographic conventions

In this guide, to make it easier to understand and read the text the following typographic and iconographic conventions are used:

- parameters and objects both of the device and the interface are coloured in GREEN;
- alarms are coloured in RED;
- states are coloured in FUCSIA.

When scrolling through the text some icons can be found on the side of the page: they are expressly designed to highlight the parts of the text which are of great interest and significance for the user. Sometimes they are used to warn against dangers or potential sources of danger arising from the use of the device. You are advised to follow strictly the instructions given in this guide in order to guarantee the safety of the user and ensure the performance of the device. In this guide the following symbols are used:

	This icon, followed by the word **WARNING**, is meant to highlight the parts of the text where information of great significance for the user can be found: user must pay the greatest attention to them! Instructions must be followed strictly in order to guarantee the safety of the user and a correct use of the device. Failure to heed a warning or comply with instructions could lead to personal injury and/or damage to the unit or other equipment.
	This icon, followed by the word **NOTE**, is meant to highlight the parts of the text where important notes needful for a correct and reliable use of the device can be found. User must pay attention to them! Failure to comply with instructions could cause the equipment to be set wrongly: hence a faulty and improper working of the device could be the consequence.
	This icon is meant to highlight the parts of the text where suggestions useful for making it easier to set the device and optimize performance and reliability can be found. Sometimes this symbol is followed by the word **EXAMPLE** when instructions for setting parameters are accompanied by examples to clarify the explanation.

Preliminary information

This guide is designed to provide the most complete and exhaustive information the operator needs to correctly and safely install and operate the **SFA-5000 and SFA-10000 absolute draw-wire encoders with EtherCAT interface**.
The cable pulling mechanism integrates a 13 x 14 bit absolute multiturn encoder (13 bits = singleturn resolution = 8,192 cpr; 14 bits = 16,384 revolutions).

SFA-5000 / SFA-10000 cable-pulling encoder is aimed at speed and position measurements and controls in a variety of industrial applications through the movement of a **5,000 mm (196.85") or 10,000 mm (393.7")** stainless steel wire. The typical back and forth travel of the moving equipment causes the wire to reel and unreel and thus the linear movement to be converted into a rotary motion detected by the encoder which is coupled to the drum.
The stroke per turn is always 200 mm (7.874"), the maximum number of turns is 25 for SFA-5000 and 50 for SFA-10000.

To make it easier to read and understand the text, this guide is divided into two main sections.
In the first section some general information concerning the safety, the mechanical installation and the electrical connection as well as tips for setting up and running properly and efficiently the SFA-5000 / SFA-10000 cable-actuated encoder are provided.
In the second section, entitled **EtherCAT Interface**, you can find detailed information on the EtherCAT interface. In this section the interface features and the objects implemented in the unit are fully described.

Glossary of EtherCAT terms

EtherCAT, like many other networking systems, has a set of unique terminology. Table below contains a few of the technical terms used in this guide to describe the EtherCAT interface. They are listed in alphabetical order.

Acknowledge telegram (AT)	Telegram, in which each Slave inserts its data.
Actual value	Value of a variable at a given instant.
Algorithm	Completely determined finite sequence of operations by which the values of the output data can be calculated from the values of the input data.
Application	Function or data structure for which data is consumed or produced. Software functional element specific to the solution of a problem in industrial-process measurement and control.
Application class	Configuration of a Drive Object with a set of functional objects and supported by standard telegrams.
Application mode	Type of application that can be requested from a PDS.
Application objects	Multiple object classes that manage and provide a run time exchange of messages across the network and within the network device.
Application process	Part of a distributed application on a network, which is located on one device and unambiguously addressed.
Application relationship	Cooperative association between two or more application-entity-invocations for the purpose of exchange of information and coordination of their joint operation. This relationship is activated either by the exchange of application-protocol-data-units or as a result of preconfiguration activities.
Attribute	Description of an externally visible characteristic or feature of an object, property or characteristic of an entity. The attributes of an object contain information about variable portions of an object. Typically, they provide status information or govern the operation of an object. Attributes may also affect the behaviour of an object. Attributes are divided into class attributes and instance attributes.
Axis	Logical element inside an automation system (e.g. a motion control system) that represents some form of movement.
Basic Slave	Slave device that supports only physical addressing of data.
Behaviour	Indication of how an object responds to particular events.
Bit	Unit of information consisting of a 1 or a 0. This is the smallest data unit that can be transmitted.

CANopen	Application layer protocol as defined in EN 50325-4.
Channel	Representation of a single physical or logical management object of a Slave to control conveyance of data.
CIP™	Common Industrial Protocol (see IEC 61158 Type 2, IEC 61784-1 and IEC 61784-2 CPF2).
Class	Description of a set of objects that share the same attributes, operations, methods, relationships, and semantics.
Client	Object which uses the services of another (Server) object to perform a task. Initiator of a message to which a Server reacts.
Clock synchronization	Representation of a sequence of interactions to synchronize the clocks of all time receivers by a time Master.
Commands	Set of commands from the application control program to the PDS to control the behaviour of the PDS or functional elements of the PDS.
Communication cycle	Accumulation of all telegrams between two Master synchronization telegrams.
Communication object	Component that manages and provides a run time exchange of messages across the network.
Connection	Logical binding between two application objects within the same or different devices.
Consume	Act of receiving data from a provider.
Consumer	Node or sink receiving data from a provider.
Control	Purposeful action on or in a process to meet specified objectives.
Control device	Physical unit that contains – in a module/subassembly or device – an application program to control the PDS.
Control unit	Control device.
Control word	Two adjacent bytes inside the Master data telegram containing commands for the addressed drive.
Controller	Controlling device which is associated with one or more drives (axes) a host for the overall automation.
Conveyance path	Unidirectional flow of APDUs across an application relationship.
Cycle time	Time span between two consecutive cyclically recurring events.
Cyclic	Events which repeat in a regular and repetitive manner.
Cyclic data	Part of the telegram which does not change its meaning during cyclic operation of the interface. High priority real-time data that is transferred by a CIP Motion connection on a periodic basis.
Data	Generic term used to refer to any information carried over a

	fieldbus.
Data consistency	Means for coherent transmission and access of the input-or output-data object between and within Client and Server.
Data exchange	Demand dependent; non cyclic transmission (service channel).
Data type	Relation between values and encoding for data of that type according to the definitions of IEC 61131-3. Set of values together with a set of permitted operations.
Data type object	Entry in the object dictionary indicating a data type.
Default gateway	Device with at least two interfaces in two different IP subnets acting as router for a subnet.
Device	Field device. Networked independent physical entity of an industrial automation system capable of performing specified functions in a particular context and delimited by its interfaces. Entity that performs control, actuating and/or sensing functions and interfaces to other such entities within an automation system. Physical entity connected to the fieldbus composed of at least one communication element (the network element) and which may have a control element and/or a final element (transducer, actuator, etc.).
Device profile	Collection of device dependent information and functionality providing consistency between similar devices of the same device type. Representation of a device in terms of its parameters and behaviour according to a device model that describes the device's data and behaviour as viewed through a network, independent from any network technology.
Diagnosis information	All data available at the Server for maintenance purposes.
Distributed clocks	Method to synchronize Slaves and maintain a global time base.
DL	Data-link-layer.
DLPDU	Data-link-protocol-data-unit.
Drive Object	Functional element of a Drive Unit.
Drive Unit	Logical device which comprises all functional elements related to one central processing unit.
Error	Discrepancy between a computed, observed or measured value or condition and the specified or theoretically correct value or condition.
Error class	General grouping for related error definitions and corresponding error codes.
Error code	Identification of a specific type of error within an error class.
EtherCAt State Machine	EtherCAT Slave is a state machine; communication and operating characteristics depend on the current state of the

	device.
Event	Instance of a change of conditions.
Event data	Medium priority real-time data that is transferred by a CIP Motion connection only after a specified event occurs.
Feed forward	Command value used to compensate the lag in the control loop.
Feedback variable	Variable which represents a controlled variable and which is returned to a comparing element.
Fieldbus memory management unit	Function that establishes one or several correspondences between logical addresses and physical memory.
Fieldbus memory management unit entity	Single element of the fieldbus memory management unit: one correspondence between a coherent logical address space and a coherent physical memory location.
Frame	Denigrated synonym for DLPDU.
FreeRun	Asynchronous communication mode.
Full Slave	Slave device that supports both physical and logical addressing of data.
Functional element	Entity of software or software combined with hardware, capable of accomplishing a specified function of a device.
HMI	Human Machine Interface.
Host	Device that covers the automation functionality of an automation device.
I/O data	Input data and output data that would typically need to be updated on a regular basis (e.g. periodic change of state), such as commands, set-points, status and actual values.
Identification number (IDN)	Designation of operating data under which a data block is preserved with its attribute, name, unit, minimum and maximum input values, and the data.
Index	Address of an object within an application process.
Input data	Data transferred from an external source into a device, resource or functional element.
Interface	Shared boundary between two entities defined by functional characteristics, signal characteristics, or other characteristics as appropriate.
Little endian	Data representation of multi-octet fields where the least significant octet is transmitted first.
Logical power drive system	Model which includes PDS and communication network accessible through the generic PDS interface.
Mapping	Correspondence between two objects in that way that one object is part of the other object.
Mapping parameters	Set of values defining the correspondence between application objects and process data objects.

Master	Device that controls the data transfer on the network and initiates the media access of the Slaves by sending messages and that constitutes the interface to the control system. Node, which assigns the other nodes the right to transmit.
Master data telegram (MDT)	Telegram, in which the Master inserts its data.
Medium	Cable, optical fibre or other means by which communication signals are transmitted between two or more points.
Message	Ordered series of octets intended to convey information. Normally used to convey information between peers at the application layer.
Model	Mathematical or physical representation of a system or a process, based with sufficient precision upon known laws, identification or specified suppositions.
Motion	Any aspect of the dynamics of an axis.
Motion Axis Object	Object that defines the attributes, services, and behaviour of a motion device based axis (or PDS) according to the CIP Motion specification, including Communications, Device Control, and Basic Drive FE elements as defined in IEC 61800-7.
Network	Set of nodes connected by some type of communication medium, including any intervening repeaters, bridges, routers and lower-layer gateways.
Node	Single DL-entity as it appears on one local link. End-point of a link in a network or a point at which two or more links meet [derived from IEC 61158-2].
Object	Abstract representation of a particular component within a device. An object can be: 1. an abstract representation of the capabilities of a device. Objects can be composed of any or all of the following components: ◦ data (information which changes with time); ◦ configuration (parameters for behaviour); ◦ methods (things that can be done using data and configuration); 2. a collection of related data (in the form of variables) and methods (procedures) for operating on that data that have clearly defined interface and behaviour.
Object dictionary	Data structure addressed by Index and Sub-index that contains description of data type objects, communication objects and application objects. List of objects with unique 16-bit index and 8-bit sub-index as defined in EN 50325-4.
Operating cycle	Period of the control loop within the drive or the control unit.
Operating mode	Characterization of the way and the extent to which the human operator intervenes in the control equipment.

Output data	Data originating in a device, resource or functional element and transferred from them to external systems.
P-Device	Field device and the host for the Drive Objects.
Parameter	Data element that represents device information that can be read from or written to a device, for example through the network or a local HMI.
PDO	Process Data Object.
PDS	Power Drive System.
Process data	Collection of application objects designated to be transferred cyclically or acyclically for the purpose of measurement and control.
Process Data Object (PDO)	Communication object with real-time capability. Structure described by mapping parameters containing one or several process data entities.
Producer	Node or source sending data to one or many consumers.
Profile	Representation of a PDS interface in terms of its parameters, parameter assemblies and behaviour according to a communication profile and a device profile.
Protocol	Convention about the data formats, time sequences, and error correction in the data exchange of communication systems.
Reference variable	Input variable to a comparing element in a controlling system which sets the desired value of the controlled variable and is deducted from the command variable.
Resource	Processing or information capability.
Segment	Collection of one real Master with one or more Slaves.
Server	Object which provides services to another (Client) object.
Service	Operation or function than an object and/or object class performs upon request from another object and/or object class.
Service data	Lower priority real-time data associated with a service message from the controller that is transferred by a CIP Motion connection on a periodic basis.
Set-point	Value or variable used as output data of the application control program to control the PDS.
Slave	DL-entity accessing the medium only after being initiated by the preceding Slave or the Master. Node, which is assigned the right to transmit by the Master.
Standard telegram	Set of input data and output data for an application mode.
Status	Set of information from the PDS to the application control program reflecting the state or mode of the PDS or a functional element of the PDS.
Status word	Two adjacent bytes inside the drive telegram containing status information.

Subindex	Sub-address of an object within the object dictionary.
Supervisor	Engineering device which manages provisions of configuration data (parameter sets) and collections of diagnosis data from P-Devices and/or controllers.
Switch	MAC bridge as defined in IEEE 802.1D.
Sync Manager	Sync Manager has the task of synchronizing data transfer between Master and Slave and prevents the same memory area from being written by different events. Collection of control elements to coordinate access to concurrently used objects.
Sync manager channel	Single control elements to coordinate access to concurrently used objects.
Synchronised	Condition where the local clock value on the drive is locked onto the Master clock of the distributed System Time.
Synchronous with DC SYNC0	In this operating mode data is sampled and then copied into Sync Manager buffer simultaneously at SYNC0 event generated by the ESC capture/compare unit.
Synchronous with SM3	In this mode data is sampled and then copied into Sync Manager buffer as soon as previous data was read from the Master (SM event); in this way new sampled data is synchronous with Master readings.
System Time	Absolute time value as defined in the CIP Sync specification in the context of a distributed time system where all devices have a local clock that is synchronised with a common Master clock.
Telegram	Message.
Time stamp	System Time stamp value associated with the CIP Motion connection data that conveys the absolute time when the associated data was captured, or that can also be used to determine when the associated data shall be applied.
Topology	Physical network architecture with respect to the connection between the stations of the communication system.
Type	Hardware or software element which specifies the common attributes shared by all instances of the type.
Use case	Class specification of a sequence of actions, including variants, that a system (or other entity) can perform, interacting with actors of the system.
Variable	Software entity that may take different values, one at a time.

1 - Safety summary

1.1 Safety

- Always adhere to the professional safety and accident prevention regulations applicable to your country during device installation and operation;
- installation and maintenance operations have to be carried out by qualified personnel only, with power supply disconnected and stationary mechanical parts;
- device must be used only for the purpose appropriate to its design: use for purposes other than those for which it has been designed could result in serious personal and/or the environment damage;
- high current, voltage and moving mechanical parts can cause serious or fatal injury;
- warning ! Do not use in explosive or flammable areas;
- failure to comply with these precautions or with specific warnings elsewhere in this manual violates safety standards of design, manufacture, and intended use of the equipment;
- Lika Electronic assumes no liability for the customer's failure to comply with these requirements.

1.2 Electrical safety

- Turn OFF the power supply before connecting the device;
- connect according to explanation in the "4 - Electrical connections" section on page 23;
- in compliance with 2014/30/EU norm on electromagnetic compatibility, following precautions must be taken:
  - before handling and installing the equipment, discharge electrical charge from your body and tools which may come in touch with the device;
  - power supply must be stabilized without noise; install EMC filters on device power supply if needed;
  - always use shielded cables (twisted pair cables whenever possible);
  - avoid cables runs longer than necessary;
  - avoid running the signal cable near high voltage power cables;
  - mount the device as far as possible from any capacitive or inductive noise source; shield the device from noise source if needed;
  - to guarantee a correct working of the device, avoid using strong magnets on or near by the unit;
  - minimize noise by connecting the shield and/or the connector housing and/or the frame to ground. Make sure that ground is not affected by noise. The connection point to ground can be situated both on the device side and on user's side. The best solution to minimize the interference must be carried out by the user. Provide the ground connection as close as possible to the encoder. We suggest using the ground point provided

in the cap, use one TCEI M3 x 6 cylindrical head screw with two tooth lock washers.

1.3 Mechanical safety

- Install the device following strictly the information in the "3 - Mechanical installation" section on page 19;
- mechanical installation has to be carried out with stationary mechanical parts;
- do not disassemble the unit;
- do not tool the unit;
- delicate electronic equipment: handle with care; do not subject the device to knocks or shocks;
- respect the environmental characteristics of the product;
- we suggest installing the unit providing protection means against waste, especially swarf as turnings, chips, or filings; should this not be possible, please make sure that adequate cleaning measures are in place in order to prevent the wire from jamming;
- to avoid failures, never exceed the maximum measuring length and prevent the wire from tangling up;
- never release the wire freely, always help the wire wind properly: risk of personal injury and/or equipment damage;
- always keep the wire aligned not to damage the equipment;
- the stroke per turn of the draw-wire unit is 200 mm (7.874").

2 - Identification

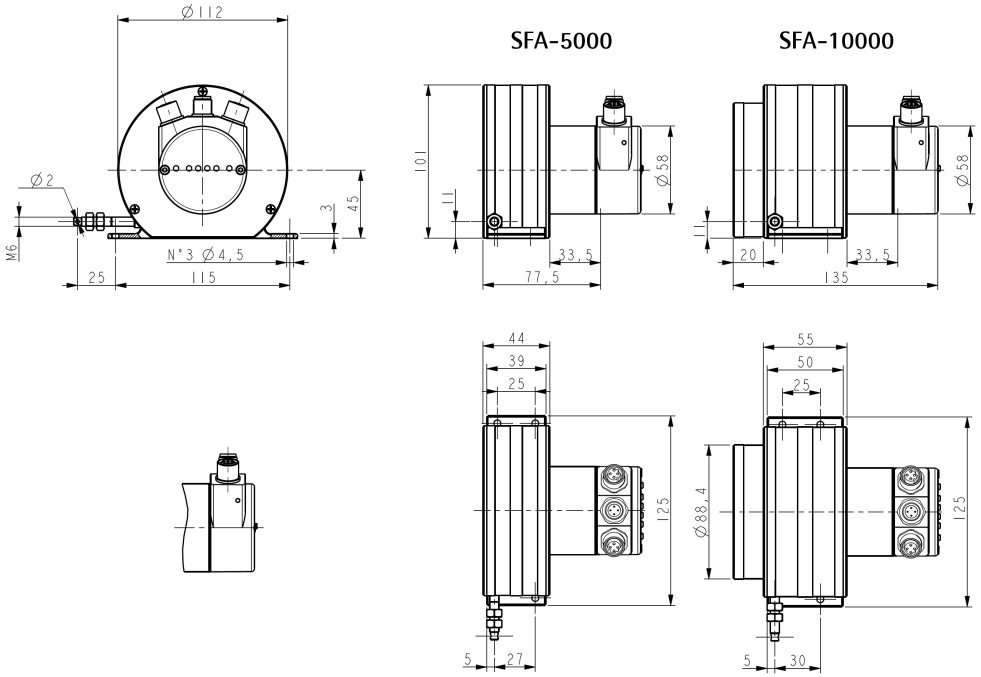
Device can be identified through the **order code** and the **serial number** printed on the label applied to its body. Information is listed in the delivery document too. Please always quote the order code and the serial number when reaching Lika Electronic. For any information on the technical characteristics of the product refer to the technical catalogue.

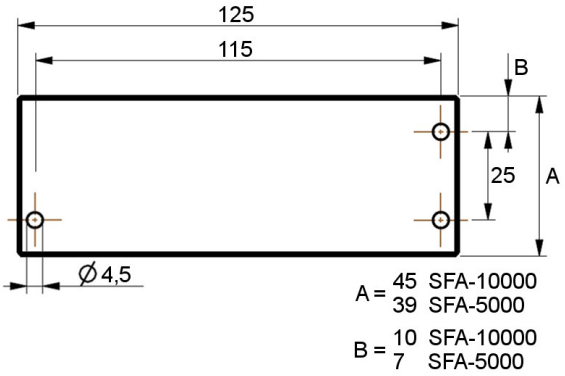
Warning: encoders having order code ending with "/Sxxx" may have mechanical and electrical characteristics different from standard and be supplied with additional documentation for special connections (Technical Info).

3 - Mechanical installation

WARNING
Installation, electrical connection and maintenance operations must be carried out by qualified personnel only, with power supply disconnected. Mechanical components must be in stop.

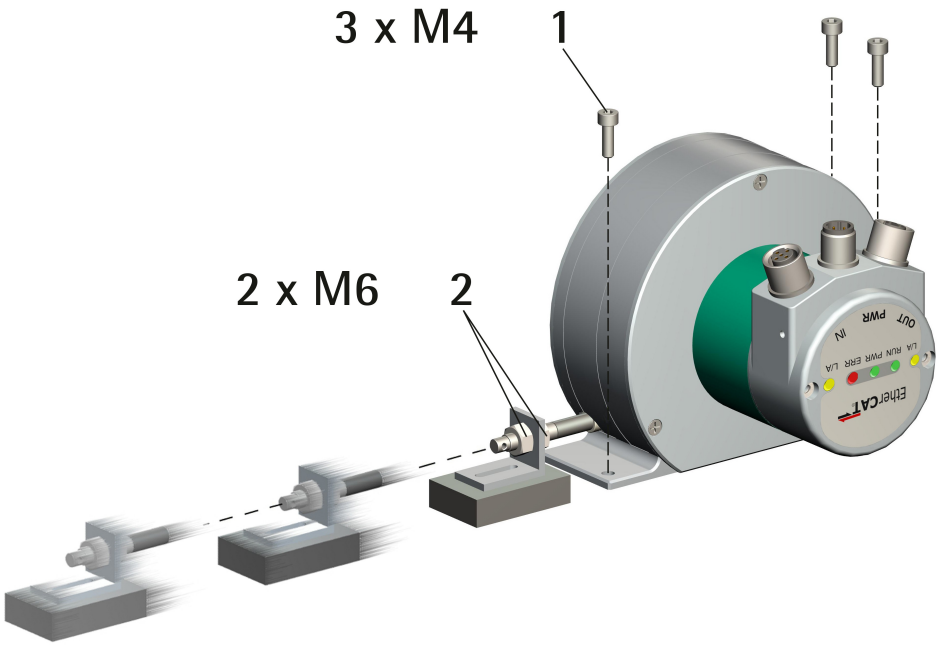
3.1 Overall dimensions





Values are expressed in mm

3.2 Mounting instructions



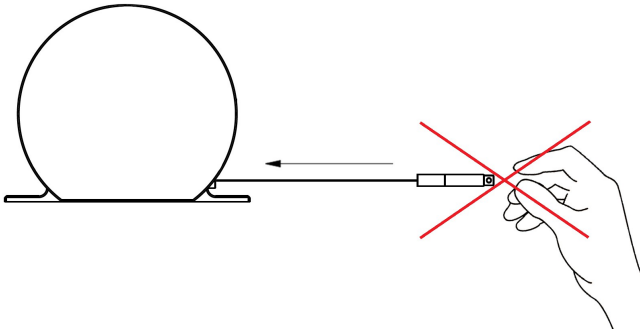
- Fasten the draw-wire unit onto a fixed support using three M4 screws **1**;
- remove the transport safety wire that pins the end of the measuring wire;
- fix the end of the measuring wire to the moving unit using the provided M6 nuts **2**.

WARNING

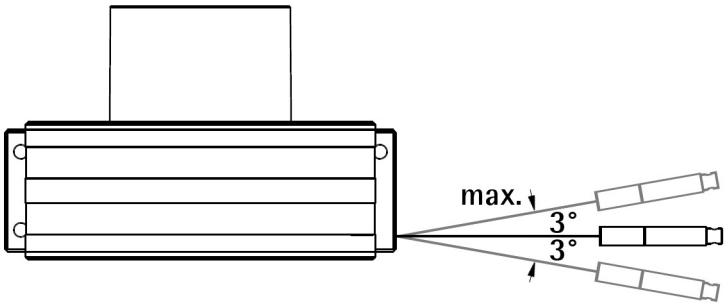
We suggest installing the unit providing protection means against waste, especially swarf as turnings, chips, or filings; should this not be possible, please make sure that adequate cleaning measures are in place in order to prevent the wire from jamming.

To avoid irreparable failures, never exceed the maximum measuring length and prevent the wire from tangling up.

Never release the wire freely, always help the wire wind properly: risk of personal injury and/or equipment damage.



Always keep the wire aligned not to damage the equipment (maximum deviation: 3°).



3.3 Useful information

If you want to know the **maximum measuring length** and the **physical linear resolution** of the draw-wire encoder please refer to the order code. The stroke per turn is always 200 mm (7.874"), the maximum number of turns is 25 for SFA-5000 and 50 for SFA-10000.

EXAMPLE 1

SFA-**5000**-**EC**-**8192**-M12 using the physical resolution (**Scaling function** in 6000-00 Operating parameters = 0)
Stroke per turn of the drum = 200 mm (7.874")
Physical resolution per turn = 13 bits = 8,192 cpr
Max. number of physical revolutions = 14 bits = 16,384 revolutions
Total physical resolution = 27 bits = 134,217,728 information
Physical linear resolution = 0.024 mm = 24 µm
Max. number of turns of the drum = 25
Max. measuring length = 5,000 mm (196.85")
Number of information = 204,800

EXAMPLE 2

SFA-**10000**-**EC**-**8192**-M12 using a custom resolution (**Scaling function** in **6000-00 Operating parameters** = 1)
Stroke per turn of the drum = 200 mm (7.874")
Physical resolution per turn = 13 bits = 8,192 cpr
Max. number of physical revolutions = 14 bits = 16,384 revolutions
Custom resolution per turn = **6001-00 Units per revolution** = 2,000 cpr (example)
6002-00 Total Measuring Range = 8,192,000 information (example)

$$\text{Custom number of encoder revolutions} = \frac{\text{6002-00 Total Measuring Range}}{\text{6001-00 Units per revolution}} = 4{,}096$$

Linear resolution = 0.1 mm = 100 µm
Max. number of turns of the drum = 50
Max. measuring length = 10,000 mm (393.7")
Number of information = 100,000

3.4 Maintenance

The measuring system does not need any particular maintenance; anyway it has to be handled with the utmost care as any delicate electronic equipment. From time to time we recommend the following operations:

- the unit and the wire have to be cleaned regularly using a soft and clean cloth to remove dust, chips, moisture etc.; do not use oil to clean the wire.

4 - Electrical connections

WARNING
Installation, electrical connection and maintenance operations must be carried out by qualified personnel only, with power supply disconnected. Mechanical components must be in stop.

For any information on the mechanical and electrical characteristics of the encoder please refer to the technical catalogue.

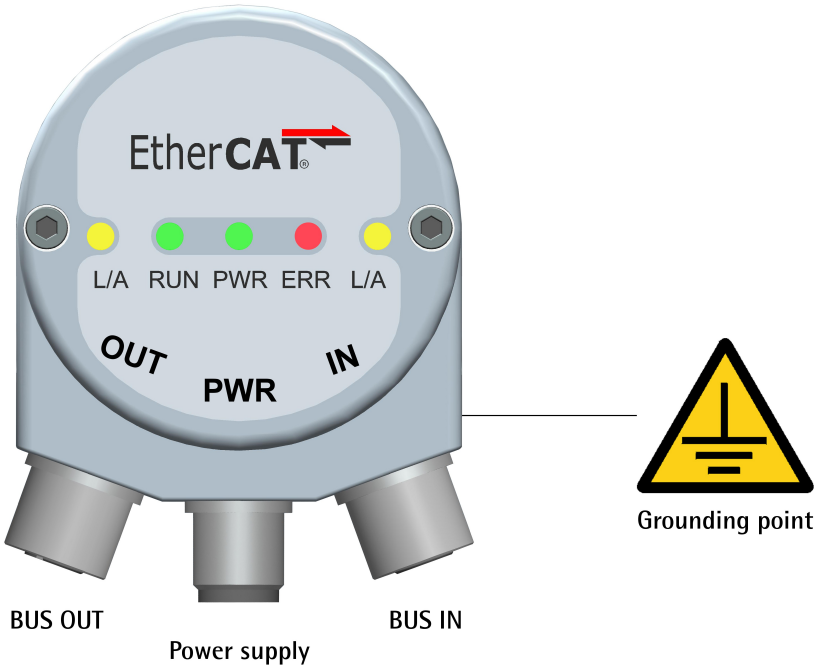
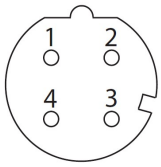
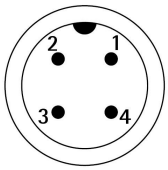


Figure 1 - Connectors and diagnostic LEDs

4.1 M12 connectors

The connection cap is fitted with three M12 connectors with pin-out in compliance with the EtherCAT® standard. Therefore you can use standard EtherCAT cables commercially available.
Input ECATIN (IN) and output ECATOUT (OUT) connectors are not interchangeable! IN connector must be networked towards the EtherCAT Master.

The Ethernet interface supports 100 Mbit/s, fast Ethernet, full duplex operation.

M12	BUS IN & BUS OUT	POWER
(frontal side)	 D coding female	 A coding male

Pin	Description	Description
1	Tx Data +	+10Vdc +30Vdc
2	Rx Data +	n.c.
3	Tx Data -	0Vdc
4	Rx Data -	n.c.

n.c.= not connected.

4.2 Network configuration: topologies, cables, hubs, switches - Recommendations

Cables and connectors comply with the EtherCAT specifications. Cables are CAT-5 shielded cables.

Line, tree or star: EtherCAT supports almost any topology. The bus or line structure known from the fieldbuses thus also becomes available for Ethernet, without the quantity limitations implied by cascaded switches or hubs.

The Fast Ethernet physics (100BASE-TX) enables a cable length of 100 m (328 ft) between two devices. Since up to 65,535 devices can be connected, the size of the network is almost unlimited.

The Ethernet protocol according to IEEE 802.3 remains intact right up to the individual device; no sub-bus is required. In order to meet the requirements of a modular device like an electronic terminal block, the physical layer in the coupling device can be converted from twisted pair or optical fiber to LVDS (alternative Ethernet physical layer, standardized in [4.5]). A modular device can thus be extended very cost-efficiently. Subsequent conversion from the backplane physical layer LVDS to the 100BASE-TX physical layer is possible at any time – as usual with Ethernet.

For a complete list of the available cordsets and connection kits please refer to the product datasheet ("Accessories" list).

4.3 Addressing

It is not necessary to assign a physical address to the device because the addressing of the Slave is automatic at power-on during the initial scanning of the hardware configuration.

The field for addressing is 32-bit long, there are three kinds of addressing:

- Auto Increment Addressing = Position Addressing: 16 bits indicate the physical position of the Slave inside the network while 16 bits are scheduled for local memory addressing; when the Slave receives the frame then it increments the position address and the Slave receiving address 0 is the addressed device;
- Fixed Addressing = 16 bits indicate the physical address of the Slave inside the network while 16 bits are scheduled for addressing the local memory;
- Logical Address = the Slave is not provided with its own individual address, but it can read and write data in a section of the total memory space available (4 Gigabytes).

For complete information refer to the "6.1.5 Addressing" section on page 52.

4.4 Line Termination

EtherCAT network needs no line termination because the line is terminated automatically; in fact every Slave is able to detect the presence of the downstream Slaves. For complete information refer to the "6.1.4 Line Termination" section on page 51.

4.5 Ground connection

To minimize noise connect properly the shield and/or the connector housing and/or the frame to ground. Connect properly the cable shield to ground on user's side. Lika's EC- pre-assembled cables are fitted with shield connection to the connector ring nut in order to allow grounding through the body of the device. Lika's E- connectors have a plastic gland, thus grounding is not possible. If metal connectors are used, connect the cable shield properly as recommended by the manufacturer. Anyway make sure that ground is not affected by noise. It is recommended to provide the ground connection as close as possible to the device. We suggest using the ground point provided in the cap (see Figure 1, use 1 TCEI M3 x 6 cylindrical head screw with 2 tooth lock washers).

4.6 Diagnostic LEDs

Five LEDs located in the rear side of the connection cap are designed to show the operating or fault status of the EtherCAT® interface.
The LEDs operation is according to the EtherCAT specifications, see ETG1300_S_R_V1i1i0_IndicatorLabelingSpecification.pdf.

LED states	Definition
ON	The indicator shall be constantly ON.
OFF	The indicator shall be constantly OFF.
Flickering	The indicator shall turn ON and OFF iso-phase with a frequency of 10 Hz: ON for 50 ms and OFF for 50 ms.
Blinking	The indicator shall turn ON and OFF iso-phase with a frequency of 2.5 Hz: ON for 200 ms followed by OFF for 200 ms.
Single flash	The indicator shall show one short flash (200 ms) followed by a long OFF phase (1000 ms).
Double flash	The indicator shall show a sequence of two short flashes (200 ms), separated by an OFF phase (200 ms), and followed by a long OFF phase (1000 ms).

L/A Link/ Activity (yellow)	It shows the state of the physical links (IN and OUT) and the activity in the links
OFF	condition: port closed, link: YES, activity: N.A.
FLICKERING	condition: port open, link: YES, activity: YES
ON	condition: port open, link: YES, activity: NO

RUN (green)	It shows the state of the EtherCAT State Machine (ESM)
OFF	The encoder is in **INIT** state
BLINKING	The encoder is in **PRE-OPERATIONAL** state
SINGLE FLASH	The encoder is in **SAFE-OPERATIONAL** state
ON	The encoder is in **OPERATIONAL** state
FLICKERING	The encoder is in **BOOT** state

PWR (green)	It shows the power supply state
OFF	The encoder is switched OFF
ON	The encoder is switched ON

ERR (red)	It shows the error state
OFF	No error
FLICKERING	Error while loading parameters from flash memory at start-up; encoder parameters have not been saved correctly on flash memory
BLINKING	Invalid configuration
SINGLE FLASH	Local error (see ETG1000.6)
DOUBLE FLASH	Watchdog timeout
ON	Memory error and ESC controller not active

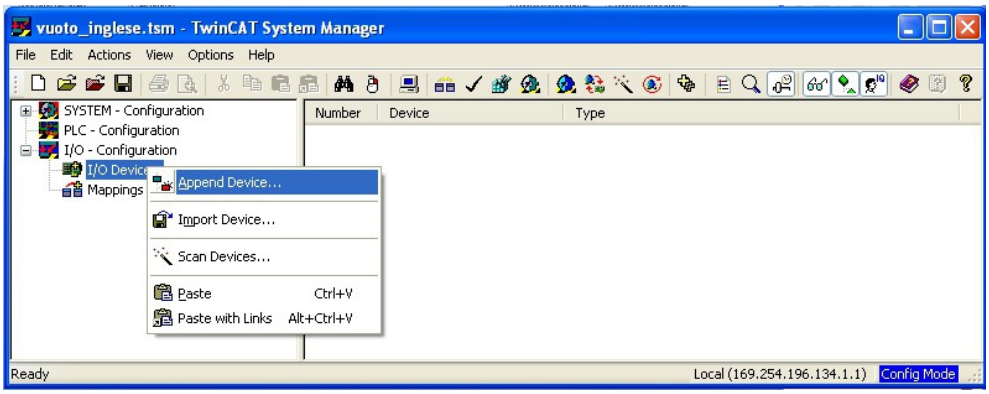
5 - Quick reference (TwinCAT)

Lika draw-wire encoders are Slave devices and support "CANopen over EtherCAT" (CoE) mode for data transfer. In particular, they support the "CANopen DS301 Communication profile".
For any omitted specification on CANopen® protocol, please refer to "CiA Draft Standard Proposal 301. Application Layer and Communication Profile" and "CiA Draft Standard 406. Device profile for encoders" documents available at the address **www.can-cia.org**.
For any omitted specification on EtherCAT® protocol, please refer to "ETG.1000 EtherCAT Specification" documents available at the address **www.ethercat.org**.

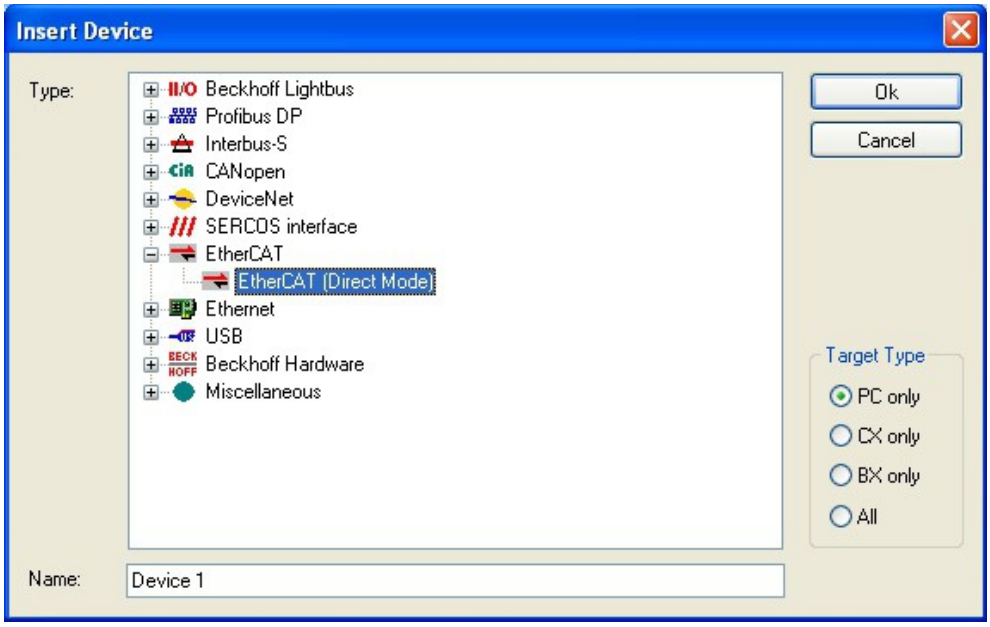
5.1 System configuration using TwinCAT software system from Beckhoff

5.1.1 Setting the Network Card

Launch **TwinCAT System Manager**.
In the left pane of the main window extend the devices tree and select the **I/O Devices** item; right-click the **I/O Devices** item and then press the **Append Device...** command.



In the **Insert Device** window select the **EtherCAT (Direct Mode)** item and confirm pressing the **OK** button.

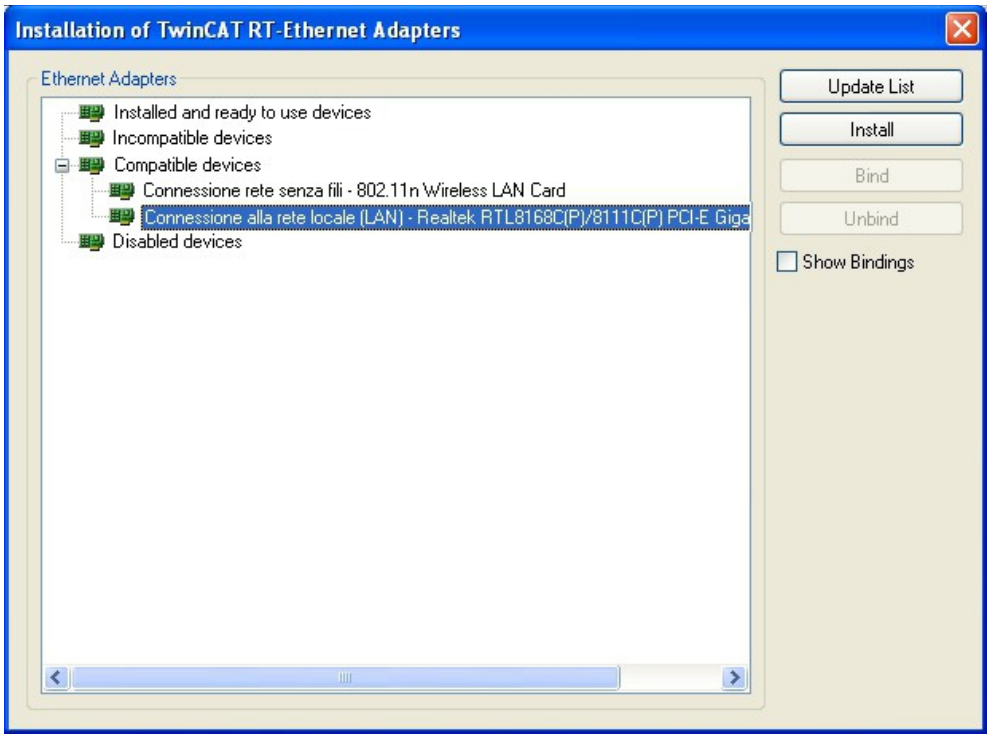


If a network card has been already installed properly, the following window will appear and show the list of the installed devices.

Select the network card you want to use and then confirm the choice pressing the **OK** button.

If there are no network cards installed, you must install one before proceeding. To do this, on the menu bar of the **TwinCAT System Manager** main window, select the **Options** menu and then press the **Show Real Time Ethernet**

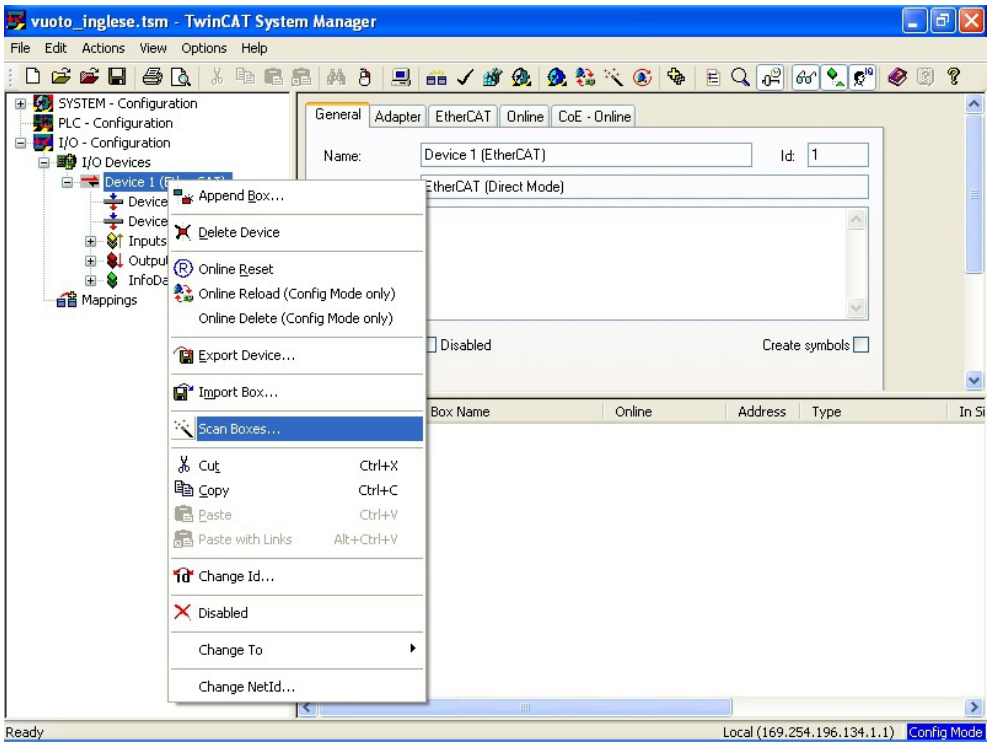
Compatible Devices... command. The **Installation of TwinCAT RT – Ethernet Adapter** window will appear.



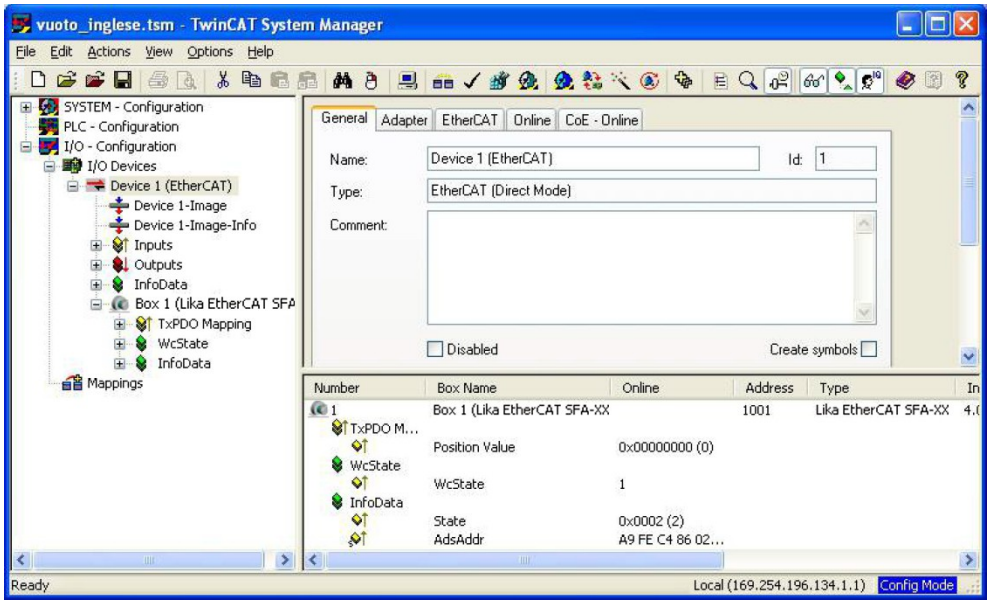
Now select the **Compatible Devices** item and choose the network card you want to install; finally press the **Install** button to confirm your choice.

5.1.2 Add new I/O modules (Boxes)

If devices are connected to the network and switched ON, right-click the **Device 1 (EtherCAT)** item in the left pane of the **TwinCAT System Manager** main window and press the **Scan Boxes...** command.

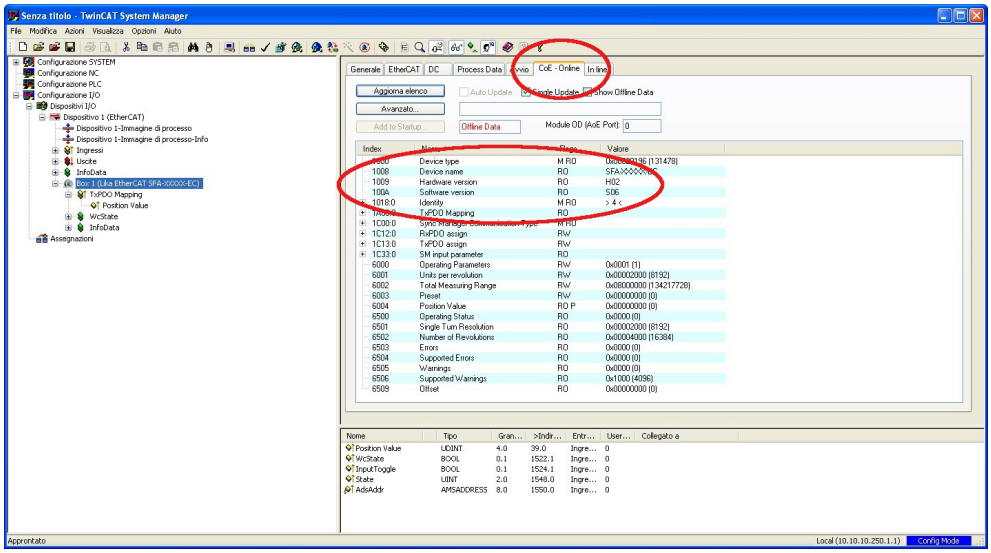


At the end of the process some information will be listed in the right page as in Figure here below.

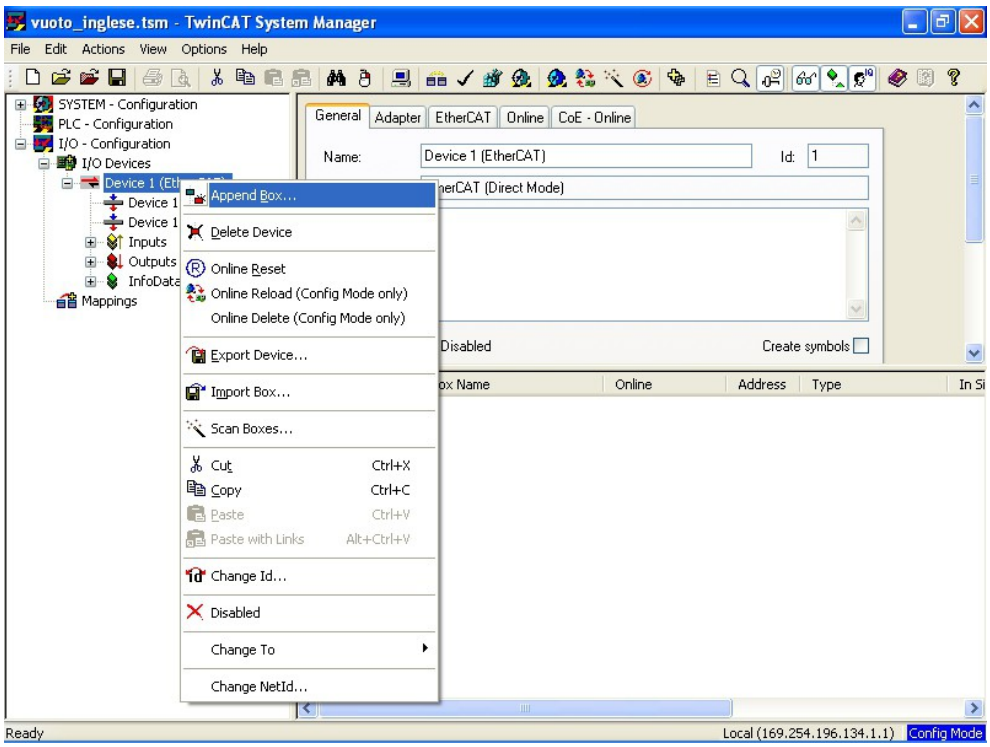


If devices are not already connected to the network it is necessary to use the XML file supplied with the draw-wire encoder: **Lika_SFA_XXXXX_ECx_Vx.xml** (V6 or higher; see at **www.lika.biz > ROTARY ENCODERS > DRAW-WIRE UNITS (DRAW-WIRE) > ABSOLUTE**.

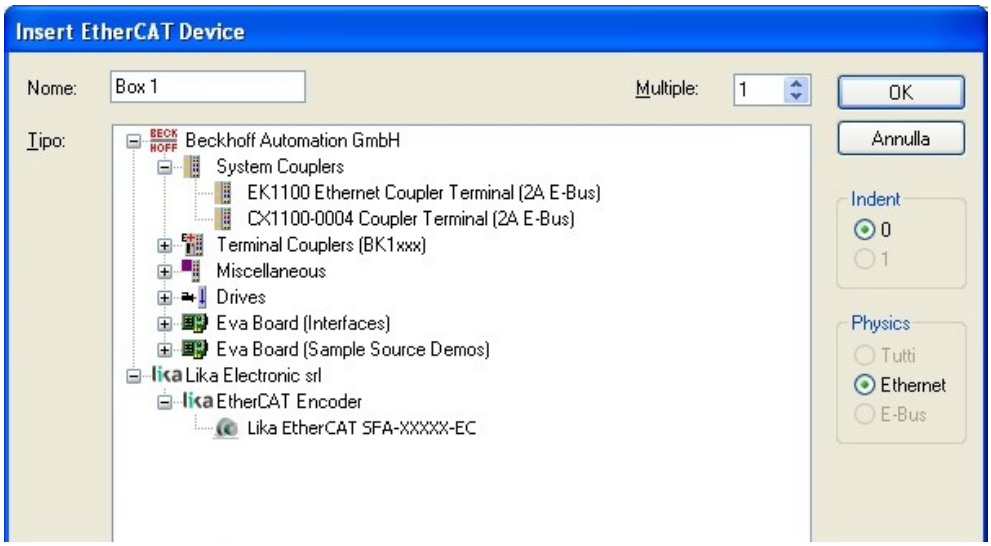
If you want to know the firmware version of a device, press the **Box (Lika EtherCAT SFA-XXXXX-EC)** item in the left pane of the **TwinCAT System Manager** main window: some tabbed pages for configuring and managing the device will appear in the right pane. Enter the **CoE - Online** page and refer to the **1009-00 Hardware version** and **100A-00 Software version** indexes.



Right-click the **Device 1 (EtherCAT)** item in the left pane of the **TwinCAT System Manager** main window and press the **Append Box...** command.



The **Insert EtherCAT Device** window will appear.



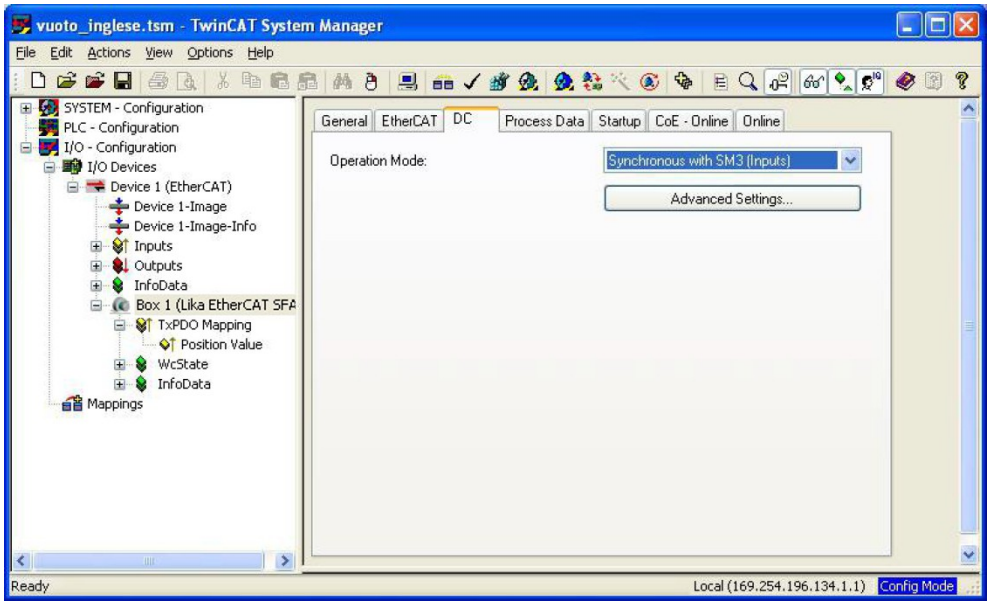
In the **Insert EtherCAT Device** window that appears select **Lika Electronic srl** and then **EtherCAT Encoder** items; now choose from the list the encoder you want to install (if several encoder models are available).

Press the **OK** button to confirm your choice.

5.2 Setting the communication mode

5.2.1 Synchronous with SM3

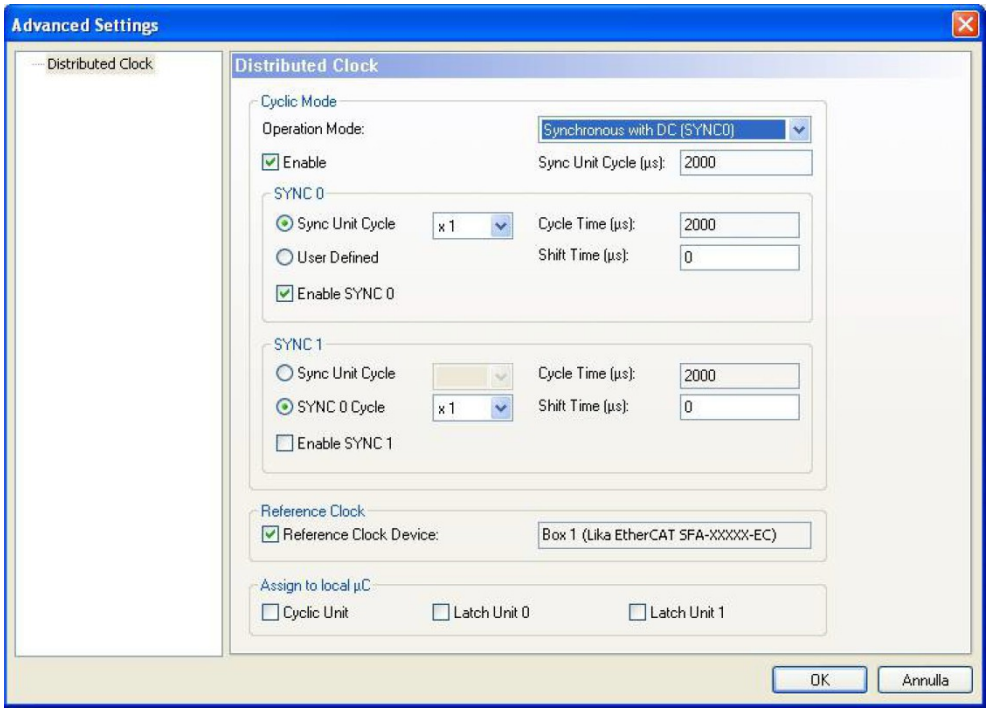
In the left pane of the **TwinCAT System Manager** main window press the **Box (Lika EtherCAT SFA-XXXXX-EC)** item: some tabbed pages for configuring and managing the device will appear in the right pane. Enter **DC** page. Select the **Synchronous with SM3 (Inputs)** option in the **Operation Mode** box.



For any further information on the Synchronous with SM3 operation mode please refer to the "Synchronous with SM3" section on page 54 and to the 1C33 SM input parameter object on page 66.

5.2.2 Synchronous with DC (SYNC0)

In the left pane of the **TwinCAT System Manager** main window press the **Box (Lika EtherCAT SFA-XXXXX-EC)** item: some tabbed pages for configuring and managing the device will appear in the right pane. Enter **DC** page.
Select the **Synchronous with DC (SYNC0)** option in the **Operation Mode** box. Then press the **Advanced Settings...** button. The **Advanced Settings** window will appear.

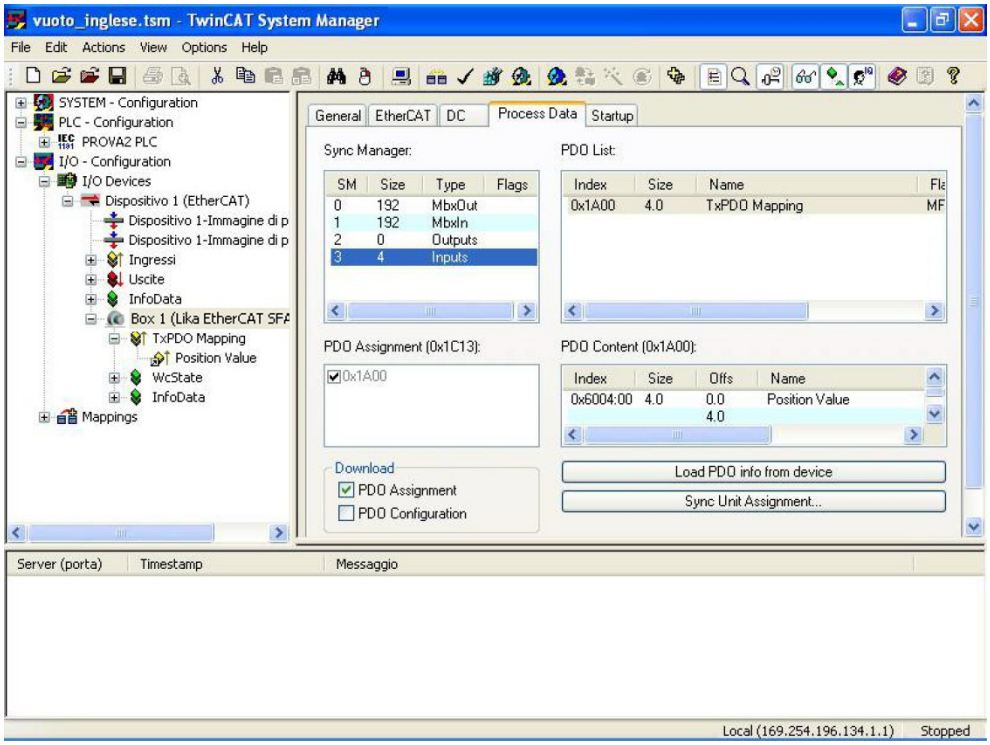


In the section tabbed **SYNC 0** set the cycle time next to the **Sync Unit Cycle** box; sync time is calculated as multiple (or sub-multiple) of the value set in the **Sync Unit Cycle (µs)** item right above.

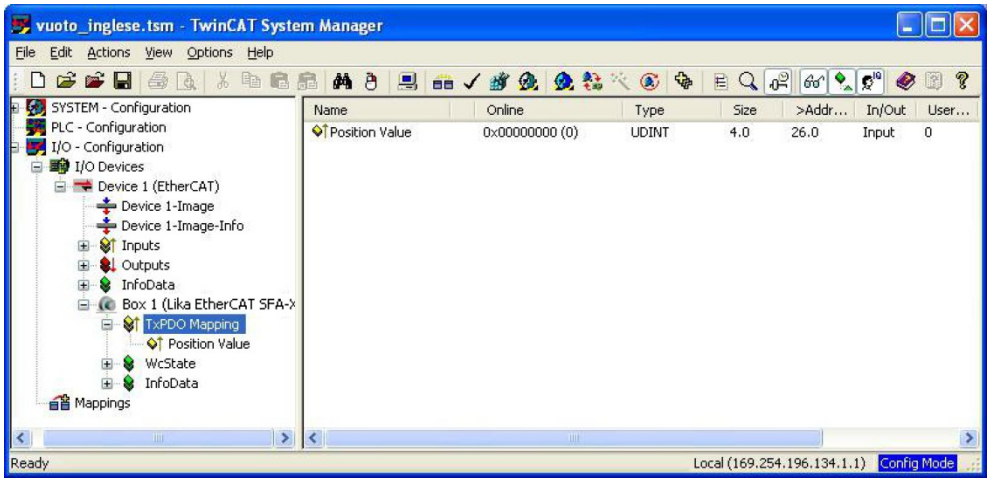
For any further information on the Synchronous with DC operation mode please refer to the "Synchronous with DC SYNC0" section on page 54 and to the **1C33 SM input parameter** object on page 66.

5.3 Process Data Objects

In the left pane of the **TwinCAT System Manager** main window press the **Box (Lika EtherCAT SFA-XXXXX-EC)** item. Expand the box to see Process Data Outputs (PDO). Some tabbed pages for configuring and managing the device will appear in the right pane. Enter the **Process Data** page. In this page process data objects (TxPDO Mapping) are shown.

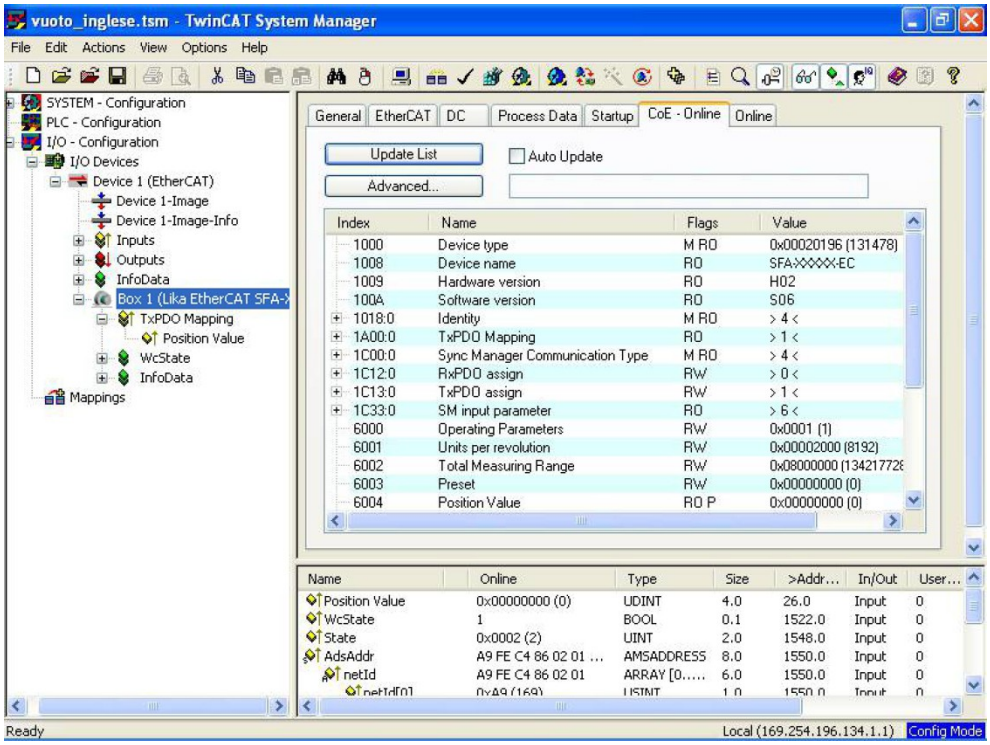


Process data objects can be displayed also by pressing the **TxPDO Mapping** item in the left pane of the **TwinCAT System Manager** main window; data is listed in the right pane.

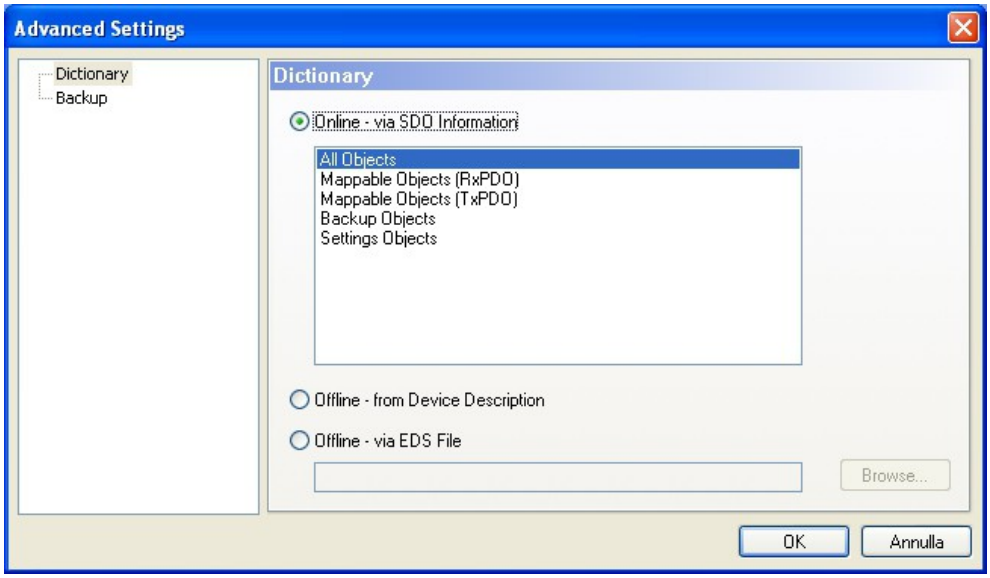


5.4 COE Object Dictionary

In the left pane of the **TwinCAT System Manager** main window press the **Box (Lika EtherCAT SFA-XXXXX-EC)** item: some tabbed pages for configuring and managing the device will appear in the right pane. Enter the **CoE - Online** page. In this page the Object Dictionary is shown. This is the offline version of the objects dictionary as read from the XML file.



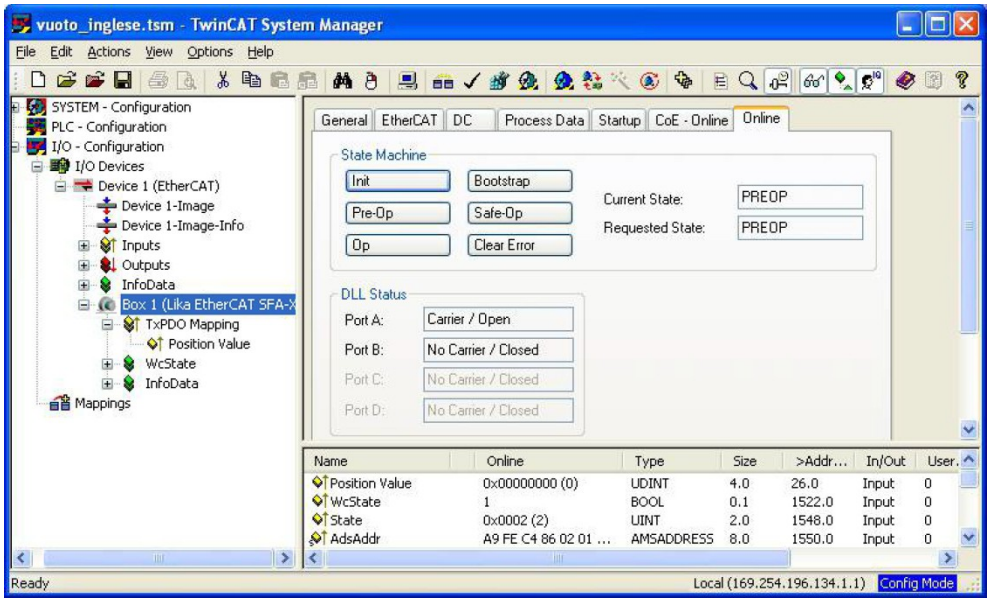
Objects can also be read directly from the encoder; to do this click the **Advanced...** button: the **Advanced Settings** window will appear.



Select the **Dictionary** item in the left pane and then choose the **Online - via SDO Information** option in the **Dictionary** page; press the **OK** button to confirm.

5.5 Online Data

In the left pane of the **TwinCAT System Manager** main window press the **Box (Lika EtherCAT SFA-XXXXX-EC)** item: some tabbed pages for configuring and managing the device will appear in the right pane. Enter the **Online** page to check the encoder status.



To display the encoder process data in real time, click the **Safe-OP** button if you want to display inputs only; click the **OP** button if you want to display both inputs and outputs.

WARNING

The structure of Data Objects (PDO and SDO) requires bytes to be sent from the Least Significant Byte (LSB) to the Most Significant Byte (MSB).
On the contrary in TwinCAT you must write and read data from MSB to LSB.
Furthermore in TwinCAT also strings must be entered in the reverse order:

- read default values: Data byte = 64 61 6F 6Chex = "daol" in ASCII code (i.e. "load" if read in reverse);
- save parameters: Data byte = 65 76 61 73hex = "evas" in ASCII code (i.e. "save" if read in reverse).

5.6 EEProm upgrade

WARNING

The EEProm upgrade process has to be accomplished by skilled and competent personnel. If the upgrade is not performed according to the instructions provided or a wrong or incompatible EEProm program is installed then the unit may not be updated correctly, in some cases preventing the unit from working.

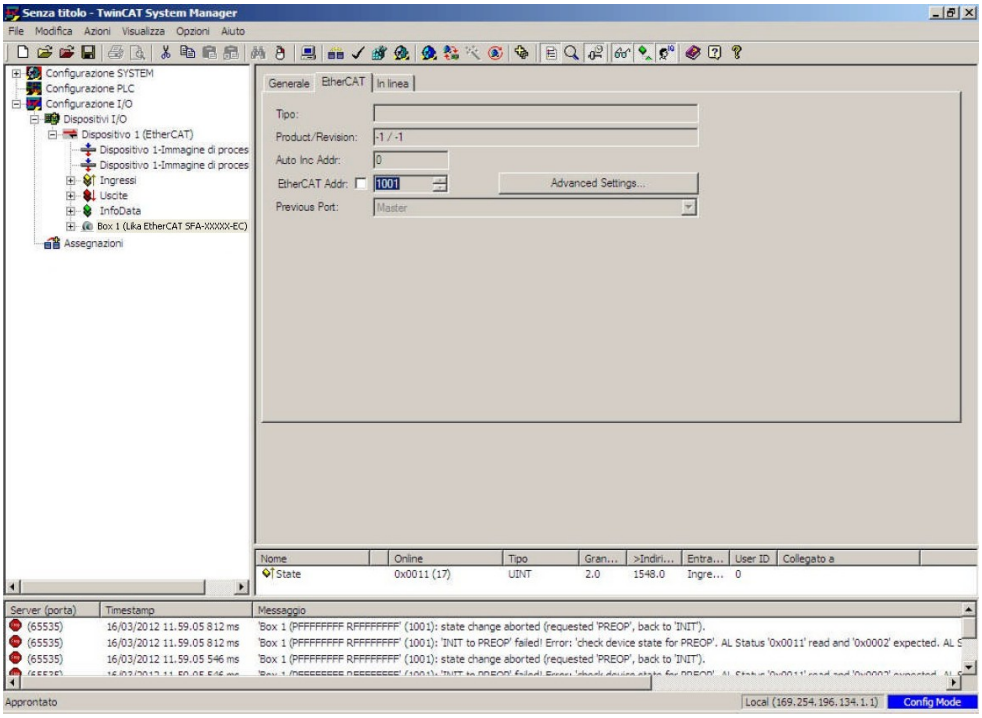
WARNING

The XML file version, the firmware version and the EEProm version must always comply. For example: if the firmware version is H02_S06 (Hardware version: 2; Software version: 6), it is mandatory that the EEProm version is S6, therefore you must then install the XML file version V6.

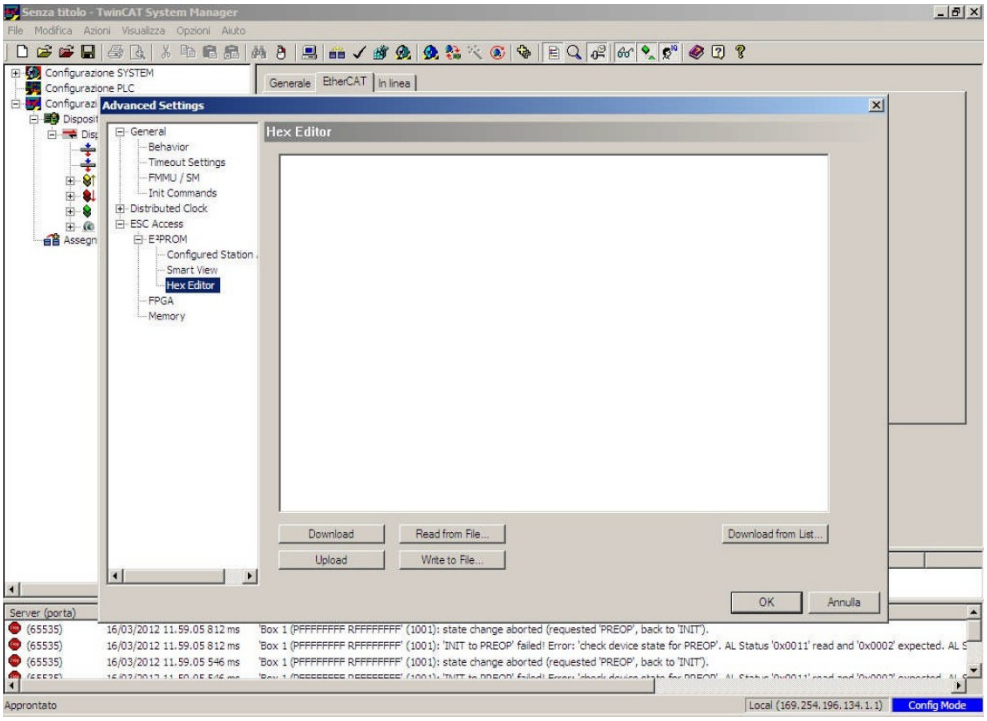
WARNING

It is mandatory that in an EtherCAT network all devices are provided with the same version of the firmware, EEProm and XML file. So when you need to replace an old encoder installed in your network, then you must either upgrade all the encoders in the network to the last version compatible with the new encoder; or you must downgrade the new encoder to the older version compatible with the encoders already installed in the network.

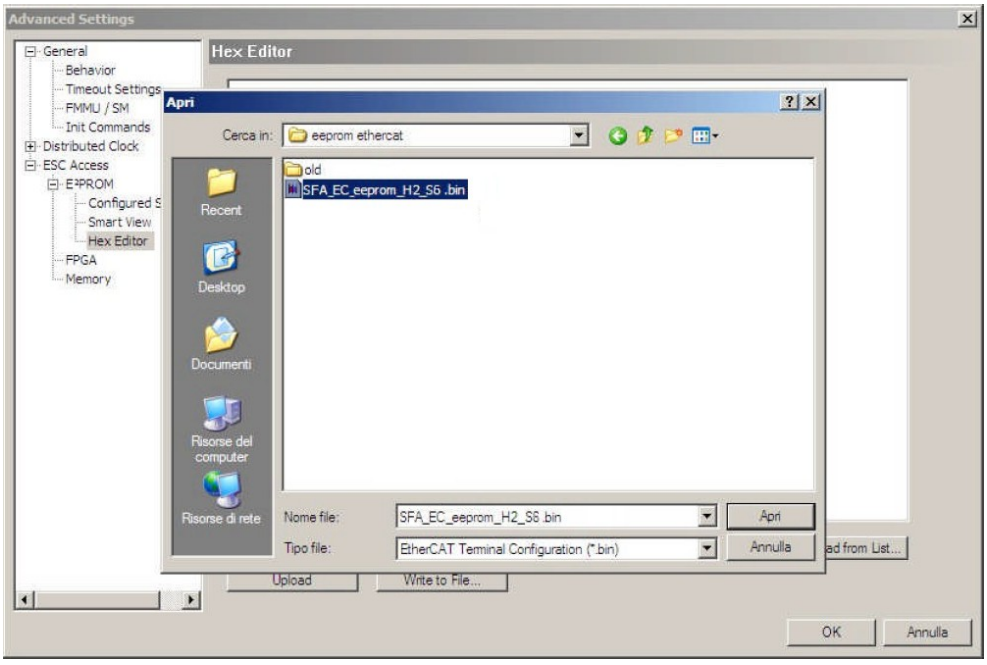
1. In the left pane of the **TwinCAT System Manager** main window press the **Box (Lika EtherCAT SFA-XXXXX-EC)** item of the encoder you need to update: some tabbed pages for configuring and managing the device will appear in the right pane. Enter the **EtherCAT** page.



2. Press the **Advanced Settings...** button; the **Advanced Settings** page will appear; in the directory tree on the left expand the **ESC Access** directory, then expand the **E²PROM** directory, finally select the **HEX Editor** item.



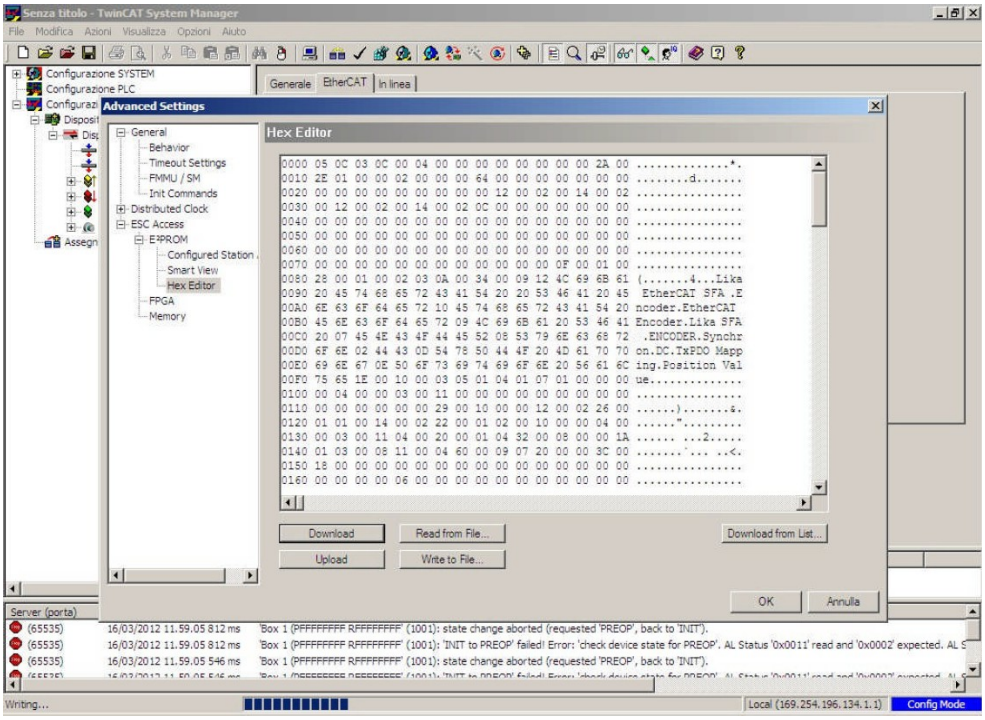
3. Press the **Read from File...** button and select the .BIN file provided by Lika Electronic to upgrade the EEProm; please make sure you select the file suitable for the model you need to upgrade (for example: if you need to upgrade an SFA draw-wire encoder then you must select the file **SFA_EC_eeprom_Hx_Sy.bin**); finally press the **Open** button.



NOTE

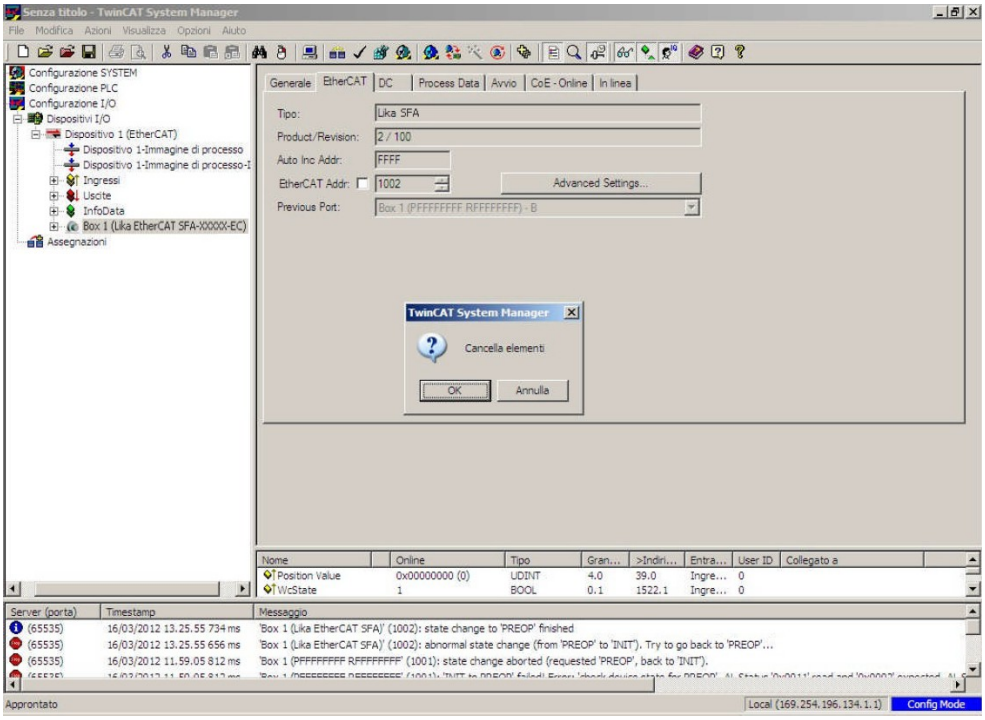
In the .BIN file Hx is the hardware version of the draw-wire encoder, while Sy is the software version.

4. Move back to the previous **Advanced Settings** page and press the **Download** button. Now wait until the EEProm writing process is carried out. The progress bar below in the page displays the progress of the operation. As soon as the process is completed press the **OK** button.

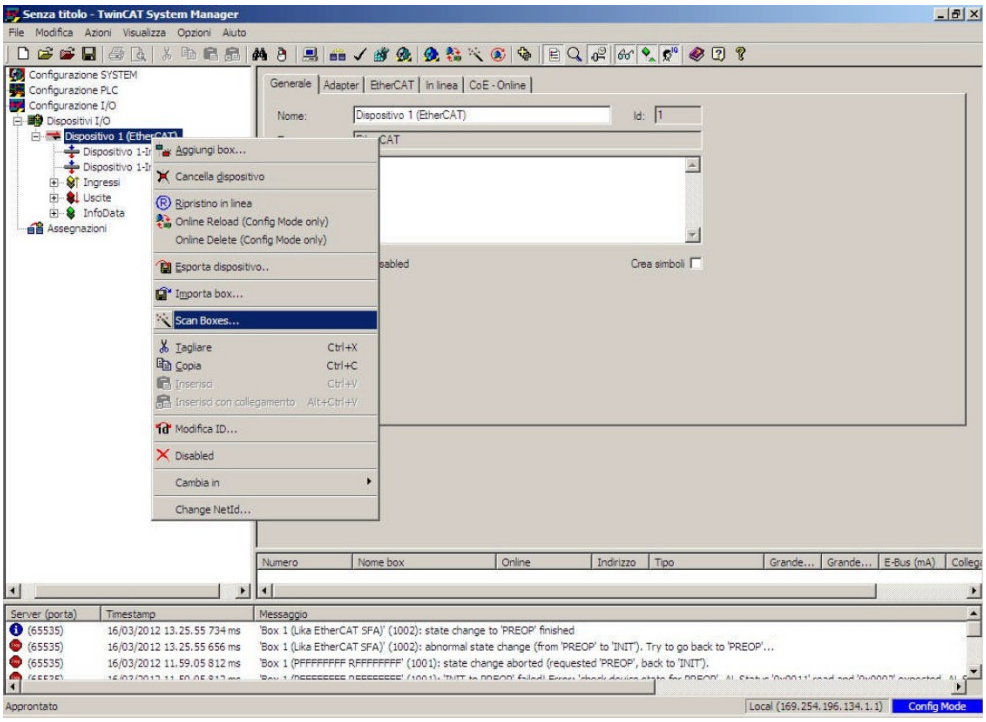


5. Now turn power off, then on again.

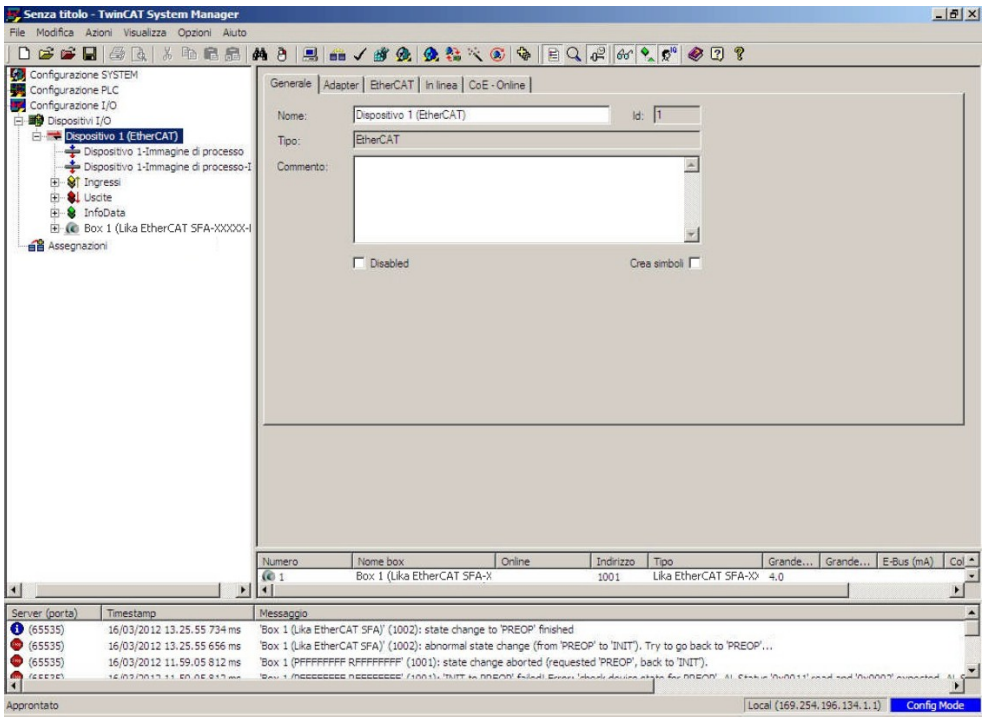
6. In the left pane of the **TwinCAT System Manager** main window delete all the **Box (Lika EtherCAT SFA-XXXXX-EC)** items in the list. Select each box and then press the **DEL** key in the PC keyboard. Press **OK** to confirm.



7. In the left pane of the **TwinCAT System Manager** main window select and right-click the **Device 1 (EtherCAT)** item; press the **Scan Boxes...** command in the menu.



8. At the end of the scanning process all the devices available in the network are listed as shown in the Figure here below.



5.7 Firmware upgrade

WARNING
Firmware upgrade process has to be accomplished by skilled and competent personnel. If the upgrade is not performed according to the instructions provided or a wrong or incompatible firmware program is installed then the unit may not be updated correctly, in some cases preventing the unit from working.

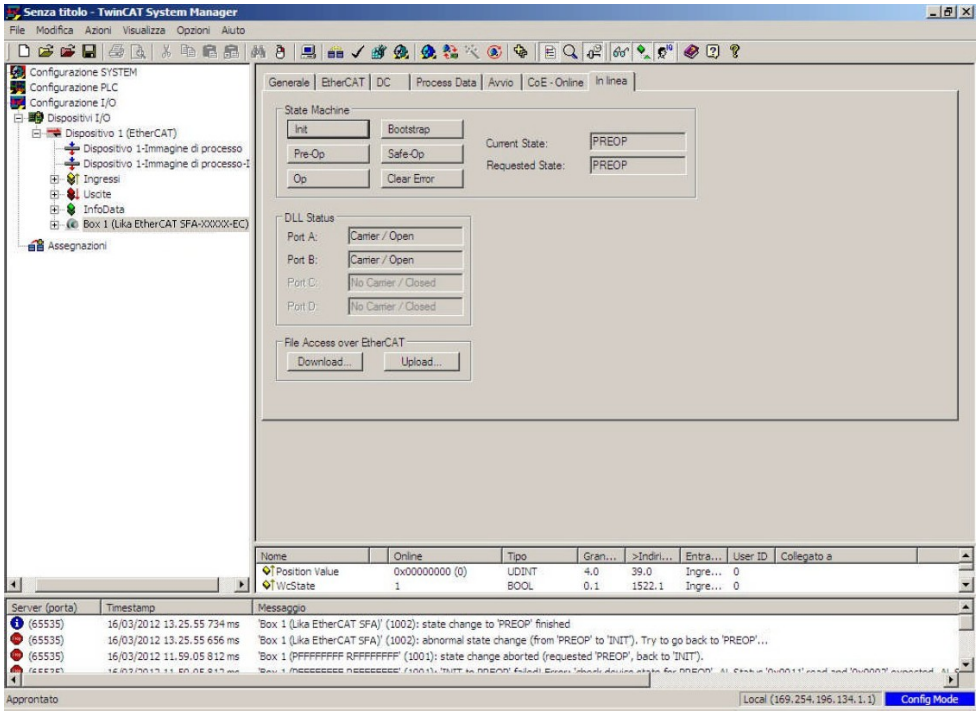
WARNING
The XML file version, the firmware version and the EEProm version must always comply. For example: if the firmware version is H02_S06 (Hardware version: 2; Software version: 6), it is mandatory that the EEProm version is S6, therefore you must then install the XML file version V6.

WARNING
It is mandatory that in an EtherCAT network all devices are provided with the same version of the firmware, EEProm and XML file. So when you need to replace an old encoder installed in your network, then you must either upgrade all the encoders in the network to the last version compatible with the new encoder; or you must downgrade the new encoder to the older version compatible with the encoders already installed in the network.

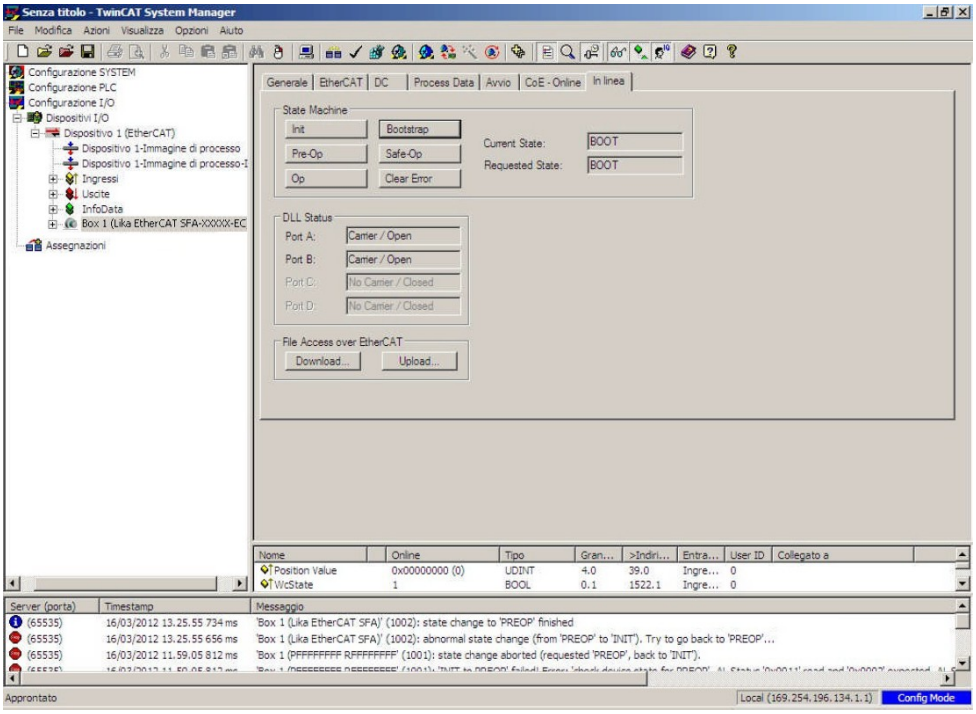
Firmware is a software program which controls the functions and operation of a device; the firmware program, sometimes referred to as "user program", is stored in the flash memory integrated inside the unit. Lika draw-wire encoders are designed so that the firmware can be easily updated by the user himself. This allows Lika Electronic to make new improved firmware programs available during the lifetime of the product.
Typical reasons for the release of new firmware programs are the necessity to make corrections, improve and even add new functionalities to the device.
The firmware upgrading program consists of a single file having .EFW extension. It is released by Lika Electronic Technical Assistance & After Sale Service.

WARNING
You must upgrade the EEProm before upgrading the firmware.

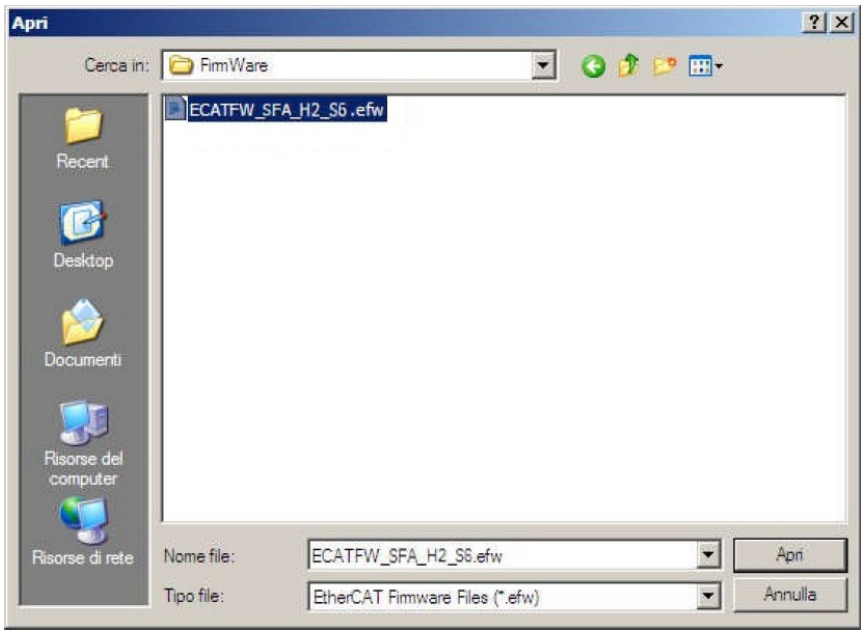
1. In the left pane of the **TwinCAT System Manager** main window press the **Box (Lika EtherCAT SFA-XXXXX-EC)** item of the encoder you need to update: some tabbed pages for configuring and managing the device will appear in the right pane. Enter the **Online** page.



2. Press the **Bootstrap** button in the **State Machine** box; in the BOOT state the encoder is ready to accept the firmware upgrade download process (the **BOOT** message appears next to the **Current State** item in the same box).

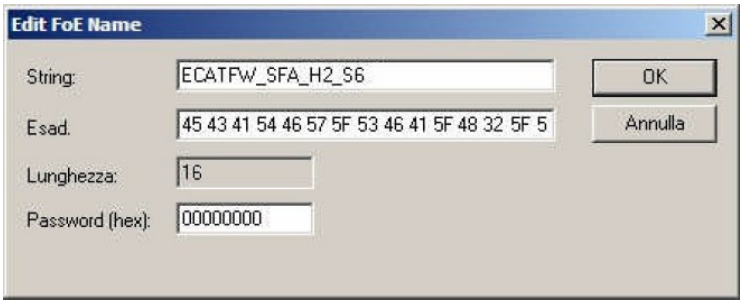


3. Now press the **Download...** button in the **File Access Over EtherCAT** box; in the **Open** window that appears select the .EFW file provided by Lika Electronic to upgrade the firmware; please make sure you select the exact file of the model you need to upgrade (for example: if you have to upgrade an SFA draw-wire encoder then you must select the file **ECATFW_SFA_Hx_Sy.efw**); finally press the **Open** button.

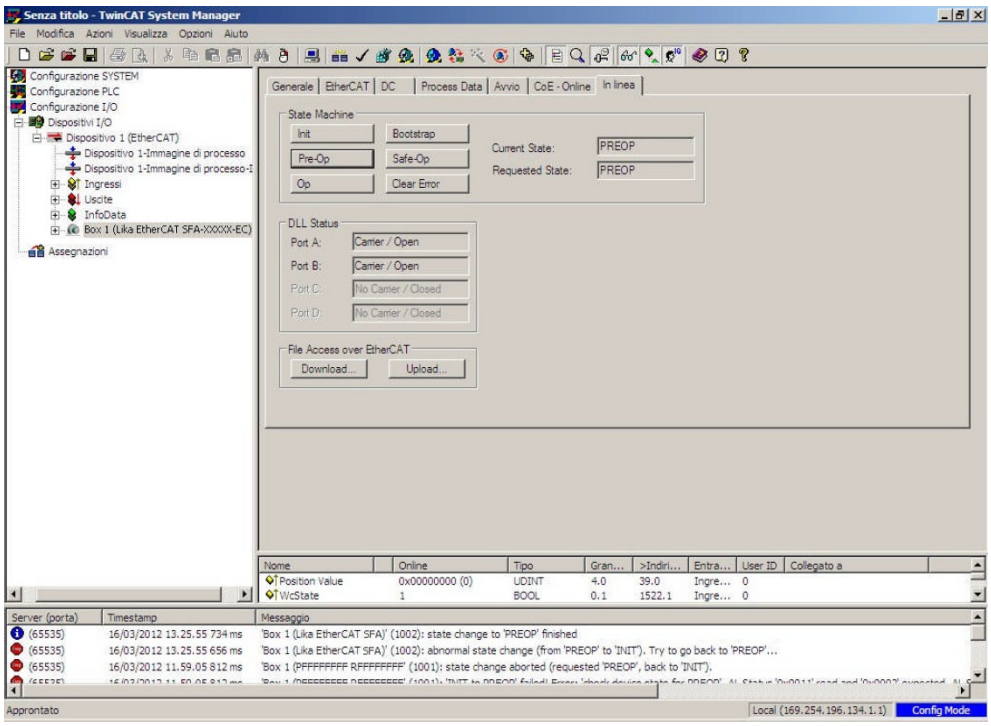


NOTE
In the .EFW file Hx is the hardware version of the encoder, while Sy is the software version.

4. In the **Edit FoE Name** page that appears on the screen enter the password 0x00000000 next to the **Password (hex)** item below in the page and then press the **OK** button to confirm. Now wait until the firmware file saving process is carried out. The progress bar below in the page displays the progress of the operation.



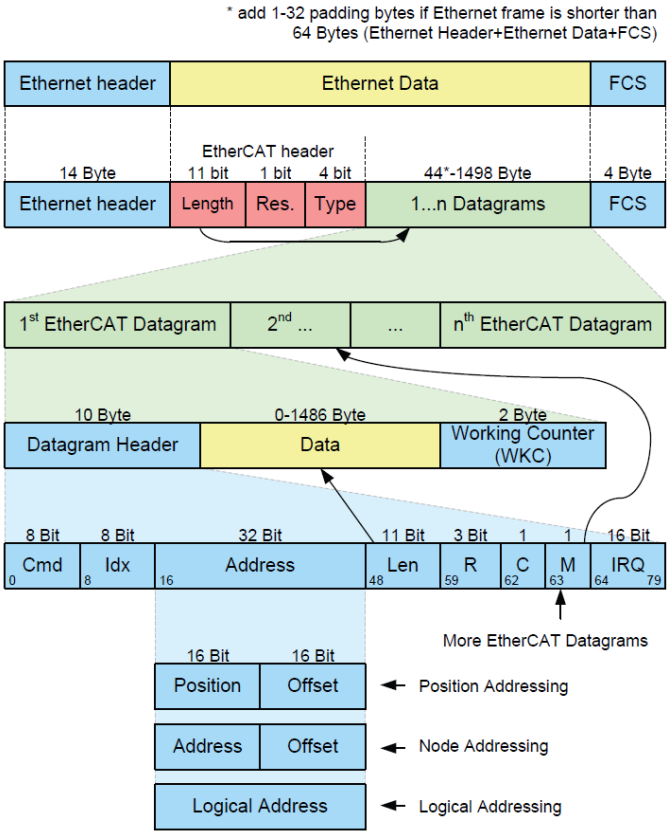
5. To check whether the firmware upgrade procedure has been completed successfully enter the **Online** page in the **TwinCAT System Manager** main window and press the **Pre-Op** button in the **State Machine** box; if everything is ok, the encoder enters the PREOPERATIONAL state (the **PREOP** message appears next to the **Current State** item in the same box).



6 - EtherCAT® interface

6.1 Basic Information on the EtherCAT® Protocol

The EtherCAT protocol is designed to use the standard Ethernet dataframes for issuing data; in addition, and as regards the hardware, it is not necessary to install dedicated Masters for establishing and managing the EtherCAT communication because standard Ethernet network cards can be used. This results in a great advantage in terms of lower costs and simplicity of use because Ethernet network cards are used in standard personal computers and are easily commercially available.
An EtherCAT bus can be viewed as a single and large Ethernet device that receives and sends Ethernet telegrams; it can be considered an Ethernet subnet supported by an Ethernet dataframes structure.
However this "subnet" must be fitted with one only EtherCAT Master controller and several EtherCAT Slaves, but no Ethernet controller with downstream microprocessor must be present.
Here follows an Ethernet frame structure with EtherCAT:

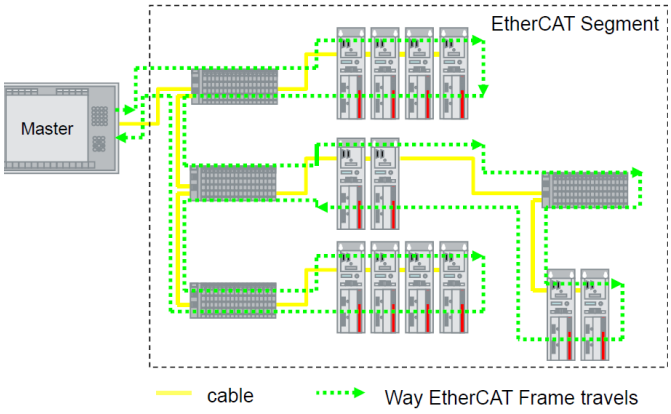


Inside the Ethernet frames, data is transmitted among Master and Slaves using PDO (Process Data Objects) protocol. Each PDO message has inside one or more addresses for issuing data to the Slaves; data + address/es (and additional elements such as a validation checksum) joined together forms an EtherCAT telegram (Datagram).

An EtherCAT frame can contain several telegrams and a complete control cycle often requires more than one frame.

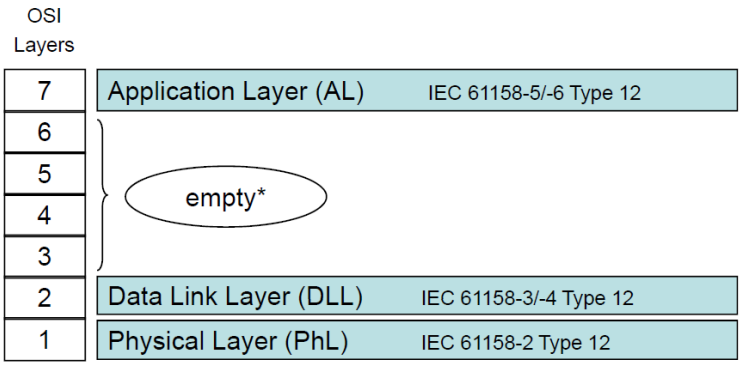
6.1.1 Data transfer

Usually, in a data bus system, Master controller sends online a data request and then waits for data to be processed and sent back from each Slave node; this does not comply with a real-time system because the Master receives data from the Slaves in different moments and the whole system cannot be synchronized.
In EtherCAT the real-time characteristic of the system is quite improved because data are processed "on-the-fly", using one single frame to acquire all data from all Slaves.
In fact the frame sent by the Master is read by each Slave node the data is addressed to while the telegram passes through the device; similarly, input data is inserted while the telegram passes through. Then the telegram is forwarded to the next device. Telegrams are only delayed by a few nanoseconds.
The last Slave issues back the complete frame to the Master with all the requested data (again passing through all the Slaves).



This efficient data flow is guaranteed by the 100BASE-TX full-duplex structure of EtherCAT bus which is fitted with two separated lines for transmitting and receiving data.
Moreover the protocols exchange takes place inside the hardware and it is thus independent from CPU and software processing.

6.1.2 ISO/OSI Layer model

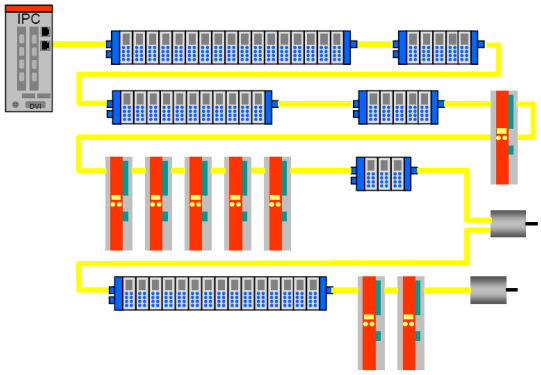


* "Empty" means that the layer behaviour exists, but it is not shown explicitly.

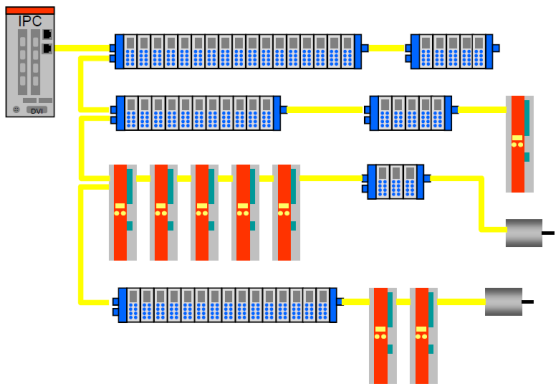
6.1.3 Topology

Several topologies of connection are supported by EtherCAT networks: line, tree, daisy-chain, star, ...). EtherCAT networks can be configured in almost any topology in the same structure. The maximum length of the cable between two Slaves is 100 m (328 ft); standard EtherCAT cables commercially available can be used.
The choice of the topology depends on the structural characteristics of the plant and it is made in order to reduce the complexity and time for cabling.
Inside an EtherCAT network up to 65,535 devices can be connected.
Some topology examples are shown in the Figures below:

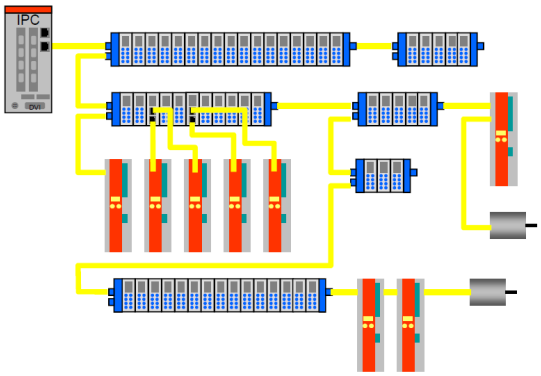
LINE topology:



TREE topology:



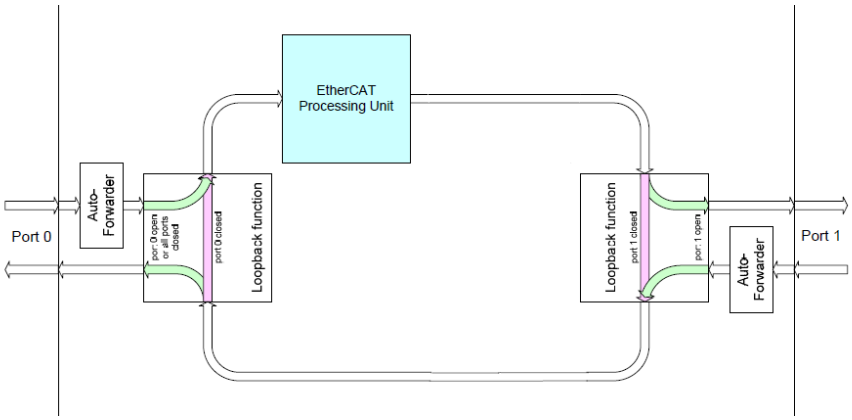
DAISY CHAIN with drop lines topology:



6.1.4 Line Termination

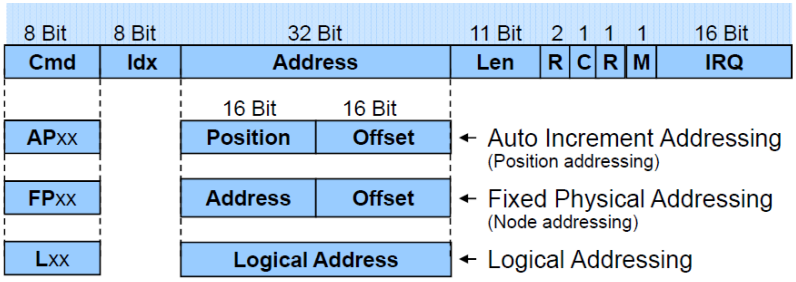
EtherCAT network needs no line termination because the line is terminated automatically; in fact every Slave is able to detect the presence of downstream Slaves.
An EtherCAT Slave is able to detect the presence of the signal in the outgoing line (Port 0) or in the return line (Port 1).
If the Slave is not able to detect the signal in its return line then it closes the communication ring by short-circuiting the TX signal of its outgoing line with the RX signal of its return line; in this way a telegram received through the outgoing line is processed and sent back through the TX of the return line.

The Slave sends a "carrier signal" or a telegram on TX of the outgoing line continuously and, once the next Slave is connected again, a signal on RX of the return line is detected again; so the short circuit is removed and the telegrams are sent on TX of the outgoing line.



6.1.5 Addressing

It is not necessary to assign a physical address to the device (for instance using a dip-switch) because the addressing of the Slave is automatic at power-on during the initial scanning of the hardware configuration.



The field for addressing is 32 bits long; there are three kinds of addressing:

- Auto Increment Addressing = Position Addressing: 16 bits indicate the physical position of the Slave inside the network while 16 bits are scheduled for local memory addressing; when the Slave receives the frame then it increments the position address and the Slave receiving address 0 is the addressed device;
- Fixed Addressing = 16 bits indicate the physical address of the Slave inside the network while 16 bits are scheduled for addressing the local memory;
- Logical Address = the Slave is not provided with its own individual address, but it can read and write data in a section of the total memory space available (4 Gigabytes).

6.1.6 Communication mode

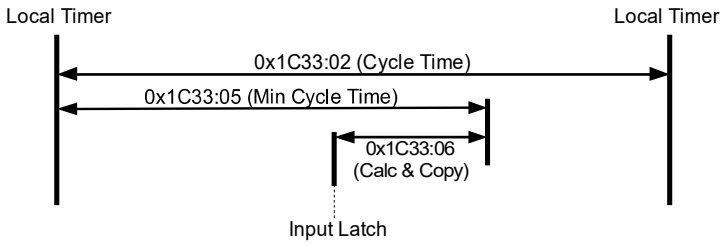
Lika encoders with EtherCAT interface support the following operating modes:

- FreeRun: asynchronous mode;
- SM3 event: synchronous mode;
- DC: distributed clock synchronization mode (synchronous mode).

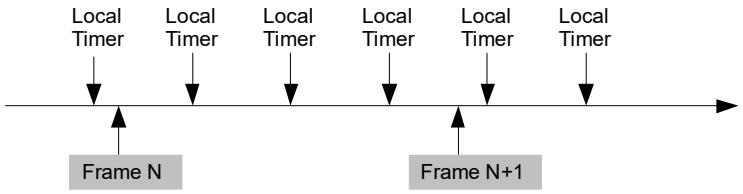
For a system that requires high performances in real time (closed-loop applications) we suggest using DC mode; if real time requirements are not so mandatory SM3 or FreeRun modes can be used instead.
A reference parameter is the "Jitter": it represents the temporal fluctuation of the instant data sampling; in other words data sampled by the micro-controller is available in ECAT DPRAM memory after a certain time and the measure of the variability over time is the "jitter".

FreeRun

Asynchronous mode; the encoder position is sampled directly from EtherCAT frame sent by the Master; the position update is performed by an internal timer of the controller every 100 microseconds.



This operating mode has a high sampling jitter (up to 100 microseconds) and can be chosen only when cycle times are quite longer than the jitter if we want to ensure a sufficient real-time system performance.

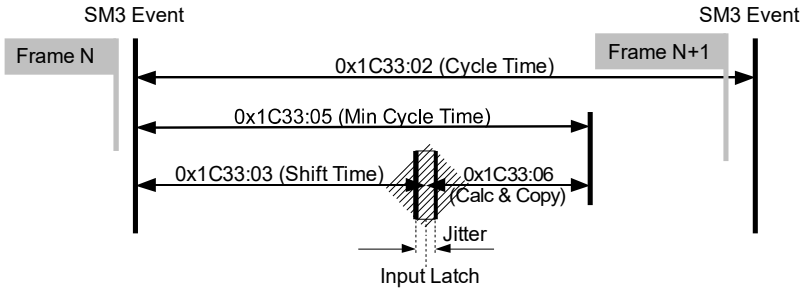


Description	Min	Typ	Max	
Jitter	0		100	µsec
Cycle Time	1000		64000	µsec

See the **1C33 SM input parameter** entry on page 66.

Synchronous with SM3

In this mode data is sampled and then copied into Sync Manager buffer as soon as previous data was read from the Master (SM event); in this way new sampled data is synchronous with Master readings.



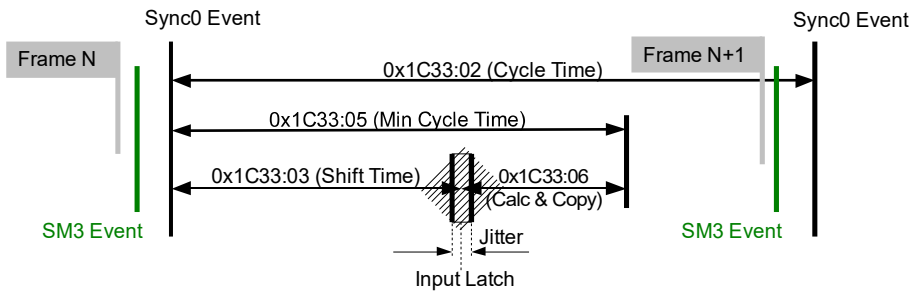
New data will be read by the Master at next cycle (following SM3 event), so if cycle time is too long, data could be relatively old for a real-time system.
The main advantage is that data is updated exactly when Master is reading (synchronous mode).

Description	Min	Typ	Max	
Jitter	0		7.2	ns
Cycle Time	62.5		64000	µs

See the **1C33 SM input parameter** entry on page 66.

Synchronous with DC SYNC0

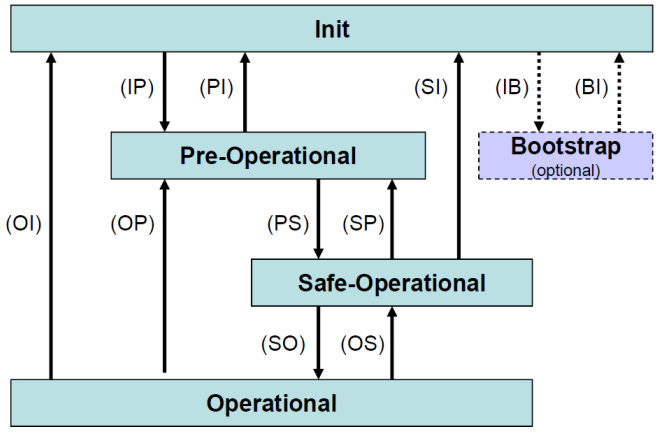
In this operating mode data is sampled and then copied into Sync Manager buffer simultaneously at SYNC0 event generated by the ESC capture/compare unit.
Time required for accomplishing these operations is set next to the **1C33 SM input parameter** object; in particular object **03 Shift time** (1C33hex, sub3) and object **06 Calc and Copy time** (1C33hex, sub6).



In this operating mode "Jitter" is a fundamental parameter in the sampling of two consecutive data. The main advantage of this mode is that there is a direct relation between the sampling instant and the absolute time of the system; in this way, if we know the shift times of the Slaves, we can have an exact image of the system at a given moment (with a tolerance equal to jitter).

Description	Min	Typ	Max	
Jitter	0	100	200	µsec
Cycle Time	62.5		64000	µsec

6.1.7 EtherCAT State Machine (ESM)



EtherCAT Slave is a state machine; communication and operating characteristics depend on the current state of the device:

- INIT: it is the default state after power-on; in this state there is not direct communication between the Master and the Slave on the Application Layer; some configuration registers are initialized and the Sync Managers are configured;
- PRE-OPERATIONAL (PREOP): in this state mailbox is active; both the Master and the Slave can use the mailbox and its protocols for exchanging specific initialization parameters of the application. Exchange of Process Data (PDO) is forbidden;
- SAFE-OPERATIONAL (SAFEOP): in this state the Master and the Slave can issue only input process data, while output process data are still in SAFE-OPERATIONAL state;
- OPERATIONAL (OP): in this state the Master and the Slave are enabled to send both input process data and output process data.
- BOOTSTRAP (BOOT): no process data communication. Communication only via mailbox on Application Layer available. Special mailbox configuration is

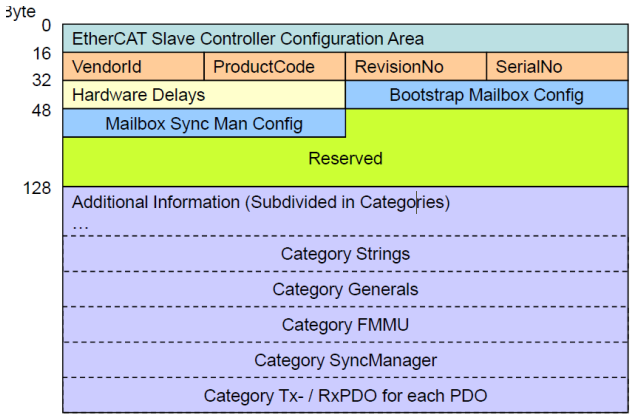
possible, e.g. larger mailbox size. In this state usually the FoE protocol is used for firmware download.

The current state of the Slave is signalled through the green **RUN** LED, see on page 26.

6.1.8 Slave configuration

The configuration of the Slave communication characteristics (Sync Manager configuration, addresses, synchronization modes, PDO mapping, ...) can be made both using the XML file (EtherCat Slave Information - ESI) or loading data directly from EEPROM (Slave Information Interface SII).

EEPROM content (SII):



6.1.9 Timing and Synchronization

The main feature of EtherCAT is its almost ideal representation of a real-time system.
Hence the Master has to synchronize all the Slaves at the same time in order to build a system where all nodes have the same reference time; this goal can be achieved by using "distributed clocks".
The Master downloads its clock into one of the Slaves (customarily the first Slave) which becomes the reference clock for all the Slaves in the network; so it has the task of synchronizing the other Slaves. The Master controller periodically sends a special synchronization-telegram where the reference Slave writes its own "current time". This telegram is then sent to all the other Slaves that, in this way, provide for a new re-synchronization of their own clock in order to avoid possible drifts.
This synchronization of the reference time is very important in order to have a snapshot of the system and accordingly to take simultaneous actions in high sensitive applications such as the coordination in axis control operations.
Besides, the EtherCAT Slave Controller (ESC) is fitted with a capture/compare unit that provides accurate synchronization signals (SYNC0 or interrupts): they

are sent to the local micro-controller so that it is able to synchronize its own clock to Slaves clock.

Sync Manager

Sync Manager has the task of synchronizing data transfer between the Master and the Slave and prevents the same memory area from being written by different events.
There are two synchronization modes:

- 3-Buffer Mode;
- 1-Buffer Mode.

The synchronisation mode is initialized through the XML file or by loading data directly from EEPROM (SII).

Buffered Mode (3-Buffer Mode)

In this mode new data can be accessed at any time by both the EtherCAT Master and the ESC controllers; no timing restrictions are imposed.
Three buffers are necessary (three consecutive memory areas); one buffer is always available to the ESC controller for writing and one buffer always contains updated data to be read by the Master.
The buffered mode is typically used for cyclic data exchange, i.e. process data.

Mailbox Mode (1-Buffer Mode)

In this mode a "handshake" between the Master and the Slave must be used; in fact one only memory buffer is available to both the Master and the Slave for writing and reading; the Master (or the Slave) is enabled to write only when the buffer is empty, that is when the Slave (or the Master) has finished reading the data buffer. And vice versa: the Master (or the Slave) is enabled to read only when the buffer is empty, that is when the Slave (or the Master) has finished writing the data buffer. The mailbox mode is typically used for application layer protocols and exchange of acyclic data (e.g. parameter settings).

The draw-wire encoder features four Sync Managers, see the **1C00 Sync Manager Communication Type** object on page 65:

- **Sync Manager 0 (SM MailBox Receive, SM0)**
  Used for mailbox write transfers (Master to Slave).
  The module has a configurable write mailbox size with default size of 276 bytes, corresponding to 255 bytes plus relevant protocol headers and padding.
- **Sync Manager 1 (SM MailBox Send, SM1)**
  Used for mailbox read transfers (Slave to Master).
  The module has a configurable read mailbox size with default size of 276 bytes, corresponding to 255 bytes plus relevant protocol headers and padding.
- **Sync Manager 2 (SM PDO output, SM2)**
  It contains the RxPDOs (i.e., Sync Manager 2 holds the Read Process Data).
- **Sync Manager 3 (SM PDO input, SM3)**

It contains the TxPDOs (i.e., Sync Manager 3 holds the Write Process Data).

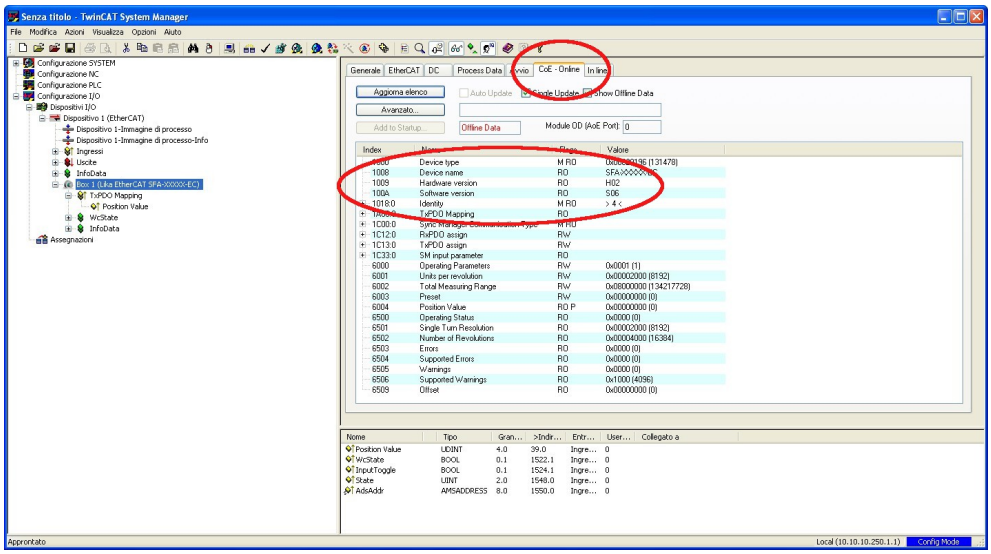
6.2 CANopen Over EtherCAT (CoE)

Lika draw-wire encoders are Slave devices and support "CanOpen Over EtherCAT" (COE) mode for data transfer. In particular, they support the "CANopen DS301 Communication profile", Class 2.
For any omitted specification on CANopen® protocol, please refer to "CiA Draft Standard Proposal 301. Application Layer and Communication Profile" and "CiA Draft Standard 406. Device profile for encoders" documents available at the address **www.can-cia.org**.
For any omitted specification on EtherCAT® protocol, please refer to "ETG.1000 EtherCAT Specification" document available at the address **www.ethercat.org**.

6.2.1 XML file

EtherCAT® draw-wire encoders are supplied with their own XML file **Lika_SFA_XXXXX_ECx_Vx.xml** (V6 or higher version; see at **www.lika.biz > ROTARY ENCODERS > DRAW-WIRE UNITS (DRAW-WIRE) > ABSOLUTE**.
The XML file must be installed in the EtherCAT® Master device.

If you want to know the firmware version of a device, press the **Box (Lika EtherCAT SFA-XXXXX-EC)** item in the left pane of the **TwinCAT System Manager** main window: some tabbed pages for configuring and managing the device will appear in the right pane. Enter the **CoE - Online** page and refer to the **1009-00 Hardware version** and **100A-00 Software version** indexes.



6.2.2 Communication messages

EtherCAT Datagram of CoE mode has the following structure:

Mbx Header	CoE Cmd			Cmd specific data
type = 3	Number	res	Type	
6 bytes	9 bits	3 bits	3 bits	0 ... 1478 bytes

Mbx Header = 3 CoE mode

Number = 0 in case of SDO messages
≠ 0 in case of PDO messages, it defines the type of service

res reserved bits

Type = 0 reserved
= 1 Emergency message
= 2 SDO request
= 3 SDO response
= 4 Transmitted PDO (TxPDO)
= 5 Received PDO (RxPDO)
= 6 Remote transmission request of TxPDO
= 7 Remote transmission request of RxPDO
= 8 SDO information
= 9 ... 15 reserved

Cmd specific data PDO messages: are the process data, e.g. position value
SDO messages: standard CANopen frame

Transmit (tx) or receive (rx) "Type" is viewed from the Slave.

6.2.3 Process Data Objects (PDO)

PDO messages are used for transmitting or receiving process data in real time; data to be transmitted or received is defined in PDO Mapping and managed by Sync Manager PDO Mapping.

6.2.4 Service Data Objects (SDO)

SDO messages are issued via Mailbox (low priority data); Segmented SDO Service and SDO Complete Access are not supported (transfer of low size data and one sub-index at a time).

"CoE Cmd type" = 2 or 3

Structure of "Cmd specific data":

Cmd specific data				
SDO control	Index	Sub index	Data	Data optional
8 bits	16 bits	8 bits	32 bits	1 ... 1470 bytes

SDO control	standard CANopen SDO Service
Index	parameter index
Sub index	parameter sub-index
Data	parameter value
Data optional	optionally, more then 4 bytes of data can be sent in one frame. Full mailbox size usable.

Index and sub-index values are described in the "Object Dictionary".

6.2.5 Object dictionary

The most important part of a device profile is the Object Dictionary. The Object Dictionary is essentially a grouping of objects accessible via the network in an ordered, pre-defined mode. Each object within the dictionary is addressed using a 16-bit index.
The Object Dictionary can contain a maximum of 65,536 entries.
The user-related objects are grouped in two main areas: the Communication Profile Area and the Standardised Device Profile Area. The objects are all described in the XML file.

The **Communication Profile Area** at indexes from 1000h to 1FFFh contains the communication specific parameters for the EtherCAT network. These entries are common to all devices. PDO objects and SDO objects are described in this section. The Communication Profile Area objects comply with the "CiA Draft Standard Proposal 301. Application Layer and Communication Profile". Refer to the "Communication Profile Area objects (DS 301)" section on page 63.

The **Standardised Device Profile Area** at indexes from 6000h to 9FFFh contains all data objects common to a class of devices that can be read or written via the network. The device profiles may use entries from 6000h to 9FFFh to describe the device parameters and the device functionality. The Standardised Device Profile Area objects comply with the "CiA Draft Standard 406 CANopen Device profile for encoders". Refer to the "Standardised Device Profile Area objects (DS 406)" section on page 68.

In the following pages the objects implemented are listed and described as follows:

Index-subindex Object name
[data types, attribute]

- Index and sub-index are expressed in hexadecimal notation.
- Attribute:

ro = read only access
rw = read and write access

Signed8 / Unsigned8 data type:

Process data bytes							
byte 4							
7	6	5	4	3	2	1	0
MSbit			...				LSbit

Signed16 / Unsigned16 data type:

Process data bytes					
byte 4			byte 5		
7	...	0	15	...	8
LSByte			MSByte		

Signed32 / Unsigned32 data type:

Process data bytes											
byte 4			byte 5			byte 6			byte 7		
7	...	0	15	...	8	23	...	16	31	...	24
LSByte			...			...			MSByte		

NOTE
Always save the new values after setting in order to store them in the non-volatile memory permanently. Use the **1010-01 Store parameters** object, see on page 63.
Should the power supply be turned off all data that has not been saved previously will be lost!

Communication Profile Area objects (DS 301)

1000-00 Device type

[Unsigned32, ro]
It contains information about the device type. The object describes the type of device and its functionality.
Default = 0002 0196h = multiturn rotary encoder, DS 406

1008-00 Device Name

[String, ro]
It shows the manufacturer device name, expressed in ASCII code.
Default = 5346412D58585858582D4543h = "SFA-XXXXX-EC" = SFA draw-wire encoder with EtherCAT interface

1009-00 Hardware version

[String, ro]
It shows the hardware version of the device, expressed in ASCII code.
Default = 483032h = "H02" = hardware version 2

100A-00 Software version

[String, ro]
It shows the software version of the device, expressed in ASCII code.
Default = 533036h = "S06" = software version 6

1010-01 Store parameters

[Unsigned32, rw]
Use this object to save all parameters on the non-volatile memory.
Write "**save**" (ASCII code in hexadecimal format) in the data bytes:

Master → Encoder

Cmd specific data							
Cmd	**Index**		**Sub**	**Data**			
23	10	10	01	73	61	76	65
				s	a	v	e

Encoder → Master (confirmation)

Cmd specific data							
Cmd	**Index**		**Sub**	**Data**			
60	10	10	01	00	00	00	00

1011-01 Restore default parameters

[Unsigned32, rw]
This object allows the operator to restore all parameters to default values. The default parameters are set at the factory by Lika Electronic engineers to allow

the operator to run the device for standard operation in a safe mode. A list of machine data and relevant default parameters preset by Lika Electronic engineers is available on page 83.
Write "**load**" (ASCII code in hexadecimal format) in the data bytes:

Master → Encoder

Cmd specific data							
Cmd	**Index**		**Sub**	**Data**			
23	11	10	01	6C	6F	61	64
				l	o	a	d

Encoder → Master (confirmation)

Cmd specific data							
Cmd	**Index**		**Sub**	**Data**			
60	11	10	01	00	00	00	00

NOTE
To save the default values execute the "Store parameters" function (see the 1010-01 Store parameters object). When the power is turned off, parameters not saved are lost.

1018 Identity

[Unsigned8, ro]
This object contains general information about the device. Sub-Index 00 contains the number of entries.
Default = 4

01 Vendor ID
[Unsigned32, ro]
It provides the manufacturer-specific vendor ID. The EtherCAT vendor ID is equal to the CANopen vendor ID.
Default = 0000 012Eh

02 Product code
[Unsigned32, ro]
The manufacturer-specific product code identifies a specific device version.
Default = 0000 0002h

03 Revision
[Unsigned32, ro]
The manufacturer-specific revision number consists of a major revision number and a minor revision number. The major revision number identifies a specific device behaviour. The minor revision number identifies different versions with the same device behaviour.
Default = device dependent

7	...	0	15	...	8	23	...	16	31	...	24
Minor revision number						Major revision number					
LSB			...			...			MSB		

04 Serial number
[Unsigned32, ro]
It provides the Serial Number of the device. It is 0 if no serial number is provided.
Default = 0000 0000h

1A00-01 TxPDO mapping

[Unsigned8, ro]
This object contains the mapping parameters for the PDOs the EtherCAT device is able to transmit. Sub-Index 00 contains the number of entries.

01 Mapped Object 001
[Unsigned32, rw]
Sub-Index 01 contains the information of the mapped application object 001. The object describes the content of the PDO by its index, sub-index and length.
The length contains the length of the application object in bits. This may be used to verify the mapping.

7 0	15 8	23 16	31 24
Length	Sub-Index	Index	
LSB			MSB

Default = 6004 0020h = **6004-00 Position value** object, length 32 bits

1C00 Sync Manager Communication Type

[Unsigned8, ro]
This object contains the number and type of Sync Manager Communication Types supported by the draw-wire encoder. Sub-Index 00 specifies the number of Sync Manager channels. Refer also to the "Sync Manager" section on page 57.

01 SM MailBox Receive (SM0)
[Unsigned8, ro]
Used for mailbox write transfers (Master to Slave).
Default = 01

02 SM MailBox Send (SM1)
[Unsigned8, ro]
Used for mailbox read transfers (Slave to Master).
Default = 02

03 SM PDO output (SM2)
[Unsigned8, ro]
It contains the RxPDOs (i.e. Sync Manager 2 holds the Read Process Data).
Default = 03

04 SM PDO input (SM3)
[Unsigned8, ro]
It contains the TxPDOs (i.e. Sync Manager 3 holds the Write Process Data).
Default = 04

1C12-00 RxPDO assign

[Unsigned8, ro]
This object specifies whether the device uses Receive PDO messages. This device does not support Receive PDO messages.
Default = 0

1C13-01 TxPDO assign

[Unsigned8, ro]
This object specifies whether the device uses Transmit PDO messages. Sub-Index 00 specifies the number of entries, i.e. the number of assigned TxPDOs.

01 SubIndex 001
[Unsigned32, ro]
This device uses TxPDO messages to send the position value.
Default = 0000 1A00h = **1A00-01 TxPDO mapping** object

1C33 SM input parameter

1C33 SM input parameter object contains the input synchronization parameters. Some of them are calculated dynamically and depend on both the encoder configuration (programmed resolution, counting direction, ...) and the selected synchronization mode (SM or DC). Sub-Index 00 contains the number of entries.

01 Sync Type
[Unsigned16, rw]
It allows to select the synchronization mode. For more information refer to page 53.
0: FreeRun; see on page 53;
1: Synchronous with SM3 Event; see on page 54;
2: DC mode synchronous with SYNC0 event; see on page 54.

Default = 1

02 Cycle time
[Unsigned32, ro]
This parameter depends on the **01 Sync Type** selected. Application cycle time, i.e. interval between two position samplings (internal timer). The value is expressed in nanoseconds (ns).
If 0 = "FreeRun": interval between two position samplings (internal timer).
If 1 = "Synchronous with SM3": minimum interval between two SM3 events.
If 2 = "DC mode synchronous with with SYNC0 event": SYNC0 cycle time.

03 Shift time
[Unsigned32, ro]
Interval between the synchronization event and the moment of inputs latching from hardware. This parameter is calculated dynamically and expressed in nanoseconds (ns).

04 Sync modes supported
[Unsigned16, ro]
It shows the list of the supported synchronization modes.
bit 0: FreeRun (supported)
bit 1: Synchronous with SM3 (supported)
bit 2: Synchronous with DC SYNC0 (supported)
Default = 7

05 Minimum cycle time
[Unsigned32, ro]
Max. duration of the encoder internal cycle time. This parameter is calculated dynamically and depends on the operating parameters and the position value. It is expressed in nanoseconds (ns).

06 Calc and Copy time
[Unsigned32, ro]
Time the internal micro-controller (DSP) needs to make calculations on latched optical reading of position and then copy updated data from local memory to ESC memory (Sync Manager) before they are available to EtherCAT. This parameter is calculated dynamically and depends on the operating parameters and the position value. It is expressed in nanoseconds (ns).

NOTE
Always save the new values after setting in order to store them in the non-volatile memory permanently. Use the **1010-01 Store parameters** object, see on page 63.
Should the power supply be turned off all data that has not been saved previously will be lost!

Standardised Device Profile Area objects (DS 406)

6000-00 Operating parameters

[Unsigned16, rw]

Bit	Function	bit = 0	bit = 1
0	**Code sequence**	Count up rewinding the wire	**Count up pulling the wire out**
1	not used		
2	**Scaling function**	**disabled**	enabled
3 ... 15	not used		

Default values are highlighted in bold
Default = 0100h

Code sequence
This is intended to set whether the count increases (count up information) when you rewind the wire or when you pull the wire out.
Setting 0 (bit 0 **Code sequence** = 0) causes the position value to increase when you rewind the wire; on the contrary, setting 1 (bit 0 **Code sequence** = 1) causes the position value to increase when you pull the wire out.
Default = 1

To know whether the **Code sequence** is currently set to "Count up rewinding the wire" or "Count up pulling the wire out", you can read the bit 0 **Code sequence** of the 6500-00 Operating status object, see on page 77.

WARNING
Every time you change the **Code sequence**, then you are required to set a new preset value (see the 6003-00 Preset value object).

Scaling function
This is meant to enable / disable the scaling parameters 6001-00 Units per revolution and 6002-00 Total Measuring Range.
When the scaling function is disabled (bit 2 **Scaling function** = 0), the encoder uses its own physical resolution (i.e. hardware counts per revolution and number of hardware revolutions, see the encoder identification label and the 6501-00 Hardware counts per revolution and 6502-00 Hardware number of turns objects; see also on page 21) to arrange the absolute position information.
On the contrary, when the scaling function is enabled (bit 2 **Scaling function** = 1), the encoder uses the custom resolution set next to the 6001-00 Units per revolution and 6002-00 Total Measuring Range objects according to the following relation:

$$\text{Transmitted position} = \frac{\text{6001-00 Units per revolution}}{\text{6501-00 Hardware counts per revolution}} \times \text{Real position} \leq \text{6002-00 Total Measuring Range}$$

The value in the **6001-00 Units per revolution** object must be less than or equal to the value in the **6501-00 Hardware counts per revolution** object. The total custom resolution in the **6002-00 Total Measuring Range** object must be less than or equal to the maximum physical value (**6501-00 Hardware counts per revolution** * **6502-00 Hardware number of turns**).
Default = 0

To know whether the **Scaling function** is currently enabled, you can read the bit 2 **Scaling function** of the **6500-00 Operating status** object, see on page 77.

WARNING
When you enable the scaling function (bit 2 **Scaling function** = 1), please enter scaled values next to the **6001-00 Units per revolution** and **6002-00 Total Measuring Range** objects that are consistent with the physical values. In the case of inconsistent values, the system will warn about the wrong parametrization and fault condition by means of the dedicated objects and visually by means of the diagnostic LEDs.

WARNING
Every time you enable the scaling function and/or change the scaled values (see the **6001-00 Units per revolution** and **6002-00 Total Measuring Range** objects), then you are required to set a new preset value (see the **6003-00 Preset value** object).

6001-00 Units per revolution

[Unsigned32, rw]

WARNING
This object is active only if the bit 2 **Scaling function** in the **6000-00 Operating parameters** object is set to "=1"; otherwise it is ignored and the system uses the physical values (**6501-00 Hardware counts per revolution** and **6502-00 Hardware number of turns**) to calculate the position information.

This object sets a custom number of distinguishable steps per revolution (custom singleturn resolution).

To avoid counting errors, check that

$$\frac{\text{6501-00 Hardware counts per revolution}}{\text{6001-00 Units per revolution}} = \text{integer value.}$$

You are allowed to set whatever integer value less than or equal to the **maximum number of physical steps per revolution** (see the hardware counts per revolution in the encoder identification label and the **6501-00 Hardware counts per revolution** object).

Default = 0000 2000h (8,192 cpr)

WARNING

When you set a new value next to the **6001-00 Units per revolution** object, please always check also the **6002-00 Total Measuring Range** object value and be sure that the resulting number of revolutions complies with the **Hardware number of revolutions** of the device (16,384 revolutions, see the **6502-00 Hardware number of turns** object).

Let's suppose that the encoder is programmed as follows:

6001-00 Units per revolution: 8,192 cpr

6002-00 Total Measuring Range = 33 554 432_{10} = 8,192 (cpr) * 4,096 (rev.)

Let's set a new singleturn resolution, for instance: **6001-00 Units per revolution** = 360 cpr.

If we do not change the **6002-00 Total Measuring Range** value at the same time, we will get the following result:

$$\text{Number of revolutions} = \frac{33\,554\,432\ (\text{6002-00 Total Measuring Range})}{360\ (\text{6001-00 Units per revolution})} = 93{,}206.755...$$

As you can see, the encoder is required to carry out more than 93,000 revolutions, this cannot be as the hardware number of revolutions is, as stated, 16,384. When this happens, the encoder falls into an error signalling the faulty condition through the diagnostic LEDs (see on page 26).

WARNING

When you enable the scaling function (bit 2 **Scaling function** = 1), please enter scaled values next to the **6001-00 Units per revolution** and **6002-00 Total Measuring Range** objects that are consistent with the physical values. In the case of inconsistent values, the system will warn about the wrong parametrization and fault condition by means of the dedicated objects and visually by means of the diagnostic LEDs.

WARNING
Every time you change the scaled values (see the **6001-00 Units per revolution** and **6002-00 Total Measuring Range** objects), then you are required to set a new preset value (see the **6003-00 Preset value** object).

6002-00 Total Measuring Range

[Unsigned32, rw]

WARNING
This object is active only if the bit 2 **Scaling function** in the **6000-00 Operating parameters** object is set to "=1"; otherwise it is ignored and the system uses the physical values (**6501-00 Hardware counts per revolution** and **6502-00 Hardware number of turns**) to calculate the position information.

This object sets a custom number of distinguishable steps over the total measuring range. In other words, this object allows to set the length of the travel the encoder has to measure expressed in number of distinguishable steps (number of information). The total resolution of the encoder results from the product of **6001-00 Units per revolution** by the required **Number of revolutions**.
You are allowed to set whatever integer value less than or equal to the **overall hardware resolution** (see the encoder identification label). The overall hardware resolution results from:
6501-00 Hardware counts per revolution * **6502-00 Hardware number of turns**.
We recommend the **Number of revolutions** to be set to a power of 2.
The set **Number of revolutions** results from the following calculation:

$$\text{Number of revolutions} = \frac{\text{6002-00 Total Measuring Range}}{\text{6001-00 Units per revolution}}$$

Setting the **Number of revolutions** to a value which is a power of 2 is meant to avoid problems when using the device in endless operations requiring the physical zero to be overstepped. If you set the **Number of revolutions** which is not a power of 2, a counting error is generated before the physical zero.
Default = 0800 0000h (134 217 728)

WARNING
When you set a new value next to the **6002-00 Total Measuring Range** object, please always check also the **6001-00 Units per revolution** object value and be sure that the resulting number of revolutions complies with the Hardware number of revolutions of the device (16,384 revolutions).
Let's suppose that the encoder is programmed as follows:

6001-00 Units per revolution: 8,192 cpr
6002-00 Total Measuring Range = 134 217 728_{10} = 8,192 (cpr) * 16,384 (rev.)
Let's set a new total resolution, for instance: **6002-00 Total Measuring Range** = 360.
As the **6002-00 Total Measuring Range** must be greater than or equal to the **6001-00 Units per revolution**, the above setting is not allowed.

EXAMPLE
We install the following draw-wire encoder: **SFA-5000-EC-8192-M12**.

The physical values are:
Stroke per turn of the drum = 200 mm (7.874")
Physical resolution per turn = 13 bits = 8,192 cpr
Max. number of physical revolutions = 16,384 revolutions
Total physical resolution = 27 bits = 134 217 728 information
Physical linear resolution = 0.024 mm = 24 µm
Max. number of turns of the drum = 25
Max. measuring length = 5,000 mm (196.85")
Number of information = 204,800

Let's suppose that we need a tenth of a millimetre linear resolution in the specific installation.

- Enable the scaling function: bit 2 **Scaling function** in the **6000-00 Operating parameters** object = 1
- Custom resolution per turn = **6001-00 Units per revolution** = 2,000 cpr
- Linear resolution = 0.1 mm = 100 µm

$$\text{Linear resolution} = \frac{\text{Stroke per turn}}{\text{6001-00 Units per revolution}} = \frac{200\text{ mm}}{2{,}000} = 0.1\text{ mm}$$

The custom number of revolutions can be as the physical number of revolutions:

$$\text{Custom number of encoder revolutions} = \frac{\text{6002-00 Total Measuring Range}}{\text{6001-00 Units per revolution}} = 16{,}384$$

- **6002-00 Total Measuring Range** = 32 768 000

NOTE

Please note that if you set a preset along the path, when the encoder moves back and cross the zero, the value immediately after 0 will be 32 768 000 - 1, i.e. 32 767 999.

...	32,767,997	32,767,998	32,767,999	0	1	2	...

EXAMPLE

Using the values in the previous example let's suppose that the travel in the application is 2 m long. As the stroke per turn is 200 mm you need 10 revolutions to cover the travel length.

- **6002-00 Total Measuring Range** = **6001-00 Units per revolution** * custom number of revolutions = 2,000 * 10 = 20,000

In fact:

$$\text{Custom number of encoder revolutions} = \frac{\text{6002-00 Total Measuring Range}}{\text{6001-00 Units per revolution}} = 10$$

In this case you will obtain several 20,000 information sections following each other all along the whole measuring length. The position information will be from 0 to 19,999; then again from 0 to 19,999 and so on.

...	19,997	19,998	19,999	0	1	2	...	19,997	19,998	19,999	0	1	2	...

← max. measuring length →

NOTE

To avoid counting errors we recommend values which are power of 2 (2^n: 2, 4, ..., 2048, 4096, 8192,...) to be set next to the **6001-00 Units per revolution** and **6002-00 Total Measuring Range** objects.

WARNING

If you have set the preset, when you change the value next to **6001-00 Units per revolution** and/or **6002-00 Total Measuring Range** object, then you must check the value in the **6003-00 Preset value** object and execute the preset operation.

6003-00 Preset value

[Unsigned32, rw]

This object allows to set the encoder position to a Preset value. The Preset function is meant to assign a desired value to a physical position of the encoder (i.e. a position in the travel of the wire). The chosen physical position will get the value set next to this item and all the previous and following positions will get a value according to it. This function can be useful, for instance, when the zero position of the encoder and the zero position of the axis need to match.

The preset value will be set for the position of the encoder (i.e. the position of the wire) in the moment when the preset value is transmitted. We suggest setting the preset value when the encoder is in stop.

Default = 0000 0000h

EXAMPLE

Let's take a look at the following example to better understand the preset function and the meaning and use of the related objects: **6003-00 Preset value** and **6509-00 Offset value**.

The encoder position which is transmitted results from the following calculation:

Transmitted value = **read position** (it does not matter whether the position is physical or scaled) + **6003-00 Preset value** - **6509-00 Offset value**.

If you never set the **6003-00 Preset value** and you never performed the preset setting, then the transmitted value and the read position are necessarily the same as **6003-00 Preset value** = 0 and **6509-00 Offset value** = 0.

When you set the **6003-00 Preset value** and then execute the preset setting, the system saves the current encoder position in the **6509-00 Offset value** object. It follows that the transmitted value and the **6003-00 Preset value** are the same as **read position** - **6509-00 Offset value** = 0; in other words, the value set next to the **6003-00 Preset value** object is paired with the current position of the encoder as you wish.

For example, let's assume that the value "50" is set next to the **6003-00 Preset value** object and you execute the preset setting when the encoder position is "1000". In other words, you want to receive the value "50" when the encoder reaches the position "1000".

We will obtain the following information sequence:

Transmitted value = **read position** (="1000") + **6003-00 Preset value** (="50") - **6509-00 Offset value** (="1000") = 50.

The following transmitted value will be:

Transmitted value = **read position** (="1001") + **6003-00 Preset value** (="50") - **6509-00 Offset value** (="1000") = 51.

And so on.

Set the Preset value 6003-00 Preset value (preset = 1000 = 03E8h)

Master → Encoder

Cmd specific data							
Cmd	**Index**		**Sub**	**Data**			
23	03	60	00	E8	03	00	00

Encoder → Master (Set confirmation)

Cmd specific data							
Cmd	**Index**		**Sub**	**Data**			
60	03	60	00	00	00	00	00

NOTE

- If the scaling function is disabled (see the bit 2 **Scaling function** in the 6000-00 Operating parameters object), the 6003-00 Preset value must be less than or equal to the "**Total hardware resolution**" - 1 (6501-00 Hardware counts per revolution * 6502-00 Hardware number of turns - 1).
- If the scaling function is enabled (see bit 2 **Scaling function** in the 6000-00 Operating parameters object), the 6003-00 Preset value must be less than or equal to the 6002-00 Total Measuring Range - 1.

WARNING

Check the value in the 6003-00 Preset value object and perform the preset operation every time you change the **Code sequence** or the scaled values (6001-00 Units per revolution and/or 6002-00 Total Measuring Range objects).

6004-00 Position value

[Unsigned32, ro]

This object contains the current position value of the encoder.

If the scaling function is enabled (see the bit 2 **Scaling function** in the 6000-00 Operating parameters object), the output value is scaled according to the scaling parameters. The 6004-00 Position value object is mapped in the 1A00-01 TxPDO mapping object, see on page 65.

WARNING

Please note that the position value issued by the encoder is expressed in pulses; thus you have then to convert the number of pulses into a linear measuring unit.

To convert the position value into millimetres (mm) or micrometres (µm) you have to multiply the number of information by the linear resolution of the encoder expressed in millimetres or micrometres.

To know the linear resolution of the encoder please consider that **the stroke per turn of the drum is 200 mm**.

The linear resolution results from the following calculation:

$$\textbf{Linear resolution} = \frac{\text{Stroke per turn of the drum}}{\text{Singleturn resolution cpr}}$$

If you want to know the linear position value you will need to multiply the transmitted position value by the linear resolution.

Linear position value = transmitted position * linear resolution

NOTE
Please note that the encoder's linear physical resolution can be read also in the order code next to the rotary resolution. Refer to the product datasheet.

EXAMPLE 1
Let's suppose that we are using the physical resolution of the SFA-5000-EC-8192-M12 draw-wire encoder (the bit 2 **Scaling function** in the **6000-00 Operating parameters** object = 0).
The physical singleturn resolution of the measuring device is 8,192 cpr (= 0.024 mm, see the order code in the product datasheet).
As stated, the linear resolution results from the following calculation:

$$\textbf{Linear resolution} = \frac{\text{Stroke per turn of the drum}}{\text{Resolution cpr}}$$

$$\textbf{Linear resolution} = \frac{200}{8192} = 0.024 \text{ mm} = 24 \text{ µm}$$

Let's say that the transmitted position value is 123.

Thus the linear position value will be as follows:
Linear position value = transmitted position * linear resolution

Linear position value = 123 * 0.024 = 2.952 mm = 2,952 µm

EXAMPLE 2
Let's suppose that we are using the SFA-5000-EC-8192-M12 draw-wire encoder. The singleturn resolution is set to the custom value of 4,000 cpr (**6001-00 Units per revolution** = 4000). The transmitted position value is 1,569.
The linear resolution can be easily calculated as follows:

$$\textbf{Linear resolution} = \frac{200}{4000} = 0.05 \text{ mm} = 50 \text{ µm}$$

Thus the linear position value will be as follows:

Linear position value = 1,569 * 0.05 = 78.45 mm = 78 450 µm

6500-00 Operating status

[Unsigned16, ro]

Bit	Function	bit = 0	bit = 1
0	**Code sequence**	Count up rewinding the wire	**Count up pulling the wire out**
1	not used		
2	**Scaling function**	**Disabled**	Enabled
3 ... 15	not used		

Code sequence

It shows the value that is currently set through the bit 0 **Code sequence** in the 6000-00 Operating parameters object. If the bit is "=0" the output encoder position value has been set to increase when you rewind the wire; if the bit is "=1" instead the output encoder position value has been set to increase when you pull the wire out. For any further information on setting and using the counting direction function refer to the 6000-00 Operating parameters object on page 68.

Scaling function

It shows the value that is currently set through the bit 2 **Scaling function** in the 6000-00 Operating parameters object. In other words, it is intended to show whether the scaling function is enabled or disabled. If the value is "=0" the scaling function is disabled; if the value is "=1" instead the scaling function is enabled. For any further information on setting and using the scaling function refer to the 6000-00 Operating parameters object on page 68.

6501-00 Hardware counts per revolution

[Unsigned32, ro]

WARNING

This object is active only if the bit 2 **Scaling function** in the 6000-00 Operating parameters object is set to "=0"; otherwise it is ignored and the system uses the custom values (6001-00 Units per revolution and 6002-00 Total Measuring Range) to calculate the position information.

This object is intended to show the number of physical distinguishable steps per each revolution provided by the hardware (physical singleturn resolution).
If you want to set a custom singleturn resolution see the **6001-00 Units per revolution** object on page 69.

6502-00 Hardware number of turns

[Unsigned32, ro]

WARNING
This object is active only if the bit 2 **Scaling function** in the **6000-00 Operating parameters** object is set to "=0"; otherwise it is ignored and the system uses the custom values (**6001-00 Units per revolution** and **6002-00 Total Measuring Range**) to calculate the position information.

This object is intended to show the number of physical revolutions provided by the hardware (number of physical revolutions).
The **Total hardware resolution** results from **6501-00 Hardware counts per revolution** * **6502-00 Hardware number of turns**.
If you want to set a custom number of revolutions see the **6001-00 Units per revolution** and **6002-00 Total Measuring Range** objects on page 69 ff.

6503-00 Errors

[Unsigned16, ro]
The corresponding bits of supported errors are set (see the **6504-00 Supported errors** object).

6504-00 Supported errors

[Unsigned16, ro]
This object contains the information on the error alarms supported by the encoder. No error alarms are supported in this encoder.
Default = 0000h (No errors supported).

6505-00 Warnings

[Unsigned16, ro]
The corresponding bits of supported warnings are set (see the **6506-00 Supported warnings** object).

6506-00 Supported warnings

[Unsigned16, ro]
This object contains the information on the warnings supported by the encoder.

Byte 0 Not used

bits 7 ... 11 Not used

Flash memory error

bit 12 Wrong parameters loaded from flash memory at power on.

bits 13 ... 15 Not used

6509-00 Offset value

[Unsigned32, ro]

As soon as you activate the preset, the current position value of the encoder is saved on this object. The offset value is then used in the preset function in order to calculate the encoder position value to be transmitted. To zero set the value in this object you must upload the factory default values (see the **1011-01 Restore default parameters** object on page 63).

For any further information on the preset function and the meaning and use of the related objects **6003-00 Preset value** and **6509-00 Offset value** please refer to page 74.

NOTE

Always save the new values after setting in order to store them in the non-volatile memory permanently. Use the **1010-01 Store parameters** object, see on page 63.

Should the power supply be turned off all data that has not been saved previously will be lost!

6.2.6 SDO Abort codes

SDO transfer could be unsuccessful; causes of error are listed and described in the SDO Abort Codes. Here follows the list of the available SDO Abort Codes. For complete information see ETG1000.6 "EtherCAT Specification – Part 6. Application Layer protocol specification", par. 5.6.2.7.2, table 40.

Abort code	Description
0503 0000h	Toggle bit not changed.
0504 0000h	SDO protocol timeout.
0504 0001h	Client/Server command specifier not valid or unknown.
0504 0005h	Out of memory.
0601 0000h	Unsupported access to an object.
0601 0001h	Attempt to read a write only object.
0601 0002h	Attempt to write a read only object.
0602 0000h	The object does not exist in the object dictionary.
0604 0041h	The object cannot be mapped into the PDO.
0604 0042h	The number and length of the objects to be mapped would exceed PDO length.
0604 0043h	General parameter incompatibility reason.
0604 0047h	General internal incompatibility in the device.
0606 0000h	Access failed due to a hardware error.
0607 0010h	Data type does not match, length of service parameter does not match
0607 0012h	Data type does not match, length of service parameter too high
0607 0013h	Data type does not match, length of service parameter too low
0609 0011h	Subindex does not exist.
0609 0030h	Value range of parameter exceeded (only for write access).
0609 0031h	Value of parameter written too high.
0609 0032h	Value of parameter written too low.
0609 0036h	Maximum value is less than minimum value.
0800 0000h	General error
0800 0020h	Data cannot be transferred or stored to the application.
0800 0021h	Data cannot be transferred or stored to the application because of local control.
0800 0022h	Data cannot be transferred or stored to the application because of the present device state.
0800 0023h	Object dictionary dynamic generation fails or no object dictionary is present.

Refer also to the "4.6 Diagnostic LEDs" section on page 26.

6.2.7 Emergency Error Codes

Emergency Service is used by server for transmitting diagnostic messages to the client using MailBox; Error Codes are listed and described in the ETG1000.6 "EtherCAT Specification – Part 6. Application Layer protocol specification", par. 5.6.4.2, table 50.

Error Code		Error Register	Diagnostic Data				
Byte (0)	Byte (1)	Byte (2)	Byte (3)	Byte (4)	Byte (5)	Byte (6)	Byte (7)

Error Code — State Transition Errors of state machine:
(for detailed description see ETG1000.6 par. 5.6.4.3)
A000hex: transition from **PRE-OPERATIONAL** to **SAFE-OPERATIONAL** not successful
A001hex: transition from **SAFE-OPERATIONAL** to **OPERATIONAL** not successful
Encoder errors:
5000hex: **Hardware error**
5001hex: **Diagnostic data** (wrong parameters loaded from flash memory)

Error Register — EtherCAT state machine current status (ESM)

Diagnostic Data — information about possible error causes (see ETG1000.6 par. 5.6.4.3.2 ... 5.6.4.3.5).

Refer also to the "4.6 Diagnostic LEDs" section on page 26.

6.2.8 AL Status Error Codes

If the state transition requested by the Master through the "AL Control Register" is unsuccessful, Slave sets to 1 the "Error Indicator Bit" in "AL Status Register" and writes the cause of the error in "AL Status Code Register".
Values and descriptions of "AL Status Code" are available in ETG1000.6 "EtherCAT Specification – Part 6. Application Layer protocol specification", par. 5.3.2 Table 11.

6.3 File Over EtherCAT (FoE)

Lika encoders are devices that allow the firmware update using the protocol "File over EtherCAT (FoE)".
For any specification on FoE protocol, please refer to "ETG.1000 EtherCAT Specification" document available at the address **www.ethercat.org**.
Please refer also to the "5.7 Firmware upgrade" section on page 44.

7 – Default parameters list

Default values are expressed in hexadecimal notation.

Parameters list	Default values		
6000-00 Operating parameters **Bit 0 Code sequence** **Bit 2 Scaling function**	0100h 1 = count up information when pulling the cable out 0 = disabled		
6001-00 Units per revolution	0000 2000h (8,192 cpr)		
6002-00 Total Measuring Range	0800 0000h (134 217 728)		
6003-00 Preset value	00000 0000h		

Document release	Release date	Description	HW	SW	XML file version
1.0	22.01.2018	First issue	2	6	From release V6 to ...

Dispose separately

Lika Electronic
Via S. Lorenzo, 25 • 36010 Carrè (VI) • Italy

Tel. +39 0445 806600
Fax +39 0445 806699

info@lika.biz • www.lika.biz