Smart encoders & actuators

User's guide

AM58S-1314-EPP9-M12
AM59H-1314-EP15-M12

EtherNet/IP™

- Up to 13-bit resolution per turn and up to 14-bit revolutions (turns)
- In compliance with ODVA specification, edition April 2023
- Class 1 Real Time Ethernet (RTE) according to IEC 61 784-2
- With Energy Harvesting Technology

General Contents

General contents

Subject index

A

B

C

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 colored in **GREEN**;
- alarms are colored in **RED**;
- states are colored 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 information the operator needs to correctly and safely install and operate the **EtherNet/IP series encoders**.

For technical specifications please refer to the encoder datasheet.

To make it easier to read the text, this guide can be divided into some main sections.

In the first section (from chapter 1 to chapter 3) general information concerning the safety, the mechanical installation and the electrical connection.

In the second section (chapter 4) information on how to install and configure the encoder under the Studio 5000 development environment as well as tips for setting up and running properly and efficiently the unit are provided.

In the third section (chapter 5) both general and specific information is given on the EtherNet/IP interface. In this section the interface features and the parameters implemented in the unit are fully described.

In the fourth section (chapter 6) the Integrated Web Server is described.

Glossary of EtherNet/IP terms

EtherNet/IP, 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 Ethernet/IP interface. They are listed in alphabetical order.

Adapter	Devices such as drives, controllers, and computers usually require an adapter to provide a communication interface between them and a network such as EtherNet/IP. An adapter reads data on the network and transmits it to the connected device. It also reads data in the device and transmits it to the network.
Adapter Class Device	An Adapter Class product emulates functions provided by traditional rack-adapter products. This type of node exchanges real-time I/O data with a Scanner Class product. It does not initiate connections on its own (see I/O Adapter).
Application I/O Trigger	The Application Trigger is one of three types of I/O triggers supported by CIP for the exchange of data on I/O connections. It is very similar to the CoS trigger and not common.
Application Objects	A reference to multiple Object Classes that implement product-specific features.
Attribute	Attributes are characteristics of an Object and/or an Object Class. They provide a description of an externally visible characteristic or feature of an object. Typically, Attributes provide status information or govern the operation of an Object. For example: the ASCII name of an object; and the repetition rate of a cyclic object. The Attribute part of an object specification is divided into two sections: • Class attributes; • Instance attributes.
Behavior	The relationship between attribute values and services, i.e. a specification of how an object acts. Actions results from different events the object detects, such as receiving service request, detecting internal faults or elapsing timers. The Behavior of an Object indicates how it responds to particular events. For example, a person can be abstractly viewed as an Instance within the Class Human. Generally speaking, all humans have the same set of attributes: age, gender, etc., yet, because the values of each attribute vary, each of us looks/behaves in a distinct fashion.
BOOTP (Bootstrap Protocol)	BOOTP lets the device configure itself dynamically at boot time if the network has a BOOTP server. The BOOTP server assigns the device a pre-configured IP address, a subnet mask, and a gateway address; therefore, you do not have to

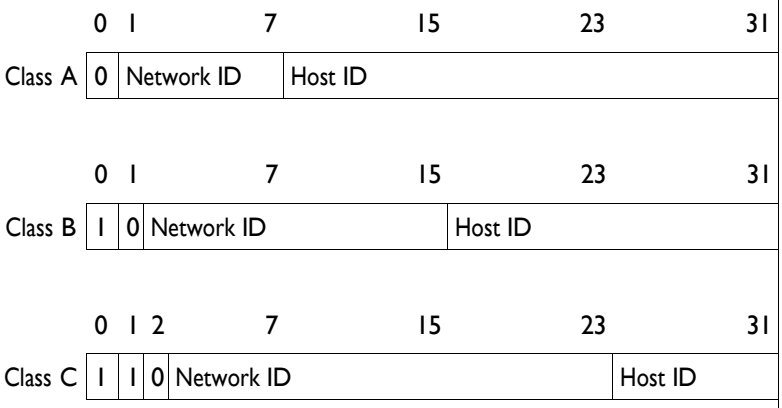
	configure these using the parameters in the device. BOOTP can make it easier to administer an EtherNet/IP network.
Bridge	A bridge refers to a network device that can route messages from one Ethernet network to another.
Broadcast	A broadcast transmission is a packet that all nodes on the network receive.
Change of State I/O Trigger	Change of State (CoS) is one of three types of I/O triggers supported by CIP for the exchange of data on Class 0 or 1 I/O connections. CoS endpoints send their messages when a change occurs. The data is also send at a background cyclic interval if no change occurs to keep the connection from timing out.
CIP (Common Industrial Protocol)	CIP is the transport and application layer protocol used for messaging over EtherNet/IP, ControlNet, and DeviceNet networks. The protocol is used for implicit messaging (real time I/O) and explicit messaging (configuration, data collection, and diagnostics).
Class	A class (of objects) is a set of objects that all represent the same kind of system component. A class is a generalization of an object. All objects in a class are identical in form and behavior, but may contain different attribute values. A class contains the objects that relate to a device, they are organized in instances. Ethernet/IP encoders from Lika supports the following classes: • Identity Object (Class Code 01h); • Message Router Object (Class Code 02h); • Assembly Object (Class Code 04h); • Connection Manager Object (Class Code 06h); • Position Sensor Object (Class Code 23h); • Time Sync Object (class Code 43h); • Device Level Ring (DLR) Object (Class Code 47h); • Quality of Service (QoS) Object (Class Code 48h) • TCP/IP Interface Object (Class Code F5h); • EtherNet Link Object (Class Code F6h); • LLDP Management Object (Class Code 109h); • Predefined Connection Object (Class Code 401h); • IO Mapping Object (Class Code 402h); • Diagnosis Object (Class Code 403h).
Class Attribute	A Class Attribute is an attribute whose scope is that of the class as a whole, rather than any one particular instance. Therefore, the list of Class Attributes is different than the list of Instance Attributes. CIP defines the Instance ID value zero (0) to designate the Class level versus a specific Instance within the Class.
Class code	A hexadecimal identifier assigned to each CIP object.
Connected Messaging	A CIP connection is a relationship between two or more application objects on different nodes. The connection

	establishes a virtual circuit between end points for transfer of data. Node resources are reserved in advance of data transfer and are dedicated and always available. Connected messaging reduces data handling of messages in the node. Connected messages can be Implicit (I/O) or Explicit.
Connection Establishment/Close	Connections are established Connection Originators using the ForwardOpen service and closed by using the ForwardClose service. Connection clean-up takes place when either connection end point times out.
Connection Originator	The source node that makes a request to a Connection Target for a connection. It can initiate either an I/O connection or explicit message connection using the ForwardOpen service.
Connection Target	Destination for I/O or explicit message connection requests. Responds to a connection request with a ForwardOpen service response.
Client	Within a client/server model, the client is the device that sends a request to a server. The client expects a response from the server.
Communication Objects	A reference to the Object Classes that manage and provide the run-time exchange of implicit (I/O) and explicit messages.
Consumer	Within the producer/consumer model, the consumer is one of potentially several consuming devices that picks up a message placed on the network by a producing device.
Controller	A controller, also called programmable logic controller, is a solid-state control system that has a user-programmable memory for storage of instructions to implement specific functions such as I/O control, logic, timing, counting, report generation, communication, arithmetic, and data file manipulation. A controller consists of a central processor, input/output interface, and memory.
Cyclic I/O Trigger	Cyclic is one of three types of I/O triggers supported by CIP for the exchange of data on Class 0 or 1 I/O connections. Endpoints send their messages at pre-determined cyclic time intervals.
Data Rate	The data rate is the speed at which data is transferred on the EtherNet/IP network. You can set the device to a data rate of 10 Mbps Full-Duplex, 10 Mbps Half-Duplex, 100 Mbps Full-Duplex, or 100 Mbps Half-Duplex. If another device on the network sets or auto-negotiates the data rate, you can set the device to automatically detect the data rate.
DSI (Drive Serial Interface)	DSI stands for Drive Serial Interface, it is based on the ModBus RTU serial communication protocol.
DSI Peripheral	A device that provides an interface between DSI and a network or user.
DSI Product	A device that uses the DSI communications interface to communicate with one or more peripheral devices. For

	example, a motor drive is a DSI product.
Duplex	Duplex describes the mode of communication. Full-duplex communications let a device exchange data in both directions at the same time. Half-duplex communications let a device exchange data only in one direction at a time. The duplex used by the adapter depends on the type of duplex that other network devices, such as switches, support.
EDS (Electronic Data Sheet) Files	EDS files are simple text files that are used by network configuration tools for EtherNet/IP to describe products so that you can easily commission them on a network. EDS files describe a product device type, revision, and configurable parameters. EDS files can be downloaded from Lika web site.
EDS File	An Electronic Data Sheet (EDS) is an ASCII text file that describes the features of an EtherNet/IP device and is used by software tools for device and network connection configuration.
EEPROM	EEPROM is the permanent memory of a device. Devices such as the encoder store parameters and other information in EEPROM so that they are not lost when the device loses power. EEPROM is sometimes called "NVS (Non-Volatile Storage)".
Encapsulation Protocol	Defines the communication relationship between two nodes known as an Encapsulation Session. The Encapsulation Protocol uses TCP/UDP Port 44818 for several Encapsulation Commands and for CIP Explicit Messaging. An example encapsulation command is the List_Identity Command that performs a "network who". An Encapsulation Session must be established before any CIP communications can take place. Data format for the Encapsulation Protocol is Little-Endian.
EtherNet/IP Network	Ethernet/IP (Industrial Protocol) is an open producer-consumer communication network based on the Ethernet standard (IEEE 802.3), TCP/IP, UDP/IP, and CIP. Designed for industrial communications, both I/O and explicit messages can be transmitted over the network. Each device is assigned a unique IP address and transmits data on the network. The number of devices that an EtherNet/IP network can support depends on the class of IP address. For example, a network with a Class C IP address can have 254 nodes. General information about EtherNet/IP and the EtherNet/IP specification are maintained by the Open DeviceNet Vendor's Association (ODVA). ODVA is online at http://www.odva.org.
Exclusive Owner Connection	This is one of three types of Implicit (I/O) Connections. It is a Class 0 or 1 bidirectional connection to an Output connection point (typically an Assembly Object), where the data of this assembly can only be controlled by one Scanner. There may be a connection to an input assembly; this data is being sent to the scanner. If the input data length is zero, then this direction becomes a Heartbeat connection.

Explicit Message Client	An explicit message client initiates request/response oriented communications with other devices. Examples of explicit message clients are HMI devices, programming tools, or PC or Linux based applications that gather data from control devices.
Explicit Message Server	An explicit message server responds to request/response oriented communications initiated by explicit message clients. An example of an explicit message server is a bar code reader.
Explicit Messaging	Explicit Messages are used to transfer data that does not require continuous updates. They are typically used to configure, monitor, and diagnose a device over the network. Explicit Messages can be sent as a connected or unconnected message. CIP defines an Explicit Messaging protocol that states the meaning of the message. This messaging protocol is contained in the message data. Explicit Messaging provide the means by which typical request/response oriented functions are performed (e.g., module configuration). These messages are typically point-to-point. Message rates and latency requirements are typically not as demanding as I/O messaging.
ForwardOpen Service Request	The ForwardOpen Service Request is sent by the Connection Originator and received by the Connection Target to open and establish explicit and I/O connections. The ForwardOpen Service request and associated response contains all of the connection parameters, including transport class, production trigger, timing information, electronic key and connection IDs.
Gateway	A gateway is a device on a network that connects an individual network to a system of networks. When a node needs to communicate with a node on another network, a gateway transfers the data between the two networks.
Hardware Address	Each Ethernet device has a unique hardware address (sometimes called a MAC address) that is 48 bits. The address appears as six digits separated by colons (for example, xx:xx:xx:xx:xx:xx). Each digit has a value between 0 and 255 (0x00 and 0xFF). This address is assigned in the hardware and cannot be changed. It is required to identify the device if you are using a BOOTP utility.
I/O Adapter	An I/O Adapter receives implicit communications requests from an I/O Scanner then produces and consumes its I/O data, typically at the requested cyclic rate. An I/O Adapter can be a simple digital input device, or something more complex such as a modular pneumatic valve system.
I/O Client	Function that uses the I/O messaging services of another (I/O Server) device to perform a task. Initiates a request for an I/O message to the server module. The I/O Client is a Connection Originator of Implicit Message connections
I/O Data	I/O data, sometimes called "implicit messages" or "input/output," transmit time-critical data. The terms "input" and "output" are defined from the controller's point of view.

	Output is transmitted by the controller and consumed by the device. Input is transmitted by the device and consumed by the controller.
I/O Messaging	Used interchangeably with the term Implicit Messaging.
I/O Scanner	An I/O scanner initiates implicit connections with I/O adapter devices, i.e., it is an I/O Client. A scanner is typically the most complex type of EtherNet/IP device, as it must deal with issues such as configuration of which connections to make, and how to configure the adapter device. Scanners also typically support initiating explicit messages, i.e., it is also an Explicit Message Client. A programmable controller is an example of an I/O scanner (used interchangeably with Scanner Class).
I/O Server	Function that provides I/O messaging services to another (I/O Client) device. Responds to a request from the I/O Client for an I/O connection. An I/O Server is the target of the implicit message connection request.
Implicit Messaging	Implicit Messages are exchanged across I/O Connections with an associated Connection ID. The Connection ID defines the meaning of the data and establishes the regular/repeated transport rate and the transport class. No messaging protocol is contained within the message data as with Explicit Messaging. Implicit Messages can be point to point (unicast) or multicast and are used to transmit application specific I/O data. This term is used interchangeably with the term I/O Messaging. Implicit Messaging on EtherNet/IP uses UDP/IP frames on port 2222. They are typically Class 0 or 1 and of the type Exclusive Owner, Input Only and Listen Only.
Input Only Connection	This is one of three types of Implicit (I/O) Connections. It is a Class 0 or 1 Connection to an Input connection point (typically an assembly object). The scanner receives input data from the target device and produces a Heartbeat to the target device. There is no Output data.
Instance	An object instance is the actual representation of a particular object within a class, i.e. it is a specific and real (physical) occurrence of an object. For example: New Zealand is an instance of the object class Country. Each instance of a class has the same attributes, but also has its own particular set of attribute values. The terms Object, Instance, and Object Instance all refer to a specific Instance.
Instance Attribute	An Instance Attribute is an attribute whose value is unique to an object instance and whose definition is shared by all instances of an object. Each instance need only support the optional attributes that apply to it. If an instance does not support an optional attribute, the Attribute Not Supported (General Status code 0x14) error shall be returned for services targeting that attribute.
IP Address	A unique IP address identifies each node on an EtherNet/IP network. An IP address consists of 32 bits that are divided into

	four segments of one byte each. It appears as four decimal integers separated by periods (xxx.xxx.xxx.xxx). Each "xxx" can have a decimal value from 0 to 255. For example, an IP address could be 192.168.0.1. An IP address has two parts: a network ID and a host ID. The class of network determines the format of the address.  The number of devices on your EtherNet/IP network will vary depending on the number of bytes that are used for the network address. In many cases you are given a network with a Class C address, in which the first three bytes contain the network address (subnet mask = 255.255.255.0). This leaves 8 bits or 256 addresses on your network. Because two addresses are reserved for special uses (0 is an address for the network usually used by the router, and 255 is an address for broadcast messages to all network devices), you have 254 addresses to use on a Class C address block. You must ensure that each device on the Internet has a unique address. You can then set the unique IP address for the device by using a BOOTP server or by manually configuring parameters in the device. The device reads the values of these parameters only at power-up.
Listen Only Connection	This is one of three types of Implicit Connections. It is a Class 0 or 1 Connection to an Input connection point (typically an assembly object). The scanner receives input data from the target device and produces a Heartbeat to the target device. There is no Output data. A Listen Only Connection can only be attached to an existing Exclusive Owner or Input Only Connection. If this underlying connection closes, then the Listen Only connection will also be closed or timed out.
Master	EtherNet/IP does not use Master/Slave technology or terminology.
Message Client	Function that uses the Explicit messaging services of another (Message Server) device to perform a task. Initiates an Explicit Message request to the server device.
Message Server	Function that provides Explicit Messaging services to another (Message Client) device. Responds to an Explicit Message request from the Message Client.
Multicast	Multicast is the single transmission of an I/O data packet that may be consumed by multiple devices using multicast IP and

	Ethernet destination addresses. See Producer/Consumer Communications Model.
Object	A CIP node is modeled as a collection of Objects. An Object provides an abstract representation of a particular component within a product. The realization of this abstract object model within a product is implementation dependent. In other words, a product internally maps this object model in a fashion specific to its implementation.
Ping	A ping is a message that is sent by a DSI product to its peripheral devices. They use the ping to gather data about the product, including whether it can receive messages and whether they can log in for control.
Point to Point (Unicast)	Point to Point or Unicast is the transmission of data to a single device.
Producer	Within the producer/consumer model, the producing device places a message on the network for consumption by one or several consumers. Generally, the produced message is not directed to a specific consumer.
Producer/Consumer Communications Model	For I/O Connections, CIP supports object-oriented Producer/Consumer communication. Connection identifiers embedded into each message are used by devices to determine which messages they should "consume" from other devices that "produce" messages. This enables efficient use of network bandwidth by transmitting information only once. Less bandwidth equates to greater efficiency and overall speed. EtherNet/IP uses IP multicast and Ethernet multicast destination addressing to implement this capability.
Requested Packet Interval (RPI)	EtherNet/IP devices typically produce or consume data based upon a Requested Packet Interval (RPI) value. Producer devices send data packets at a predetermined time interval based on the RPI, whereas consumer devices will listen for a packet of data at a given RPI.
Scanner Class	A Scanner Class product exchanges real-time I/O data with Adapter Class and Scanner Class products. This type of node can respond to connection requests and can also initiate connections to target devices (see I/O Scanner).
Server	Within a client/server model, the server is the device that receives a request from a client. The server is expected to give a response to the client.
Service (common service)	A list of the common services defined for the object. A function supported by an object and/or object class.
Service (object-specific service)	The full specifications of any services unique to the object.
Service code	Service codes are used to define the action that is requested to take place when an object or parts of an object are addressed through explicit messages. They are used to access

	classes or the attributes of a class or to generate specific events.
Slave	EtherNet/IP does not use Master/Slave technology or terminology.
Subnet Mask	A subnet mask is an extension to the IP addressing scheme that lets you use a single network ID for multiple physical networks. A bit mask identifies the part of the address that specifies the network and the part of the address that specifies the unique node on the network. A "1" in the subnet mask indicates the bit is used to specify the network. A "0" in the subnet mask indicates that the bit is used to specify the node. For example, a subnet mask on a Class C address may appear as follows: 11111111 11111111 11111111 11000000 (255.255.255.192). This mask indicates that 26 bits are used to identify the network and 6 bits are used to identify devices on each network. Instead of a single physical Class C network with 254 devices, this subnet mask divides it into four networks with up to 62 devices each.
Switches	Switches are network devices that provide virtual connections that help to control collisions and reduce traffic on the network. They are able to reduce network congestion by transmitting packets to an individual port only if they are destined for the connected device. In a control application, in which real time data access is critical, network switches may be required in place of hubs.
TCP (Transmission Control Protocol)	EtherNet/IP uses this protocol to transfer Explicit Messaging packets using IP. TCP guarantees delivery of data through the use of retries.
Transport Classes	CIP defines several Transport Classes for messaging connections. Within EtherNet/IP, I/O data sent on Class 1 connections is pre-pended with a 16-bit sequence count, while data on Class 0 connections is not. Class 3 connections are used for Explicit Messaging Connections.
UDP (User Datagram Protocol)	EtherNet/IP uses this protocol to transfer I/O packets using IP. UDP provides a simple, but fast capability to send I/O messaging packets between devices. This protocol ensures that devices transmit the most recent data because it does not use acknowledgments or retries.
Unconnected Messaging	Provides a means for a node to send message requests without establishing a CIP connection prior to data transfer. More overhead is contained within each message and the message is not guaranteed destination node resources. Unconnected Messaging is used for non-periodic requests (e.g., network "Who" function). Applies to explicit messages only.
Unicast (Point to Point)	Unicast or Point to Point is a connection for the transmission of data to a single device.

List of abbreviations

Table below contains a list of abbreviations (in alphabetical order) which may be used in this guide to describe the EtherNet/IP interface.

API	Actual Packet Interval
ASCII	American Standard Code for Information Interchange
ASN.1	Abstract Syntax Notation
CIP	The Common Industrial Protocol defined in this volume of the CIP Networks Library. CIP includes both connected and unconnected messaging.
CID	Connection Identifier
DLL	Data Link Layer
EPR	Expected Packet Rate
ISO	International Standards Organization
MAC ID	Media Access Control Identifier
PDU	Protocol Data Unit
ODVA	ODVA, Inc.
O ➡ T	Originator to Target (used to describe packets that are sent from the originator to the target)
OSI	Open Systems Interconnection (see ISO 7498)
RPI	Requested Packet Interval
SDU	Service Data Unit
SEM	State Event Matrix
SEMI	Semiconductor Equipment Materials International
STD	State Transition Diagram, used to describe object behavior
T ➡ O	Target to Originator (used to describe packets that are sent from the target to the originator)
UCMM	Unconnected Message Manager

References

[1] THE CIP NETWORKS LIBRARY, Volume 1, Common Industrial Protocol (CIP™), Edition 3.34, April 2023

[2] THE CIP NETWORKS LIBRARY, Volume 2, EtherNet/IP Adaptation of CIP, Edition 1.32, April 2023

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 the explanation in the "3 -Electrical connections" section on page 36;
- connect +Vdc and 0Vdc and check the power supply is correct first before connecting the communication ports;
- in compliance with the 2014/30/EU norm on electromagnetic compatibility, the following precautions must be taken:

  - before handling and installing, 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 housing, use one TCEI UNI M3 x 6 cylindrical head screw with two tooth lock washers.

1.3 Mechanical safety

- Install the device following strictly the information in the "2 -Mounting instructions" section on page 32;
- mechanical installation has to be carried out with stationary mechanical parts;
- do not disassemble the encoder;
- do not tool the encoder or its shaft;
- delicate electronic equipment: handle with care; do not subject the device and the shaft to knocks or shocks;
- respect the environmental characteristics declared by manufacturer
- unit with solid shaft: in order to guarantee maximum reliability over time of mechanical parts, we recommend a flexible coupling to be installed to connect the encoder and user's shaft; make sure the misalignment tolerances of the flexible coupling are respected;
- unit with hollow shaft: the encoder can be mounted directly on a shaft whose diameter has to respect the technical characteristics specified in the purchase order and clamped by means of the collar and, when requested, the anti-rotation pin.

2 Mounting instructions

WARNING
Installation and maintenance operations have to be carried out by qualified personnel only, with power supply disconnected. Shaft and mechanical components must be in stop.

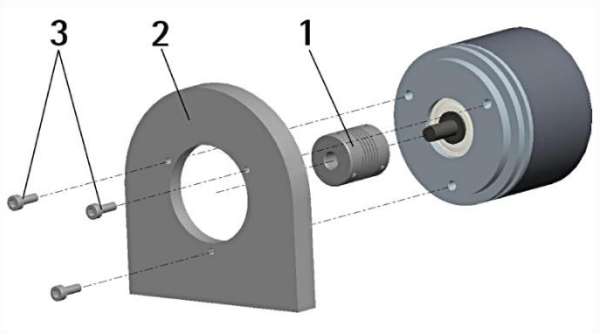
For any information on the mechanical data and the electrical characteristics of the encoder please refer to the encoder datasheet.

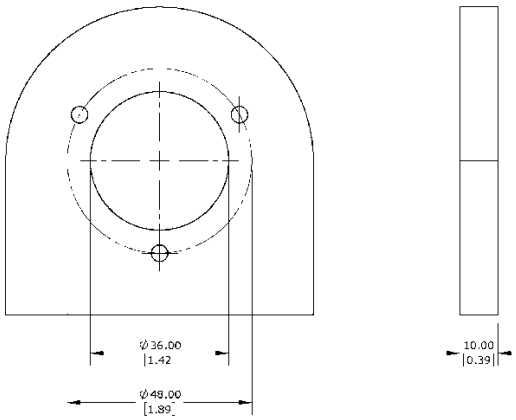
Values are expressed in millimeters (mm).

2.1 Solid shaft encoders

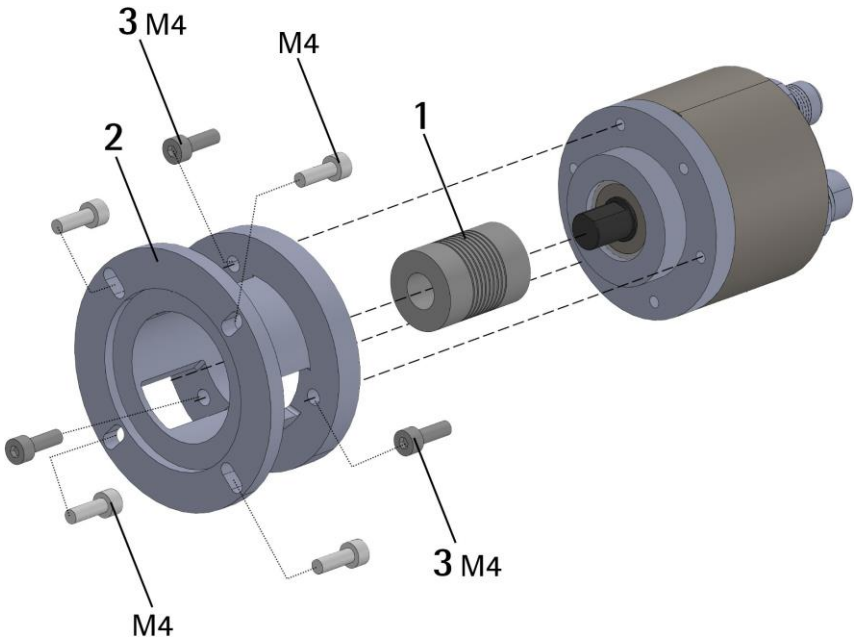
- Mount the flexible coupling **1** on the encoder shaft;
- fix the encoder to the flange **2** (or to the mounting bell) by means of the M4 screws **3**;
- mount the flexible coupling **1** on the motor shaft;
- secure the flange **2** to the support (or the mounting bell to the motor);
- make sure the misalignment tolerances of the flexible coupling **1** are met.

2.1.1 Solid shaft standard installation





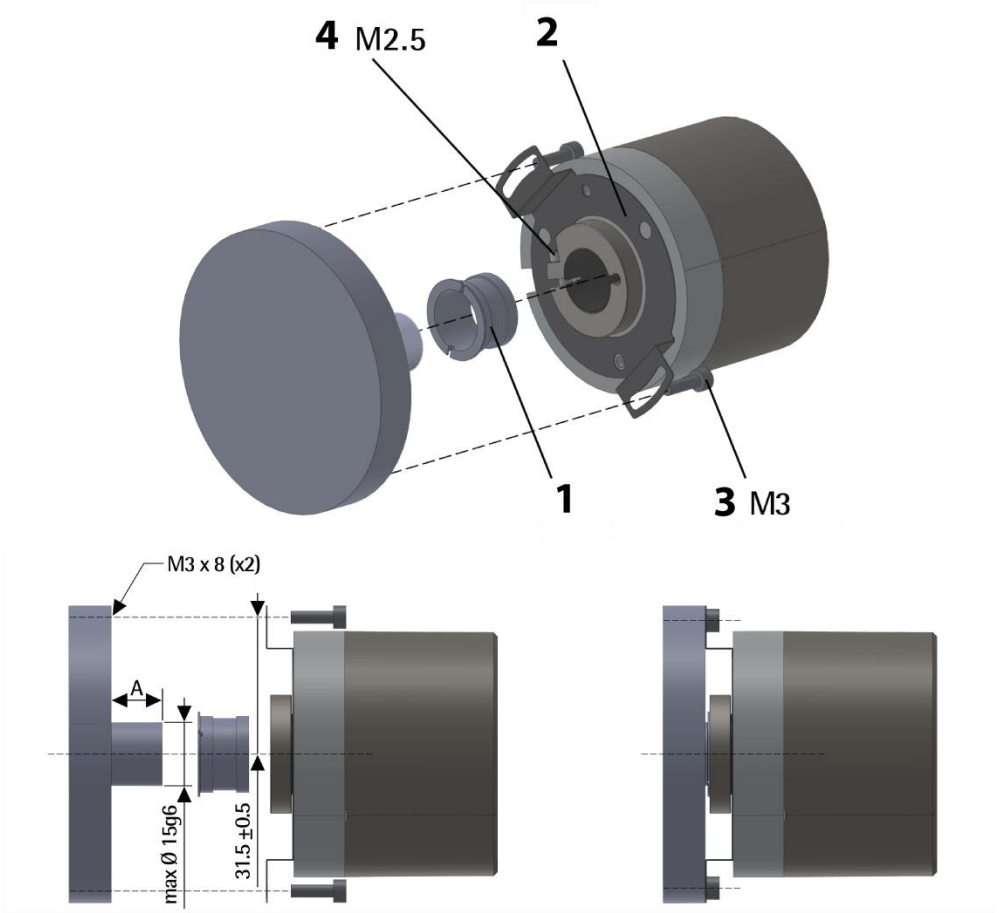
2.1.2 Solid shaft installation using PF4256 mounting bell



NOTE
In order to guarantee reliability over time of the encoder mechanical parts, we recommend a flexible coupling to be installed between the encoder and the motor shaft. Make sure the misalignment tolerances of the flexible coupling are met.

2.2 Hollow shaft encoders

- Mount the encoder on the motor shaft using the BR1-xx reducing sleeve **1** (if supplied). Avoid forcing the encoder shaft;
- fasten the fixing plate **2** to the rear of the motor using two M3 cylindrical head screws **3**;
- fix the collar to the encoder shaft (apply threadlocker to M2.5 screw **4**).



A = min. 8 mm, max. 18 mm

NOTE
You are strongly advised not to carry out any mechanical operations (drilling, milling, etc.) on the encoder shaft. This could cause serious damages to the internal parts and an immediate warranty loss. Please contact our technical personnel for the complete availability of "custom made" shafts.

3 Electrical connections

WARNING
Power supply must be turned off before performing any electrical connection! Installation, electrical connection, and maintenance operations must be carried out by qualified personnel only, with power supply disconnected. Mechanical components must be in stop.
Do not remove the plug **A** on the rear of the encoder unless otherwise indicated. Damage may be caused to internal components.

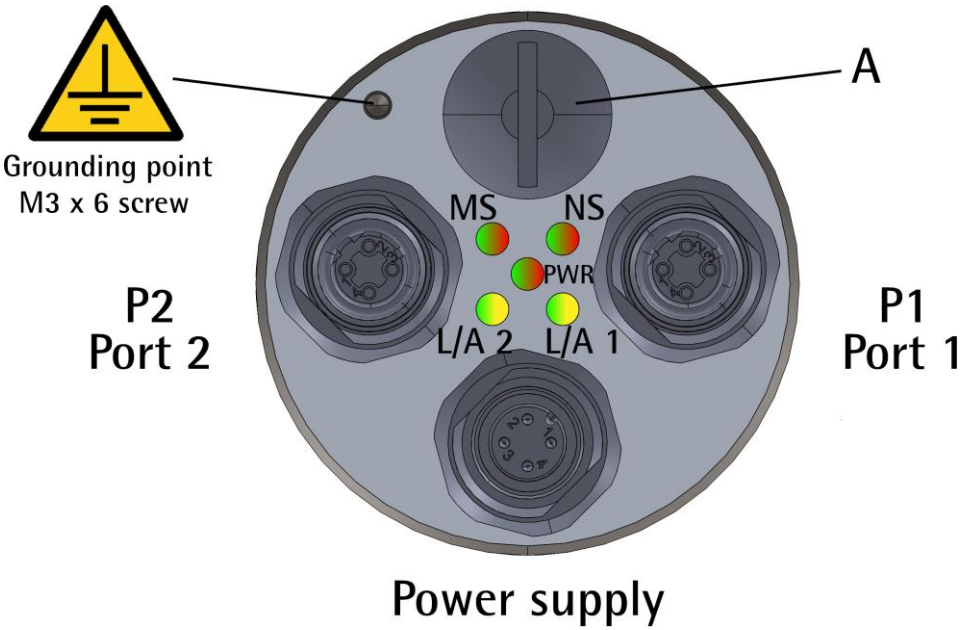
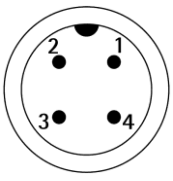


Figure 1 - Connectors and diagnostic LEDs

3.1 PWR Power supply connector (Figure 1)

M12 4-pin male connector with A coding is used for power supply.



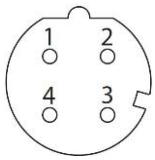
Description	Pin
+5Vdc +30Vdc	1
n.c.	2
0Vdc	3
n.c.	4

n.c. = not connected

WARNING
Connect +Vdc and 0Vdc and check the power supply is correct first before connecting the communication ports.

3.2 P1 Port 1 and P2 Port 2 connectors (Figure 1)

Two M12 4-pin female connectors with D coding are used for Ethernet connection through port 1 and port 2.



Description	Pin
Tx Data +	1
Rx Data +	2
Tx Data -	3
Rx Data -	4

The Ethernet interface supports 100 Mbit/s, half-duplex/full-duplex operation.
P1 PORT 1 and P2 PORT 2 M12 connectors have pin-out in compliance with the Ethernet standard. Therefore you can use standard Ethernet cables commercially available, for more information see later.
P1 PORT 1 and P2 PORT 2 connectors are interchangeable.

WARNING
Connect +Vdc and 0Vdc and check the power supply is correct first before connecting the communication ports.

3.3 Network configuration: cables, hubs, switches - Recommendations

Cables and connectors comply with the Ethernet specifications.
Standard Ethernet cables type CAT-5, CAT-5e and CAT-6 commercially available can be used.
The minimum cabling performance that will support EtherNet/IP is Category 5 as defined by ANSI/TIA/EIA-568-B.2 Annex N. There are reasons to select one category of cabling over another. In general, the higher the category, the better the cabling performance. Another consideration is balance. Category 5e, 6 and the newest proposed category, known as augmented 6 or Category 6a, will support current applications such as 1Gb/s and 10 Gb/s. Generally speaking, the greater the cabling category, the less EMC protection that is needed. Consult your cable supplier for guidance on EMC protection for the specific cable being used.
For complete information please refer to IEC 61918, IEC 61784-5-13 and IEC 61076-2-101.

The maximum cable length (100 meters) predefined by Ethernet 100Base-TX must be compulsorily fulfilled.

Regarding wiring and EMC measures, the IEC 61918 and IEC 61784-5-13 must be considered.
Compliance with IEEE Ethernet standards provides users with a choice of network interface speeds — e.g., 10, 100 Mbps, 1 Gbps and beyond — and a flexible network architecture compatible with commercially available Ethernet installation options including copper, fiber, fiber ring and wireless, and topologies including star, linear and ring.
A hub is an inexpensive connectivity method that provides an easy method of connecting devices on information networks (shared Ethernet). A switch reduces collisions and is recommended for real-time control installations (switched Ethernet). Routers are used to isolate control data traffic from other types of office data traffic, to isolate information traffic on the plant floor from control traffic on the plant floor, and for security purposes, i.e., firewalls. Repeaters extend the overall network cable length. They can also connect networks with different media types.

3.4 Line termination

EtherNet/IP 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.

3.5 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.

3.6 MAC address and IP address

The unit can be identified in the network through the **MAC address** and the **IP address**.
The MAC address has to be intended as a permanent and globally unique identifier assigned to the unit for communication on the physical layer; while the IP address is the name of the unit in a network using the Internet protocol.
MAC address is 6-byte long and cannot be modified. It consists of two parts, numbers are expressed in hexadecimal notation: the first three bytes are used to identify the manufacturer (OUI, namely Organizationally Unique Identifier), while the last three bytes are the specific identifier of the unit. The MAC address can be found on the label applied to the encoder.
The IP address (and the subnet mask) must be assigned by the user to each interface of the unit to be connected in the network.
For additional information on the MAC address refer to the "4.4 MAC address" section on page 53.
For additional information on the IP address refer to the "3.8 EtherNet/IP Node ID" section below.

3.7 EtherNet/IP Node ID

By default, the encoder is configured so that it uses the IP address, Subnet mask, and Gateway address that are saved internally. The use of a DHCP Server to allocate the IP address is disabled.
The IP address, the Subnet mask, and the Gateway address are set next to the **IP Address**, **Network Mask**, and **Gateway Address** parameters in the **F5-01-05 Interface Configuration** attribute, see the "5.12.9 Class F5h: TCP/IP Interface Object" section on page 140. For more information on setting the node ID *via software* refer to the "3.8.1 Setting the node ID via software" section hereafter.

The following table summarizes the default software IP parameters.

IP Parameter	IP address
IP address	192.168.1.10
Subnet mask	255.255.255.0
Gateway address	0.0.0.0
DHCP	Disabled

As an alternative, the node address can be set *via hardware* by using the DIP switch located inside the enclosure. For more information on setting the node ID via hardware refer to the "3.8.2 Setting the node ID via hardware (DIP A DIP switch)" section below.

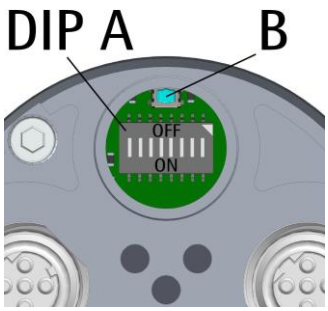
3.7.1 Setting the node ID via software

As stated, by default, the encoder is configured so that it uses the IP address saved internally. The sliding levers in the DIP A DIP switch located inside the enclosure are all set to OFF (value 0_{10}, 00000000_2) so meaning that the software values saved internally are used, see the next section.
The software values can be changed by using a software tool such as Studio 5000 or by means of the integrated web server (see the "6.7 Network configuration" section on page 182) or by enabling a DHCP server (see the "3.8.2 Setting the node ID via hardware (DIP A DIP switch)" section hereafter).
Any Net ID value and Host ID value can be set via software.

3.7.2 Setting the node ID via hardware (DIP A DIP switch)

WARNING
Power supply must be turned off before setting the DIP switch!
Be careful not to press the **B** tactile switch.

The EtherNet/IP node ID can be set *via hardware* using the DIP A DIP switch located inside the enclosure. To access the DIP A DIP switch remove the **A** plug (see Figure 1). Be careful with the internal electronics. Always replace the **A** plug at the end of the operation.



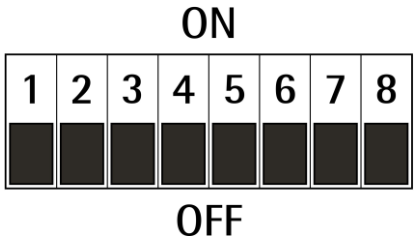
The DIP A DIP switch allows to set the Host ID; the Net ID is fixed, as defined in the following table:

192.168.1.	EtherNet/IP Node
Net ID	Host ID

Allowed node addresses range between 1_{10} (00000001_2) and 254_{10} (11111110_2).
The subnet mask is 255.255.255.0.

Value 0_{10} (00000000_2) means that the system uses the software IP address, Subnet mask, and Gateway address that are saved internally (default value, see the "3.8.1 Setting the node ID via software" section on page 40).
Value 255_{10} (11111111_2) enables the use of a DHCP Server. The IP address and the Subnet mask are assigned by a DHCP Server.
The DIP switches are evaluated only during switching the operating voltage on or when resetting the encoder.
Changes in the position of the switches when the encoder is switched on are taken into consideration only after switching the encoder off and then on again.

DIP A:

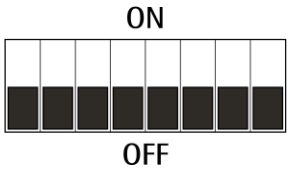


Set the EtherNet/IP node ID in binary value: ON = 1, OFF = 0

bit	1 LSB	2	3	4	5	6	7	8 MSB
	2^0	2^1	2^2	2^3	2^4	2^5	2^6	2^7

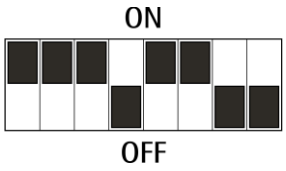
EXAMPLE
Enabling the software node ID = 0:
0_{10} = 0000 0000$_2$ (binary value)

bit	1	2	3	4	5	6	7	8
	2^0	2^1	2^2	2^3	2^4	2^5	2^6	2^7
	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF



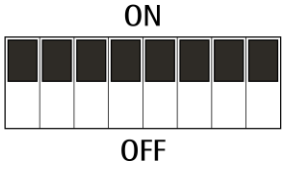
Setting the node ID = 55:
55_{10} = 0011 0111$_2$ (binary value)

bit	1	2	3	4	5	6	7	8
	2^0	2^1	2^2	2^3	2^4	2^5	2^6	2^7
	ON	ON	ON	OFF	ON	ON	OFF	OFF



Enabling the DHCP server = 255:
255_{10} = 1111 1111$_2$ (binary value)

bit	1	2	3	4	5	6	7	8
	2^0	2^1	2^2	2^3	2^4	2^5	2^6	2^7
	ON	ON	ON	ON	ON	ON	ON	ON



3.8 Diagnostic LEDs (Figure 1)

Five LEDs located in the rear of the encoder (see Figure 1) are meant to show visually the operating or fault status of the encoder and the EtherNet/IP interface. The meaning of each LED is explained in the following tables.

MS Module Status LED (green / red)

It shows the state of the EtherNet/IP device.

MS LED	Description
OFF	The power supply is switched OFF.
ON green	The device is operational, it is operating correctly.
FLASHING (1 Hz) green	The device is in standby, it has not been configured.
FLASHING FAST green / red / green	Self-test. The device performs a self-test after power on. The following sequence is displayed during the self-test: • The NS LED is OFF. • The MS LED turns ON green for approximately 250 ms, then it turns ON red for approximately 250 ms, and again it turns ON green for approximately 250 ms (finally it holds the state until the power-up test has been completed). • The NS LED turns ON green for approximately 250 ms, then it turns ON red for approximately 250 ms, and again it turns OFF (finally it holds the state until the power-up test has been completed).
FLASHING SEQUENCE red / green / OFF	The flashing sequence is used to visually identify the device. The scanner can start the flashing sequence in Identity object 1 of the device. The MS LED and the NS LED perform the flashing sequence simultaneously.
FLASHING (1 Hz) red	Major recoverable fault. The device has detected a major recoverable fault, e.g. an incorrect or inconsistent configuration can be considered a major recoverable fault. See the 01-01-05 Status attribute on page 92.
ON red	Major unrecoverable fault. The device has detected a major unrecoverable fault (FATAL error, etc.). See the 01-01-05 Status attribute on page 92.

NS Network Status LED (green / red)

It shows the current state of the network.

LED	Description
OFF	• The device is switched OFF. • No IP address has been set, the device does not have an IP address.
ON green	The device is connected, an IP address is configured, at least one CIP connection (any transport class) is established, and an Exclusive Owner connection has not timed out.
FLASHING (1 Hz) green	There is no connection, an IP address is configured, but no CIP connection has been established, and an Exclusive Owner connection has not timed out.
FLASHING FAST green / red / OFF	Self-test. The device performs a self-test after power on. The following sequence is displayed during the self-test: • The NS LED is OFF. • The MS LED turns ON green for approximately 250 ms, then it turns ON red for approximately 250 ms, and again it turns ON green for approximately 250 ms (finally it holds the state until the power-up test has been completed). • The NS LED turns ON green for approximately 250 ms, then it turns ON red for approximately 250 ms, and again it turns OFF (finally it holds the state until the power-up test has been completed).
FLASHING SEQUENCE red / green / OFF	The flashing sequence is used to visually identify the device. The scanner can start the flashing sequence in Identity object 1 of the device. The MS LED and the NS LED perform the flashing sequence simultaneously.
FLASHING (1 Hz) red	Connection timeout. An IP address is configured and an Exclusive Owner connection for which the device is the target has timed out. The NS LED returns to the solid green state only when all timed out Exclusive Owner connections are established again.
ON red	Duplicate IP address conflict has occurred, two devices on the network have been assigned the same IP address.

PWR Power LED (green / red)

It shows the power supply and system state. It is also referred to as SYS (System) LED.

PWR LED	Description	Meaning
OFF	Power OFF	The encoder power supply is switched OFF. No supply voltage for the device or hardware fault.
ON green	Power ON	The encoder power supply is switched ON. The firmware is running.
BLINKING red	No firmware program installed, firmware update mode	At power ON the LED blinks red at 1 Hz. The firmware program is not installed, the encoder enters the firmware update mode and waits for the firmware file to be installed.

L/A Link/Activity LED for port 2 P2 (green / yellow)

It shows the state and the activity of the physical link (port 2 P2).

L/A LED	Description	Meaning
OFF	No link No activity	The device has no link to the Ethernet, the link through port 2 P2 is not active. There is no activity on port 2 P2, the device does not send/receive Ethernet frames through port 2 P2.
ON green	Link active No activity	Port 2 P2 link active, the device is linked to the Ethernet, there is no activity on port 2 P2.
FLICKERING yellow	Activity	Port 2 P2 link is active, there is activity on port 2 P2, the device sends/receives Ethernet frames through port 2 P2.

L/A Link/Activity LED for port 1 P1 (green / yellow)

It shows the state and the activity of the physical link (port 1 P1).

LED	Description	Meaning
OFF	No link No activity	The device has no link to the Ethernet, the link through port 1 P1 is not active. There is no activity on port 1 P1, the device does not send/receive Ethernet frames through port 1 P1.
ON green	Link active No activity	Port 1 P1 link active, the device is linked to the Ethernet, there is no activity on port 1 P1.
FLICKERING yellow	Activity	Port 1 P1 link is active, there is activity on port 1 P1, the device sends/receives Ethernet frames through port 1 P1.

After power on, the MS module status indicator and the NS network status indicator shall perform a self-test sequence. For more information see the description of each LED operation.

3.9 LED state definition

LED state	Description
Flashing, 1 Hz	The LED turns ON and OFF with a frequency of 1 Hz: "ON" for 500 ms, followed by "OFF" for 500 ms.
Flashing fast green / red / green	The MS LED or NS LED turns ON green "ON" for 250 ms, then red "ON" for 250 ms, then green "ON" for 250 ms (until the test is completed).
Flashing sequence red / green / off	The MS LED and NS LED turn ON red "ON" for 500 ms, then green "ON" for 500 ms, then "OFF" for 500 ms. The flashing sequence is repeated at least 6 times.
Flickering (load dependent)	The LED turns ON and OFF with a frequency of approximately 10 Hz to indicate high Ethernet activity: "ON" for approximately 50 ms, followed by "OFF" for 50 ms. The LED turns ON and OFF in irregular intervals to indicate low Ethernet activity.

3.10 Tactile switch (Figure 2)

WARNING
Be careful not to press the **B** tactile switch unless specifically requested.

The **B** tactile switch is located inside the enclosure. You must remove the **A** plug (see Figure 1) to access it. It has no useful function to the operator under normal usage conditions, so never press it unless specifically requested by Lika Electronic's technicians.

4 Quick reference

4.1 Quick setting and main functions

The following instructions allow the operator to quickly and safely set up the encoder in a standard operational mode and to execute its main functions.
Sometimes a function or a procedure can be accomplished by using alternative ways:

- by means of a software tool such as Studio 5000 from Rockwell Automation (see the "4.5 Encoder installation under Studio 5000 design environment" section on page 54 ff);
- by means of the Integrated Web Server (see the "Integrated Web Server" section on page 171);
- or via hardware by means of the internal DIP switches (see the "3.8.2 Setting the node ID via hardware (DIP A DIP switch)" section on page 40).

They are all mentioned whenever available.
For complete and detailed information please read the mentioned pages thoroughly.

- Mechanically install the device, see on page 32 ff;
- execute the electrical connection and switch on the +5Vdc +30Vdc power supply, see on page 36 ff; check the soundness of the connection;
- switch off the power supply and execute the network connection, then switch on the power supply again, see on page 36 ff; check the soundness of the connection;
- in the software tool install the EDS file corresponding to the encoder to be installed, see on page 61 ff; please note that specific EDS files are provided to each encoder model. They are:
  - **Lika AM58X-1314-EP_V1_1.eds**: it is intended for installation of **AM58x-1314 EtherNet/IP encoders.**
- in the software tool insert the Lika module and select the encoder type, see on page 64 ff;
- in the software tool set the device name, see on page 64 ff;

- if required, set the IP address and the subnet mask to the node, see here later for alternative setting modes; the default address (software address) set by Lika is **192.168.1.10**;
- the attributes used to specifically configure the encoder are grouped in the Position Sensor Object, see the "5.12.5 Class 23h: Position Sensor Object" section on page 105; they allow, for example, to set the singleturn resolution or the total resolution, to enable the scaling function or to change the counting direction; the complete list of the default parameters is available on page 184.

4.1.1 Setting the node address

The node address and the network-related parameters can be set either via software or via hardware.
Software configuration:

- set the **IP Address**, **Network Mask**, and **Gateway Address** parameters in the **F5-01-05 Interface Configuration** attribute, see the "5.12.9 Class F5h: TCP/IP Interface Object" section on page 140; the sliding levers in the DIP A DIP switch are all set to OFF (value 0_{10}, 00000000_2), see the "3.8 EtherNet/IP Node ID" section on page 39);
- set the parameters in the Integrated Web Server, see the "6.7 Network configuration" section on page 182; the sliding levers in the DIP A DIP switch are all set to OFF (value 0_{10}, 00000000_2), see the "3.8 EtherNet/IP Node ID" section on page 39);
- enable a DHCP Server as follows (the sliding levers in the DIP A DIP switch are all set to OFF -value 0_{10}, 00000000_2-; or all set to ON -value 255_{10}, 11111111_2-):
  - see the **F5-01-03 Configuration Control** attribute, see the "5.12.9 Class F5h: TCP/IP Interface Object" section on page 140.

Hardware configuration:

- set the sliding levers in the DIP A DIP switch to value 0_{10} (00000000_2) to enable the software IP address, Subnet mask, and Gateway address that are saved internally, see software configuration above;
- set the sliding levers in the DIP A DIP switch to any value in the range between 1_{10} (00000001_2) and 254_{10} (11111110_2). The Subnet mask is 255.255.255.0;
- set the sliding levers in the DIP A DIP switch to value 255_{10} (11111111_2) to enable the use of a DHCP Server.

4.1.2 Setting scaling function and custom resolution

- If you want to use the physical resolution of the encoder, please check that the **23-01-0E Scaling Function Control** attribute is disabled (="0"), see on page 107; in this case, the device uses the physical resolution (see the **23-01-2A Physical Resolution Span** and **23-01-2B Number of Spans** attributes) to arrange the absolute position value.

You can also use the Integrated Web Server, see the "6.4 Setting the attributes" section on page 175; or a software tool, see the "4.5.11 Configuring the encoder" section on page 67;

- on the contrary, if you need a custom resolution, you must enable the scaling function by setting the **23-01-0E Scaling Function Control** attribute to ="1" first and then set the required resolution parameters:
  - set the singleturn resolution next to the **23-01-10 Measuring Units per Span** attribute, see on page 107;
  - set the total resolution next to the **23-01-11 Total Measuring Range 32 bit** parameter, see on page 109.

  You can also use the Integrated Web Server, see the "6.4 Setting the attributes" section on page 175; or a software tool, see the "4.5.11 Configuring the encoder" section on page 67.

4.1.3 Reading the absolute position

To read the position value you can choose among the following methods.

- To read the absolute position of the encoder see the **23-01-03 Position value 32 bit** attribute on page 106;
- open the Integrated Web Server, see the "6.3 Encoder position and speed" section on page 173; see the "6.5 Encoder information (EtherNet/IP attributes)" section on page 177;
- open the **Monitor Tags** tabbed page in your project, see the "4.5.9 Checking the communication" section on page 66.

4.1.4 Reading the velocity value

To read the velocity value you can choose among the following methods.

- To read the velocity value of the encoder see the **23-01-18 Velocity Value** attribute on page 112;
- open the Integrated Web Server, see the "6.3 Encoder position and speed" section on page 173; see the "6.5 Encoder information (EtherNet/IP attributes)" section on page 177;
- open the **Monitor Tags** tabbed page in your project, see the "4.5.9 Checking the communication" section on page 66.

4.1.5 Setting and executing the preset

To set and execute the preset you can choose among the following methods.

- Enter a suitable value next to the **23-01-13 Preset Value 32 bit** attribute, see on page 111; the preset value is activated as soon as the value is confirmed.
- If you need to activate in a different physical position of the encoder shaft the value that has been already set next to the **23-01-13 Preset Value 32 bit** attribute, you can use the bit 0 **Activate Preset** in the **23-01-68 Command Register** attribute, see on page 120.
- Open the **Set Encoder Attributes** page in the Integrated Web Server, see the "6.4 Setting the attributes" section on page 175.

- Use the Test_EXM1314_Lika.acd sample program, you can find it in the **Examples_EXM1314_EP.zip** compressed file. Refer also to the "4.5.12 How to create a sample program and send the parameters" section on page 67.

4.1.6 Saving data

To save the parameters permanently you can choose among the following methods.

- Use the Class Service 16h available for the Position Sensor Object, see on page 105.
- Set the bit 6 **Save Parameters** in the **23-01-68 Command Register** attribute to 1 and then back to 0, see on page 121.
- Use the **Save Param.** function in the **Set Encoder Attributes** page of the Integrated Web Server, see the "6.4 Setting the attributes" section on page 175.

4.1.7 Restoring defaults

To restore the default parameters you can choose among the following methods.

- Use the Class Service 15h available for the Position Sensor Object, see on page 105.
- Set the bit 7 **Restore Parameters to Defaults** in the **23-01-68 Command Register** attribute to 1 and then back to 0, see on page 121.
- Use the **Load Default** function in the **Set Encoder Attributes** page of the Integrated Web Server, see the "6.4 Setting the attributes" section on page 175.

4.2 About Lika encoders

Lika encoders are **22 hex type devices** and comply with the specifications reported in the Chapter 6 "Device Profiles, Encoder Device Type 22 hex " of the publication "THE CIP NETWORKS LIBRARY, Volume 1, Common Industrial Protocol (CIP™)".
The Object Model of an encoder device is represented in the following picture:

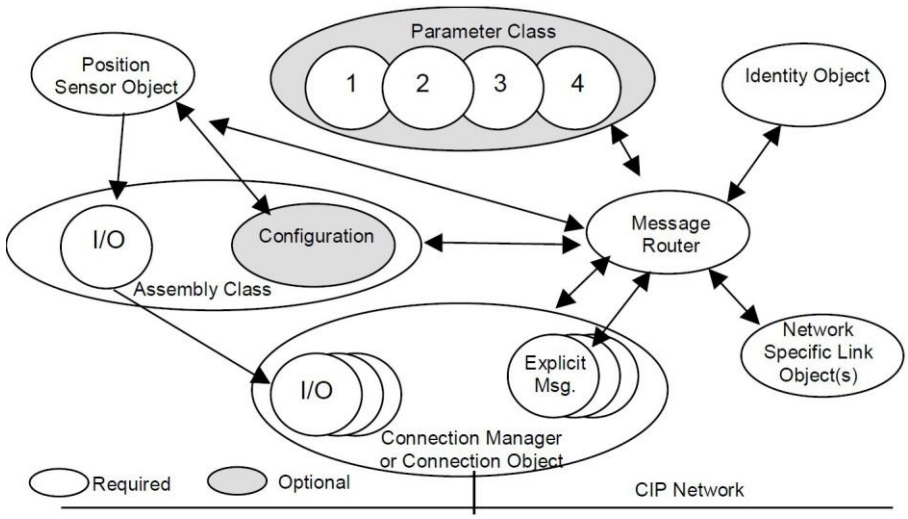


Figure 3 - Object model

The Parameter Object data mapping complies with information in the "Mapping Parameter Object Data" section.
The attributes that are used to specifically configure the encoder and make it operational in order to provide the absolute position value and the velocity value are all grouped in the Position Sensor Object, refer to the "5.12.5 Class 23h: Position Sensor Object" section on page 105.

4.2.1 Network identity

Lika EtherNet/IP encoders use the following identity settings available in the Identity Object, see the "5.12.1 Class 01h: Identity Object" section on page 90:

Identity Name: **Vendor ID**
Attribute: **01-01-01 Vendor ID**
Setting: **0299h = 665dec = Lika Electronic Srl**

Identity Name: **Device Type**
Attribute: **01-01-02 Device type**
Setting: **0022h: Encoder Device Profile**

Identity Name: **Product Code**
Attribute: **01-01-03 Product code**
Setting: **0082h AM58x-1314 27-bit multiturn encoder**

Identity Name: **Revision**
Attribute: **01-01-04 Revision**
Setting: **device dependent**

Identity Name: **Serial Number**
Attribute: **01-01-06 Serial number**
Setting: **device dependent**

Identity Name: **Product Name**
Attribute: **01-01-07 Product name**
Setting: **AM58X-1314 AM58x 27-bit multiturn encoder**

4.2.2 Network and communication settings

The **MAC address** of the device is always reported in the label applied to the encoder enclosure. See on page 39.
The **EtherNet/IP Node ID** can set both via software and via hardware using the DIP A DIP switch located inside the encoder enclosure. By default it is set via software and its value is 192.168.1.10. See on page 39.

4.3 Configuring the encoder with Studio 5000 V30.00 from Rockwell Automation

In this manual some screenshots are shown to explain how to install and configure the encoder in a supervisor. In the specific example the development environment is Studio 5000 V30.00 from Rockwell Automation; it is used in combination with CompactLogix 5370 L1 Controller "1769-L16ER-BB1B/B" series from Allen Bradley. Therefore, the information on the installation of the EDS file, the assignment of the IP address and the device name, the configuration of the encoder in the network, topology, diagnostics, etc. will always refer to the aforementioned design environment. If you need to install the encoder using a different configuration tool, please read and follow carefully the instructions given in the documentation provided by the manufacturer.
In the following pages the Controller is assumed to have 192.168.1.20 IP address and 255.255.255.0 Subnet mask.

Lika Electronic EtherNet/IP encoder documentation is complete with a **sample project** supplied free of charge. This program is designed to make your own project planning, programming, communication and diagnostics with Studio 5000 V30.00 design environment user-friendly and reliable. For instance it allows to execute the following functions: setting the Preset value and execute it; setting both the singleturn and the multiturn resolutions; setting an output (refer to page 67 ff). You can find it in the **Examples_EXM1314_EP.zip** compressed file.

4.4 MAC address

The MAC address is an identifier unique worldwide.
The MAC-ID consists of two parts: the first three bytes are the manufacturer ID and are provided by IEE standard authority; the last three bytes represent a consecutive number of the manufacturer.

NOTE
The MAC address is always printed on the encoder label for commissioning purposes.

The MAC address has the following structure:

Bit value 47 ... 24			**Bit value 23 ... 0**		
10	B9	FE	X	X	X
Company code (OUI)			**Consecutive number**		

The MAC address can also be read next to the **F6-01-03 Physical Address** attribute. Refer to the "5.12.10 Class F6h: Ethernet Link Object" section on page 150.
It is further shown in the **Encoder Information** page of the web server. Refer to the "6.5 Encoder information (EtherNet/IP attributes)" section on page 177.

4.5 Encoder installation under Studio 5000 design environment

4.5.1 Description of the EDS file

The functionality of an EtherNet/IP device is always described in an EDS file (Electronic Data Sheet file). The Electronic Data Sheet file provides information about the device basic communication and functional properties. It must be installed in the Controller.

EtherNet/IP encoders from Lika Electronic are supplied with their own EDS file. Specific EDS files are provided to each encoder model.
They are:

- **Lika AM58X-1315-EP_v1_1.eds**: it is intended for installation of **AM58x-1314 EtherNet/IP encoders.**

The version of the EDS file is reported under the Version item inside the file.
EDS files can be paired with the **EXO58_EXM58_48x48.ico** picture file available inside the file folder (the picture is also integrated into the EDS file).

Go to Support Resources on the encoder item page at AutomationDirect.com to get the correct EDS file.

4.5.2 Configuring the network interface controller (NIC) of the computer

To set the computer's IP address in Windows, type *network and sharing* into the **Search** box in the **Start** menu and select **Network and Sharing Center** when the **Control Panel** comes up.

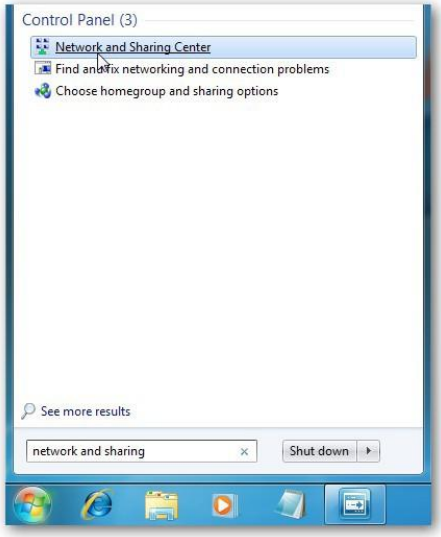


Figure 4 - Network and Sharing Center

Then when the **Network and Sharing Center** opens, click on **Change adapter settings**.

Figure 5 - Change adapter settings

Right-click on your local adapter and select **Properties.**

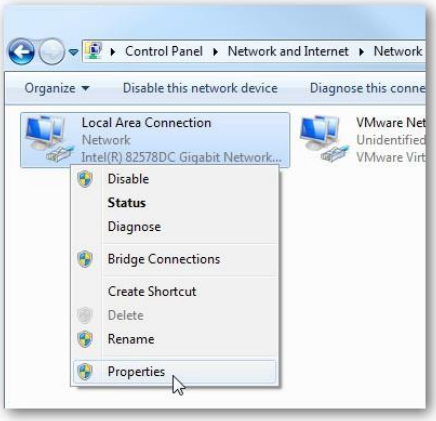


Figure 6 - Local Area Connection properties

In the **Local Area Connection Properties** window highlight *Internet Protocol Version 4 (TCP/IPv4)*, then click the **Properties** button.

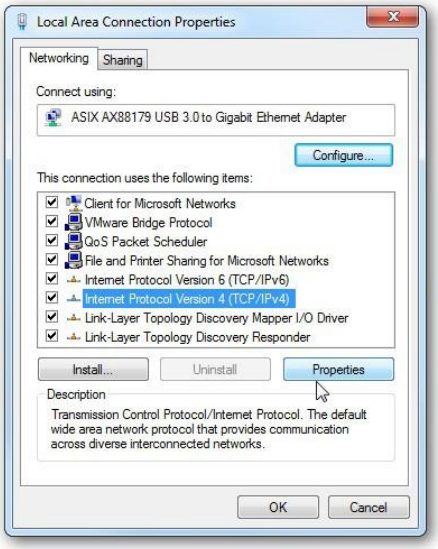


Figure 7 - Internet Protocol Version 4 properties

Now select the **Use the following IP address** radio button and enter in the correct IP, Subnet mask, and Default gateway that corresponds with your network setup. Then, if required, enter your Preferred and Alternate DNS Server addresses. We suggest setting a simple Class C network configuration such as 192.168.1.xx as the default software IP address of the encoder has this NET ID. Check **Validate settings upon exit** so Windows can find any problems with the addresses you entered. When you are finished click **OK**.

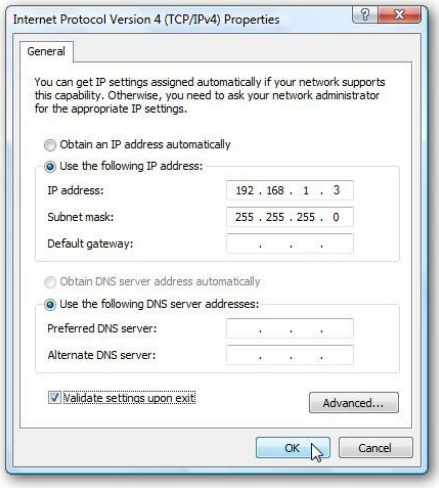


Figure 8 - Setting the IP Address

4.5.3 Networking the PC and the Controller

Use a Category 5 minimum cable to network the Ethernet port of the PC to the Ethernet port of the Controller.

4.5.4 Configuring the driver

Launch the **RSLinx Classic** communication software and then open **RSWho** by pressing **Communication** and then the **RSWho** command.
Again in the menu bar of the main page press **Communication** and then the **Configure Drivers** command.
The **Configure Drivers** dialog box will appear.

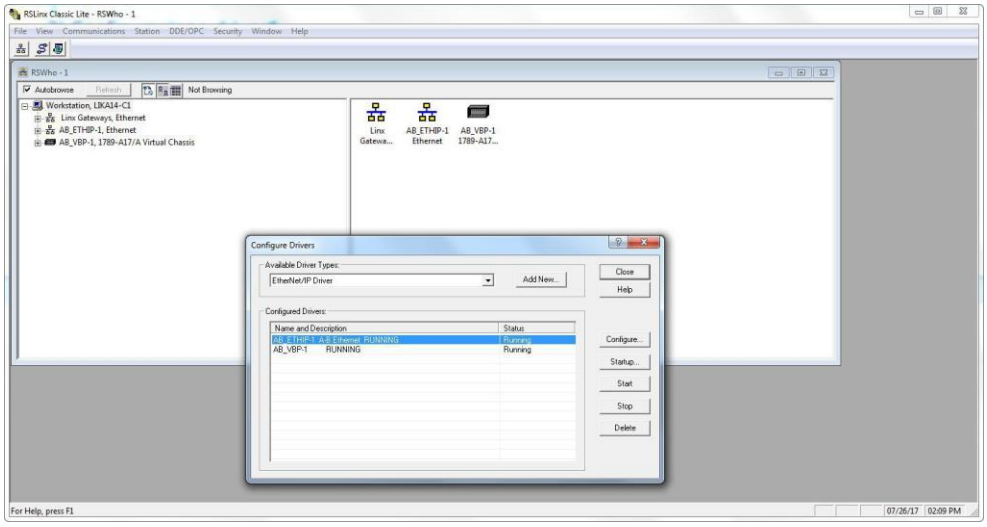


Figure 9 - Configure Drivers

From the **Configure Drivers** dialog box, select the desired driver from the **Available Driver Types** list.
Click **Add New**. The **Add New RSLinx Classic Driver** dialog box opens.
Enter a name for the selected driver (15 characters at maximum), and click **OK**.
The **Configuration** dialog box for that driver shows.
In the **Configuration** dialog box, enter the appropriate parameters for the desired driver.
Click **OK** to close the **Configuration** dialog box. The new driver now appears in the **Configured Drivers** list.
Press **Close** to close the dialog box.

Now right-click the driver you have just installed and press **Configure Driver**.

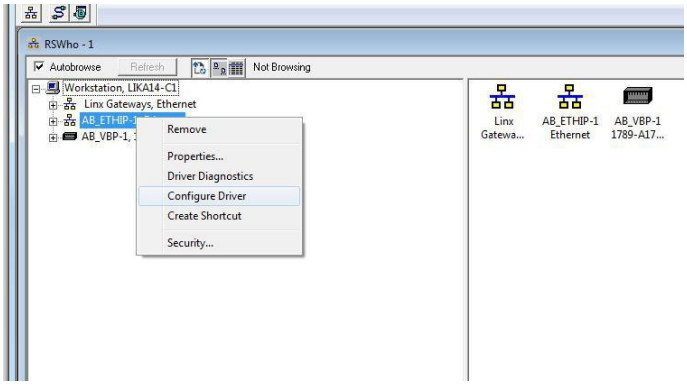


Figure 10 - Configure Driver

In the **Configure Driver** dialog box, select the network interface controller you configured and connected to the PLC; finally press **OK** to confirm.

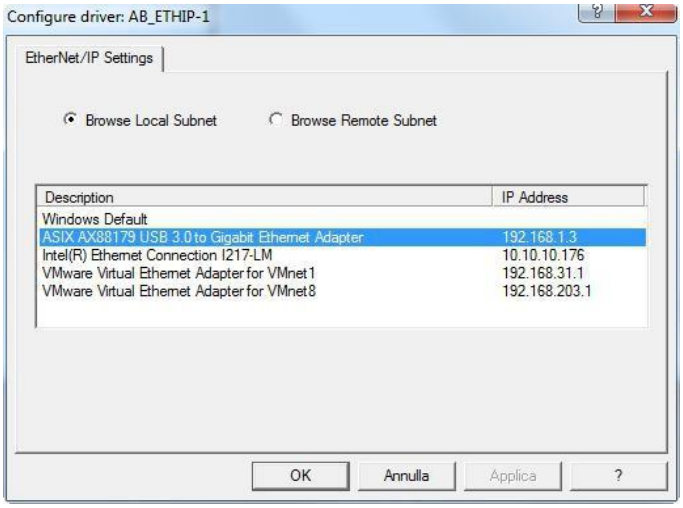


Figure 11 - Browse Local Subnet

4.5.5 Starting a new project

Double-click on the **Studio 5000** icon on your Desktop to launch Studio 5000 software. The Studio 5000 Splash Screen appears.
Select **New Project** under the **Create** section.

Figure 12 - Studio 5000 New Project

When the **New Project** pop-up is displayed, select **Logix** and the type of controller (such as "1769-L16ER-BB1B", in the example). Enter the name of the project and the path where the file has to be saved.

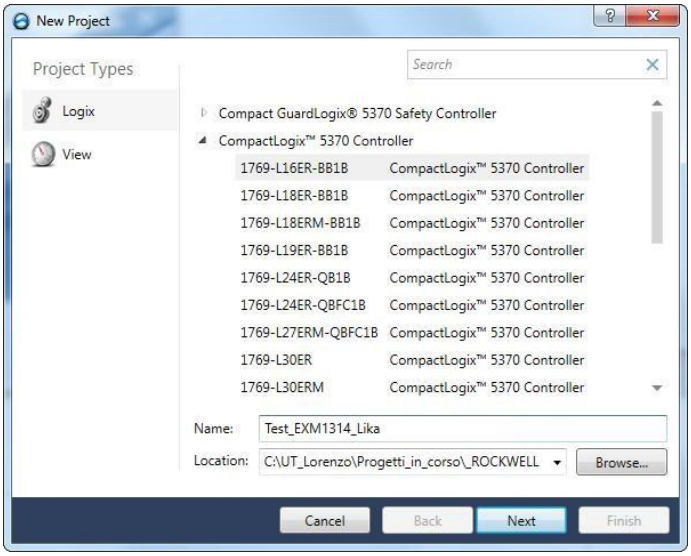


Figure 13 - New Project

Press the **Next** button and then set the **Revision** and the **Expansion I/O** settings. Finalize by pressing the **Finish** button.

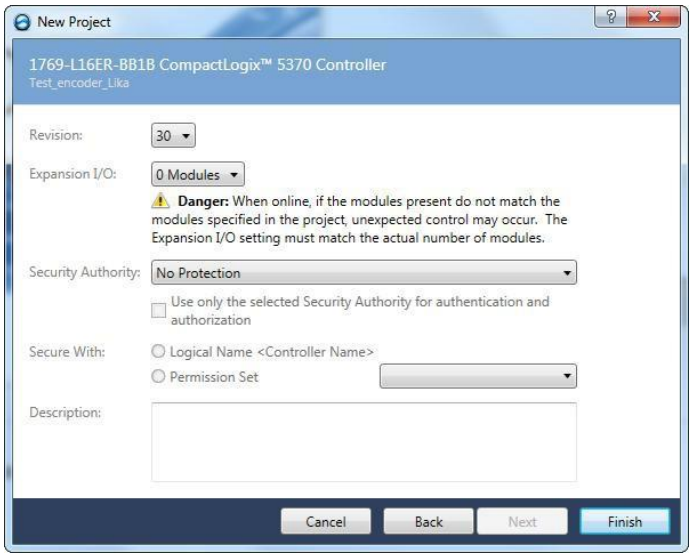


Figure 14 - Controller's settings

4.5.6 Installing the EDS file

To manually register the EDS files of the encoder in the **EDS Hardware Installation Tool**, perform the following steps.

Launch the **EDS Hardware Installation Tool** by pressing **Tools** and then the **EDS Hardware Installation Tool** command.
The **Rockwell Automation's EDS Wizard** dialog box opens.
On the **Options** screen select **Register an EDS file(s)**, then press **Next**.

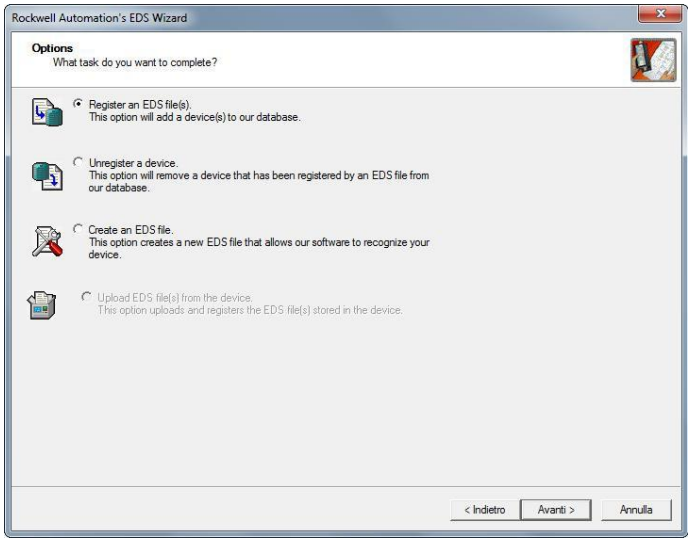


Figure 15 - EDS Wizard

On the **Registration** screen select **Register a single file** to register one EDS file at a time, and click **Browse** to select the EDS file corresponding to the encoder to be installed (such as Lika EXM5XX-13-14-EP_V0.eds in the screenshot Figure 16, please check the order code) and press the **Next** button until the registration is finalized.

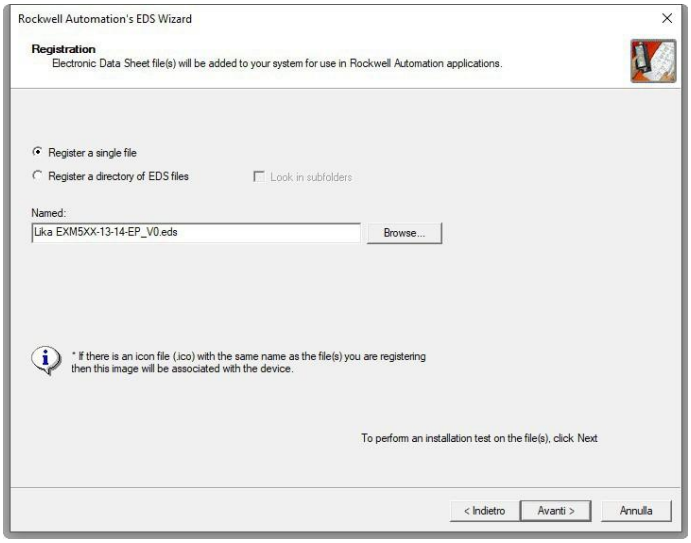


Figure 16 - EDS Wizard

4.5.7 Defining the communication path

To define a path to the controller click on the icon shown in Figure 17.

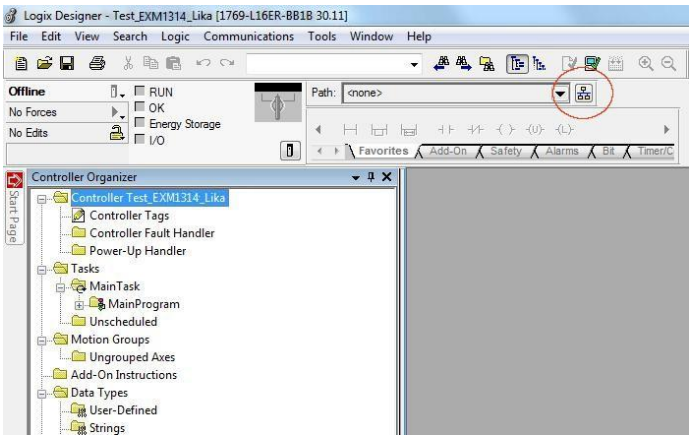


Figure 17 - Path to Controller

Browse to the Controller, select it and click the **Set Project Path** button.

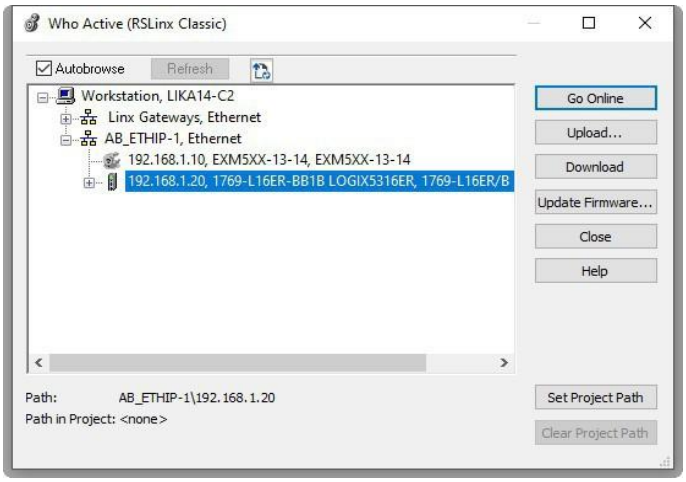


Figure 18 - Set Project Path

Close the dialog box: the selected path will appear on the main page.

Figure 19 - Project Path set

4.5.8 Adding the encoder to the project

On the **Controller Organizer**, right-click on **Ethernet** and select **New Module ...** from the pull-down menu.

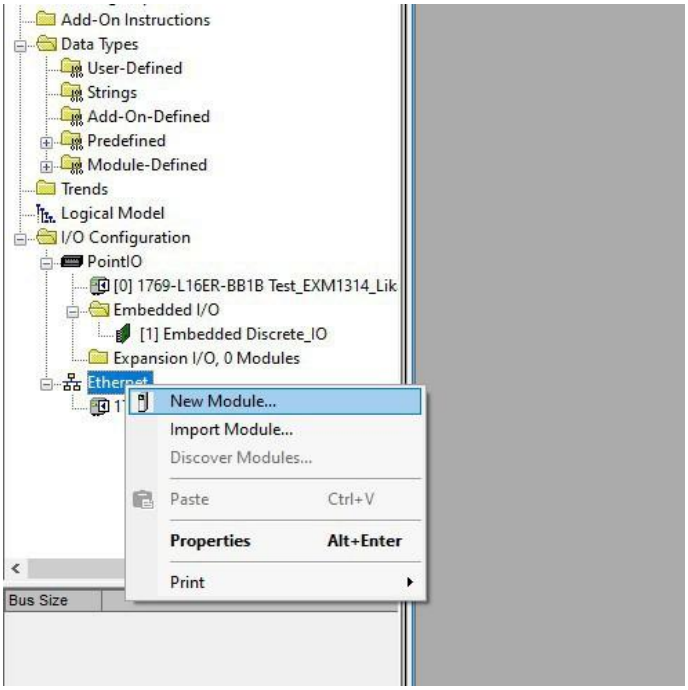


Figure 20 - New Module

On the **Select Module Type** dialog box select the installed encoder module Example: (EXM5X... in the screenshot, Figure 21). Click **Create**.

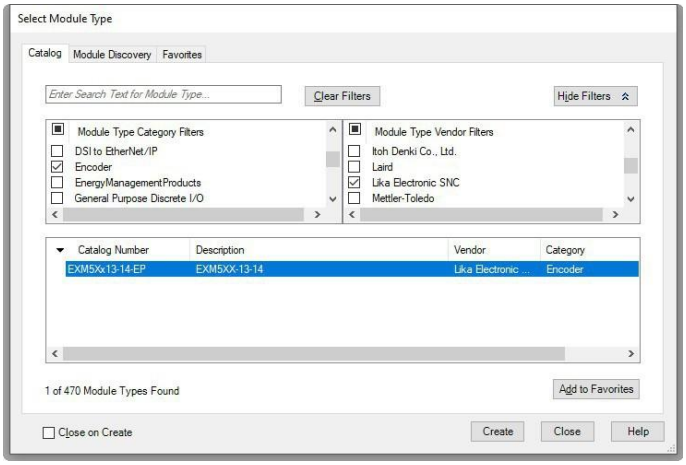


Figure 21 - Select Module Type

Configure the encoder module by setting the required parameters **Name** and **Ethernet Address**. Then press the **Change...** button to select the connection type.

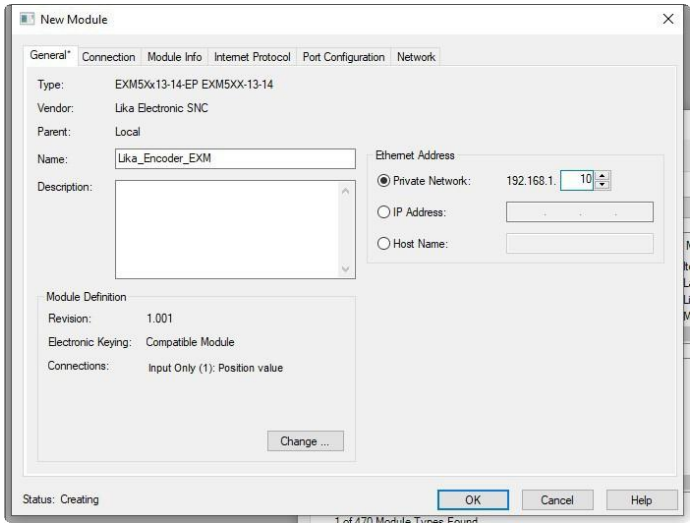


Figure 22 - New module configuration

Select the required connection type and then click **OK**. For more information on the available connection types refer to the "5.12.3.4 Supported connection types" section on page 100.

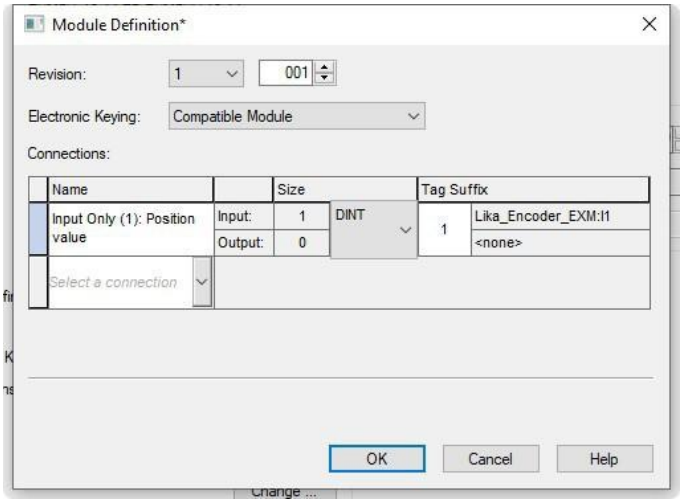


Figure 23 - Select connection type

In the example an Input Only connection has been set: the encoder will send both the position and velocity values (i.e. it will produce instances), while the Controller will send parameters configuration at switching on (the encoder will receive configuration data).
Set DINT data type in order to display properly the position and velocity values.
Press **OK** to finalize and **YES** in the next dialog box.

Close the **New Module** and **Select Module Type** dialog boxes.

4.5.9 Checking the communication

You can check whether the communication between the Controller and the encoder is established properly by displaying the encoder parameters.
On the **Controller Organizer**, double-click on **Controller Tags** in the **Controller Test_EXM1314_Lika** folder: the encoder parameters will be displayed in the **Monitor Tags** tabbed page. The **Monitor Tags** page displays the tags.

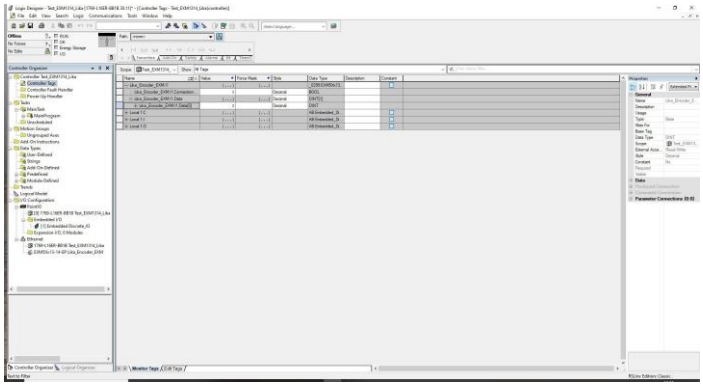


Figure 24 - Monitor Tags

4.5.10 Downloading the configuration to the Controller

To download the configuration to the Controller you must go online first.
Press the drop-down box between the **Offline** and **RUN** items and select **Go Online** in the pull-down menu.

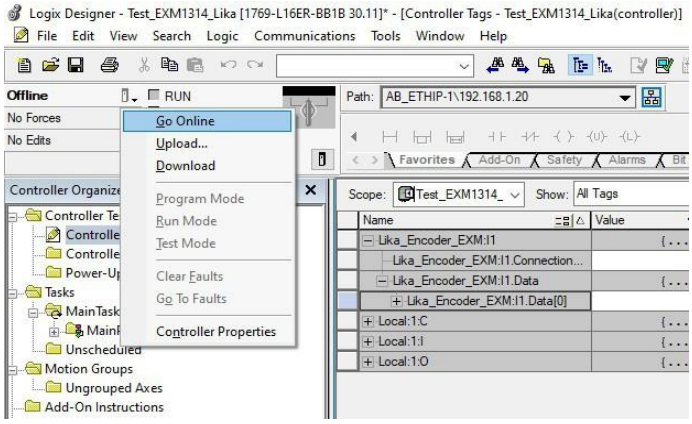


Figure 25 - Going online

Press **Download** in the **Who Active** window to start the download process; the **Download** window will be displayed. Before pressing the **Download** button once more please note the cautionary messages. Click **Download** to continue the download process.
When the download process is completed, the Controller may return to Remote Program mode or ask whether you want to return to Run mode. The message you see is determined by the state the Controller was in at the beginning of the download process.

4.5.11 Configuring the encoder

Before executing the download process, you can set the configuration parameters of the encoder.
On the **Controller Organizer**, right-click **Controller Tags** and choose **Monitor Tags**: the Tag Monitor displays the tags.
A blue arrow indicates that when you change the value, it immediately takes effect.
To see a value in a different style, select the desired style.
To change a value, click the **Value** cell, type the new value, and click **ENTER**.
To expand a tag and show its members, click the + sign.

WARNING
Parameters are not saved on the non-volatile memory. At next power-on you are required to send them again.
To save the parameters permanently you can choose among the following methods:

- by means of the Class Service 16h, see on page 105;
- or by setting the bit 6 **Save Parameters** in the **23-01-68 Command Register** attribute to 1 and then back to 0, see on page 121;
- or by using the **Save Parameters** function in the **Set Encoder Attributes** page of the Integrated Web Server, see the "6.4 Setting the attributes" section on page 175.

4.5.12 How to create a sample program and send the parameters

Here follows a description of a simple program created using Ladder programming language. The program allows to send a preset "1000" to the encoder by means of EtherNet/IP explicit messages with CIP protocol. See also the **23-01-13 Preset Value 32 bit** attribute on page 111.
Described program can be used as a base to build further programs: the procedure is exactly the same, you have just to change the Attribute value and the type of variable of the parameter.

NOTE
As previously stated, Lika Electronic EtherNet/IP encoder documentation is complete with a **sample project** supplied free of charge. This program is designed to make your own project planning, programming, communication and diagnostics with Studio 5000 V30.00 design environment user-friendly and reliable. You can find it in the **Examples_EXM1314_EP.zip** compressed file.
Three demo programs are available.

- **Test_EXM1314_Lika.acd** program allows the user to set and execute the preset (**23-01-13 Preset Value 32 bit**). It is fully described in the following pages.
- **Test_EXM5XX_13_14_demo.acd** program allows the user to set and execute the preset (**23-01-13 Preset Value 32 bit**); to set the singleturn resolution (**23-01-10 Measuring Units per Span**) and the total resolution (**23-01-11 Total Measuring Range 32 bit**); and to send the **23-01-68 Command Register** attribute.
- **Test_EXM5XX_13_14_ST_2.acd** program allows the user to set and execute the preset (**23-01-13 Preset Value 32 bit**); to set high the output 0 when the **23-01-03 Position value 32 bit** attribute is greater than 10,000 cps; to set high the output 1 when the **23-01-18 Velocity Value** attribute is greater than 10,000 cps; and to set high the output 2 when the **23-01-18 Velocity Value** attribute is lower than (-10,000) cps.

Each program requires a main routine. Once you create your routines, assign a main routine for each program.
On the **Controller Organizer**, expand the **Tasks**, **MainTask** and **MainProgram** folders and double-click on **MainRoutine**: the **MainProgram – MainRoutine** ladder window appears.

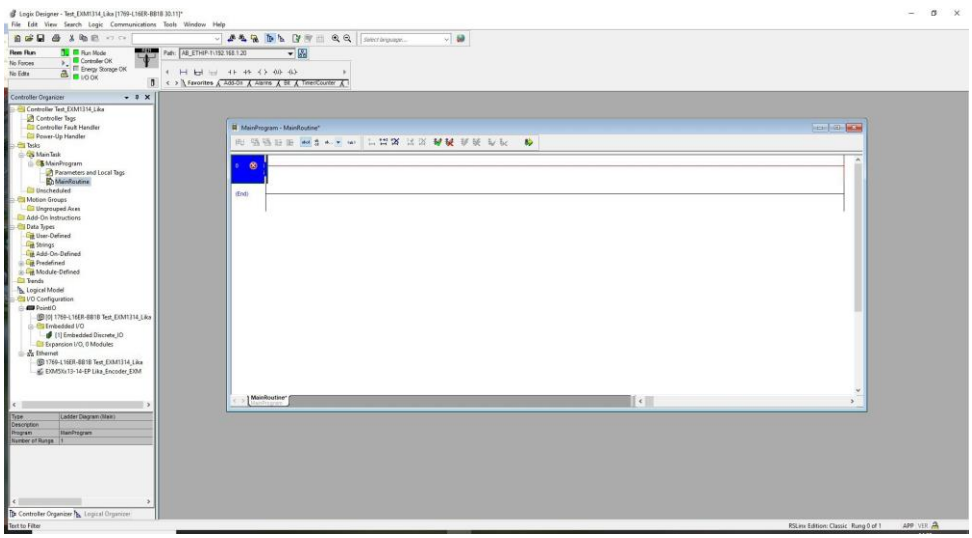


Figure 26 - MainProgram – MainRoutine ladder window

We need to create some tags (variables) that are needful for the program.
On the **Controller Organizer**, right-click on **Controller Tags** and select **New Tag...** from the pull-down menu.

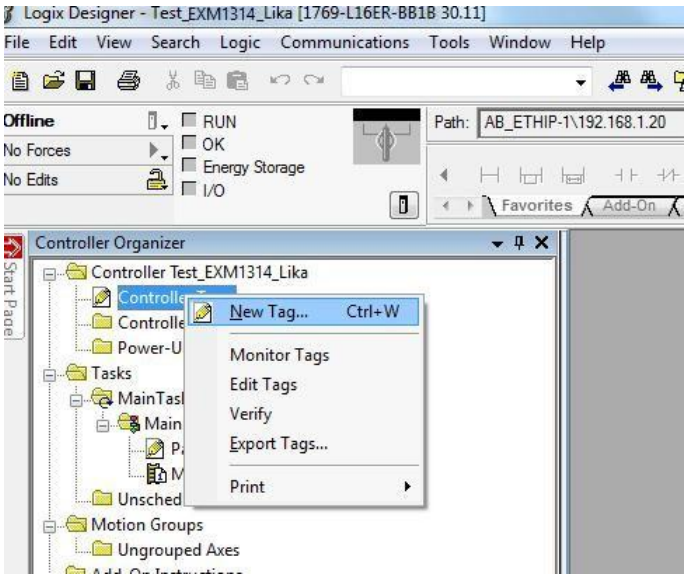
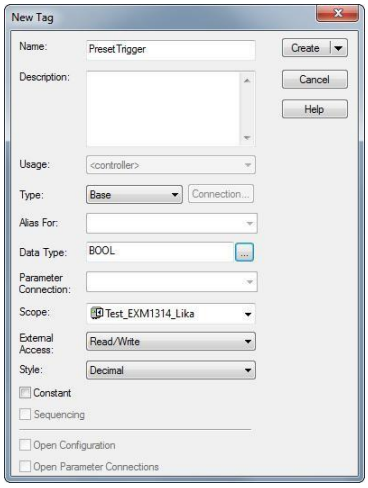


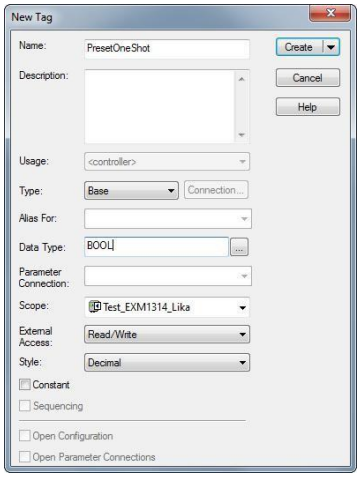
Figure 27 - New Tag

The following tags need to be created:

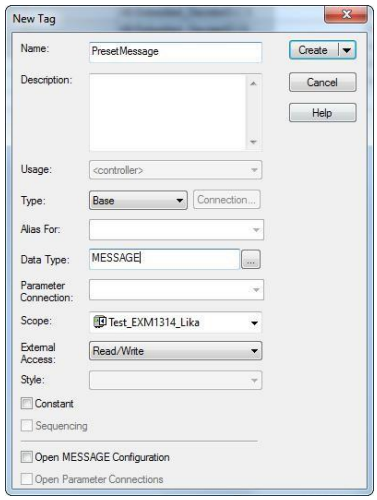
PresetTrigger tag, data type BOOL



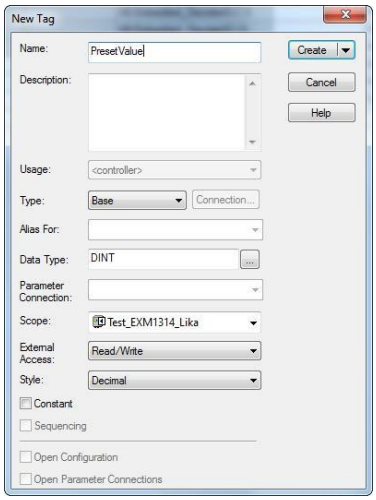
PresetOneShot tag, data type BOOL



PresetMessage tag, data type MESSAGE



PresetValue tag, data type DINT



NOTE
You can type any name for the tags.

Now we need to add ladder logics to the program. To enter logics you must drag buttons from the **Logic Element** toolbar to the desired location. A green dot shows a valid placement location (drop point).

Drag the "**Examine ON (XIC, Examine If Closed)**" logic element onto rung 0 until the green dot appears. Release the mouse button at the location you wish to place your instruction.
Repeat the operation to add a "**One Shot Block (ONS)**" logic element.

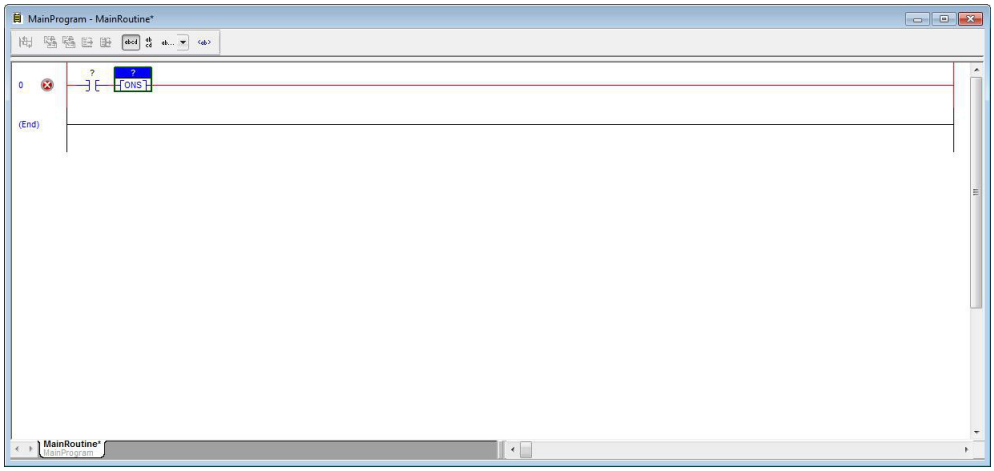


Figure 28 - Dragging logic elements

NOTE
If you place an instruction in the wrong location on a rung, simply click and hold on the instruction and drag it to the correct location.

Double-click the question mark in the "**Examine ON (XIC)**" logic to assign the **PresetTrigger** tag. Choose the tag from the variable list in the drop-down menu.

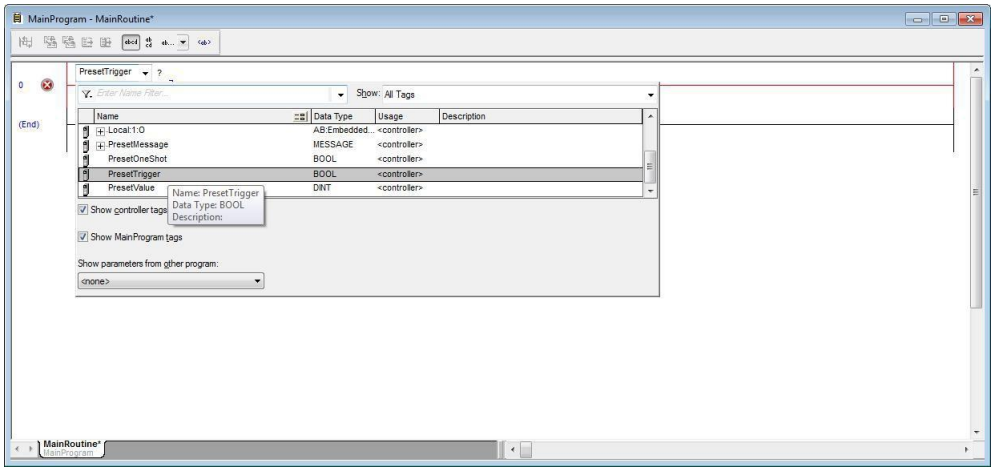


Figure 29 - Assigning tags

Double-click the question mark in the "**One Shot Block (ONS)**" logic to assign the **PresetOneShot** tag. Choose the tag from the variable list in the drop-down menu.

Finally you will get the following situation:

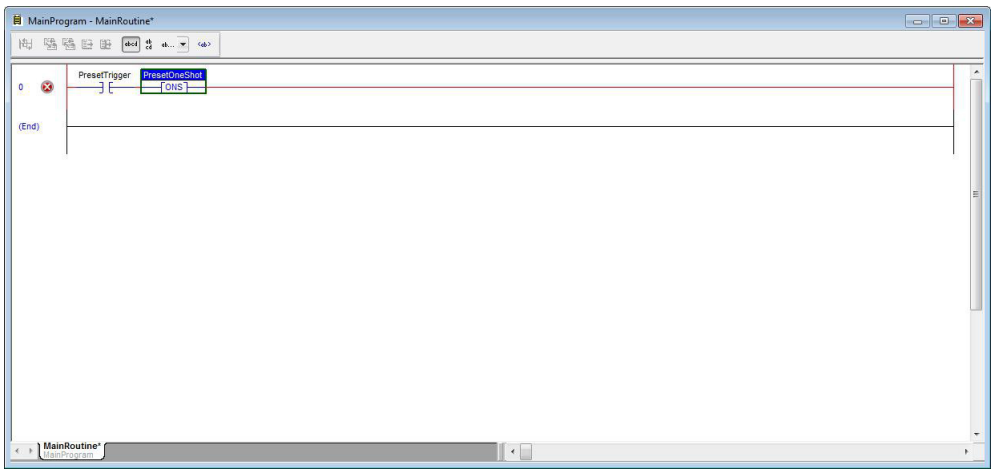


Figure 30 - Tags

Now drag a "**Message (MSG)**" logic from the **Logic Element** toolbar to the desired location. Assign the **PresetMessage** tag as described above.

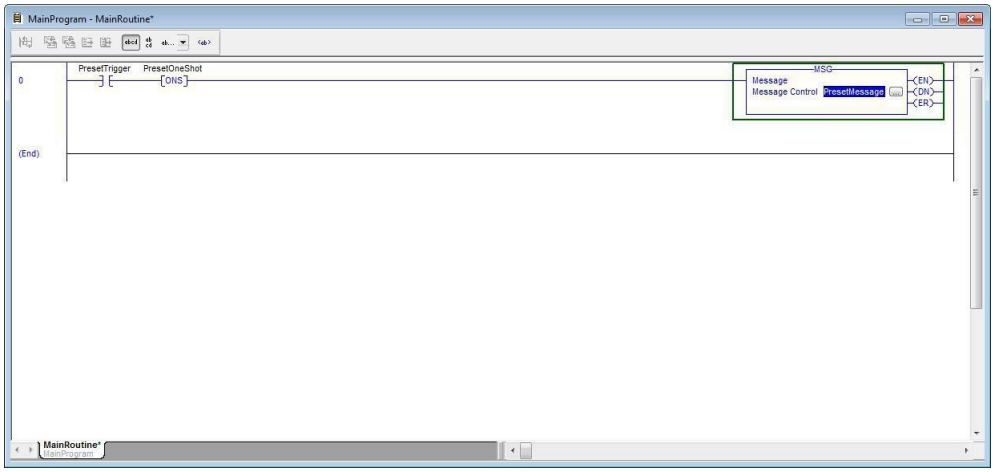
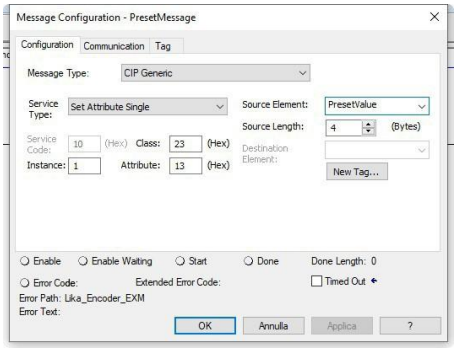


Figure 31 - Message logic element

Configure the message, press the  ellipsis icon next to the **PresetMessage** label to open the **Message Configuration** dialog box.
Configure both the **Configuration** and the **Communication** tabbed pages as shown in the following screenshots, Figure 32.



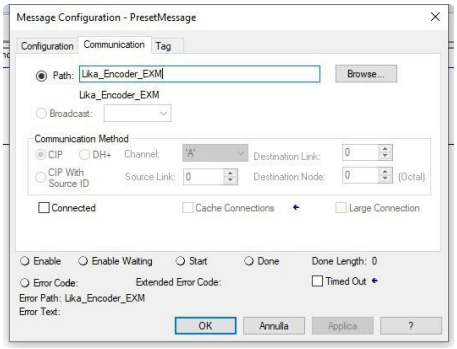


Figure 32 - Message Configuration

See the **23-01-13 Preset Value 32 bit** attribute in the "5.12.5 Class 23h: Position Sensor Object" section on page 111.

On the **Controller Organizer**, double-click on **Controller Tags** in the **Controller Test_EXM1314_Lika** folder: the encoder parameters will be displayed in the **Monitor Tags** tabbed page. The **Monitor Tags** page displays the tags.

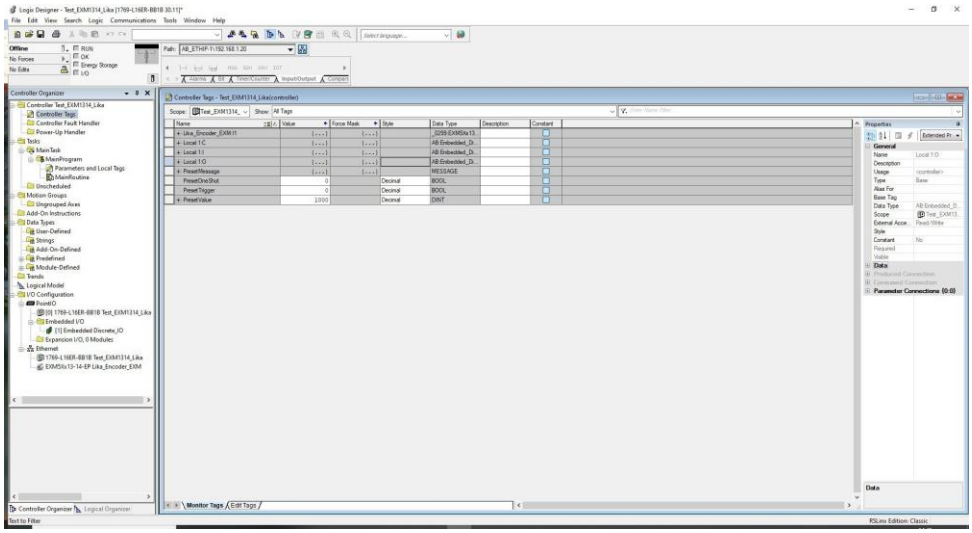


Figure 33 - Setting variables

Set the **PresetOneShot** variable to "0", the **PresetTrigger** variable to "0" and the **PresetValue** variable to "1000", as shown in the screenshot above, Figure 33. To change a value, click the **Value** cell, type the new value, and click **ENTER**. Click the **Style** cell and set the three variables to "Decimal".

Now go online, download data to the Controller and then put the Controller in Run mode.

On the **Controller Organizer**, expand the **Tasks**, **MainTask** and **MainProgram** folders and double-click on **MainRoutine**: the **MainProgram – MainRoutine** ladder window appears.

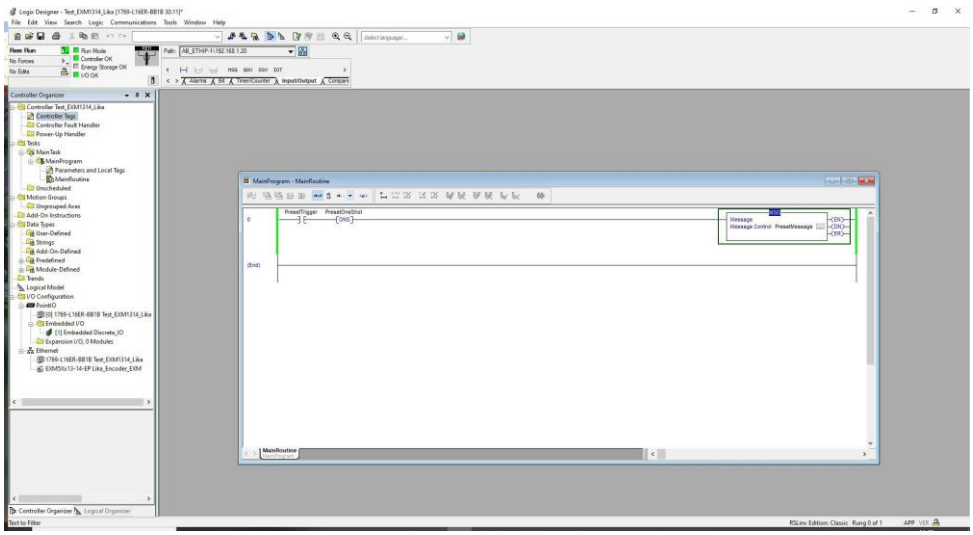


Figure 34 - MainProgram – MainRoutine ladder window

Right-click on the **PresetTrigger** logic element and select **Toggle Bit** from the pull-down menu. The position of the encoder will be preset to the value "1000".

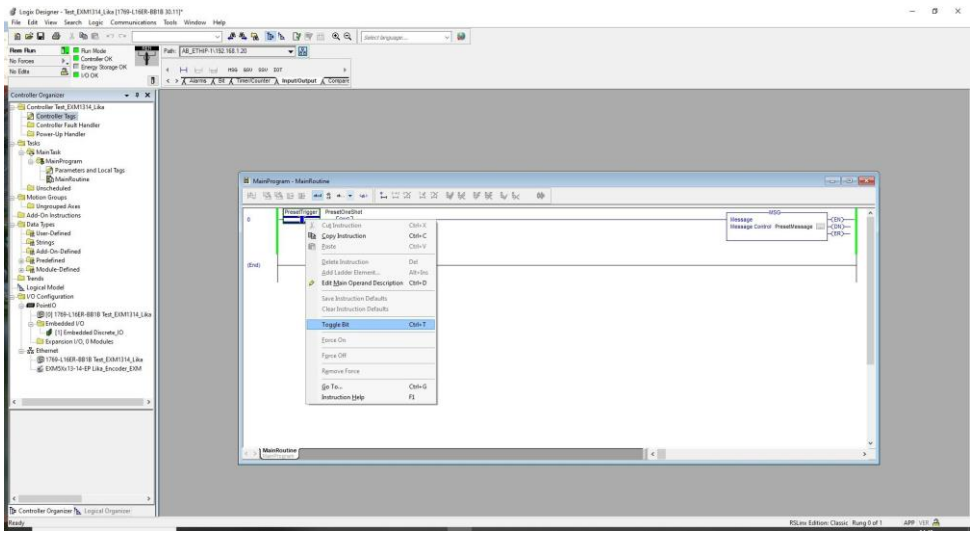


Figure 35 - Setting the Toggle bit

Both the **PresetOneShot** variable and the **PresetTrigger** variable in the **Monitor Tags** tabbed page will be set to "1".

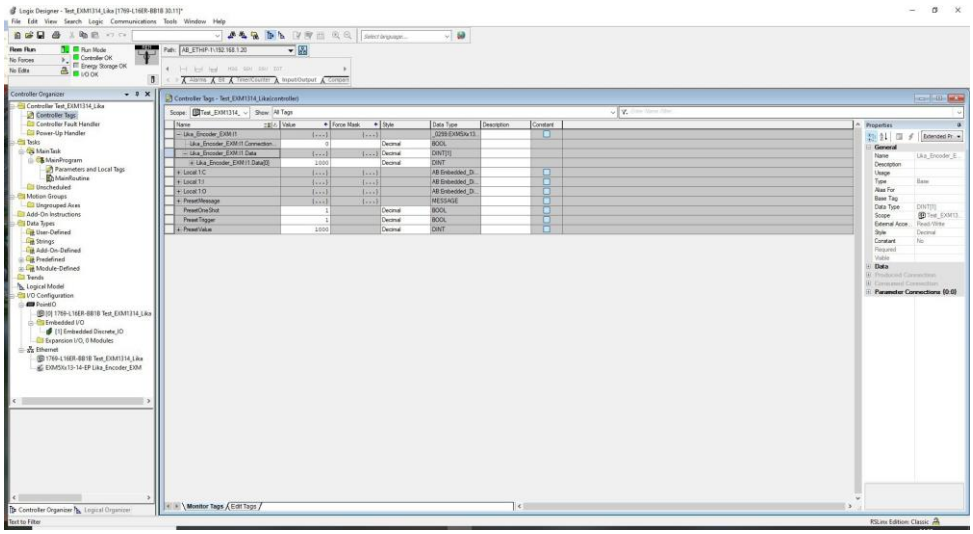


Figure 36 - Monitor Tags

Again right-click on the **PresetTrigger** logic element and select **Toggle Bit** from the pull-down menu to set the variables back to 0.

5 EtherNet/IP interface

5.1 Introduction to EtherNet/IP

EtherNet/IP is the name given to the Common Industrial Protocol (CIP), as implemented over standard Ethernet (IEEE 802.3 and the TCP/IP protocol suite). EtherNet/IP was introduced in 2001 and today is the most developed, proven and complete industrial Ethernet network solution available for manufacturing automation, with rapid growth as users seek to harness the advantages of open technologies and the Internet. EtherNet/IP is a member of a family of networks that implements CIP at its upper layers.
EtherNet/IP and CIP are managed by ODVA, see later. ODVA publishes "The EtherNet/IP™ Specification" and helps ensure compliance through conformance testing.

5.2 CIP protocol

The Common Industrial Protocol (CIP) is a media independent, connection-based, object-oriented protocol designed for automation applications. It encompasses a comprehensive set of communication services for automation applications: control, safety, synchronization, motion, configuration and information. It allows users to integrate these applications with enterprise-level Ethernet networks and the Internet. CIP provides users with a unified communication architecture throughout the manufacturing enterprise. CIP allows users to benefit from the many advantages of open networks while protecting their existing automation investments when upgrading in the future.
CIP brings:

- Coherent integration of I/O control, device configuration and data collection.
- Seamless flow of information across multiple networks.
- Ability to implement multi-layer networks without the added cost and complexity of bridges and proxies.
- Minimized investment in system engineering, installation and commissioning.

The "IP" in "EtherNet/IP" refers to "Industrial Protocol". EtherNet/IP utilizes CIP over standard IEEE 802.3 and the TCP/IP protocol suite. Since EtherNet/IP uses standard Ethernet and TCP/IP technologies, it allows compatibility and coexistence with other applications and protocols.

5.3 CIP and International Standards

CIP technologies are compliant with a number of fieldbus-related international standards, and are generally referred to as members of CPF 2 (Communication Profile Family 2) of IEC 61158.

- IEC 61158: Specifies various fieldbus protocols for applications ranging from discrete manufacturing to process control. It includes the

specifications for CIP, as well as EtherNet/IP and ControlNet-specific protocol elements, as Type 2.

- IEC 61784-1 and IEC 61784-2: Specify general-purpose and real time Ethernet fieldbus Communication Profiles (CPs) (i.e., how to build a specific communication network using IEC 61158 and other standards). ControlNet, EtherNet/IP and DeviceNet are defined respectively as CP 2/1, CP 2/2 (CP 2/2.1 with CIP Sync), and CP 2/3.
- IEC 61784-3: Specifies Functional Safety Communication Profiles (FSCPs), i.e., extensions of fieldbusses for use in safety related applications. CIP Safety is included as FSCP 2/1.
- IEC 61918 & IEC 61784-5: Specify general and fieldbus-specific cabling installation guidelines. IEC 61784-5 includes specific guidelines for ControlNet, EtherNet/IP and DeviceNet.
- IEC 61800-7: Specifies profiles for power drive systems and their mapping to existing communication systems by use of a generic interface. It includes CIP Motion and its mapping on ControlNet, EtherNet/IP and DeviceNet.
- ISO 15745: Defines elements and rules for application integration, including communication network profiles and the communication aspects of device profiles for some fieldbus technologies. EDS files used for device and network integration of DeviceNet, ControlNet or EtherNet/IP applications are compliant with the relevant parts of ISO 15745 (respectively Parts 2, 3 and 4).

Also:

- The lower layers of EtherNet/IP are based on the various RFC Internet standards for the TCP/UDP/IP suite, on the IEEE 802.3 and ISO Ethernet standards (ISO/IEC 8802-3), without modification or extension.
- CIP Safety (on EtherNet/IP) has been certified for use in applications in systems needing to meet the requirements of IEC 61508 up to and including SIL3.

5.4 EtherNet/IP adaptation to CIP

EtherNet/IP, like other CIP Networks, follows the Open Systems Interconnection (OSI) model, which defines a framework for implementing network protocols in seven layers: physical, data link, network, transport, session, presentation and application. Networks that follow this model define a complete suite of network functionality from the physical implementation through the application or user interface layer. As with all CIP Networks, EtherNet/IP implements CIP at the Session layer and above and adapts CIP to the specific EtherNet/IP technology at the Transport layer and below. This network architecture is shown in Figure 37. Ethernet has the unique characteristic of being a network with an active infrastructure. Therefore, unlike typical device or control level networks—which generally have a passive infrastructure that limits the number of devices that can be connected and the way they can be connected—the EtherNet/IP network infrastructure can accommodate a virtually unlimited number of point-to-point nodes, providing users with unsurpassed flexibility in designing networks that accommodate their current requirements while enabling easy, cost-effective expansion in the future.

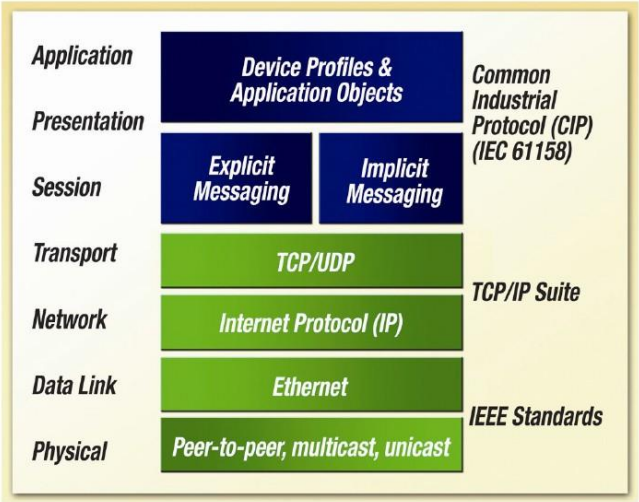


Figure 37 - EtherNet/IP adaptation to CIP

To further decrease complexity, EtherNet/IP systems require only a single point of connection for both configuration and control, because EtherNet/IP supports both I/O (or implicit) messages—those that typically contain time-critical control data—and explicit messages—those in which the data field carries both protocol information and instructions for service performance (see the "5.9.4 Types of EtherNet/IP communications" section on page 84). And, as a producer-consumer network that supports multiple communication hierarchies and message prioritization, EtherNet/IP provides more efficient use of bandwidth than a device network based on a source-destination model. EtherNet/IP systems can be configured to operate either in a Master/Slave or distributed control architecture using peer-to-peer communication.

5.5 The Physical Layer

EtherNet/IP uses standard IEEE 802.3 technology at the Physical and Data Link Layers. This standard provides a specification for physical media, defines a simple frame format for moving packets of data between devices and supplies a set of rules for determining how network devices respond when two devices attempt to use a data channel simultaneously. This is known as CSMA/CD (Carrier Sense Multiple Access/Collision Detection).
As a network with an active infrastructure, EtherNet/IP is typically configured using a series of network segments constructed of point-to-point connections in a star configuration. The core of this network topology is an interconnection of Ethernet Layer 2 and Layer 3 switches that can accommodate an unlimited number of point-to-point nodes.

5.6 The Data Link Layer

IEEE's 802.3 specification is also the standard used for transmitting packets of data from device to device on the EtherNet/IP Data Link Layer. Ethernet employs

a CSMA/CD media access mechanism that determines how networked devices share a common bus (i.e., cable), and how they detect and respond to data collisions.
Originally, Ethernet worked in a half-duplex mode of operation, meaning that a node could send or receive data, but it could not do both at the same time. This caused data traffic jams, which are unacceptable in time-critical control applications. With full-duplex Ethernet, networked devices can both send and receive packets of Ethernet data at the same time. This is one of several advances in Ethernet technology that has increased its level of determinism to the point where Ethernet can be used in an ever-increasing number of manufacturing applications.
The Media Access Control (MAC) protocol of the IEEE 802.3 specification is what actually allows devices to "talk" on the Ethernet network. Each device has a unique MAC address comprised of a 6-byte number that is regulated by IEEE and the product manufacturer to maintain uniqueness (refer also to the "4.4 MAC address" section on page 53). This MAC address is used in the source address (SA) field of the frame to indicate what node sent the frame, and it is used in the destination address (DA) field to indicate the destination of the frame. Setting the first bit to a "1" in the DA field indicates a packet of data for multiple destinations, and enables an Ethernet node to transmit a single data packet to broadcast to the various destinations.
A single frame of industrial EtherNet/IP can contain up to 1,500 bytes of data, depending on the application requirements. The combination of real-time control with high-data capacity makes industrial Ethernet increasingly attractive, as more intelligence is embedded into smaller and less-expensive devices.

5.7 Ethernet data packets

Ethernet data packets are sent in the format shown in Figure 38.

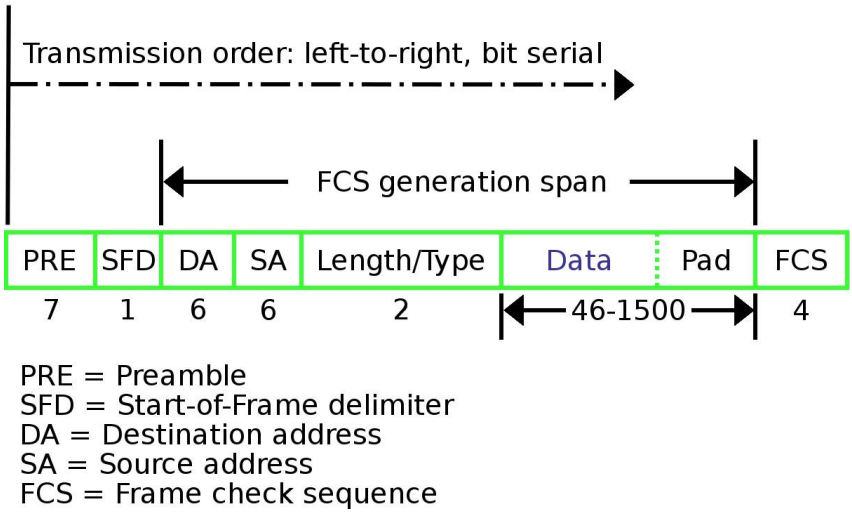


Figure 38 - Typical layout of an Ethernet Data Packet

This data format is used to implement the **Media Access Control (MAC)** protocol that allows a device to "talk" on the Ethernet network. Each MAC device has a unique **Source Address (SA)** comprised of a 6-byte number (48 bits or 12 hexadecimal digits) that was assigned to it at the time of manufacture. The **Destination Address (DA)** is the target MAC address for which the packet of data is intended. Setting the first bit to a "1" in the DA field, indicates a packet of data for multiple destinations. This enables an Ethernet device to transmit one packet that can be received by multiple other devices.
There are a number of different types of Ethernet packets that can be sent and received on an Ethernet network. Some of these protocols are Novell's IPX/SPX, DECNET, UDP, TCP/IP, FTP, TELNET, and so on. All of these unique protocols use the MAC to do the physical sending and receiving of data packets. However, by defining how the "DATA" portion of the data packet is organized, different protocols and functions are created.

5.8 The Network and Transport Layers

At the Network and Transport Layers, EtherNet/IP utilizes the Internet standard known as the Transmission Control Protocol/Internet Protocol (TCP/IP) Suite to send messages between one or more devices. TCP/IP provides the necessary communication protocol features needed to implement fully functional networks (i.e., an addressing scheme and mechanisms for establishing a connection with a device and exchanging data) that the IEEE specification in and of itself lacks.
Also, at these layers, the standard CIP messages used by all CIP Networks are encapsulated. TCP/IP encapsulation allows a node on the network to embed a message as the data portion in an Ethernet message. The node then sends the message—TCP/IP protocol with the message inside—to an Ethernet communication chip (the Data Link Layer). By using TCP/IP, EtherNet/IP is able to send **explicit messages**, which are used to perform Client-Server type transactions between nodes.

The TCP/IP Suite consists of the following:

- The TCP portion of the TCP/IP protocol is a connection-oriented, unicast transport mechanism that provides data flow control, fragmentation reassembly and message acknowledgments. Nodes must interpret each message, execute the requested task and generate responses. Since TCP is ideal for the reliable transmission of large quantities of data, EtherNet/IP uses TCP/IP to encapsulate CIP explicit messages, which are generally used to transmit configuration, diagnostic and event data.
- The IP portion of the TCP/IP protocol is the mechanism that enables packet routing through multiple possible paths. The ability to send messages to their destinations even when the primary path is disrupted is the basis of the Internet. This same type of routing is used in industrial networks to maintain proper separation of control elements and other factory infrastructure through the use of managed switches and Layer 3 routers. All devices and infrastructure components with added diagnostic capabilities (managed switches and routers) on an industrial

Ethernet-based system must be assigned an IP address. This is most commonly identified by the four-byte address listed in the "network properties" on personal computers that use TCP/IP as their Ethernet network connection (e.g., 192.168.1.10). IP addresses must be unique on a given network (see also the "3.8 EtherNet/IP Node ID" section on page 39).

For real-time messaging, EtherNet/IP also employs UDP over IP, which allows messages to be multicast to a group of destination addresses. This is how CIP I/O data transfers (**implicit messaging**, see the "5.9.4 Types of EtherNet/IP communications" section later) are sent on EtherNet/IP. With implicit messaging, the data field contains no protocol information, only real-time I/O data. Since the meaning of the data is pre-defined at the time the connection is established, processing time is minimized during runtime. UDP is connectionless and makes no guarantee that data will get from one device to another; however, UDP messages are smaller and can be processed more quickly than explicit messages. As a result, EtherNet/IP uses UDP/IP to transport I/O messages that typically contain time-critical control data. The CIP Connection mechanism provides timeout mechanisms that can detect data delivery problems, a capability that is essential for reliable control system performance.

5.9 Upper Layers: Objects, Services, and Application Data

5.9.1 EtherNet/IP services

The CIP application layer defines a set of **application objects** and **device profiles** that define common interfaces and behaviors. In addition, CIP communication services enable end-to-end communication between devices on the different CIP networks. EtherNet/IP maps the CIP communication services to Ethernet and TCP/IP, enabling multi-vendor interoperability between devices on Ethernet as well as with the other CIP networks.

5.9.2 Simplified EtherNet/IP Object Model Overview

Within the CIP application layer, devices are represented using an object model (Figure 39). **Application objects** define how device data is represented and accessed in a common way. **Network-specific objects** define how parameters such as IP addresses are configured and EtherNet/IP specific functions.
Communication objects and services provide the means to establish communication associations and access device data and services over the network.

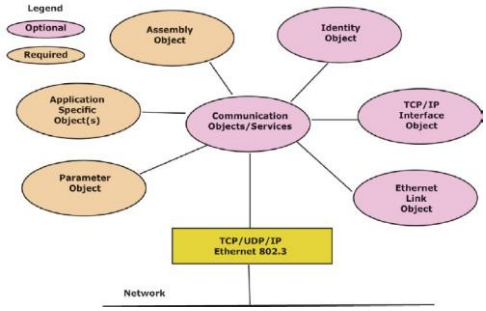


Figure 39 - EtherNet/IP Object Model

5.9.3 Exposing Application Data with CIP

Objects within a device are groups of related data and behavior associated with this data. CIP requires certain objects to describe a device, how it functions, communicates and its unique identity. The **Identity Object** (see on page 90), for example, contains identity data values called **attributes** that are used to store the identity information of a device. Attributes for the Identity Object include the Vendor ID, Device Type, device serial number and other identity data. CIP does not specify how object data is implemented, rather, which data values or attributes must be supported and made available to other CIP devices.

There are three types of objects defined by CIP:

- **Required Objects** must be included in all CIP devices. These objects include the Identity Object (page 90), the Message Router Object (page 97) and network-specific objects such as TCP/IP Interface Object (page 140) and Ethernet Link Object (page 152) for EtherNet/IP protocol.
- **Application Objects** describe how data is encapsulated by a device. These objects are specific to the Device Type and function. For example, an input device would have an input object with attributes that describe the value and fault status of a particular input point. See the Position Sensor Object (page 105).
- **Vendor-specific Objects** describe services that are specific to a particular vendor; they are optional and not described in a predefined Device Profile. See for instance the Diagnosis Object (page 168).

Addressing data within a CIP device utilizes the same object-oriented view. A **class** (of objects) is a set of objects that represent the same type of system component (Figure 40). Sometimes it is necessary to have more than one "copy"

of an object, called **object instances**, within a device. This set of objects is called an **object class**. Each instance of the object class will have the same set of attributes, but will have a unique set of values. An object instance or an object class has **attributes**, providing services and implementing behavior.

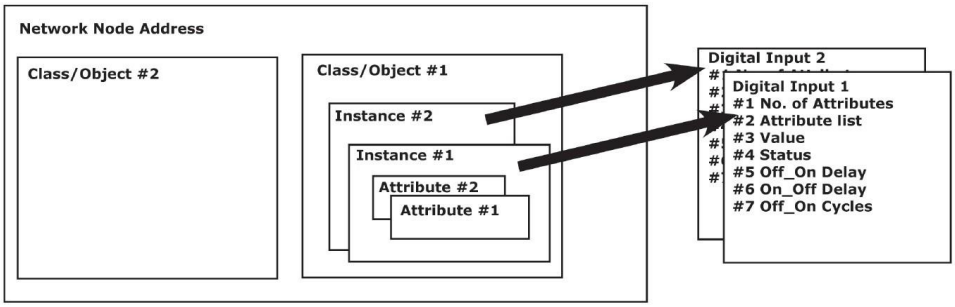


Figure 40 - CIP Object-oriented view of application data

The following Object Modeling related terms are used when describing CIP services and protocol.

- **Object** – An abstract representation of a particular component within a product.
- **Class** – A set of objects that all represent the same kind of system component. A class is a generalization of an object. All objects in a class are identical in form and behavior, but may contain different attribute values.
- **Instance** – A specific and real (physical) occurrence of an object. For example: New Zealand is an instance of the object class Country. The terms Object, Instance, and Object Instance all refer to a specific Instance.
- **Attribute** – A description of an externally visible characteristic or feature of an object. Typically, attributes provide status information or govern the operation of an Object. For example: the ASCII name of an object; and the repetition rate of a cyclic object.
- **Instantiate** - To create an instance of an object with all instance attributes initialized to zero unless default values are specified in the object definition.
- **Behavior** – A specification of how an object acts. Actions result from different events the object detects, such as receiving service requests, detecting internal faults or elapsing timers.
- **Service** – A function supported by an object and/or object class. CIP defines a set of common services and provides for the definition of Object Class and/or Vendor Specific services.
- **Communication Objects** - A reference to the Object Classes that manage and provide the runtime exchange of implicit (I/O) and explicit messages.
- **Application Objects** - A reference to multiple Object Classes that implement product-specific features.

Accessing data within a device using a non-time critical message (an explicit message – see "5.9.4 Types of EtherNet/IP communications" section on page 84) typically contains the following address information:

- Device network address
- Class ID
- Instance ID
- Attribute ID
- Service code (describing the action/service required)

The Class/Instance/Attribute ID form of addressing is also used in Electronic Data Sheets (EDS) to identify configurable parameters within a device.
In addition to specifying how device data is represented, CIP also specifies methods by which I/O data can be accessed, using triggers, such as cyclic or change-of-state. Vendors can also describe how data from different objects can be combined in an I/O or configuration message using the Assembly Object, refer to the "5.12.3 Class 04h: Assembly Object" section on page 99.

5.9.4 Types of EtherNet/IP communications

EtherNet/IP defines two primary types of communications: **explicit** and **implicit**.

CIP Message Type	CIP Communication Relationship	Transport Protocol	Communication Type	Typical Use	Example
Explicit	Connected or Unconnected	TCP/IP	Request/reply transactions	Non time-critical information data	Read/Write configuration parameters
Implicit	Connected	UDP/IP	I/O data transfers	Real-time I/O data	Real-time control data from a remote I/O device

- **Explicit Messaging** in general has a request/reply (or Client/Server) nature. This type of communication is **used for non-real-time data**, normally for information. Explicit messages include a description of their meaning (expressed explicitly), so the transmission is less efficient, but very flexible. It may be used by an HMI to collect data, or by a device programming tool. In CIP terms, with Explicit Messaging you request a service of a particular object, e.g., a read or a write service. For EtherNet/IP, Explicit Messaging uses TCP. Explicit Messaging can be done with or without prior establishment of a CIP connection.
- **Implicit Messaging** is also often referred to as **"I/O" and is time-critical** in nature. Typically this type of communication is used for **real-time data exchange**, where speed and low latency are important. Implicit messages include very little information about their meaning, so the transmission is more efficient, but less flexible than explicit. The interpretation of the transmitted data is fast. With Implicit Messaging

you establish an association (a "CIP connection") between two devices and produce the Implicit Messages according to a predetermined trigger mechanism, typically at a specified packet rate. The devices both know and agree on the data formats they will use (i.e., the format is "implied"). For EtherNet/IP, Implicit Messaging uses UDP and can be multicast or unicast.

Connections are established using the ForwardOpen Request service of the Connection Manager Object, see the "5.12.4 Class 06h: Connection Manager Object" section on page 103. The ForwardOpen Request contains all of the connection parameters, including transport class, production trigger, timing information, electronic key and connection IDs. Connection clean-up takes place when a ForwardClose Request service request is issued or when either connection end point times out.
Implicit messaging can make use of the CIP Producer/Consumer communication model. With **Producer/Consumer**, the producing device transmits data once, regardless of the number of consumers. All interested consuming devices receive the same data. For EtherNet/IP the produced data is identified by the IP multicast address and the CIP Connection ID. The Producer/Consumer model leads to greater network efficiency when multiple consumers need to receive the same data from a producer. For I/O connections, once the connection is established there is no request/response, the data with the ConnectionID is just produced and consumed at intervals determined by the Production Trigger which was specified at connection establishment. Triggers can be Cyclic (most common), Change of State (CoS) or Application.

5.9.5 Types of EtherNet/IP devices

Several device classifications, based on their general behavior and types of EtherNet/IP communications they support, have been defined:

- **Explicit Message Server**: An explicit message server responds to request/response oriented communications initiated by explicit message clients. An example of an explicit message server is a bar code reader.
- **Explicit Message Client**: An explicit message client initiates request/response oriented communications with other devices. Message rates and latency requirements are typically not too demanding. Examples of explicit message clients are HMI devices, programming tools, or PC or Linux based applications that gather data from control devices.
- **I/O Adapter**: An I/O adapter receives implicit communication connection requests from an I/O scanner then produces its I/O data at the requested rate. An I/O adapter is also an explicit message server. An I/O adapter can be a simple digital input device, or something more complex such as a modular pneumatic valve system.
- **I/O Scanner**: An I/O scanner initiates implicit communications with I/O adapter devices. A scanner is typically the most complex type of EtherNet/IP device, as it must deal with issues such as configuration of which connections to make, and how to configure the adapter device.

Scanners also typically support initiating explicit messages. A programmable controller is an example of an I/O scanner.

5.10 ODVA

ODVA is an international association comprising members from the world's leading automation companies. Collectively, ODVA and its members support network technologies based on the Common Industrial Protocol (CIP™). These currently include DeviceNet™, EtherNet/IP™, CompoNet™, and ControlNet™, along with the major extensions to CIP — CIP Safety™, CIP Sync™ and CIP Motion™. ODVA manages the development of these open technologies, and assists manufacturers and users of CIP Networks through its activities in standards development, certification, vendor education and industry awareness.

For further information on ODVA, see the ODVA website: www.odva.org.

5.11 EDS file

The functionality of an EtherNet/IP device is always described in an EDS file (Electronic Data Sheet file). The Electronic Data Sheet File provides information about the device basic communication and functional properties. It must be installed in the Controller.

EtherNet/IP encoders from Lika Electronic are supplied with their own EDS file. Specific EDS files are provided to each encoder model.
They are:

- **Lika AM58X-1314-EP_v1_1.eds**: it is intended for installation of **AM58x-1314 EtherNet/IP encoders.**

The version of the EDS file is reported under the Version item inside the file.
EDS files can be paired with the **EXO58_EXM58_48x48.ico** picture file available inside the file folder (the picture is also integrated into the EDS file).

Go to Support Resources on the encoder item page at AutomationDirect.com to get the correct EDS file.

5.12 Object Library

As previously stated, object modeling is used to represent the network visible behavior of devices (i.e. the encoder). Devices are modeled as a collection of objects. Each class of objects is a collection of related services, attributes and behaviors. Services are the procedures that an object performs. Attributes are characteristics of objects represented by values, which can vary. An object's behavior is an indication of how the object responds to particular events. For more information refer to the "5.9.3 Exposing Application Data with CIP" section on page 82.
This section contains the description of the objects specific to Lika encoders, including services and attributes.

The following objects are implemented:

Class Code	Object Class	Page
01h	Identity Object	92
02h	Message Router Object	99
04h	Assembly Object	101
06h	Connection Manager Object	105
23h	Position Sensor Object	107
43h	Time Sync Object	125
47h	Device Level Ring (DLR) Object	135
48h	Quality of Service (QoS) Object	138
F5h	TCP/IP Interface Object	142
F6h	Ethernet Link Object	152
109h	LLDP Management Object	160
401h	Predefined Connection Object	162
402h	IO Mapping Object	168
403h	Diagnosis Object	170

In the following pages the Class Attributes are listed and described as follows:

Class-Attribute ID Attribute name
[Data type, Access Rule, NV]

While the Instance Attributes are listed and described as follows:
Class-Instance-Attribute ID Attribute name
[Data type, Access Rule, NV]

- Class, instance and attribute are expressed in hexadecimal notation.
- Data types are as shown in the following table:

Data type	Code	Name	Range
BOOL	C1h	Boolean	0 (FALSE) and 1 (TRUE)
SINT	C2h	Signed 8-bit integer	-128 to 127
INT	C3h	Signed 16-bit integer	-32,768 to 32,767
DINT	C4h	Signed 32-bit integer	-2^{31} to $2^{31}-1$
LINT	C5h	Signed 64-bit integer	-2^{63} to $2^{63}-1$
USINT	C6h	Unsigned 8-bit integer	0 to 255
UINT	C7h	Unsigned 16-bit integer	0 to 65,535
UDINT	C8h	Unsigned 32-bit integer	0 to $2^{31}-1$
ULINT	C9h	Unsigned 64-bit integer	0 to $2^{63}-1$
STRING	D0h	Character string	1 byte per character
BYTE	D1h	Bit string – 8 bits	$2\#b_{N-1}b_{N-2}...b_2b_1b_0$, where N is the number of bits in the bit string, b_{N-1} is the "most significant bit", and b_0 is the "least significant bit"
WORD	D2h	Bit string - 16 bits	
SHORT_STRING	DAh	Character string	1 byte per character, 1 byte length indicator
ENGUNIT	DDh	Engineering unit	0 to 65,535

- Access rule can be:

Get (Gettable): the same as "ro" = read only access. The attribute can be accessed by at least one of the get services.

Set (Settable): the same as "rw" = read and write access. The attribute shall be accessed by at least one of the set services. Settable attributes, unless otherwise specified by the object definition, shall also be accessed by get services.

- NV / V

It indicates whether an attribute value is maintained through power cycles. An entry of 'NV' indicates value shall be saved, 'V' means not saved.

- Default, Min. and Max. values

Default, Min. and Max. values are expressed in hexadecimal notation, unless otherwise indicated.

NOTE
All data bytes are sent from least significant byte (LSB) to most significant byte (MSB).

5.12.1 Class 01h: Identity Object

Class Code	Object Class	Access	Nr. of Instances
01h	Identity Object	Get	1

The Identity Object provides identification of and general information about the encoder (e.g. Vendor ID, device type, product code, etc.). Instance 1, which is the only mandatory instance, describes the whole product. It is used by applications to determine what nodes are on the network and to match an EDS file with a product on the network. The EtherNet/IP protocol stack implements the Identity object at class level and a single instance with Instance ID 1.

5.12.1.1 Supported Class Services

The supported **Class Services** of the Identity Object are:
01h = Get_Attribute_All: used to read the value of all attributes.
0Eh = Get_Attribute_Single: used to read the value of an attribute.
10h = Set_Attribute_Single: used to write connection class attribute value.
05h = Reset: the following types of reset are defined:

0 = Power Cycle Reset	It emulates a power cycling of the encoder.
1 = Return to Factory Defaults Reset	It returns to the factory default configuration of the encoder parameters and communication link parameters and emulates a power cycling of the encoder.

NOTE
After executing a Return to Factory Defaults reset (type 1), if the DIP A DIP switches are all set to 0, the encoder restarts using the IP address saved internally. If the DIP A DIP switches are set to any value between 1 and 254, then the encoder restarts using the address 192.168.1."DIP switch setting". For more information refer to page 40.

5.12.1.2 Class Attributes

01-01 Revision
[UINT, Get, NV]
Object revision. The current value assigned to this attribute is 0002h.
Default = 0002h

01-02 Max Instance
[UINT, Get, NV]
The largest instance number of a created object in this class.
Default = 0001h

01-03 Number of Instances

[UINT, Get, NV]
The number of object instances in this class.
Default = 0001h

01-06 Maximum ID Number Class Attributes

[UINT, Get, NV]
The attribute ID number of the last class attribute of the class definition implemented in the encoder.
Default = 0007h

01-07 Maximum ID Number Instance Attributes

[UINT, Get, NV]
The attribute ID number of the last instance attribute of the class definition implemented in the encoder.
Default = 0013h

5.12.1.3 Supported Instance Services

The supported **Instance Services** of the Identity Object are:
01h = Get_Attribute_All: used to read the value of all attributes.
0Eh = Get_Attribute_Single: used to read the value of an attribute.
10h = Set_Attribute_Single: used to write connection class attribute value.
4Bh = Flash_LEDs: used to force the LEDs to flash for identification.
05h = Reset: the following types of reset are defined:

0 = Power Cycle Reset	It emulates a power cycling of the encoder.
1 = Return to Factory Defaults Reset	It returns to the factory default configuration of the encoder parameters and communication link parameters and emulates a power cycling of the encoder.

NOTE
After executing a Return to Factory Defaults reset (type 1), if the DIP A dip switches are all set to 0, the encoder restarts using the IP address saved internally. If the DIP A dip switches are set to any value between 1 and 254, then the encoder restarts using the address 192.168.1."dip switch setting". For more information refer to page 40.

5.12.1.4 Instance Attributes

01-01-01 Vendor ID

[UINT, Get, NV]
Identification of the vendor by its own number. Lika Vendor ID is 0299h = 665. Vendor IDs are managed by ODVA.
Default = 0299h = Lika Electronic Srl

01-01-02 Device type

[UINT, Get, NV]
The Device Type value is used to identify the device profile that a particular product is using. Device profiles are managed by ODVA and define minimum requirements a device must implement, as well as common options.
Default = 0022h: Encoder Device Profile

01-01-03 Product code

[UINT, Get, NV]
Product Code identifies a particular product within the encoder device type.
The available product codes are:

- 0082h for 27-bit multiturn encoder

01-01-04 Revision

[USINT, Get, NV]
The Revision attribute, which consists of Major and Minor Revisions, identifies the Revision of the item the Identity Object is representing. It is displayed as majorXX.minorYY, so representing the hardware (XX) and software (YY) revisions.

LSByte XX	MSByte YY
Major revision	Minor revision

Default = device dependent

01-01-05 Status

[WORD, Get, V]
This attribute represents the current status of the device. Its value changes as the state of the device changes. The Status attribute is a WORD, with the following bit definitions:

Bit(s)	Called	Definition
0	Owned	TRUE indicates the device (or an object within the device) has an owner. Within the Master/Slave paradigm the setting of this bit means that the Predefined Master/Slave Connection Set has been allocated to a Master. Outside the Master/Slave paradigm the meaning of this bit is TBD. 0 = no connection to the Master 1 = connection to the Master established
1	Reserved	Reserved, shall be 0
2	Configured	TRUE indicates the application of the device has been configured to do something

		different than the "out–of–box" default. This shall not include configuration of the communications. 0 = encoder is set to default parameters 1 = encoder is not set to default parameters	
3	Reserved	Reserved, shall be 0	
4-7	**Extended device status**	Bits are defined as follows:	
		0000	Self Testing or Unknown
		0001	Firmware update in progress
		0010	At least one faulted I/O connection
		0011	No I/O connection established
		0100	Non-Volatile Configuration bad (EEPROM)
		0101	Major Fault – either bit 10 or bit 11 is TRUE (1)
		0110	At least one I/O connection in run mode
		0111	At least one I/O connection established, all in idle mode
		1000	The Status attribute is not applicable to this instance. Valid only for instances greater than one
		1001 ... 1111	Reserved
8	**Minor recoverable fault**	TRUE indicates that the device detected a problem with itself, which is thought to be recoverable. The problem does not cause the device to go into one of the faulted states. Not implemented. For Alarms list refer to page 114	
9	**Minor unrecoverable fault**	TRUE indicates that the device detected a problem with itself, which is thought to be unrecoverable. The problem does not cause the device to go into one of the faulted states. Not implemented. For Alarms list refer to page 114	
10	**Major recoverable fault**	TRUE indicates that the device detected a problem with itself, which caused the device to go into the "Major Recoverable Fault" state. Not implemented. For Alarms list refer to page 114	
11	**Major unrecoverable fault**	TRUE indicates that the device detected a problem with itself, which caused the device to go into the "Major Unrecoverable Fault" state. Not implemented. For Alarms list refer to page 114	
12 ...15	Reserved	Reserved, shall be 0	

For any further information on status instance attribute refer to the publication "The CIP Networks Library. Volume I. Common Industrial Protocol (CIP™)".

01-01-06 Serial number

[UDINT, Get, NV]
This attribute is a number used in conjunction with the Vendor ID to form a unique identifier for each device on any CIP network.
The Serial Number is shown in the following format: YYwwnnnnn.
YY = Year
ww = week
nnnnn = unique number in ascending order assigned by Lika Electronic
Default = device dependent

EXAMPLE
172100123 has to be intended as follows:
17 = Year of production = 2017
21 = Week of production = week 21
00123 = unique number in ascending order assigned by Lika Electronic

01-01-07 Product name

[SHORT_STRING, Get, NV]
This text string represents a short description of the product represented by the Product Code in the **01-01-03 Product code** attribute.
ADC Default = AM58X-1314 for 27-bit multiturn encoder

01-01-08 State

[USINT, Get, V]
This attribute is an indication of the present state of the device as represented by the state transition diagram.
The states are according to the following table:

Value	State	Meaning
0	Nonexistent	The device is without power.
1	Device Self Testing	The device is executing its self tests.
2	Standby	The device needs commissioning due to an incorrect or incomplete configuration.
3	Operational	The device is operating in a fashion that is normal for the device.
4	Major Recoverable Fault	The device has experienced a fault that is believed to be recoverable.
5	Major Unrecoverable Fault	The device has experienced a fault that is believed to be unrecoverable.

6 ... 254	Reserved	
255	Default Value 1	The Default Value shall be used in the Get_Attributes_All response if the attribute is not implemented.

NOTE
Please note that the nature of a **Major Unrecoverable Fault** could be such that it may not be accurately reflected by the **01-01-08 State** attribute.

01-01-09 Configuration Consistency Value

[UINT, Get, NV]
A product may automatically modify the *Configuration Consistency Value* whenever any non-volatile attribute is altered. A client node may, or may not, compare this value to a value within its own memory prior to system operation. The client node's behavior, upon detection of a mismatch, is vendor specific. The *Configuration Consistency Value* may be a CRC, incrementing count or any other mechanism. The only requirement is that if the configuration changes, the *Configuration Consistency Value* shall be different to reflect the change.
Default = 0000h

01-01-13 Protection Mode

[WORD, Get, V]
This is the current protection mode of the device.

Bit	Name	Definition	
0 ... 2	Implicit Protection Setting	It indicates the current implicit protection setting of the device.	Bit 0 = Implicit Protection (clear if Not Protected; set if Protected) Bits 1-2 = Reserved for future use
3	Explicit Protection Setting	It indicates the current explicit setting of the device	Explicit Protection (clear if Not Protected; set if Protected)
4 ... 15	Reserved	Reserved for future use and shall be set to zero.	

Two protection settings are defined: **Implicit Protection** (bits 0-2) and **Explicit Protection** (bit 3). The default value for both the Implicit and Explicit Setting is Not Protected. When both the Implicit and Explicit Protection Settings are Not Protected, the device shall accept all CIP explicit messages that are valid for the

device, subject to any object-specific or device-specific rules regarding state conflicts.
The Implicit Protection setting shall indicate that the device has entered a state in which it rejects explicit message requests that would disrupt its operation. The conditions under which a device enters the Implicit Protection setting are device-specific.
Default = 0

5.12.2 Class 02h: Message Router Object

Class Code	Object Class	Access	Nr. of Instances
02h	Message Router Object	Get	1

The Message Router object provides a messaging connection point through which a Client may address a service to any object class or instance residing in the encoder. It is responsible for dispatching service requests toward the addressed object class or object class instance. The EtherNet/IP protocol stack implements the Message Router object exclusively at class level.
In Lika encoders it is used internally to direct object requests.

5.12.2.1 Supported Class Services

The supported **Class Services** of the Message Router Object are:
0Eh = Get_Attribute_Single: used to read the value of an attribute.

5.12.2.2 Class Attributes

02-01 Revision

[UINT, Get, NV]
Object revision. The current value assigned to this attribute is 0001h.
Default = 0001h

02-02 Max Instance

[UINT, Get, NV]
The largest instance number of a created object in this class.
Default = 0001h

02-03 Number of Instances

[UINT, Get, NV]
The number of object instances in this class.
Default = 0001h

02-06 Maximum ID Number Class Attributes

[UINT, Get, NV]
The attribute ID number of the last class attribute of the class definition implemented in the encoder.
Default = 0007h

02-07 Maximum ID Number Instance Attributes

[UINT, Get, NV]
The attribute ID number of the last instance attribute of the class definition implemented in the encoder.
Default = 0000h

5.12.2.3 Supported Instance Services

The supported **Instance Services** of the Message Router Object are:
0Eh = Get_Attribute_Single: used to read the value of an attribute.
10h = Set_Attribute_Single: used to write connection class attribute value.

5.12.2.4 Instance Attributes

The EtherNet/IP protocol stack implements the Message Router object exclusively at class level. It does not provide any instances.

5.12.3 Class 04h: Assembly Object

Class Code	Object Class	Access	Nr. of Instances
04h	Assembly Object	Get	4

The Assembly Object binds attributes of multiple objects, which allows data to or from each object to be sent or received over a single connection. Assembly objects can be used to bind input data or output data. The terms ”input” and ”output” are defined from the network’s point of view. An input will produce data on the network and an output will consume data from the network.
Assembly objects instances are static: assemblies with member lists defined by the open device profile or vendor specific device profile. The Instance number, number of members, and member list are fixed.

5.12.3.1 Supported Class Services

The supported **Class Services** of the Assembly Object are:
0Eh = Get_Attribute_Single: used to read the value of an attribute.

5.12.3.2 Class Attributes

04-01 Revision

[UINT, Get, NV]
Object revision. The current value assigned to this attribute is 0002h.
Default = 0002h

04-02 Max Instance

[UINT, Get, NV]
The largest instance number of a created object in this class.
Default = 0000h

04-03 Number of Instances

[UINT, Get, NV]
The number of object instances in this class.
Default = 0000h

04-06 Maximum ID Number Class Attributes

[UINT, Get, NV]
The attribute ID number of the last class attribute of the class definition implemented in the encoder.
Default = 0007h

04-07 Maximum ID Number Instance Attributes

[UINT, Get, NV]
The attribute ID number of the last instance attribute of the class definition implemented in the encoder.
Default = 0004h

5.12.3.3 Supported Instance Services

The supported **Instance Services** of the Assembly Object are:

0Eh = Get_Attribute_Single: used to read the value of an attribute.

10h = Set_Attribute_Single: used to write connection class attribute value.

18h = Get_Member: used to get a member of **04-01-02 Member List** instance attribute.

5.12.3.4 Supported connection types

Lika EtherNet/IP encoders support "Input only" connections.

Input Only connection

This type of connection is used to read data from the encoder without controlling the outputs. It does not depend on other connections.

Connection point O → T Assembly Object, instance C1h

Connection point T → O Assembly Object, instances 01h, 02h, 03h

T is the Target, i.e. the encoder

O is the Origin, i.e. the Master

Refer also to the "5.12.4 Class 06h: Connection Manager Object" section on page 103.

5.12.3.5 Instance Attributes

04-01-01 Number of Members in List

[UINT, Get, NV]

Number of members in List.

04-01-02 Member List

[ARRAY of STRUCT, Get, NV]

Member list.

04-01-03 Data

[ARRAY of BYTE, Get, NV]

Current process data snapshot.

04-01-04 Size

[ARRAY of BYTE, Get, NV]

Process data size in number of bytes.

04-01-300 Member Data List

Data of assembly members.

04-01-301 Parameter
Assembly parameter.

04-01-302 Status
Status of the assembly.

5.12.3.6 I/O Assembly Instances

The following table identifies the I/O Assembly instances, which are supported by the encoder device.

Instance ID	Attribute	Access	Description	Bits	Bytes
01h	03h	Get	**23-01-03 Position value 32 bit**	32	4
02h	03h	Get	**23-01-03 Position value 32 bit** & **23-01-69 Warning/Alarm Flags**	32 8	5
03h	03h	Get	**23-01-03 Position value 32 bit** & **23-01-18 Velocity Value**	32 32	8
6Eh	03h	Set/Get	**Configuration Assembly**	96	12

23-01-03 Position value 32 bit, **23-01-18 Velocity Value**, **23-01-69 Warning/Alarm Flags**, **23-01-0C Direction Counting Toggle**, **23-01-0E Scaling Function Control**, **23-01-10 Measuring Units per Span**, **23-01-11 Total Measuring Range 32 bit** and **23-01-19 Velocity Format** attributes are fully described in the "5.12.5 Class 23h: Position Sensor Object" section on page 105.

5.12.3.7 I/O Assembly Data Attribute Format

The I/O assembly data attributes have the format shown below.

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
01h	0	**23-01-03 Position value 32 bit**							(low byte)
	1								
	2								
	3								(high byte)
02h	0	**23-01-03 Position value 32 bit**							(low byte)
	1								

	2	(high byte)							
	3								
	4							23-01-31 Warning Flag	23-01-2E Alarm Flag
03h	0	(low byte)							
	1	23-01-03 Position value 32 bit							
	2								
	3	(high byte)							
	4	(low byte)							
	5	23-01-18 Velocity Value							
	6								
	7	(high byte)							

5.12.3.8 Configuration Assembly

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Attribute ID
6Eh	0	23-01-0C Direction Counting Toggle								0Ch
	1	23-01-0E Scaling Function Control								0Eh
	2	23-01-10 Measuring Units per Span							(low byte)	10h
	3									
	4									
	5								(high byte)	
	6	23-01-11 Total Measuring Range 32 bit							(low byte)	11h
	7									
	8									
	9								(high byte)	
	10	23-01-19 Velocity Format							(low byte)	19h
	11								(high byte)	

5.12.4 Class 06h: Connection Manager Object

Class Code	Object Class	Access	Nr. of Instances
06h	Connection Manager Object	Get	1

The Connection Manager Class allocates and manages the internal resources associated to both "I/O Messages" and "Explicit Messaging Connections".

For complete information refer to "THE CIP NETWORKS LIBRARY, Volume 1, Common Industrial Protocol (CIP™), Chapter 3: Communication Object Classes".

Six types of connection are supported:

1. Connection 1 = Input Only (1): **23-01-03 Position value 32 bit**
2. Connection 2 = Input Only (2): **23-01-03 Position value 32 bit** + **23-01-69 Warning/Alarm Flags**
3. Connection 3 = Input Only (3): **23-01-03 Position value 32 bit** + **23-01-18 Velocity Value**
4. Connection 4 = Input Only (1) + Configuration: **23-01-03 Position value 32 bit** + **Configuration Assembly**
5. Connection 5 = Input Only (2) + Configuration: **23-01-03 Position value 32 bit** + **23-01-69 Warning/Alarm Flags** + **Configuration Assembly**
6. Connection 6 = Input Only (3) + Configuration: **23-01-03 Position value 32 bit** + 23-01-18 Velocity Value + **Configuration Assembly**

Refer also to the previous "5.12.3 Class 04h: Assembly Object" section on page 99.

5.12.4.1 Supported Class Services

The supported **Class Services** of the Connection Manager Object are:
0Eh = Get_Attribute_Single: used to read the value of an attribute.

5.12.4.2 Class Attributes

06-01 Revision

[UINT, Get, NV]
Object revision. The current value assigned to this attribute is 0001h.
Default = 0001h

06-02 Max Instance

[UINT, Get, NV]
The largest instance number of a created object in this class.
Default = 0001h

06-03 Number of Instances

[UINT, Get, NV]
The number of object instances in this class.
Default = 0001h

06-06 Maximum ID Number Class Attributes

[UINT, Get, NV]
The attribute ID number of the last class attribute of the class definition implemented in the encoder.
Default = 0007h

06-07 Maximum ID Number Instance Attributes

[UINT, Get, NV]
The attribute ID number of the last instance attribute of the class definition implemented in the encoder.
Default = 0000h

5.12.4.3 Supported Instance Services

The supported **Instance Services** of the Connection Manager Object are:
0Eh = Get_Attribute_Single: used to read the value of an attribute.
10h = Set_Attribute_Single: used to write connection class attribute value.
4Eh = Forward_Close: used to close a connection.
54h = Forward_Open: used to open a new connection.

5.12.4.4 Instance Attributes

06-01-01 Open Requests

[UINT, Set, V]
It shows the number of Forward Open service requests received.
Default = 0000h

5.12.5 Class 23h: Position Sensor Object

Class Code	Object Class	Access	Nr. of Instances
23h	Position Sensor Object	Set/Get	1

The Position Sensor Object is meant to describe the attributes used by the device to calculate the transmitted position values. It contains all attributes for acyclic process data and for setting the encoder.

5.12.5.1 Supported Class Services

The supported **Class Services** of the Position Sensor Object are:

05h = Reset: resets all parameter values to the factory default values and saves them on flash memory. The following types of reset are defined:

0 = Power Cycle Reset — It emulates a power cycling of the encoder.

1 = Return to Factory Defaults Reset — It returns to the factory default configuration of the encoder parameters and communication link parameters and emulates a power cycling of the encoder.

NOTE

After executing a Return to Factory Defaults reset (type 1), if the DIP A dip switches are all set to 0, the encoder restarts using the IP address saved internally. If the DIP A dip switches are set to any value between 1 and 254, then the encoder restarts using the address 192.168.1."dip switch setting". For more information refer to page 40.

0Eh = Get_Attribute_Single: used to read connection class attribute value.

15h = Restore: restores all parameter values from flash memory and saves them.

16h = Save: saves all parameters to non-volatile memory.

5.12.5.2 Class Attributes

23-01 Revision

[UINT, Get, NV]

Object revision. The current value assigned to this attribute is 0002h.

Default = 0002h

5.12.5.3 Supported Instance Services

The supported **Instance Services** of the Position Sensor Object are:

01h = Get_Attribute_All: used to read the value of all attributes.

10h = Set_Attribute_Single: used to write connection class attribute value.

5.12.5.4 Instance Attributes

23-01-03 Position value 32 bit

[UDINT, Get, V]
This attribute represents the absolute position detected by the encoder conditioned by the scaling value attributes (see on page 107).

23-01-0 B Position Sensor type

[UINT, Get, NV]
This attribute specifies the type of device.
0001h: **singleturn** absolute rotary encoder;
0002h: **multiturn** absolute rotary encoder.
Default = 0001h (min. value 0001h, max. value 0001h) for singleturn encoders
Default = 0002h (min. value 0002h, max. value 0002h) for multiturn encoders

23-01-0 C Direction Counting Toggle

[BOOL, Set, NV]
This attribute defines whether the position value output by the encoder increases (count up information) when the encoder shaft rotates clockwise (0 = CW) or counter-clockwise (1 = CCW). If the attribute is set to 0, the absolute position value **increases** when the encoder shaft rotates **clockwise**; on the contrary, if the attribute is set to 1, the absolute position value **increases** when the encoder shaft rotates **counter-clockwise**. CW and CCW rotations are viewed from shaft end.
00: turning the shaft CW the position will increase;
01: turning the shaft CCW the position will increase.

NOTE
Please note that the value of the **23-01-0C Direction Counting Toggle** attribute also affects the sign of the velocity, see the **23-01-18 Velocity Value** attribute on page 112.

WARNING
Changing this value causes also the position calculated by the controller to be necessarily affected. Therefore it is mandatory to execute a new preset and save the attributes after setting this attribute.

NOTE
To know whether the counting is currently increasing or decreasing (in real time), you can read the bit 0 **Counting direction** of the **23-01-29 Operating Status** attribute, see on page 113.

Default = 00h (min. value 00h, max. value 01h)

23-01-0 E Scaling Function Control

[BOOL, Set, NV]

If this attribute is disabled (00 = OFF), the device uses the physical resolution (see the **23-01-2A Physical Resolution Span** and **23-01-2B Number of Spans** attributes) to calculate the absolute position value; if it is enabled (01 = ON, default), the device uses the custom resolution set next to the **23-01-10 Measuring Units per Span** and **23-01-11 Total Measuring Range 32 bit** attributes in compliance with the following relation:

$$\text{Transmitted position} = \frac{\text{23-01-10 Measuring Units per Span}}{\text{23-01-2A Physical Resolution Span}} * \text{real position} \leq \text{23-01-11 Total Measuring Range 32 bit}$$

NOTE

To know whether the **23-01-0E Scaling Function Control** is currently enabled, you can read the bit 1 **Scaling function** of the **23-01-29 Operating Status** attribute, see on page 113.

WARNING

Every time you enable the scaling function and/or change the scaling values (see the **23-01-10 Measuring Units per Span** and **23-01-11 Total Measuring Range 32 bit** attributes), then you are required to set the preset again (see the **23-01-13 Preset Value 32 bit** attribute) and finally save the new parameters (by means of the Class Service 16h, see on page 105; or by setting the bit 6 **Save Parameters** in the **23-01-68 Command Register** attribute to 1, see on page 121).

Default = 01h (min. value 00h, max. value 01h)

23-01-10 Measuring Units per Span

[UDINT, Set, NV]

WARNING

This attribute is active only if the **23-01-0E Scaling Function Control** attribute is enabled; otherwise it is ignored and the system uses the physical resolution values (**23-01-2A Physical Resolution Span** and **23-01-2B Number of Spans**) to calculate the position information.

This attribute sets the custom number of measuring steps per revolution that are output for the absolute singleturn position value.
If you enter an out-of-range value, the number of measuring units per revolution is forced to the physical singleturn resolution and the **23-01-2F**

Warnings attribute signals the error (see the bit 0 **Measuring Units per Span exceeded** in the **23-01-67 Wrong Parameters List** and the LEDs).

To avoid counting errors, check that:

$$\frac{\text{23-01-2 A Physical Resolution Span}}{\text{23-01-10 Measuring Units per Span}} = \text{integer value.}$$

Allowed values must be less than or equal to the physical singleturn resolution (**23-01-2A Physical Resolution Span**).

Default = 8,192 (min. = 1, max. = 8,192)

WARNING

When you set a new value next to the **23-01-10 Measuring Units per Span** attribute, please always check also the **23-01-11 Total Measuring Range 32 bit** attribute value and be sure that the resulting number of revolutions complies with the physical number of revolutions of the device (see the **23-01-2B Number of Spans** attribute).

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

23-01-10 Measuring Units per Span: 8,192

23-01-11 Total Measuring Range 32 bit = 33,554,432 = 8,192 (cpr) * 4,096 (rev.)

Let's set a new singleturn resolution, for instance: **23-01-10 Measuring Units per Span** = 360.

If we do not change the **23-01-11 Total Measuring Range 32 bit** value at the same time, we will get the following result:

$$\textbf{Number of revolutions} = \frac{\text{33,554,432 (23-01-11 Total Measuring Range 32 bit)}}{\text{360 (23-01-10 Measuring Units per Span)}} = 93{,}206.755\ldots$$

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 (see the **23-01-2B Number of Spans** attribute). When this happens, the **23-01-2F Warnings** attribute signals the error (see also the **23-01-67 Wrong Parameters List** attribute and the LEDs).

WARNING
Every time you change the value in this attribute then you are required to set the preset again (see the **23-01-13 Preset Value 32 bit** attribute) and finally save the new parameters (by means of the Class Service 16h, see on page 105; or by setting the bit 6 **Save Parameters** in the **23-01-68 Command Register** attribute to 1, see on page 121).

23-01-11 Total Measuring Range 32 bit

[UDINT, Set, NV]

WARNING
This attribute is active only if the **23-01-0E Scaling Function Control** attribute is enabled; otherwise it is ignored and the system uses the physical resolution values (**23-01-2A Physical Resolution Span** and **23-01-2B Number of Spans**) to calculate the position information.

This attribute sets a custom number of distinguishable steps over the total measuring range. The total resolution of the encoder results from the product of **23-01-10 Measuring Units per Span** by the required **Number of revolutions**. Allowed values are less than or equal to **23-01-2A Physical Resolution Span** * **23-01-2B Number of Spans**.
If you enter an out-of-range value, the total resolution is forced to the max. physical number of measuring steps over the total measuring range and the **23-01-2F Warnings** attribute signals the error (see the bit 1 **Total Measuring Range exceeded** in the **23-01-67 Wrong Parameters List** and the LEDs).

Default = 134,217,728 (min. = 1, max. = 134,217,728)

WARNING
When you set a new value next to the **23-01-11 Total Measuring Range 32 bit** attribute, please always check also the **23-01-10 Measuring Units per Span** attribute value and be sure that the resulting number of revolutions complies with the physical **23-01-2B Number of Spans** of the device.
Let's suppose that the encoder is programd as follows:
23-01-10 Measuring Units per Span: 8,192
23-01-11 Total Measuring Range 32 bit = 33,554,432 = 8,192 (cpr) * 4,096 (rev.)
Let's set a new total resolution, for instance: **23-01-11 Total Measuring Range 32 bit** = 360.

As the **23-01-11 Total Measuring Range 32 bit** must be greater than or equal to the **23-01-10 Measuring Units per Span**, the above setting is not allowed. When this happens, the **23-01-2F Warnings** attribute signals the error (see also the **23-01-67 Wrong Parameters List** attribute and the LEDs).

WARNING
Every time you change the value in this attribute, then you are required to set a new preset value (see the **23-01-13 Preset Value 32 bit** attribute) and finally save the new parameters (by means of the Class Service 16h, see on page 105; or by setting the bit 6 **Save Parameters** in the **23-01-68 Command Register** attribute to 1, see on page 121).

EXAMPLE

We install the multiturn encoder.

Its physical resolution is as follows (see the order code):

- Hardware counts per revolution: **23-01-2A Physical Resolution Span** = 13 bits = 8192 (2^{13})
- Hardware number of revolutions: **23-01-2B Number of Spans** = 14 bits = 16,384 (2^{14})
- Total hardware resolution: **23-01-2A Physical Resolution Span** * **23-01-2B Number of Spans** = 27 bits = 134,217,727 ($2^{13+14=27}$)

In the specific installation 2,048 counts/rev. * 1,024 turns are required:

- Enable the scaling function: **23-01-0E Scaling Function Control** attribute = "1"
- Counts per revolution: **23-01-10 Measuring Units per Span** = 2,048 (0000 0800h)
- Total resolution: **23-01-11 Total Measuring Range 32 bit** = 2,048 * 1,024 = 2,097,152 (0020 0000h)

NOTE
We suggest setting values which are power of 2 (2^n: 2, 4, …, 2048, 4096, 8192,…) to be set in the **23-01-10 Measuring Units per Span** and **23-01-11 Total Measuring Range 32 bit** attributes to avoid counting errors.

WARNING
If **23-01-10 Measuring Units per Span** and/or **23-01-11 Total Measuring Range 32 bit** values change, the **23-01-13 Preset Value 32 bit** must be set accordingly. A new preset operation is required.

23-01-13 Preset Value 32 bit

[UDINT, Set, NV]

This attribute 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 shaft. The chosen position will get the value set next to this attribute and all the previous and the following positions will get a value according to it. This function is useful, for example, when the zero position of the encoder and the zero position of the axis need to match. The preset value will be set and activated for the position of the encoder in the moment when the preset value is transmitted. We suggest setting the preset value when the encoder is in stop. The preset value is activated as soon as the value is set.

If you need to activate a value already set next to the **23-01-13 Preset Value 32 bit** in a different physical position of the encoder shaft, you can use the bit 0 **Activate Preset** in the **23-01-68 Command Register** attribute, see on page 120.

Default = 0 (min. = 0, max. = 134,217,727 *)

* See the NOTE below.

EXAMPLE

Let's take a look at the following example to better understand the preset function and the meaning and use of the related attributes: **23-01-13 Preset Value 32 bit** and **23-01-33 Offset Value signed 32 bit**.

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) + **23-01-13 Preset Value 32 bit** - **23-01-33 Offset Value signed 32 bit**.

If you never set the **23-01-13 Preset Value 32 bit** or performed the preset setting, then the transmitted value and the read position are necessarily the same as **23-01-13 Preset Value 32 bit** = 0 and **23-01-33 Offset Value signed 32 bit** = 0.

When you set the **23-01-13 Preset Value 32 bit** or execute the preset setting, the system saves the current encoder position in the **23-01-33 Offset Value signed 32 bit** attribute. It follows that the transmitted value and the **23-01-13 Preset Value 32 bit** are the same as **read position** - **23-01-33 Offset Value signed 32 bit** = 0; in other words, the value set next to the **23-01-13 Preset Value 32 bit** attribute 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 **23-01-13 Preset Value 32 bit** attribute 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") + **23-01-13 Preset Value 32 bit** (="50") - **23-01-33 Offset Value signed 32 bit** (="1000") = 50.
The following transmitted value will be:
Transmitted value = **read position** (="1001") + **23-01-13 Preset Value 32 bit** (="50") - **23-01-33 Offset Value signed 32 bit** (="1000") = 51.
And so on.

NOTE

- If the scaling function is disabled (**23-01-0E Scaling Function Control** attribute = 0), then the **23-01-13 Preset Value 32 bit** must be less than or equal to the "Total hardware resolution" - 1, i.e. (**23-01-2A Physical Resolution Span** * **23-01-2B Number of Spans**) - 1.
- If the scaling function is enabled (**23-01-0E Scaling Function Control** attribute = 1), then the **23-01-13 Preset Value 32 bit** must be lower than or equal to the **23-01-11 Total Measuring Range 32 bit** - 1.

WARNING

Check the value in the **23-01-13 Preset Value 32 bit** attribute and perform the preset operation every time you set a new **23-01-0C Direction Counting Toggle** or change the scaled values (**23-01-10 Measuring Units per Span** and/or **23-01-11 Total Measuring Range 32 bit**).

23-01-18 Velocity Value

[DINT, Get, V]
This attribute shows the current output speed value detected by the position encoder and calculated every 100 ms.
The value can be expressed in several engineering units according to the setting next to the following **23-01-19 Velocity Format** attribute. As a default the velocity value is expressed in counts per second (cps).

23-01-19 Velocity Format

[UINT, Set, NV]
This attribute defines the engineering unit for the velocity value (see the previous **23-01-18 Velocity Value** attribute).
1F04h = counts per second: number of steps per second; the minimum resolution is 10 cps (default);

1F05h = counts per millisecond: number of steps per millisecond; the minimum resolution is 1 cpms;
1F07h = counts per minute: number of steps per minute; the minimum resolution is 600 cpm;
1F0Eh = revolutions per second: number of revolutions per second; the minimum resolution is 1 rps;
1F0Fh = revolutions per minute: number of revolutions per minute; the minimum resolution is 1 rpm;
1F10h = revolutions per hour: number of revolutions per hour; the minimum resolution is 4 rph;
Default = 1F04h (min. = 1F04h, max. = 1F10h)

23-01-29 Operating Status

[BYTE, Get, V]
This attribute contains the operating status of the encoder according to definitions in the following table.

Bit	Function	bit = 0	bit = 1
0	**Counting direction**	Count up information with CW rotation	Count up information with CCW rotation
1	**Scaling function**	Disabled	Enabled
2 ... 7	not used		

Counting direction
It shows whether the **23-01-0C Direction Counting Toggle** attribute is set to 0 = CW = count up information when the encoder shaft rotates clockwise; or to 1 = CCW = count up information when the encoder shaft rotates counter-clockwise. If the bit is "=0" the CW option is currently set; if the bit is "=1" the CCW option is currently set. See the **23-01-0C Direction Counting Toggle** attribute on page 106.

Scaling function
It shows whether the scaling function is disabled or enabled. If the value is "=0" the scaling function is disabled (i.e. the system uses the physical resolution values -**23-01-2A Physical Resolution Span** and **23-01-2B Number of Spans**- to calculate the position information); if the value is "=1" the scaling function is enabled (i.e. the system uses the custom values -**23-01-10 Measuring Units per Span** and **23-01-11 Total Measuring Range 32 bit**- to calculate the position information). To disable / enable the scaling function you must set the **23-01-0E Scaling Function Control** attribute to 0 / 1, see on page 107.

23-01-2 A Physical Resolution Span

[UDINT, Get, NV]

WARNING
This attribute is active only if the **23-01-0E Scaling Function Control** attribute is set to "=0"; otherwise it is ignored and the system uses the custom resolution values (**23-01-10 Measuring Units per Span** and **23-01-11 Total Measuring Range 32 bit**) to calculate the position information.

This attribute is intended to show the number of physical distinguishable steps each turn provided by the hardware (physical singleturn resolution).
If you want to set a custom resolution see the **23-01-10 Measuring Units per Span** attribute.

Default = 8,192

23-01-2 B Number of Spans

[UINT, Get, NV]

WARNING
This attribute is active only if the **23-01-0E Scaling Function Control** attribute is set to "=0"; otherwise it is ignored and the system uses the custom resolution values (**23-01-10 Measuring Units per Span** and **23-01-11 Total Measuring Range 32 bit**) to calculate the position information.

This attribute is intended to show the number of physical distinguishable turns provided by the hardware (number of physical revolutions).
The **Total hardware resolution** results from **23-01-2A Physical Resolution Span** * **23-01-2B Number of Spans**.
If you want to set a custom number of turns see the **23-01-10 Measuring Units per Span** and **23-01-11 Total Measuring Range 32 bit** attributes.
Default = 16,384

23-01-2 C Alarms

[UINT, Get, V]
An alarm indicates that a malfunction has occurred which could lead to an incorrect position value. It is set when a bit indicating a fault is set to true (high). The alarm remains active until the alarm is cleared and the device is able to provide an accurate position value. When an alarm is active, also the **23-01-**

2E Alarm Flag attribute is set to 1. The attribute is defined according to the following table.
Refer also to the following 23-01-2D Supported Alarms attribute.

Bit	Function	bit = 0	bit = 1
0	Position error	Alarm not active	Alarm active
1	Diagnostic error	Alarm not active	Alarm active
2 ... 11	not used		
12	Flash memory error	Alarm not active	Alarm active
13 ... 15	not used		

Position error
Fault and malfunction of the encoder position measurement system or the measured value processing unit. This error causes an invalid position and speed actual value, it may be due to the hardware or the signal quality.

Diagnostic error
It warns about an error that is specified in the Vendor Specific bits 12 … 15, o nly the Flash memory error alarm is available.

Flash memory error
Internal error, it cannot be restored. The flash memory contains corrupted data; or maybe the flash memory is damaged.

23-01-2 D Supported Alarms

[WORD, Get, NV]
This attribute contains information on the supported alarms. Refer to the previous 23-01-2C Alarms attribute.

Bit	Function	bit = 0	bit = 1
0	Position error	Not supported	Supported
1	Diagnostic error	Not supported	Supported
2 ... 11	not used		
12	Flash memory error	Not supported	Supported
13 ... 15	not used		

Default = 1003h (= 0001 0000 0000 0011 = alarms at bits 0, 1, and 12 are supported and displayed next to the previous 23-01-2C Alarms attribute).

23-01-2 E Alarm Flag

[BOOL, Get, V]
When its value is "1", it indicates that a fault occurred and an alarm has been triggered, see the description of the supported alarms in the previous 23-01-2C Alarms attribute.

23-01-2 F Warnings

[UINT, Get, V]

The **23-01-2F Warnings** attribute indicates that tolerances for certain internal parameters of the encoder have been exceeded. It does not imply incorrect position values. The warning is cleared if the tolerances are again within normal parameters. When a warning is active, also the **23-01-31 Warning Flag** attribute is set to 1. The attribute is defined according to the following table. Refer also to the following **23-01-30 Supported Warnings** attribute.

Bit	Function	bit = 0	bit = 1
0 ... 12	not used		
13	**Errors in config parameters**	Warning not active	Warning active
14	**Position Warning**	Warning not active	Warning active
15	not used		

Errors in config parameters

An out-of-tolerance parameter has been set. For more details about the specific out-of-tolerance parameter refer to the **23-01-67 Wrong Parameters List** attribute, see on page 118.

Position Warning

Fault and malfunction of the encoder position measurement system or the measured value processing unit. This warning does not cause an invalid position and speed actual value, it may be due to the hardware or the signal quality.

23-01-30 Supported Warnings

[WORD, Get, NV]

This attribute contains information on the supported warnings. Refer to the previous **23-01-2F Warnings** attribute.

Bit	Function	bit = 0	bit = 1
0 ... 12	not used		
13	**Errors in config parameters**	Not supported	Supported
14	**Position Warning**	Not supported	Supported
15	not used		

Default = 6000h (= 0110 0000 0000 0000 = warnings at bits 13 and 14 are supported and displayed next to the previous **23-01-2F Warnings** attribute).

23-01-31 Warning Flag

[BOOL, Get, V]

The attribute indicates whether any of the defined warnings are active (1) or not (0).

23-01-32 Operating Time

[UDINT, Get, NV]

This attribute contains the information on the operating time and is incremented as long as the encoder is powered. It is expressed in tenths of an hour. This attribute is not used currently.

23-01-33 Offset Value signed 32 bit

[DINT, Get, NV]

The **23-01-33 Offset Value signed 32 bit** attribute is calculated by the preset function and shifts the **23-01-03 Position value 32 bit** attribute with the calculated value. It is stored automatically by the device and can be read from the encoder for diagnostic purposes. To zero set the value in this attribute you must upload the factory default values (see the Class Service 15h on page 105 and the **Restore Parameters to Defaults** bit option in the **23-01-68 Command Register** attribute).

23-01-64 Application FW Version

[UDINT, Get, NV]

This attribute contains the version of the firmware that is currently installed in the device.

The meaning of the 32 bits in the attribute is as follows:

Bit	31 … 16	15 … 0
	Major version	Minor version

For example, the value 0001 0001 hex in hexadecimal notation corresponds to the binary representation 0000 0000 0000 0001 0000 0000 0000 0001 and has to be interpreted as: firmware version 1.1.

Default = Device dependent

23-01-65 Hardware Version

[UDINT, Get, NV]
This attribute contains the version of the electronics the device is currently equipped with.
The meaning of the 32 bits in the attribute is as follows:

Bit	31 … 16	15 … 0
	Major version	Minor version

For example, the value 0002 0001 hex in hexadecimal notation corresponds to the binary representation 0000 0000 0000 0002 0000 0000 0000 0001 and has to be interpreted as: hardware version 2.1.

Default = Device dependent

23-01-66 Reserved

This attribute is not used and reserved for future use.

23-01-67 Wrong Parameters List

[UINT, Get, NV]
The operator has entered invalid data and the **Errors in config parameters** warning in the **23-01-2F Warnings** attribute has been triggered. This variable is meant to show (bit value = HIGH) the list of the wrong parameters, according to the following table.
Please note that the normal work status can be restored only after having set proper values.

Bit	Function	bit = 0	bit = 1
0	**Measuring Units per Span exceeded**	Warning not active	Warning active
1	**Total Measuring Range exceeded**	Warning not active	Warning active
2	**Preset Value exceeded**	Warning not active	Warning active
3	**Offset Value exceeded**	Warning not active	Warning active
4 … 15	not used		

Byte 0

Measuring Units per Span exceeded

bit 0 — Wrong data has been set next to the **23-01-10 Measuring Units per Span** attribute. The tolerances for the parameter have been exceeded. Set proper values to restore the normal work condition. The warning is cleared if the tolerances are again within normal parameters.

Total Measuring Range exceeded

bit 1 — Wrong data has been set next to the **23-01-11 Total Measuring Range 32 bit** attribute. The tolerances for the parameter have been exceeded. Set proper values to restore the normal work condition. The warning is cleared if the tolerances are within normal parameters again.

Preset Value exceeded

bit 2 — Wrong data has been set next to the **23-01-13 Preset Value 32 bit** attribute. The tolerances for the parameter have been exceeded. Set proper values to restore the normal work condition. The warning is cleared if the tolerances are within normal parameters again.

Offset Value exceeded

bit 3 — Wrong data has been set next to the **23-01-13 Preset Value 32 bit** attribute and the calculated **23-01-33 Offset Value signed 32 bit** is out-of-tolerance. The tolerances for the parameter have been exceeded. Set proper values to restore the normal work condition. The warning is cleared if the tolerances are within normal parameters again.

bits 4 … 7 — Not used

Byte 1 — **Not used**

23-01-68 Command Register

[BYTE, Set, V]

This attribute contains some commands to be sent in real time to the encoder in order to manage it.

Bit	Function	bit = 1	bit = 0
0	**Activate Preset**	Activate	Finalize
1 … 5	not used		
6	**Save Parameters**	Activate	Finalize
7	**Restore Parameters to Defaults**	Activate	Finalize

Byte 0

Activate Preset

bit 0

This command is used to activate a preset value in the encoder. As soon as the command is sent, the position value which is transmitted for the current encoder position is the one set next to the **23-01-13 Preset Value 32 bit** attribute and all the previous and following positions will get a value according to it. The operation is performed at each rising edge of the bit, i.e. each time this bit is switched from logic level low ("0") to logic level high ("1"). Then the bit must be switched back to logic level low ("0") to finalize the command. When the command is sent, the current encoder position is saved temporarily in the **23-01-33 Offset Value signed 32 bit** attribute. For any further information on the preset function and the meaning and use of the related attributes and commands **23-01-13 Preset Value 32 bit**, **23-01-33 Offset Value signed 32 bit**, and **Activate Preset** refer to page 111.

NOTE

Please note that as soon as the preset value is entered next to the **23-01-13 Preset Value 32 bit** attribute, it is also automatically activated, so you do not need to use this command. Use the **Activate Preset** command to activate a preset value that has been already set next to the **23-01-13 Preset Value 32 bit** attribute and you want to set for a different shaft position.

WARNING

To save permanently the current encoder position in the **23-01-33 Offset Value signed 32 bit** attribute, please execute the **Save Parameters** command. Should the power supply be turned off without saving data, the **23-01-33**

Offset Value signed 32 bit that has not been saved will be lost!

bits 1 … 5 Not used

Save Parameters

bit 6 This function allows to save all parameters on non-volatile memory. Data is saved on non-volatile memory at each rising edge of the bit; in other words, data save is performed each time this bit is switched from logic level low ("0") to logic level high ("1"). Then the bit must be switched back to logic level low ("0") to finalize the command.

NOTE
Always save the new values after setting in order to store them in the non-volatile memory permanently.
Should the power supply be turned off all data that has not been saved previously will be lost!

NOTE
To save the new values in the non-volatile memory permanently you can use also the Class Service 16h, see on page 105.

Restore Parameters to Factory Defaults

bit 7 This function allows the operator to restore all parameters to default values. The default values are set at the factory by Lika Electronic engineers to allow the operator to run the device for standard operation in a safe mode. This function can be useful, for instance, to restore the factory values in case the encoder is set incorrectly and you are not able to resume the proper operation.
Default parameters are restored at each rising edge of the bit; in other words, the default parameters uploading operation is performed each time this bit is switched from logic level low ("0") to logic level high ("1"). Then the bit must be switched back to logic level low ("0") to finalize the command. The complete list of machine data and relevant default parameters preset by Lika Electronic engineers is available on page 184.

WARNING
The execution of this command causes all parameters which have been set previously to be overwritten!

23-01-69 Warning/Alarm Flags

[BYTE, Get, NV]
This attribute is used in conjunction with the I/O assembly data attributes, refer to the "5.12.3.7 I/O Assembly Data Attribute Format" section on page 101.
Its value is **00h** ($0000\ 0000_2$) when neither warnings nor alarms are active.
Its value is **01h** ($0000\ 0001_2$) when alarms are active.
Its value is **02h** ($0000\ 0010_2$) when warnings are active.
Its value is **03h** ($0000\ 0011_2$) when both warnings and alarms are active.
See the byte 4 of Instance 02h on page 101.

23-01-6A Encoder Serial Number

[UDINT, Get, NV]
This attribute contains the serial number of the encoder assigned by the manufacturer. This is not the same as the Identity Object's serial number (see the **01-01-06 Serial number** attribute on page 94) which is used to uniquely identify the device in the network environment. It can be read in the label applied to the device enclosure.
The meaning of the 32 bits in the attribute is as follows:

Bit	31 … 24	23 … 16	15 … 0
	YoP	WoP	Serial number

YoP: year of production.
Wop: week of production.
Serial number: serial number in ascending order.

Default = Device dependent

5.12.6 Class 43h: Time Sync Object

Class Code	Object Class	Access	Nr. of Instances
43h	Time Sync Object	Get	1

The Time Sync Object provides a CIP interface to the IEEE 1588 (IEC 61588) Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems, commonly referred to as the Precision Time Protocol (PTP). Refer to the IEEE 1588 (IEC 61588) Standard for additional details.
Any device supporting CIP Sync shall provide a single instance (Instance 1) of the Time Sync Object.
The object provides attributes and services to:

1. Get clock status and properties such as synchronized state, current offset to master, and grandmaster identity.
2. Access PTP clock management functions such as clock priority.
3. Access the PTP network of devices via native PTP management messages.

5.12.6.1 Supported Class Services

The supported **Class Services** of the Time Sync Object are:
0Eh = Get_Attribute_Single: used to read the value of an attribute.

5.12.6.2 Class Attributes

43-01 Revision
[UINT, Get, NV]
Object revision. The current value assigned to this attribute is 0003h.
Default = 0003h

43-02 Max Instance
[UINT, Get, NV]
The largest instance number of a created object in this class.
Default = 0001h

43-03 Number of Instances
[UINT, Get, NV]
The number of object instances in this class.
Default = 0001h

43-06 Maximum ID Number Class Attributes
[UINT, Get, NV]
The attribute ID number of the last class attribute of the class definition implemented in the encoder.
Default = 0007h

43-07 Maximum ID Number Instance Attributes

[UINT, Get, NV]
The attribute ID number of the last instance attribute of the class definition implemented in the encoder.
Default = 0300h

5.12.6.3 Supported Instance Services

The supported **Instance Services** of the Time Sync Object are:
03h = Get_Attribute_List: the Get_Attribute_List service returns the contents of the selected attributes of the specified object class or instance.
04h = Set_Attribute_List: the Set_Attribute_List service sets the contents of the selected attributes of the specified object class or instance.
0Eh = Get_Attribute_Single: used to read the value of an attribute.
10h = Set_Attribute_Single: used to write connection class attribute value.

5.12.6.4 Instance Attributes

43-01-05 Offset from Master

[LINT, Get, V]
Offset between local clock and Master clock.
Default = 0000h

43-01-06 Max Offset from Master

[ULINT, Set, V]
Maximum offset between local clock and Master clock since last reset of this value.
Default = 0000h

43-01-07 Mean path delay to Master

[LINT, Get, V]
Mean path delay to Master.
Default = 0000h

43-01-08 Grandmaster Clock Info

[Struct of, Get, V]
43-01-08 Grandmaster Clock Info, **43-01-09 Parent Clock Info**, and **43-01-0A Local Clock Info** specify clock property information for the Grandmaster, Parent and Local PTP clock respectively. The data is extracted from the PTP data sets maintained by the PTP device.
The **Clock Identity** provides a unique identifier for the clock. The clock **Time Source**, **Clock Class**, **Offset Scaled Log Variance**, and other attributes provide additional information about the properties of the clock.

Default = 0000h

Clock Identity
[USINT] It specifies the unique identifier for the clock. The format of the identifier depends on the network protocol. Ethernet encodes the MAC address into the identifier.

Clock Class
[UINT] It specifies the class of the clock quality. The clock class represents a relative measure of the clock quality used by the Best Master algorithm to determine the grandmaster. The class is a value between 0 and 255, with 0 as the best clock. These are the values most likely to be used in CIP Sync.

Time Accuracy
[UINT] It specifies the expected absolute accuracy of the clock relative to the PTP epoch. **Time Accuracy** is the accuracy measure of clock quality used by the Best Master algorithm to determine the grandmaster. The accuracy is specified as a graduated scale starting at 25 nsec and ending at greater than 10 seconds or unknown. A GPS time source will have an accuracy of approximately 250 nanoseconds. A HAND set clock will typically have accuracy less than 10 seconds. The lower the accuracy value, the better the clock.

Offset Scaled Log Variance
[UINT] It specifies a measure of the inherent stability properties of the clock. **Offset Scaled Log Variance** is the variance measure of clock quality used by the Best Master algorithm to determine the grandmaster. The value is represented in offset scaled log units. The lower the variance, the better the clock.

Current UTC Offset
[UINT] It specifies the current UTC offset in seconds from International Atomic Time (TAI) of the clock. As of 0 hours 1 January 2006 UTC, the offset was 33 seconds.

Time Property Flags
[WORD] It specifies the time property flags of the clock.

Time Source
[UINT] It specifies the primary time source of the clock.

Priority 1
[UINT] **Priority 1** and **Priority 2** values specify the relative priority of the grandmaster clock to other clocks in the system.

Priority 2
[UINT] **Priority 1** and **Priority 2** values specify the relative priority of the grandmaster clock to other clocks in the system.

43-01-09 Parent Clock Info

[Struct of, Get, V]

43-01-08 Grandmaster Clock Info, **43-01-09 Parent Clock Info**, and **43-01-0A Local Clock Info** specify clock property information for the Grandmaster, Parent and Local PTP clock respectively. The data is extracted from the PTP data sets maintained by the PTP device.

The **Clock Identity** provides a unique identifier for the clock. The **Observed Offset Scaled Log Variance**, and other attributes provide additional information about the properties of the clock.

Default = 0000h

Clock Identity

[USINT] It specifies the unique identifier for the clock. The format of the identifier depends on the network protocol. Ethernet encodes the MAC address into the identifier.

Port Number

[UINT] It specifies the port number of the port identity.

Observed Offset Scaled Log Variance

[UINT] It specifies an estimated measure of the parent clock's variance as observed by the Slave clock.

Observed Phase Change Rate

[UDINT] It specifies an estimated measure of the parent clock's drift as observed by the Slave clock.

43-01-0 A Local Clock Info

[Struct of, Get, V]

43-01-08 Grandmaster Clock Info, **43-01-09 Parent Clock Info**, and **43-01-0A Local Clock Info** specify clock property information for the Grandmaster, Parent and Local PTP clock respectively. The data is extracted from the PTP data sets maintained by the PTP device.

The **Clock Identity** provides a unique identifier for the clock. The clock **Time Source**, **Clock Class**, **Offset Scaled Log Variance**, and other attributes provide additional information about the properties of the clock.

Default = 0000h

Clock Identity

[USINT] It specifies the unique identifier for the clock. The format of the identifier depends on the network protocol. Ethernet encodes the MAC address into the identifier.

Clock Class

[UINT] It specifies the class of the clock quality. The clock class represents a relative measure of the clock quality used by the Best Master algorithm to

determine the grandmaster. The class is a value between 0 and 255, with 0 as the best clock. These are the values most likely to be used in CIP Sync.

Time Accuracy
[UINT] It specifies the expected absolute accuracy of the clock relative to the PTP epoch. **Time Accuracy** is the accuracy measure of clock quality used by the Best Master algorithm to determine the grandmaster. The accuracy is specified as a graduated scale starting at 25 nsec and ending at greater than 10 seconds or unknown. A GPS time source will have an accuracy of approximately 250 nanoseconds. A HAND set clock will typically have accuracy less than 10 seconds. The lower the accuracy value, the better the clock.

Offset Scaled Log Variance
[UINT] It specifies a measure of the inherent stability properties of the clock. **Offset Scaled Log Variance** is the variance measure of clock quality used by the Best Master algorithm to determine the grandmaster. The value is represented in offset scaled log units. The lower the variance, the better the clock.

Current UTC Offset
[UINT] It specifies the current UTC offset in seconds from International Atomic Time (TAI) of the clock. As of 0 hours 1 January 2006 UTC, the offset was 33 seconds.

Time Property Flags
[WORD] It specifies the time property flags of the clock.

Time Source
[UINT] It specifies the primary time source of the clock.

43-01-0 B Number of ports
[UINT, Get, V]
It specifies the number of PTP ports on the device. PTP Ordinary clocks have one port. PTP Boundary and Transparent clocks have more than one port. A hybrid clock that contains both an ordinary clock and an end-to-end transparent clock has a value of one (1) for this attribute.
Default = 0001h

43-01-0 C Port State Info
[Struct of, Get, V]
It specifies the current state of each PTP port on the device.
Default = 0003h (Disabled)

43-01-0 D Port Enable Cfg

[Struct of, Set, NV]
It specifies the port enable configuration of each port on the device. The Port Enable member is set to 1 if the port is enabled and 0 if the port is disabled. The default value is Enabled.
Default = 0001h (Enabled)

43-01-0 E Port Log Announce Interval Cfg

[Struct of, Set, NV]
It specifies the PTP announce interval between successive "Announce" messages issued by a Master clock on each PTP port of the device. The units of the **43-01-0E Port Log Announce Interval Cfg** member are log base 2 seconds.
Default = 0000h

43-01-0 F Port Log Sync Interval Cfg

[Struct of, Set, NV]
It specifies the PTP sync interval between successive "Sync" messages issued by a Master clock on each PTP port of the device. The units of the **43-01-0F Port Log Sync Interval Cfg** member are log base 2 seconds.
Default = 0000h

43-01-12 Domain Number

[USINT, Set, NV]
It specifies the PTP clock domain.
Default = 0000h

43-01-13 Clock Type

[WORD, Get, V]
The value of **43-01-13 Clock Type** shall indicate the PTP functions that the node supports. A value of one (1) for the bit indicates that the capability applies to this node.
Default = 0000h

NOTE
More than one bit may be set - for example an ordinary clock combined with an end-to-end transparent clock. When bit 8 is set, bit 7 shall also be set, as Slave Only functionality is a subset of the Ordinary Clock function.

43-01-14 Manufacturer Identity

[USINT, Get, V]
It specifies the manufacture identity of the clock. The first 3 octets specify the IEEE OUI (Organization Unique ID) for the manufacturer. The last octet is reserved.
Default = 0000h

43-01-15 Product Description

[Struct of, Get, V]
It specifies the product description of the device that contains the clock. The format is:

- the name of manufacturer of the device followed by a semicolon;
- the model number of the device followed by a semicolon;
- the serial number.

For example: Lika Electronic;EXM58;123456
The format is UTF-8 Unicode. The maximum number of symbols is 64. The **Size** field of the data type is the total number of bytes for the **Description** field. Convert the number of symbols into bytes.

Size
[UDINT] It specifies the size of the product description.

Description
[ARRAY of USINT] Description.

43-01-16 Revision Data

[Struct of, Get, V]
It specifies the revision data of the device that contains the clock. The format is:

- the hardware revision of the clock followed by a semicolon;
- the firmware revision of the clock followed by a semicolon;
- the software revision of the clock.

For example: 1.2;2.3;3.0.1
The format is UTF-8 Unicode. The maximum number of symbols is 32. The **Size** field of the data type is the total number of bytes for the **Revision** field. Convert the number of symbols into bytes. Subfields that do not apply may be null or blank (e.g. "1.2;2.3;" or ";;3.4").

Size
[UDINT] It specifies the size of the revision data.

Revision
[ARRAY of USINT] Revision.

43-01-17 User Description

[Struct of, Get, V]
It specifies the user description of the device that contains the clock. The format is:

- a user defined name or description of the device followed by a semicolon;
- a user defined physical location of the device.

For example: Encoder-1;Axis 3
The format is UTF-8 Unicode. The maximum number of symbols is 128. The **Size** field of the data type is the total number of bytes for the **Description** field. Convert the number of symbols into bytes.

Size
[UDINT] It specifies the size of the user description.

Description
[ARRAY of USINT] Description.

43-01-18 Port Profile Identity Info

[Struct of, Get, V]
It specifies the PTP profile of each port of the device. The attribute returns the profile identity of the currently active profile. The profile identity is contained in the first 6 bytes of the array. The last two octets should be set to zero.
Default = 00-21-6C-00-01-00 (Profile identifier)

43-01-19 Port Physical Address Info

[Struct of, Get, V]
It specifies the Physical Protocol and Physical Address of each port of the device. The Physical Protocol is an array of ASCII characters and is limited to a maximum of 16 characters. Unused array characters are zero-filled. For example a Physical Protocol of "IEEE 802.3" = 49 45 45 45 20 38 30 32 2E 33 00 00 00 00 00 00.
The Physical Address is an array of bytes and is limited to a maximum of 16 bytes. Unused array bytes are zero-filled. For example a Physical Address for MAC address 01 02 03 04 05 06 = 01 02 03 04 05 06 00 00 00 00 00 00 00 00 00 00 and has a size of 6 bytes.
Default = according to the MAC address of the device

43-01-1 A Port Protocol Address Info

[Struct of, Get, V]
It specifies the network protocol and protocol address of each port of the device (e.g. IP address). The Network Protocol specifies the protocol for the network. The maximum number of bytes for the Port Protocol Address is 16. Unused array bytes are zero-filled. For example a Network Protocol of UDP/IPv4 = 0001 and an IP protocol address of 192.168.1.2 = C0 A8 01 02 00 00 00 00 00 00 00 00 00 00 00 00 and has a size of 4 bytes.
Default = according to the IP address of the device

43-01-1 B Steps Removed

[UINT, Get, V]
It specifies the number of communication paths traversed between the local clock and the grandmaster clock.
Default = 0000h

43-01-1 C System Time and Offset

[Struct of, Get, V]
It specifies the System Time in microseconds and the Offset to the local clock value. The responding device will return the current System Time and Offset. For complete information on the CIP Sync Clock Model refer to the ODVA documentation.
Default = 0000h

43-01-1 D Associated Interface Objects

[Struct of, Get, NV]
PTP port numbers must start with 1 and be sequential, according to IEEE1588. When it is not possible for the PTP and CIP port numbers to be the same or when the PTP port is associated with a physical port, the device needs to identify the associations. The **43-01-1D Associated Interface Objects** attribute specifies for each PTP port whether it is associated with a CIP port or a physical port.
If the PTP Port is associated with a CIP port, the Associated Object shall specify the object instance that represents the CIP port. That is, the Associated Object shall specify the Port object instance (20 F4 24 xx, where xx is the Port object instance).
If the PTP Port is associated with a physical Ethernet port (e.g. PRP case), the Associated Object shall specify the Ethernet Link object instance (20 F6 24 xx, where xx is the Ethernet Link object instance).
Default = CIP path to Ethernet Link object

43-01-300 Sync Parameters

[Struct of, Get, NV]

The **43-01-300 Sync Parameters** attribute is not available through the Get_Attribute_List and the Set_Attribute_List services.

It controls synchronization-related parameters. These parameters are used to adjust intervals and offsets of the hardware synchronization signals Sync 0 and Sync 1.

The Sync 0 signal is the interrupt that the host application will receive in order to retrieve the current system time. On each event, the EtherNet/IP stack writes the current system time into the extended data area of the Dual Port Memory interface.

NOTE

Currently, only Sync 0 can be used.

Here follows the description of the **43-01-300 Sync Parameters** attribute.

ulSync0Interval

[UINT32] Sync 0 Interval expressed in nanoseconds. This parameter specifies the interval of the Sync 0 signal in nanoseconds. The value 0 means the signal is deactivated. The starting point of the Sync 0 signal is dependent on the Sync 0 Offset (see the **ulSync0Offset** parameter).

Default = 500000000 (0, 10000 … 999999999)

ulSync0Offset

[UINT32] Sync 0 Offset in nanoseconds. This parameter specifies a nanosecond offset for the Sync 0 signal relative to the system time (Time of the Sync Master).

Default = 0 (smaller than **ulSync0Interval**)

ulSync1Interval

[UINT32] Sync 1 Interval expressed in nanoseconds. This parameter specifies the interval of the Sync 1 signal in nanoseconds. The value 0 means the signal is deactivated. The starting point of the Sync 1 signal is dependent on the Sync 1 Offset (see the **ulSync1Offset** parameter).

Default = 500000000 (0, 10000 … 999999999)

ulSync1Offset

[UINT32] Sync 1 Offset expressed in nanoseconds. This parameter specifies a nanosecond offset for the Sync 1 signal relative to the system time (Time of the Sync Master).

Default = 150 (smaller than **ulSync1Interval**)

ulPulseLength

[UINT32] Pulse length of the trigger signals expressed in microseconds.

Default = 4 (1 … 500 AND smaller than the minimum of the values **ulSync0Interval** and **ulSync1Interval**, when converted to microseconds)

5.12.7 Class 47h: Device Level Ring (DLR) Object

Class Code	Object Class	Access	Nr. of Instances
47h	Device Level Ring (DLR) Object	Get	1

The Device Level Ring (DLR) Object provides the configuration and status information interface for the DLR protocol. The DLR protocol is a layer 2 protocol that enables the use of an Ethernet ring topology. The DLR Object provides the CIP application-level interface to the protocol. The DLR protocol is fully specified in Chapter 9 of the publication "THE CIP NETWORKS LIBRARY, Volume 2, EtherNet/IP Adaptation of CIP".

5.12.7.1 Supported Class Services

The supported **Class Services** of the Device Level Ring (DLR) Object are:
0Eh = Get_Attribute_Single: used to read the value of an attribute.

5.12.7.2 Class Attributes

47-01 Revision

[UINT, Get, NV]
Object revision. The current value assigned to this attribute is 0003h.
Default = 0003h

47-02 Max Instance

[UINT, Get, NV]
The largest instance number of a created object in this class.
Default = 0001h

47-03 Number of Instances

[UINT, Get, NV]
The number of object instances in this class.
Default = 0001h

47-06 Maximum ID Number Class Attributes

[UINT, Get, NV]
The attribute ID number of the last class attribute of the class definition implemented in the encoder.
Default = 0007h

47-07 Maximum ID Number Instance Attributes

[UINT, Get, NV]
The attribute ID number of the last instance attribute of the class definition implemented in the encoder.
Default = 000Ch

5.12.7.3 Supported Instance Services

The supported **Instance Services** of the Device Level Ring (DLR) Object are:
01h = Get_Attribute_All: used to read the value of all attributes.
0Eh = Get_Attribute_Single: used to read the value of an attribute.

5.12.7.4 Instance Attributes

47-01-01 Network Topology

[USINT, Get, V]
It indicates the current network topology mode. A value of "0" indicates "Linear" topology; a value of "1" indicates "Ring" topology.
Default = 0 (Linear)

47-01-02 Network Status

[USINT, Get, V]
This attribute provides current status of the network based on the device's view of the network, according to the following table.

Network Status value	Description
0	**Normal operation** in both Ring and Linear Network Topology modes.
1	**Ring Fault**. A ring fault has been detected. Valid only when **47-01-01 Network Topology** is "1" = Ring.
2	**Unexpected Loop Detected**. A loop has been detected in the network. Valid only when **47-01-01 Network Topology** is "0" = Linear.
3	**Partial Network Fault**. A network fault has been detected in one direction only. Valid only when **47-01-01 Network Topology** is "1" = Ring and the node is the active ring supervisor.
4	**Rapid Fault/Restore Cycle**. A series of rapid ring fault/restore cycles has been detected. Similar to the **Partial Network Fault** status (3), the supervisor remains in a state with forwarding blocked on its ring ports. The condition must be cleared explicitly via the "Clear_Rapid_Faults" service.

Default = 0 (Normal operation)

47-01-0 A Active Supervisor Address

[Struct of, Get, V]
This attribute contains the IP address (IPv4) and/or the Ethernet MAC address of the active ring supervisor. The initial values of IP address and Ethernet MAC address shall be 0, until the active ring supervisor is determined.

47-01-0 C Capability Flags

[DWORD, Get, NV]

The **47-01-0C Capability Flags** attribute describes the DLR capabilities of the device, according to the following table.

Bit(s)	Called	Definition
0	**Announce-based Ring Node**	It sets if device's ring node implementation is based on processing of Announce frames. Bits 0 and 1 are mutually exclusive. Exactly only one of these bits shall be set in the attribute value that a device reports.
1	**Beacon-based Ring Node**	It sets if device's ring node implementation is based on processing of Beacon frames. Bits 0 and 1 are mutually exclusive. Exactly only one of these bits shall be set in the attribute value that a device reports.
2 ... 4	Reserved	Reserved, shall be 0
5	**Supervisor Capable**	It sets if device is capable of providing the supervisor function.
6	**Redundant Gateway Capable**	It sets if device is capable of providing the redundant gateway function.
7	**Flush_Table Frame Capable**	It sets if device is capable of supporting the Flush_Tables frame.
8 ... 31	Reserved	Reserved, shall be 0

Default = 0082h = **Beacon-based Ring Node** + **Flush_Table Frame Capable**

5.12.8 Class 48h: Quality of Service (QoS) Object

Class Code	Object Class	Access	Nr. of Instances
48h	Quality of Service (QoS) Object	Get	1

The Quality of Service (QoS) Object is used to treat traffic streams with different relative priorities or other delivery characteristics. Standard QoS mechanisms include IEEE 802.1D/Q (Ethernet frame priority) and Differentiated Services (DiffServ) in the TCP/IP protocol suite.
The QoS Object provides a means to configure certain QoS-related behaviors in EtherNet/IP devices.
The QoS Object is required for devices that support sending EtherNet/IP messages with non-zero DiffServ code points (DSCP), or sending EtherNet/IP messages in 802.1Q tagged frames.

5.12.8.1 Supported Class Services

The supported **Class Services** of the Quality of Service (QoS) Object are:
0Eh = Get_Attribute_Single: used to read the value of an attribute.

5.12.8.2 Class Attributes

48-01 Revision

[UINT, Get, NV]
Object revision. The current value assigned to this attribute is 0001h.
Default = 0001h

48-02 Max Instance

[UINT, Get, NV]
The largest instance number of a created object in this class.
Default = 0001h

48-03 Number of Instances

[UINT, Get, NV]
The number of object instances in this class.
Default = 0001h

48-06 Maximum ID Number Class Attributes

[UINT, Get, NV]
The attribute ID number of the last class attribute of the class definition implemented in the encoder.
Default = 0007h

48-07 Maximum ID Number Instance Attributes

[UINT, Get, NV]
The attribute ID number of the last instance attribute of the class definition implemented in the encoder.
Default = 0008h

5.12.8.3 Supported Instance Services

The supported **Instance Services** of the Quality of Service (QoS) Object are:
0Eh = Get_Attribute_Single: used to read the value of an attribute.
10h = Set_Attribute_Single: used to write connection class attribute value.

5.12.8.4 Instance Attributes

48-01-01 802.1Q Tag Enable

[USINT, Set, NV]
This attribute enables (1) or disables (0) sending 802.1Q frames on CIP and IEEE 1588 messages. When the attribute is enabled, the device shall send 802.1Q frames for all CIP and IEEE 1588 messages.
A value of 1 shall indicate enabled. A value of 0 shall indicate disabled. The default value for the attribute shall be 0. A change to the value of the attribute shall take effect the next time the device restarts.
Default = 0000h

NOTE
Devices shall always use the corresponding DSCP values regardless of whether 802.1Q frames are enabled or disabled.

48-01-02 DSCP PTP Event

[USINT, Set, NV]
Attributes 2 through 8 contain the DSCP values that shall be used for the different types of EtherNet/IP traffic.
Refer to the ODVA documentation for the format of the DSCP value within the IP header. Since the DSCP field has a size of 6 bits, the valid range of values for these attributes is 0-63. Note that the DSCP value, if placed directly in the ToS field in the IP header, must be shifted left 2 bits.
This attribute contains the DSCP value for the PTP Event frames.
Default = 003Bh

48-01-03 DSCP PTP General

[USINT, Set, NV]
Attributes 2 through 8 contain the DSCP values that shall be used for the different types of EtherNet/IP traffic.
Refer to the ODVA documentation for the format of the DSCP value within the IP header. Since the DSCP field has a size of 6 bits, the valid range of values for these attributes is 0-63. Note that the DSCP value, if placed directly in the ToS field in the IP header, must be shifted left 2 bits.
This attribute contains the DSCP value for the PTP General frames.
Default = 002Fh

48-01-04 DSCP Urgent

[USINT, Set, NV]
Attributes 2 through 8 contain the DSCP values that shall be used for the different types of EtherNet/IP traffic.
Refer to the ODVA documentation for the format of the DSCP value within the IP header. Since the DSCP field has a size of 6 bits, the valid range of values for these attributes is 0-63. Note that the DSCP value, if placed directly in the ToS field in the IP header, must be shifted left 2 bits.
DSCP value for CIP transport class 1 Urgent priority messages (implicit messages with urgent priority).
Default = 0037h

48-01-05 DSCP Scheduled

[USINT, Set, NV]
Attributes 2 through 8 contain the DSCP values that shall be used for the different types of EtherNet/IP traffic.
Refer to the ODVA documentation for the format of the DSCP value within the IP header. Since the DSCP field has a size of 6 bits, the valid range of values for these attributes is 0-63. Note that the DSCP value, if placed directly in the ToS field in the IP header, must be shifted left 2 bits.
DSCP value for CIP transport class 1 Scheduled priority messages (implicit messages with scheduled priority).
Default = 002Fh

48-01-06 DSCP High

[USINT, Set, NV]
Attributes 2 through 8 contain the DSCP values that shall be used for the different types of EtherNet/IP traffic.
Refer to the ODVA documentation for the format of the DSCP value within the IP header. Since the DSCP field has a size of 6 bits, the valid range of values for these attributes is 0-63. Note that the DSCP value, if placed directly in the ToS field in the IP header, must be shifted left 2 bits.
DSCP value for CIP transport class 1 High priority messages (implicit messages with high priority).
Default = 002Bh

48-01-07 DSCP Low

[USINT, Set, NV]
Attributes 2 through 8 contain the DSCP values that shall be used for the different types of EtherNet/IP traffic.
Refer to the ODVA documentation for the format of the DSCP value within the IP header. Since the DSCP field has a size of 6 bits, the valid range of values for these attributes is 0-63. Note that the DSCP value, if placed directly in the ToS field in the IP header, must be shifted left 2 bits.
DSCP value for CIP transport class 1 Low priority messages (implicit messages with low priority).

Default = 001Fh

48-01-08 DSCP Explicit

[USINT, Set, NV]
Attributes 2 through 8 contain the DSCP values that shall be used for the different types of EtherNet/IP traffic.
Refer to the ODVA documentation for the format of the DSCP value within the IP header. Since the DSCP field has a size of 6 bits, the valid range of values for these attributes is 0-63. Note that the DSCP value, if placed directly in the ToS field in the IP header, must be shifted left 2 bits.
DSCP value for CIP explicit messages (transport class 3 and UCMM) and all other EtherNet/IP encapsulation messages.
Default = 0018h

5.12.9 Class F5h: TCP/IP Interface Object

Class Code	Object Class	Access	Nr. of Instances
F5h	TCP/IP Interface Object	Get	1

The TCP/IP Interface Object provides the mechanism to configure the TCP/IP network interface of a device. Examples of configurable items include the device's IP Address, Network Mask, and Gateway Address.
The underlying physical communications interface associated with the TCP/IP Interface Object shall be any interface that supports the TCP/IP protocol. For example, a TCP/IP Interface Object may be associated with any of the following: an IEEE 802.3 interface, an ATM interface, a serial port running SLIP, a serial port running PPP, etc. The TCP/IP Interface Object provides an attribute that identifies the link-specific object for the associated physical communications interface. The link-specific object is generally expected to provide link-specific counters as well as any link-specific configuration attributes.
Each device shall support exactly one instance of the TCP/IP Interface Object for each TCP/IP-capable communications interface on the module.
For complete information on TCP/IP Interface Object attributes refer to the publication "The CIP Networks Library. Volume 2. EtherNet/IP Adaptation of CIP".

5.12.9.1 Supported Class Services

The supported **Class Services** of the TCP/IP Interface Object are:
0Eh = Get_Attribute_Single: used to read the value of an attribute.

5.12.9.2 Class Attributes

F5-01 Revision

[UINT, Get, NV]
Object revision. The current value assigned to this attribute is 0004h.
Default = 0004h

F5-02 Max Instance

[UINT, Get, NV]
The largest instance number of a created object in this class.
Default = 0001h

F5-03 Number of Instances

[UINT, Get, NV]
The number of object instances in this class.
Default = 0001h

F5-06 Maximum ID Number Class Attributes

[UINT, Get, NV]

The attribute ID number of the last class attribute of the class definition implemented in the encoder.

Default = 0007h

F5-07 Maximum ID Number Instance Attributes

[UINT, Get, NV]

The attribute ID number of the last instance attribute of the class definition implemented in the encoder.

Default = 000Eh

5.12.9.3 Supported Instance Services

The supported **Instance Services** of the TCP/IP Interface Object are:

01h = Get_Attribute_All: used to read the value of all attributes.

0Eh = Get_Attribute_Single: used to read the value of an attribute.

10h = Set_Attribute_Single: used to write connection class attribute value.

5.12.9.4 Instance Attributes

F5-01-01 Status

[DWORD, Get, V]

This attribute represents the current status of the TCP/IP network interface. Its value changes as the state of the interface changes. The **F5-01-01 Status** attribute is a DWORD, with the following bit definitions:

Bit(s)	Called	Definition
0 … 3	**Interface Configuration Status**	It indicates the status of the **F5-01-05 Interface Configuration** attribute. 0 = the **F5-01-05 Interface Configuration** attribute has not been configured. 1 = the **F5-01-05 Interface Configuration** attribute contains configuration obtained from DHCP or non-volatile storage. 2 = the **F5-01-05 Interface Configuration** attribute contains configuration from hardware settings. 3 … 15 = reserved for future use
4	**Mcast Pending**	If set to 1 it indicates a multicast pending configuration in the **F5-01-08 TTL Value** and/or **F5-01-09 Mcast Config** attributes. This bit shall be set when either the **F5-01-08 TTL Value** or **F5-01-09 Mcast Config** attribute is set, and shall be cleared the next

		time the device starts.
5	**Interface Configuration Pending**	It indicates a pending configuration change in the **F5-01-05 Interface Configuration** attribute. This bit shall be 1 (TRUE) when the **F5-01-05 Interface Configuration** attribute is set and the device requires a reset in order for the configuration change to take effect (as indicated in the **F5-01-02 Configuration Capability** attribute). The intent of the **Interface Configuration Pending** bit is to allow client software to detect that a device's IP configuration has changed, but will not take effect until the device is reset.
6	**AcdStatus**	It indicates when an IP address conflict has been detected by ACD. To enable/disable the ACD refer to **F5-01-0A SelectAcd** attribute on page 148.
7	**AcdFault**	It indicates when an IP address conflict has been detected by ACD or the defense failed, and that the current **F5-01-05 Interface Configuration** cannot be used due to this conflict.
8	**IANA Port Admin Change Pending**	It indicates a pending configuration change in the **F5-01-0E IANA Port Admin** attribute. This bit shall be set when the device requires a reset in order for the configuration change to take effect.
9 ... 31	Reserved	Reserved, shall be 0

F5-01-02 Configuration Capability

[DWORD, Get, NV]
It indicates the method of obtaining an initial IP address.

Bit(s)	Called	Definition
0	**BOOTP Client**	1 (TRUE) shall indicate that the device is capable of obtaining its network configuration via BOOTP.
1	**DNS Client**	1 (TRUE) shall indicate that the device is capable of obtaining its network configuration via DNS.
2	**DHCP Client**	1 (TRUE) shall indicate that the device is capable of obtaining its network configuration via DHCP.
3	**DHCP-DNS Update**	It is 0, behavior to be defined in a future specification edition.
4	**Configuration Settable**	If set to 1 (TRUE), it indicates that the **F5-**

		01-05 Interface Configuration attribute is settable.
5	**Hardware Configurable**	1 (TRUE) shall indicate that the IP Address member of the F5-01-05 Interface Configuration attribute can be obtained from hardware settings. If this bit is 0 (FALSE) the F5-01-01 Status Instance Attribute (1), Interface Configuration Status field value shall never be 2 (the F5-01-05 Interface Configuration attribute contains valid configuration, obtained from hardware settings).
6	**Interface Configuration Change requires reset**	1 (TRUE) shall indicate that the device requires a restart in order for a change to the F5-01-05 Interface Configuration attribute to take effect. If this bit is 0 (FALSE) a change in the F5-01-05 Interface Configuration attribute will take effect immediately.
7	**AcdCapable**	If set to 1 (TRUE), the encoder is capable of detecting address conflicts (ACD capable). See the F5-01-0A SelectAcd attribute on page 148.
8 ... 31	Reserved	Reserved, shall be 0.

Default = 0095h

F5-01-03 Configuration Control

[DWORD, Get/Set, NV]
It is used to control network configuration options.

Bit(s)	Called	Definition
0 ... 3	**Configuration Method**	It determines how the device shall obtain its IP-related configuration. 0 = The device shall use statically-assigned IP configuration values. 1 = The device shall obtain its interface configuration values via BOOTP. 2 = The device shall obtain its interface configuration values via DHCP. 3 … 15 = Reserved for future use. See also the NOTE below.
4	**DNS Enable**	If it is set to 1 (TRUE), the device shall resolve host names by querying a DNS server.
5 ... 31	Reserved	Reserved for future use, shall be 0.

Default = 0000h

NOTE

As previously stated, the **Configuration Method** determines how a device shall obtain its IP-related configuration:

- If the **Configuration Method** is 0, the device shall use statically-assigned IP configuration contained in the **F5-01-05 Interface Configuration** attribute (or assigned via non-CIP methods).
- If the **Configuration Method** is 1, the device shall obtain its IP configuration via BOOTP. The BOOTP client behavior shall be as defined in the relevant RFCs (RFC 951, RFC 1542, RFC 2132) or their successors.
- If the **Configuration Method** is 2, the device shall obtain its IP configuration via DHCP. The DHCP client behavior shall be as defined in the relevant RFCs (RFC 2131, RFC 2132) or their successors.
- Devices that optionally provide hardware means to configure IP addressing behavior shall set the **Configuration Method** to reflect the configuration set via hardware: 0 if a static IP address has been configured, 1 if BOOTP has been configured, 2 if DHCP has been configured.

If a device has been configured to obtain its configuration via BOOTP or DHCP it shall continue sending requests until a response from the server is received. Devices that elect to use default IP configuration in the event of no response from the server shall continue issuing requests until a response is received, or until the **Configuration Method** is changed to 0 (static).
Once the device receives a response from the server it shall stop sending the BOOTP/DHCP client requests (DHCP clients shall follow the lease renewal behavior per the RFC). It is recommended that devices implement the means to detect a link up and upon a link up detection restart the initial BOOTP or DHCP sequence. For multiport devices the restart of the initial BOOTP or DHCP sequence shall only be triggered if all external links have been down and when the first link up is detected.
Setting the **Configuration Method** to 0 (static address) shall cause the **F5-01-05 Interface Configuration** to be saved to NV storage.
It is recommended that setting the **Configuration Method** to 1 (BOOTP) or 2 (DHCP) cause the device to start the BOOTP / DHCP client to obtain new IP address configuration. If the device requires a reset in order to start the BOOTP / DHCP client, it shall set the **Interface Configuration Pending** bit, and upon device reset start the BOOTP / DHCP client.

F5-01-04 Physical Link Object

[Struct of, Get, NV]
This attribute identifies the object associated with the underlying physical communications interface (e.g., an 802.3 interface). There are two components to the attribute: a **Path size** (in UINTs) and a **Path**. The **Path** shall contain a Logical Segment, type Class, and a Logical Segment, type Instance that identifies

the physical link object. The maximum **Path size** is 6 (assuming a 32 bit logical segment for each of the class and instance).
Default = 20 F6 24 01h

Path size
[UINT] Size of path (0002h).

Path
[Padded EPATH] Path to Ethernet Link Object, **F6-01-03 Physical Address** instance, see on page 152 (20 F6 24 01h).

F5-01-05 Interface Configuration

[Struct of, Get/Set, V/NV]
The **F5-01-05 Interface Configuration** attribute contains the configuration parameters required for a device to operate as a TCP/IP node. The contents of the **F5-01-05 Interface Configuration** attribute shall depend upon how the device has been configured to obtain its IP parameters:

- If configured to use a static IP address (**Configuration Method** value in the **F5-01-03 Configuration Control** attribute is set to 0), the **F5-01-05 Interface Configuration** values shall be those which have been statically assigned and stored in NV storage.
- If configured to use BOOTP or DHCP (**Configuration Method** value is set to 1 or 2), the **F5-01-05 Interface Configuration** values shall contain the configuration obtained from the BOOTP or DHCP server. The **F5-01-05 Interface Configuration** attribute shall be 0 until the BOOTP/DHCP reply is received.
- Some devices optionally provide additional, non-CIP mechanisms for setting IP-related configuration (e.g., a web server interface, rotary switch for configuring IP address, etc.). When such a mechanism is used, the **F5-01-05 Interface Configuration** attribute shall reflect the IP configuration values in use.

Default = 0000h

IP Address
[UDINT] The device's IP address (192.168.1.10).

Network Mask
[UDINT] The device's network mask (255.255.255.0). The network mask is used when the IP network has been partitioned into subnets. The network mask is used to determine whether an IP address is located on another subnet.

Gateway Address
[UDINT] The IP address of the device's default gateway (0.0.0.0). When a destination IP address is on a different subnet, packets are forwarded to the default gateway for routing to the destination subnet.

Name Server
[UDINT] The IP address of the primary name server (primary DNS). The name server is used to resolve host names. For example, that might be contained in a CIP connection path.

Name Server 2
[UDINT] The IP address of the secondary name server (secondary DNS). The secondary name server is used when the primary name server is not available, or is unable to resolve a host name.

Domain Name
[STRING] The default domain name. The default domain name is used when resolving host names that are not fully qualified. For example, if the default domain name is "odva.org", and the device needs to resolve a host name of "plc", then the device will attempt to resolve the host name as "plc.odva.org".

F5-01-06 Host Name
[STRING, Get/Set, NV]
It contains the device's host name, which can be used for informational purposes.

F5-01-07 Safety Network Number
[6 octets]
For any information see the CIP Safety Specification, Volume 5, Chapter 3.
Default = FF FF FF FF FF FFh

F5-01-08 TTL Value
[USINT, Get/Set, NV]
The device shall use the TTL value for the IP header Time-to-live field when sending EtherNet/IP packets via IP multicast. By default, **F5-01-08 TTL Value** is 1. The maximum value for **F5-01-08 TTL Value** is 255. Note that unicast packets shall use the TTL as configured for the TCP/IP stack, and not the TTL Value configured in this attribute.
When set, the **F5-01-08 TTL Value** attribute shall be saved in non-volatile memory. If a device does not support applying the TTL Value immediately, the **Mcast Pending** bit in the **F5-01-01 Status** attribute shall be set, indicating that there is pending configuration. For devices that support applying the TTL Value immediately, if there are existing multicast connections, an **Object State Conflict** error (0xC) shall be returned and the **Mcast Pending** bit shall not be set. When a new TTL Value is pending, Get_Attribute_Single or Get_Attributes_All requests shall return the pending value. The **Mcast Pending** bit shall be cleared the next time the device starts.
Users should exercise caution when setting the **F5-01-08 TTL Value** greater than 1, to prevent unwanted multicast traffic from propagating through the network.

Default = 0001h

F5-01-09 Mcast Config

[Struct of, Get, NV]
It contains the configuration of the device's IP multicast addresses to be used for EtherNet/IP multicast packets.
Default = 0000h

Alloc Control
[USINT] 0 = multicast addresses shall be generated using the default allocation algorithm according to specifications. 1 = multicast addresses shall be allocated according to the values specified in **Num Mcast** and **Mcast Start Addr** parameters.

(reserved)
[USINT] set to 0, do not change.

Num Mcast
[UINT] Number of IP multicast addresses allocated (1). The maximum number of multicast addresses is device specific, but shall not exceed the number of EtherNet/IP multicast connections supported by the device.

Mcast Start Addr
[UDINT] Starting multicast address from which **Num Mcast** addresses are allocated. When set, the **F5-01-09 Mcast Config** attribute shall be saved in non-volatile memory. If a device does not support applying the **F5-01-09 Mcast Config** attribute immediately, the **Mcast Pending** bit in the **F5-01-01 Status** attribute shall be set, indicating that there is pending configuration. For devices that support applying the **F5-01-09 Mcast Config** attribute immediately, if there are existing multicast connections an **Object State Conflict** error (0xC) shall be returned and the **Mcast Pending** bit shall not be set. When a new **F5-01-09 Mcast Config** value is pending, Get_Attribute_Single or Get_Attributes_All requests shall return the pending value. The **Mcast Pending** bit shall be cleared the next time the device starts.
When the multicast addresses are generated using the default algorithm, **Num Mcast** and **Mcast Start Addr** shall report the values generated by the algorithm.

F5-01-0A SelectAcd

[BOOL, Set, NV]
It allows to enable (1) / disable (0) the Address Conflict Detection (ACD). If ACD is enabled, as soon as an address conflict is detected, the bit 6 **AcdStatus** in the **F5-01-01 Status** attribute will be set to 1 and NS Network State Error LED will light on red (refer to page 42).
The default value of **F5-01-0A SelectAcd** shall be 1 indicating that ACD is enabled.
When the value of **F5-01-0A SelectAcd** is changed by a Set_Attribute service, the new value of **F5-01-0A SelectAcd** shall not be applied until the device executes a restart.
0 = Disable ACD
1 = Enable ACD
Default = 0001h

F5-01-0B LastConflictDetected

[Struct of, Set, NV]
It is a diagnostic attribute presenting information about the ACD state when the last IP address conflict was detected.

AcdActivity
[USINT] State of the ACD algorithm when the last IP address conflict was detected.

RemoteMAC
[Array of 6 USINTs] The IEEE 802.3 source MAC address from the header of the received Ethernet packet sent by the device when reporting the conflict.

ArpPDU
[Array of 28 USINTs] The ARP Response PDU in binary format.

F5-01-0C EtheNet/IP QuickConnect

[BOOL, Set, NV]
It shall enable (1) or disable (0) the EtherNet/IP QuickConnect feature. If EtherNet/IP QuickConnect is enabled, it will direct EtherNet/IP target devices to quickly power up and join an EtherNet/IP network.
Default = 0

F5-01-0D Encapsulation Inactivity Timeout

[UINT, Set, NV]
Number of seconds with no Encapsulation activity before the TCP connection or the DTLS is closed. It is used to enable TCP socket or DTLS session cleanup (closing) when the defined number of seconds have elapsed with no Encapsulation activity. When set, the **F5-01-0D Encapsulation Inactivity Timeout** attribute shall be saved in non-volatile memory and applied to all

subsequently-opened connections. It is disabled by setting 0. The timeout can be set in the range 1 to 3600 and is expressed in seconds.
Default = 0078h

F5-01-0E IANA Port Admin

[Struct of, Get, NV]
The TCP and UDP port numbers are a part of the transport layer. The port numbers are used to identify the sending and receiving applications within the communicating devices. Port numbers are divided into three ranges: well-known, registered, and dynamic or private. The registered ports are assigned by Internet Assigned Numbers Authority (IANA), www.iana.org. Both well-known and registered ports are listed in the Service Name and Transport Protocol Port Number Registry at IANA.
The **F5-01-0E IANA Port Admin** attribute lists TCP and UDP ports used by the device where it acts as a server. It is recommended that all TCP and UDP ports used by the device be exposed by the attribute. At a minimum all EtherNet/IP-related ports supported by the devices shall be exposed.
The text provided in the Port Name member is vendor specific but for well-known and registered ports it is recommended to use the IANA description. It is valid to have a NULL string in the **Port Name** attribute.
For complete information refer to the ODVA documentation.
Default = TCP : AF12h
UDP : AF12h
UDP : 08AEh

5.12.10 Class F6h: Ethernet Link Object

Class Code	Object Class	Access	Nr. of Instances
F6h	Ethernet Link Object	Get	1

The EtherNet Link Object maintains link-specific counters and status information for an IEEE 802.3 communications interface such as transmission speed, interface status and the MAC address.

5.12.10.1 Supported Class Services

The supported **Class Services** of the Ethernet Link Object are:
0Eh = Get_Attribute_Single: used to read the value of an attribute.

5.12.10.2 Class Attributes

F6-01 Revision

[UINT, Get, NV]
Object revision. The current value assigned to this attribute is 0004h.
Default = 0004h

F6-02 Max Instance

[UINT, Get, NV]
The largest instance number of a created object in this class.
Default = 0002h

F6-03 Number of Instances

[UINT, Get, NV]
The number of object instances in this class.
Default = 0002h

F6-06 Maximum ID Number Class Attributes

[UINT, Get, NV]
The attribute ID number of the last class attribute of the class definition implemented in the encoder.
Default = 0007h

F6-07 Maximum ID Number Instance Attributes

[UINT, Get, NV]
The attribute ID number of the last instance attribute of the class definition implemented in the encoder.
Default = 0300h

5.12.10.3 Supported Instance Services

The supported **Instance Services** of the Ethernet Link Object are:
01h = Get_Attribute_All: used to read the value of all attributes.
0Eh = Get_Attribute_Single: used to read the value of an attribute.
10h = Set_Attribute_Single: used to write connection class attribute value.
4Ch = Get_And_Clear: used to get and then clear the specified attribute; it retrieves the attribute value and then sets it to zero.

5.12.10.4 Instance Attributes

F6-01-01 Interface Speed

[UDINT, Get, V]
Interface speed currently in use, expressed in Mbps (e.g., 0, 10, 100, 1000, etc.).
Default = 0064h

F6-01-02 Interface Flags

[DWORD, Get, V]
Interface status flags, according to the following table.

Bit(s)	Called	Definition
0	Link Status	It indicates whether or not the IEEE 802.3 communications interface is connected to an active network. 0 indicates an inactive link; 1 indicates an active link.
1	Half/Full Duplex	It indicates the duplex mode currently in use. 0 indicates the interface is running half duplex; 1 indicates full duplex. If the Link Status flag is 0, then the value of the Half/Full Duplex flag is indeterminate.
2 ... 4	Negotiation Status	It indicates the status of link auto-negotiation. 0 = Auto-negotiation in progress 1 = Auto-negotiation and speed detection failed. Using default values. Recommended defaults are 10 Mbps and half duplex. 2 = Auto-negotiation failed but detected speed. Duplex was defaulted. 3 = Successfully negotiated speed and duplex. 4 = Auto-negotiation not attempted. Forced speed and duplex.
5	Manual Setting Requires Reset	It is 0 when the interface can activate changes to link parameters during runtime. It is 1 when reset is required in order for changes to take effect.

6	Local Hardware Fault	0 indicates the interface detects no local hardware fault; 1 indicates a local hardware fault is detected.
7 ... 31	Reserved	Reserved, shall be 0

Default = 0020h

F6-01-03 Physical Address

[Array of 6 UINTs, Get, NV]
MAC ID. This attribute contains the physical network address, i.e. the assigned MAC address.
Default = according to the specific device

F6-01-04 Interface Counters

[Struct of, Get, V]
This attribute contains counters relevant to the receipt of packets on the interface.

In Octets
[UDINT] Octets received on the interface.

In Ucast Packets
[UDINT] Unicast packets received on the interface.

In NUcast Packets
[UDINT] Non-unicast packets received on the interface.

In Discards
[UDINT] Inbound packets received on the interface but discarded.

In Errors
[UDINT] Inbound packets that contain errors (does not include **In Discards**).

In Unknown Protos
[UDINT] Inbound packets with unknown protocol.

Out Octets
[UDINT] Octets sent on the interface.

Out Ucast Packets
[UDINT] Unicast packets sent on the interface.

Out NUcast Packets
[UDINT] Non-unicast packets sent on the interface.

Out Discards
[UDINT] Outbound packets discarded.

Out Errors
[UDINT] Outbound packets that contain errors (does not include **Out Discards**).

F6-01-05 Media Counters

[Struct of, Get, V]
This attribute contains counters specific to Ethernet media.

Alignment Errors
[UDINT] Frames received that are not integral number of octets in length.

FCS Errors
[UDINT] Frames received that do not pass the FCS check.

Single Collisions
[UDINT] Successfully transmitted frames which experienced exactly one collision.

Multiple Collisions
[UDINT] Successfully transmitted frames which experienced more than one collision.

SQE Test Errors
[UDINT] Number of times SQE test error message is generated.

Deferred Transmissions
[UDINT] Frames for which first transmission attempt is delayed because the medium is busy.

Late Collisions
[UDINT] Number of times a collision is detected later than 512 bit-times into the transmission of a packet.

Excessive Collisions
[UDINT] Frames for which transmission fails due to excessive collisions.

MAC Transmit Errors
[UDINT] Frames for which transmission fails due to an internal MAC sublayer transmit error.

Carrier Sense Errors
[UDINT] Times that the carrier sense condition was lost or never asserted when attempting to transmit a frame.

Frame Too Long
[UDINT] Frames received that exceed the maximum permitted frame size.

MAC Receive Errors
[UDINT] Frames for which reception on an interface fails due to an internal MAC sublayer receive error.

F6-01-06 Interface Control

[Struct of, Get/Set, NV]
This attribute is a structure consisting of the following parameters.

Control Bits
[WORD] Interface control bits.

Bit(s)	Called	Definition
0	**Auto-negotiate**	0 indicates that 802.3 link auto-negotiation is disabled. 1 indicates that auto-negotiation is enabled. If auto-negotiation is disabled, then the device shall use the settings indicated by the **Forced Duplex Mode** and **Forced Interface Speed** bits.
1	**Forced Duplex Mode**	If the **Auto-negotiate** bit is 0, the **Forced Duplex Mode** bit indicates whether the interface shall operate in full or half duplex mode. 0 indicates that the interface duplex should be half duplex. 1 indicates that the interface duplex should be full duplex. Interfaces not supporting the requested duplex shall return status code 0x09 (**Invalid Attribute Value**). If auto-negotiation is enabled, attempting to set the **Forced Duplex Mode** bit shall result in status code 0x0C (**Object State Conflict**).
2 ... 15	Reserved	Reserved, shall be 0

Forced Interface Speed
[UINT] If the **Auto-negotiate** bit is 0, the **Forced Interface Speed** bits indicate the speed at which the interface shall operate. Speed is specified in megabits per second (e.g., for 10 Mbps Ethernet, the **Forced Interface Speed** shall be 10).

Default = 0000h

F6-01-07 Interface Type

[USINT, Get, NV]
This attribute indicates the type of the physical interface according to the following table.

Instance	Value	Type of interface
1	2	Twisted-pair
2	2	Twisted-pair
3	1	The interface is internal to the device

Default = 0002h

F6-01-08 Interface State

[USINT, Get, V]
This attribute indicates the current operational state of the interface according to the following table.

Value	Interface State
0	Unknown interface state
1	The interface is enabled and is ready to send and receive data
2	The interface is disabled
3	The interface is testing
4 ... 255	Reserved

Default = 0000h

F6-01-09 Admin State

[USINT, Set, NV]
This attribute allows administrative setting of the interface state according to the following table.

Value	Admin State
0	Reserved
1	Enable the interface
2	Disable the interface
3 ... 255	Reserved

Default = 2 (Disable the interface)

F6-01-0A Interface Label

[SHORT_STRING, Get, NV]
This attribute is a string that describes the interface according to the following table.

Instance	Value
1	Port 1
2	Port 2
3	Internal

Default = "Port 1" or "Port 2"

F6-01-0B Interface Capability

[Struct of, Get, NV]
This attribute indicates the set of capabilities for the interface according to the following table.

Bit(s)	Called	Definition
0	**Manual Setting Requires Reset**	It indicates whether or not the device requires a reset to apply changes made to the F6-01-06 Interface Control attribute. 0 = It indicates that the device automatically applies changes made to the F6-01-06 Interface Control attribute and, therefore, does not require a reset in order for changes to take effect. This is the value this bit shall have when the F6-01-06 Interface Control attribute is not implemented. 1 = It indicates that the device does not automatically apply changes made to the F6-01-06 Interface Control attribute and, therefore, will require a reset in order for changes to take effect. Note: this bit shall also be replicated in the F6-01-02 Interface Flags attribute in order to retain backwards compatibility with previous object revisions.
1	**Auto-negotiate**	0 = It indicates that the interface does not support link auto-negotiation (internal interface) 1 = It indicates that the interface supports link auto-negotiation (external interface)
2	**Auto-MDIX**	0 = It indicates that the interface does not support auto MDIX operation (internal interface) 1 = It indicates that the interface supports

		auto MDIX operation (external interface)
3	**Manual Speed/Duplex**	0 = It indicates that the interface does not support manual setting of speed/duplex. The **F6-01-06 Interface Control** attribute shall not be supported (internal interface) 1 = It indicates that the interface supports manual setting of speed/duplex via the **F6-01-06 Interface Control** attribute (external interface)
4 ... 31	Reserved	Reserved, shall be set to 0

Default = 10 / HD
10 / FD
100 / HD
100 / FD

F6-01-300 MDIX

[USINT, Get, NV]
This attribute sets the MDIX configuration.

Value	MDIX	
1	EIP_EN_INTF_MDIX_AUTO	Auto-detect
2	EIP_EN_INTF_MDIX_MDI	Explicit MDI
3	EIP_EN_INTF_MDIX_MDIX	Explicit MDIX

Default = 0001h

5.12.11 Class 109h: LLDP Management Object

Class Code	Object Class	Access	Nr. of Instances
109h	LLDP Management Object	Get	1

The LLDP Management Object provides administrative information for the LLDP protocol. It functions as an interface to configure aspects of the LLDP protocol that is running in the device.
All information about neighboring devices that is stored in the data tables of the LLDP protocol stack can be accessed via the SNMP LLDP MIB (OID 1.0.8802.1.1.2.1).

5.12.11.1 Supported Class Services

The supported **Class Services** of the LLDP Management Object are:
0Eh = Get_Attribute_Single: used to read the value of an attribute.

5.12.11.2 Class Attributes

109-01 Revision

[UINT, Get, NV]
Object revision. The current value assigned to this attribute is 0001h.
Default = 0001h

109-02 Max Instance

[UINT, Get, NV]
The largest instance number of a created object in this class.
Default = 0001h

109-03 Number of Instances

[UINT, Get, NV]
The number of object instances in this class.
Default = 0001h

109-06 Maximum ID Number Class Attributes

[UINT, Get, NV]
The attribute ID number of the last class attribute of the class definition implemented in the encoder.
Default = 0007h

109-07 Maximum ID Number Instance Attributes

[UINT, Get, NV]
The attribute ID number of the last instance attribute of the class definition implemented in the encoder.
Default = 0005h

5.12.11.3 Supported Instance Services

The supported **Instance Services** of the LLDP Management Object are:
0Eh = Get_Attribute_Single: used to read the value of an attribute.
10h = Set_Attribute_Single: used to write connection class attribute value.

5.12.11.4 Instance Attributes

109-01-01 LLDP Enable

[Struct of, Set, NV]
It enables / disables LLDP global or per port.
Default = all ports enabled

109-01-02 MsgTxInterval

[UINT, Set, NV]
From 802.1AB-2016. It is the interval for transmitting LLDP frames from the device. The value is expressed in seconds.
Default = 001Eh

109-01-03 MsgTxHold

[USINT, Set, NV]
From 802.1AB-2016. It is a multiplier of the **109-01-02 MsgTxInterval** attribute value to determine the value of the TTL TLV sent to neighboring devices.
Default = 0004h

109-01-04 LLDP Datastore

[UINT, Get, NV]
An indication of the retrieval methods for the LLDP database supported by the device.
Default = 0002h (SNMP)

109-01-05 Last Change

[UDINT, Get, NV]
The value of sysUpTime taken the last time any entry in the local LLDP database changed.
Default = 0000h

5.12.12 Class 401h: Predefined Connection Object

Class Code	Object Class	Access	Nr. of Instances
401h	Predefined Connection Object	Get	1

The Predefined Connection Object (PDC) defines and maintains the implicit (class 0/1) connections of the EtherNet/IP Adapter. It is a manufacturer-specific CIP object, which is not covered by the CIP specification.
The PDC object has two purposes:

1. During the configuration phase, it lets the host application define the set of implicit connections the EtherNet/IP Adapter supports. For each connection, the following parameters are defined:
   - the connection endpoints, a.k.a. Assembly instances for the Input and Output data directions;
   - the allowed range of packet intervals (RPI), further limiting the range the protocol stack is technically capable of, if intended;
   - the set of connection trigger types supported by the connection;
   - the connection type (class 0, class 1, listen only, input only).
2. During the runtime phase, it provides information about the current state of the connections.

5.12.12.1 Supported Class Services

The supported **Class Services** of the Predefined Connection Object are:
0Eh = Get_Attribute_Single: used to read the value of an attribute.
08h = Create: used to create new predefined connection instance.

5.12.12.2 Class Attributes

401-01 Revision

[UINT, Get, NV]
Object revision. The current value assigned to this attribute is 0001h.
Default = 0001h

401-02 Max Instance

[UINT, Get, NV]
The largest instance number of a created object in this class.
Default = 0003h (in default configuration)

401-03 Number of Instances

[UINT, Get, NV]
The number of object instances in this class.
Default = 0003h (in default configuration)

401-06 Maximum ID Number Class Attributes

[UINT, Get, NV]
The attribute ID number of the last class attribute of the class definition implemented in the encoder.
Default = 0007h

401-07 Maximum ID Number Instance Attributes

[UINT, Get, NV]
The attribute ID number of the last instance attribute of the class definition implemented in the encoder.
Default = 0003h

5.12.12.3 Supported Instance Services

The supported **Instance Services** of the Predefined Connection Object are:
0Eh = Get_Attribute_Single: used to read the value of an attribute.
08h = Create: used to create new predefined connection instance.
09h = Delete: used to delete predefined connection instance.

Create
The "Create" service creates a new instance of the Predefined Connection object.

NOTE
This service may be executed successfully with faulty configurations, e.g. invalid assembly instances. However, inconsistent configuration might lead to a non-functioning connection.

Request Service Data Field Parameters
The request service data equals the PDC instance attribute 3 structure (see the structure of PDC, **401-01-03 Configuration**, below).

Successful Response Service Data Field Parameters
The response data to the "Create" service provides the CIP instance number of the newly created Predefined Connection object instance.

Name	Byte size	Description
CIP Instance number that has been created	2	CIP Instance that has been created inside the Predefined Connection class.

Unsuccessful Response Service Data Field Parameters
The unsuccessful response does not provide any data.

Delete
The Delete service deletes an instance of the Predefined Connection object. Deleting of an instance is only possible if the instance is not participating in an

active connection. Otherwise, the service will be answered with general status code 0x0C (**Bad Object Mode**).

Request Service Data Field Parameters
The service does not accept any parameters.

Success Response Service Data Field Parameters
The service has no response parameters.

Unsuccessful Response Service Data Field Parameters
The unsuccessful response does not provide any data.

5.12.12.4 Instance Attributes

401-01-01 State
[Get]
This attribute provides information about the current state of the connection.
0 = FREE
1 = UNCONNECTED
2 = CONNECTED
3 = TIMEOUT

401-01-02 Count
[Get]
This attribute indicates how many connections of that type are currently opened.

401-01-03 Configuration
[Get]
This attribute indicates a specific implicit connection that can be opened to the EtherNet/IP Adapter.

Consumer Connection Point
[Byte size = 4]
Connection point addressing the O2T (Originator to Target) direction.
Typically, this is an assembly instance number. The value 0xFFFFFFFF serves a wildcard (don't care) purpose. If the wildcard is given, any Assembly of the proper data direction, type, and size will be accepted as the connection endpoint. Specifying explicit connection endpoints is to be preferred over using the wildcard feature for the sake of a clearer system design.

Producer Connection Point
[Byte size = 4]
Connection point addressing the T2O (Target to Originator) direction.
Typically, this is an assembly instance number. The value 0xFFFFFFFF serves a wildcard (don't care) purpose. If the wildcard is given, any Assembly of the proper data direction, type, and size will be accepted as the connection endpoint. Specifying explicit connection endpoints is to be preferred over using the wildcard feature for the sake of a clearer system design.

Configuration Connection Point
[Byte size = 4]
Connection point addressing a configuration assembly instance.
The value 0xFFFFFFFF serves a wildcard (don't care) purpose. If the wildcard is given, any Configuration Assembly of the proper size will be accepted. Specifying explicit connection endpoints is to be preferred over using the wildcard feature for the sake of a clearer system design.

Minimum O2T RPI
[Byte size = 4]
Minimum Requested Packet Interval (RPI) of the consuming direction expressed in microseconds.

Maximum O2T RPI
[Byte size = 4]
Maximum Requested Packet Interval (RPI) of the consuming direction expressed in microseconds.

Minimum T2O RPI
[Byte size = 4]
Minimum Requested Packet Interval (RPI) of the producing direction expressed in microseconds.

Maximum T2O RPI
[Byte size = 4]
Maximum Requested Packet Interval (RPI) of the producing direction expressed in microseconds.

Supported Trigger Types
[Byte size = 1]
Supported trigger types of the connection. There can be up to three trigger types supported.
Use the following flags:

```
#define CIP_PDC_TTYPE_CYCLIC        0x01 /* Cyclic */
#define CIP_PDC_TTYPE_COS           0x02 /* Change of State */
#define CIP_PDC_TTYPE_APPLICATION   0x04 /* Application Triggered */
```

NOTE
The **Supported Trigger Types** field only affects the message production in the T2O (producing) direction. Which trigger type is used for the connection

depends on what the originator of the connection (e.g. the PLC) is requesting. Here, we only configure what types the specific connection supports. The following description of the different trigger types reference the ***Transmission Trigger Timer*** and the ***Production Inhibit Timer***. These timers are described in more detail below.

Cyclic

The ***Transmission Trigger Timer*** triggers the Message production.
In that case, the message production on the network is completely independent from the moment when the host application updates the data in the DPM (e.g. via xChannelloWrite). Therefore, the host application can update the producing data at its own rate without having influence on the frames sent on the network.

Application Triggered

Message production is triggered when the application updates the application production data (e.g. via xChannelloWrite) and by the ***Transmission Trigger Timer***. The message production triggered by the application additionally depends on the ***Production Inhibit Timer*** (see below).

Change of State

Message production is triggered when the application production data has changed (e.g. via xChannelloWrite) and by the ***Transmission Trigger Timer***.

NOTE

The protocol stack will not check for production data changes. Therefore, the host application is responsible to update production data only if it has changed. The message production triggered by the application additionally depends on the ***Production Inhibit Timer*** (see below).

Transmission Trigger Timer

The ***Transmission Trigger Timer*** is using the RPI rate the connection originator (e.g. the PLC) requested during connection establishment. The expiration of this timer will result in the production of the producing data on the network regardless of the connection's trigger type.

Production Inhibit Timer

The **Production Inhibit Timer** applies only to ***Change of State*** or ***Application Triggered*** connections. The timer is started only when the application updates the production data (e.g. xChannelloWrite). Data produced due to the expiration of the ***Transmission Trigger Timer*** will not result in a restart of the ***Production Inhibit Timer*** (one shot). While the timer is running, the protocol stack suppresses new message production to the network. If one or more new data events occur while this timer is running, the protocol stack will produce the most recent new data immediately when it expires. The mechanism intends to limit the production intervals to the lower levels.

The originator of the connection can configure the timer via a "Production Inhibit Time" segment attached to the ForwardOpen message. If this segment is not present, the stack will set the timer value to ¼ of the RPI (as defined by CIP).

Connection Type

[Byte size = 1]

This field specifies the connection application type. The following types are available:

```
#define CIP_CTYPE_EXCLUSIVE_OWNER  0x01
#define CIP_CTYPE_LISTEN_ONLY      0x03
#define CIP_CTYPE_INPUT_ONLY       0x04
```

For more information about application types see the THE CIP NETWORKS LIBRARY, Volume 1, Common Industrial Protocol (CIP™), Edition 3.34, April 2023.

5.12.13 Class 402h: IO Mapping Object

Class Code	Object Class	Access	Nr. of Instances
402h	IO Mapping Object	Get	1

The IO Mapping Object is responsible for partitioning of the DPM I/O input and output areas and mapping of those partitions, i.e. members, to the related instances of the Assembly object (see the "5.12.3 Class 04h: Assembly Object" section on page 99). It is a manufacturer-specific CIP object, which is not covered by the CIP specification.

5.12.13.1 Supported Class Services

The supported **Class Services** of the IO Mapping Object are:
0Eh = Get_Attribute_Single: used to read the value of an attribute.

5.12.13.2 Class Attributes

402-01 Revision

[UINT, Get, NV]
Object revision. The current value assigned to this attribute is 0001h.
Default = 0001h

402-02 Max Instance

[UINT, Get, NV]
The largest instance number of a created object in this class.
Default = 0001h

402-03 Number of Instances

[UINT, Get, NV]
The number of object instances in this class.
Default = 0001h

402-06 Maximum ID Number Class Attributes

[UINT, Get, NV]
The attribute ID number of the last class attribute of the class definition implemented in the encoder.
Default = 0007h

402-07 Maximum ID Number Instance Attributes

[UINT, Get, NV]
The attribute ID number of the last instance attribute of the class definition implemented in the encoder.
Default = 0003h

5.12.13.3 Supported Instance Services

The supported **Instance Services** of the IO Mapping Object are:
0Eh = Get_Attribute_Single: used to read the value of an attribute.
10h = Set_Attribute_Single: used to write connection class attribute value.

5.12.13.4 Instance Attributes

402-01-01 Status

[Get]
This attribute provides information about the current status of I/O data (Data direction, State of Connection).

402-01-02 Length

[Get]
This attribute provides the length of I/O data.

402-01-03 Data

[Get]
This attribute provides I/O data.

5.12.14 Class 403h: Diagnosis Object

Class Code	Object Class	Access	Nr. of Instances
403h	Diagnosis Object	Get	1

The Diagnosis object provides diagnostic information on the product. Any user may read the diagnostic information through the EtherNet/IP network or the host interface and provide it to the support team, precisely identifying the affected product. The Diagnosis object is a manufacturer-specific CIP object, which is not covered by the CIP specification.

5.12.14.1 Supported Class Services

The supported **Class Services** of the Diagnosis Object are:
0Eh = Get_Attribute_Single: used to read the value of an attribute.

5.12.14.2 Class Attributes

403-01 Revision

[UINT, Get, NV]
Object revision. The current value assigned to this attribute is 0001h.
Default = 0001h

403-02 Max Instance

[UINT, Get, NV]
The largest instance number of a created object in this class.
Default = 0001h

403-03 Number of Instances

[UINT, Get, NV]
The number of object instances in this class.
Default = 0001h

403-06 Maximum ID Number Class Attributes

[UINT, Get, NV]
The attribute ID number of the last class attribute of the class definition implemented in the encoder.
Default = 0007h

403-07 Maximum ID Number Instance Attributes

[UINT, Get, NV]
The attribute ID number of the last instance attribute of the class definition implemented in the encoder.
Default = 0009h

5.12.14.3 Supported Instance Services

The supported **Instance Services** of the Diagnosis Object are:
0Eh = Get_Attribute_Single: used to read the value of an attribute.

6.12.13.4 Instance Attributes

403-01-01 Chip info
[SHORT_STRING, Get]
This attribute provides the name of the used EtherNet/IP chip.
Default = device-specific

403-01-02 OS info
[SHORT_STRING, Get]
This attribute provides the name of the used operating system.
Default = device-specific

403-01-03 Stack info
[SHORT_STRING, Get]
This attribute provides the name/version of the used protocol stack core component.
Default = device-specific

403-01-04 Firmware info
[SHORT_STRING, Get]
This attribute provides the name/version of the used EtherNet/IP firmware.
Default = device-specific

403-01-06 Build date
[SHORT_STRING, Get]
This attribute provides information on the build date of the used EtherNet/IP firmware.
Default = device-specific

403-01-07 Build type
[SHORT_STRING, Get]
This attribute provides information on the build type of the used EtherNet/IP firmware.
Default = "release"

403-01-08 Build host

[SHORT_STRING, Get]
This attribute provides information on the build machine name of the used EtherNet/IP firmware.
Default = device-specific

403-01-09 Uptime

[UDINT, Get]
This attribute informs about the device uptime expressed in seconds.
Default = 0000h

6 Integrated Web Server

6.1 Integrated web server – Preliminary information

EtherNet/IP encoders from Lika Electronic integrate a web server. This web-based user interface is designed to offer helpful functions and deliver complete information on the device that can be accessed through the Internet.
In particular it allows:

- to display the current position and speed values;
- to set some parameters such as the preset and the code sequence;
- to display and check the parameters set currently;
- to monitor the encoder;
- to upgrade the firmware;
- to set the network communication parameters.

The web server can be accessed from any PC running a web browser. Since its only requirement is a HTTP connection between the web browser and the web server running on the device, it is perfectly fitted also for remote access scenarios.

Before opening the web server of the EtherNet/IP encoder please ascertain that the following requirements are fully satisfied:

- the encoder is connected to the network;
- the encoder has valid IP address;
- the PC is connected to the network;
- a web browser (Internet Explorer, Mozilla Firefox, Google Chrome, Opera, …) is installed in the PC or in the device used for connection.

NOTE
This web server has been tested and verified using the following web browsers:

- Internet Explorer IE11 version 11.1593.14393.0
- Mozilla Firefox version 116.0.1
- Google Chrome version 115.0.5790.111
- Opera version 68.0.3618.165

NOTE
Please note that the appearance of the snapshots may vary depending on the web browser used. The following snapshots were taken from Google Chrome.

6.2 Web server Home page

To open the EtherNet/IP encoder web server proceed as follows:

1. type the IP address of the encoder you want to connect to (in the example: 192.168.1.10, this is the default software IP address set at Lika, see on page 39) in the address bar of your web browser and confirm by pressing **ENTER**;

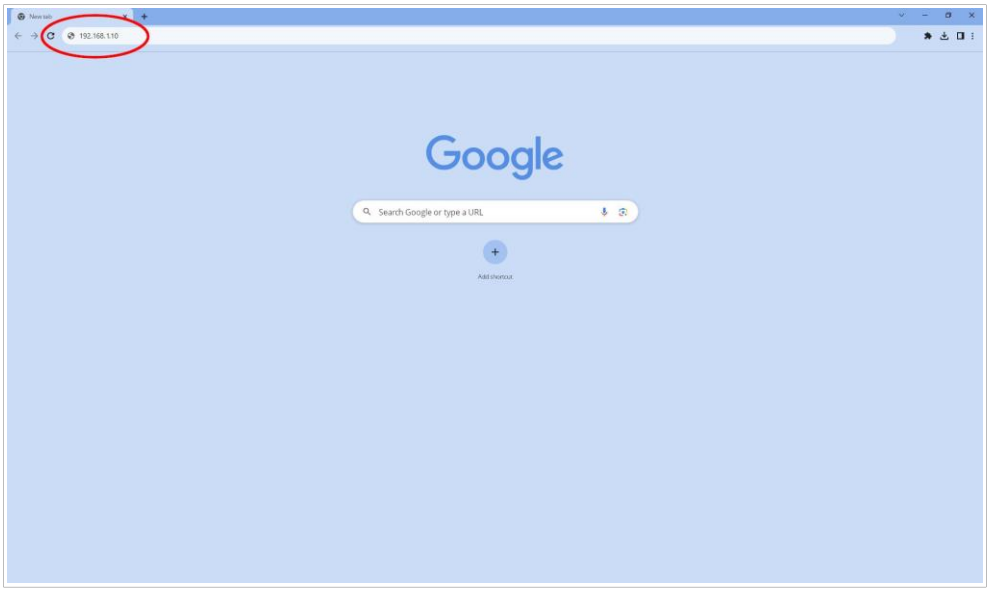


Figure 41 - Opening the web server

2. as soon as the connection is established, the web server **Home** page will appear on the screen;

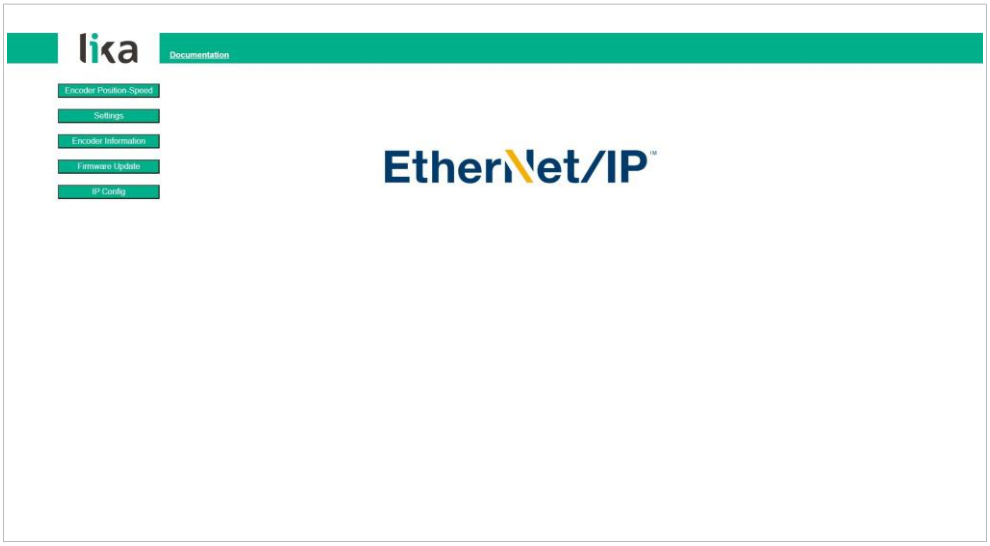


Figure 42 - Web server Home page

Some commands are available in the menu bar of the **Home** page.

Press the **Lika logo** to enter Lika's web site (www.lika.biz).

Press the **DOCUMENTATION** button to enter the EtherNet/IP encoder technical documentation page available on Lika's web site (https://www.lika.it/eng/products/rotary-encoders/absolute/ethernet/) where specific technical information and documentation concerning the EtherNet/IP encoder can be found.

Furthermore some buttons are available in the left navigation bar. All the pages except the **Upgrade firmware** page are freely accessible through the buttons in the bar. The **Upgrade firmware** page is protected and requires a password.
These buttons allow to enter specific pages where information and diagnostics on the connected encoder as well as useful functions can be achieved.
They are described in the following sections.

6.3 Encoder position and speed

Press the **ENCODER POSITION-SPEED** button in the left navigation bar of the Web server **Home** page to enter the page where the current encoder position and the current encoder speed are displayed.

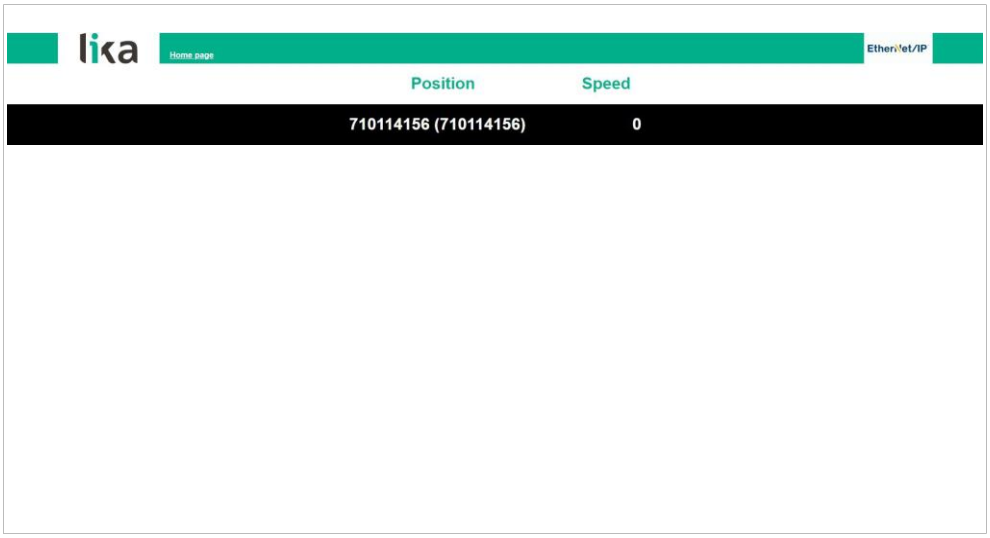


Figure 43 - Encoder position and speed page

The first value (under the Position item) is the absolute position calculated considering scaling and preset functions, if activated; the value in brackets is the raw value (physical absolute position). Both encoder positions are expressed in counts. For any information refer to the **23-01-03 Position value 32 bit** attribute on page 106.

The current encoder speed (under the Speed item) is expressed according to the setting next the **23-01-19 Velocity Format** attribute on page 112 (by default

it is expressed in counts per second). For any information refer to the **23-01-18 Velocity Value** attribute on page 112.

NOTE
The current encoder position and speed values are real-time processed and updated continuously (every 200 msec. on the screen).

Press the **HOMEPAGE** button to move back to the Web server **Home** page.

6.3.1 Specific notes on using Internet Explorer

The following options must be set properly on Internet Explorer in order to get the **Encoder position and speed** page to be updated continuously.

- Open the **Settings** menu;
- open the **Internet Options** property sheet;
- in the **General** tabbed page, press the **Setting** button available in the **History Browsing** section;
- under **Check for newer versions of stored pages**, click **Every time I visit the webpage**;
- press the **OK** button to confirm whenever requested.

6.4 Setting the attributes

Press the **SETTINGS** button in the left navigation bar of the Web server **Home** page to enter the **Set Encoder Attributes** page. In this page the read-write (Set) access EtherNet/IP encoder attributes available in the Position Sensor Object (Class 23h) are displayed and their value can be changed.
For complete information on the encoder attributes please refer to the "5.12.5 Class 23h: Position Sensor Object" section on page 105.

Figure 44 - Set Encoder Attributes page

The values that are currently set in the encoder are displayed in the fields.

To change any value enter a suitable value next to the desired parameter and then press the **SEND** button on the right. The values have to be set in either decimal or hexadecimal notation.
For complete information on the available attributes please refer to the "5.12.5 Class 23h: Position Sensor Object" section on page 105.

EXAMPLE
The **23-01-10 Measuring Units per Span** attribute is currently set to "**65536**" (see the box next to the **Units/Span** item the Figure above). To change the set value enter a suitable value in the box and then press the **SEND** button on the right in the same line to confirm.

NOTE
Please note that, after pressing the **SEND** button, the set value is saved temporarily in the attributes. To save it permanently, please press the **SAVE** button next to the **Save Param.** item. Should the power supply be turned off without saving data, the values that have not been saved on the Flash EEProm

will be lost! For more information refer to the "4.1.6 Saving data" section on page 50.

Press the **LOAD** button next to the **Load Default** item to restore all parameters to default values. Default values are set at the factory by Lika Electronic engineers to allow the operator to run the device for standard operation in a safe mode. This function can be useful, for instance, to restore the factory values in case the encoder is set incorrectly and you are not able to resume the proper operation. For more information refer to the "4.1.7 Restoring defaults" section on page 50.

WARNING
The execution of this command causes all parameters which have been set previously to be overwritten!

NOTE
At each confirmation of the set parameters, a message will appear under the buttons. It informs whether the operation has been accomplished properly or an error occurred (for example **Setting executed correctly!** if everything went well).

Press the **HOMEPAGE** button to move back to the Web server **Home** page.

6.5 Encoder information (EtherNet/IP attributes)

Press the **ENCODER INFORMATION** button in the left navigation bar of the Web server **Home** page to enter the **Encoder Information** page. In this page the list of the most useful EtherNet/IP attributes available for the encoder is displayed. Values of the attributes are expressed in either hexadecimal or decimal notation or in a string format.

Figure 45 - Encoder Information page

For a complete description of the available encoder attributes please refer to the "5.12.5 Class 23h: Position Sensor Object" section on page 105.

NOTE
Please note that the values shown in the **Encoder Information** page are "frozen" in the moment when the page is displayed. To update the values you must refresh the web page.

NOTE
The attributes in the **Encoder Information** page cannot be changed even though they are read-write access attributes. To change the set values please enter the **Set Encoder Attributes** page (see on page 175).

Press the **HOMEPAGE** button to move back to the Web server **Home** page.

6.6 Firmware update

Press the **FIRMWARE UPDATE** button in the left navigation bar of the Web server **Home** page to enter the **Firmware Update** page. Please note that this is a password protected page, thus a password is requested to access the page.

Password: **LiKa** ("L" and "K" in uppercase letters; "i" and "a" in lowercase letters)

WARNING
Firmware updating process has to be accomplished by skilled and competent personnel. It is mandatory to perform the update according to the instructions provided in this section.
Before installation always ascertain that the firmware program is compatible with the hardware and software of the device. Furthermore never turn off the power supply during the flash update. In case of flash update error, the program is lost irreversibly (there is not a bootloader) and the device must be sent back to Lika Electronic for restoring.

This operation allows to update the unit firmware by downloading updating data to the flash memory.
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. These 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 .ZIP extension.

If the latest firmware version is already installed in the unit, you do not need to proceed with any new firmware installation. The firmware version currently installed can be read next to the **Software revision** item in the **Encoder Information** page after connection to the web server (see on page 177).

Before proceeding with the firmware update please ascertain that the following requirements are fully satisfied:

- the encoder is connected to the Ethernet network;
- the encoder has valid IP address;
- the PC is connected both to the network and to the IO controller;

- a web browser (Internet Explorer, Mozilla Firefox, Google Chrome, Opera, …) is installed in the PC or device used for connection;
- you have the .ZIP file for firmware update.

To update the firmware program please proceed as follows:

1. press the **FIRMWARE UPDATE** button in the left navigation bar of the Web server **Home** page to enter the **Firmware Update** page;
2. the operator is requested to submit a password before starting the firmware update procedure;

Figure 46 - Firmware Update page

3. in the **Insert password** text box type the password **LiKa** (“L” and “K” in uppercase letters; “i” and “a” in lowercase letters) and then press the **SEND** button;
4. if the password you typed is wrong, the following warning message will appear on the screen: **WRONG PASSWORD INSERTED. RETRY**. Please retype the password and confirm;

5. if the password you typed is correct, the **Firmware Update** page will appear on the screen;

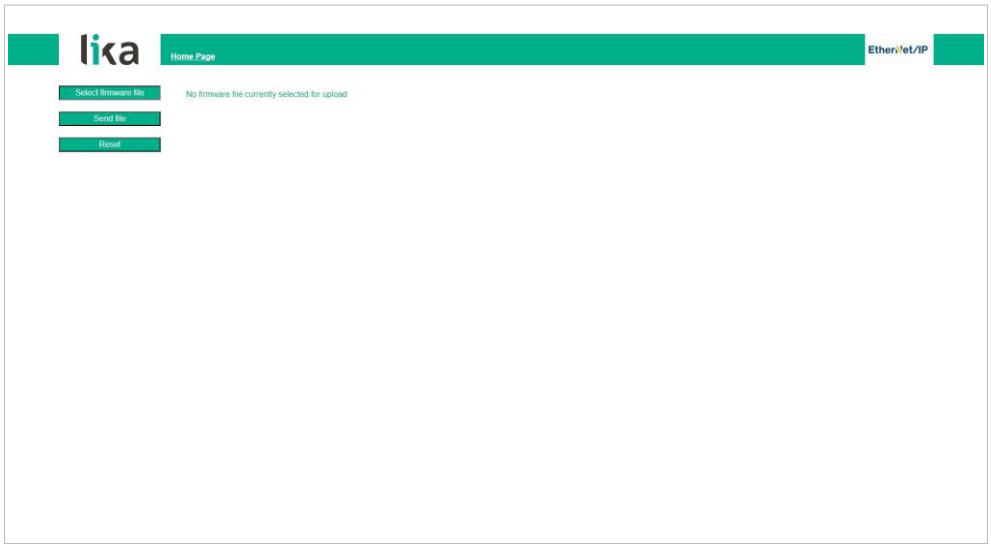


Figure 47 - Firmware Update page

6. press the **SELECT FIRMWARE FILE** button; once you press the **SELECT FIRMWARE FILE** button an **OPEN** dialog box appears on the screen: open the folder where the firmware updating .ZIP file released by Lika Electronic is located, select the file and confirm. Please check the file properties and ascertain that you are installing the correct update file;

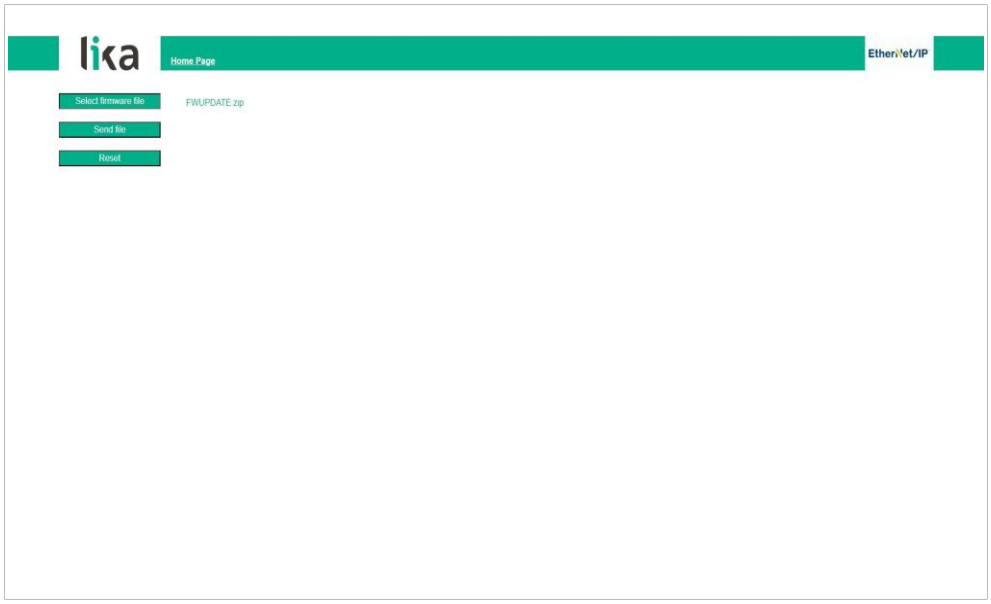


Figure 48 - Selecting the firmware update .zip file

WARNING
Before installation always ascertain that the firmware program is compatible with the hardware and software of the device.
Never turn the power supply off during the flash update operation.

7. press the **SEND FILE** button to start the upload of the firmware program;
8. during the operation and as soon as the operation is carried out successfully, some messages will appear on the screen;

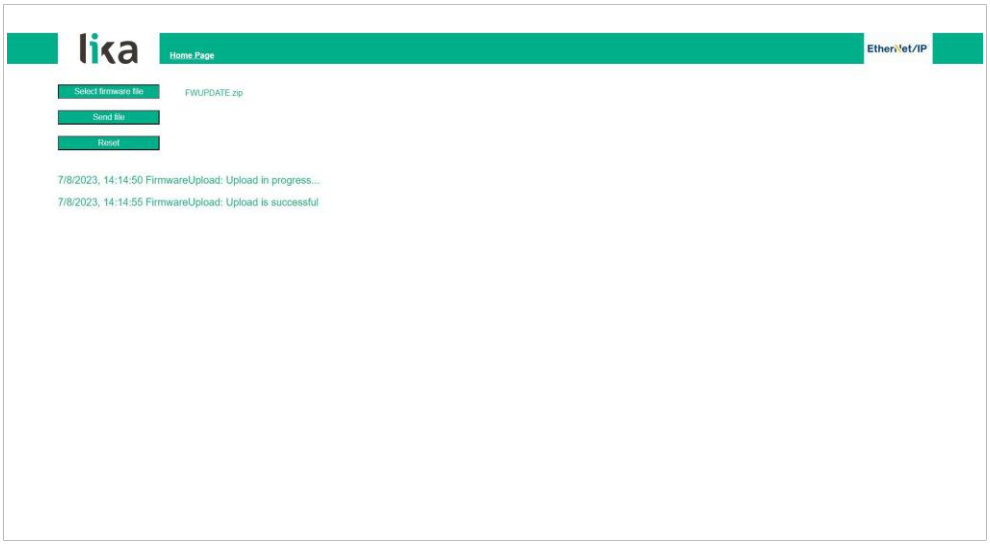


Figure 49 - Messages during firmware upload

9. finally press the **RESET** button to automatically reset and restart the encoder and complete the operation.

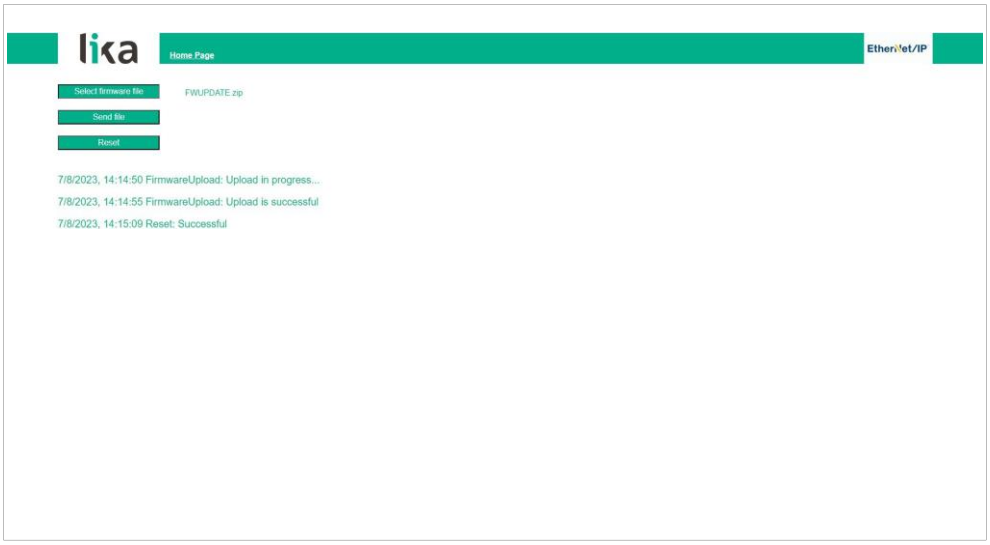


Figure 50 - Firmware update process accomplished

NOTE
While downloading the firmware updating program, unexpected conditions may arise which could lead to a failure of the installation process. When such a matter occurs, the download process cannot be carried out successfully and thus the operation is aborted. In case of flash update error, please switch the encoder off and then on again and retry the operation.

Press the **HOMEPAGE** button to move back to the Web server **Home** page.

6.7 Network configuration

Press the **IP CONFIG** button in the left navigation bar of the Web server **Home** page to enter the **Network IP Configuration** page. This page allows the operator to configure the TCP/IP properties, that is how the encoder communicates with other devices in the network.
For further information on the network communication parameters please refer to the "3.8 EtherNet/IP Node ID" section on page 39.

WARNING
The network configuration must be accomplished by skilled and competent personnel.

Figure 51 - Network Configuration page

WARNING
Only competent technicians, who are properly trained, have adequate experience and are familiar with computer architecture, network design and operating systems should configure the network communication parameters.
The inappropriate setting of the network parameters results in an incorrect operation of the system.

In this page it is possible to set the parameters that affect the proper communication of the encoder in the TCP/IP network: IP address, Subnet mask, and Gateway.
After setting each parameter, press the **SEND** button on the right in the same line to confirm the value and save it permanently. A message will appear on the screen to inform about the correctness of the operation (for example, **IP Config setting executed correctly**).

The following table summarizes the default software IP address and the network configuration parameters.

IP Parameter	Value
IP address	192.168.1.10
Subnet mask	255.255.255.0
Default Gateway	0.0.0.0

WARNING
After any setting please note down the configuration values to have access to the encoder and the Web server pages in the future.

Press the **HOMEPAGE** button to move back to the Web server **Home** page.

7 Default parameters list

Default values are expressed in hexadecimal (h) notation, unless otherwise indicated.

7.1 Attributes of the Class 01h Identity Object

Parameters list	Default values		
01-01-01 Vendor ID	0299h = Lika Electronic srl		
01-01-02 Device type	0022h = Encoder Device Profile		
01-01-03 Product code	0082h = AM58x 27-bit multiturn encoder		
01-01-04 Revision	Device dependent		
01-01-06 Serial number	Device dependent		
01-01-07 Product name	AM58X-1314 = 27-bit multiturn encoder		

7.2 Attributes of the Class 23h Position Sensor Object

Parameters list	Default values		
23-01-0B Position Sensor type	0001h = singleturn absolute rotary encoder 0002h = multiturn absolute rotary encoder		
23-01-0C Direction Counting Toggle	00h = CW rotation		
23-01-0E Scaling Function Control	01h = enabled		
23-01-10 Measuring Units per Span	8,192		
23-01-11 Total Measuring Range 32 bit	134,217,728		

23-01-13 Preset Value 32 bit	0		
23-01-19 Velocity Format	1F04h = cps		
23-01-2A Physical Resolution Span	8,192		
23-01-2B Number of Spans	16,384r		
23-01-2D Supported Alarms	1003h		
23-01-30 Supported Warnings	6000h		
23-01-64 Application FW Version	Device dependent		
23-01-65 Hardware Version	Device dependent		
23-01-6A Encoder Serial Number	Device dependent		

This device is to be supplied by a Class 2 Circuit or Low-Voltage Limited Energy or Energy Source not exceeding 30 Vdc. Refer to the order code for supply voltage rate.
Ce dispositif doit être alimenté par un circuit de Classe 2 ou à très basse tension ou bien en appliquant une tension maxi de 30Vcc. Voir le code de commande pour la tension d'alimentation.

Dispose separately