# Industrial Protocols

## Modbus TCP

## EtherNet/IP

# User Manual

**Version 1.0**

**Jan, 2018**

www.oringnet.com

# COPYRIGHT NOTICE



## TRADEMARKS

ORing is a registered trademark of ORing Industrial Networking Corp.
All other trademarks belong to their respective owners.

## REGULATORY COMPLIANCE STATEMENT

Product(s) associated with this publication complies/comply with all applicable regulations. Please refer to the Technical Specifications section for more details.

## WARRANTY

ORing warrants that all ORing products are free from defects in material and workmanship for a specified warranty period from the invoice date (5 years for most products). ORing will repair or replace products found by ORing to be defective within this warranty period, with shipment expenses apportioned by ORing and the distributor. This warranty does not cover product modifications or repairs done by persons other than ORing-approved personnel, and this warranty does not apply to ORing products that are misused, abused, improperly installed, or damaged by accidents.
Please refer to the Technical Specifications section for the actual warranty period(s) of the product(s) associated with this publication.

## DISCLAIMER

Information in this publication is intended to be accurate. ORing shall not be responsible for its use or infringements on third-parties as a result of its use. There may occasionally be unintentional errors on this publication. ORing reserves the right to revise the contents of this publication without notice.

## CONTACT INFORMATION

**ORing Industrial Networking Corp.**
3F., NO.542-2, Jhongjheng Rd., Sindian District, New Taipei City 231, Taiwan, R.O.C.
Tel: + 886 2 2218 1066 // Fax: + 886 2 2218 1014
Website: www.oringnet.com

**Technical Support**
E-mail: support@oringnet.com

**Sales Contact**
E-mail: sales@oringnet.com (Headquarters)
sales@oring-china.com (China)

# Table of Content

# Modbus TCP

## 1.1 Introduction

Modbus TCP is a variant of the Modbus family protocol which simply the Modbus RTU protocol with a TCP interface. It's is a vendor-neutral communication protocols intended for supervision and control of automation equipment. Modbus TCP uses TCP/IP and Ethernet to carry the data of the Modbus message between compatible devices. That is, it allows the transmission of information over Ethernet between the switch and devices such as PLCs, sensors, and meters. Users can read the information through the Modbus TCP based software and monitor the status of the switch easily. For more information, please visit http://www.modbus.org/

## 1.2 Modbus object types

Modbus is a Master-Slave architecture, where the Master device like a PLC sends request to Slaves device like a switch, and then the Slave device responds data to Master device.

There are four object types provided by a Slave device to a Master device, and the ORing switches supports object type 4: Input register.

| Code | Object type | Access | Size |
|---|---|---|---|
| 1 | Coil | Read-write | 1 bit |
| 2 | Discrete input | Read-only | 1 bit |
| 3 | Holding register | Read-write | 16 bits |
| 4 | Input register | Read-only | 16 bits |

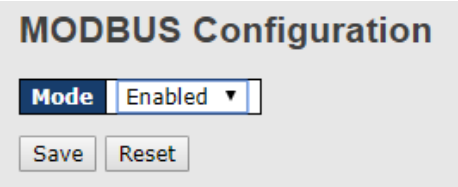
## 1.3 Configuring Modbus TCP on Oring Switches

Follow the steps below to configure Modbus TCP on Oring Switches.

Step 1: Login the web page of the switch

Step 2: Find Modbus Configuration Page on Basic Setting→Modbus TCP

Step 3: Select Enable and click Save to enable the Modbus TCP.

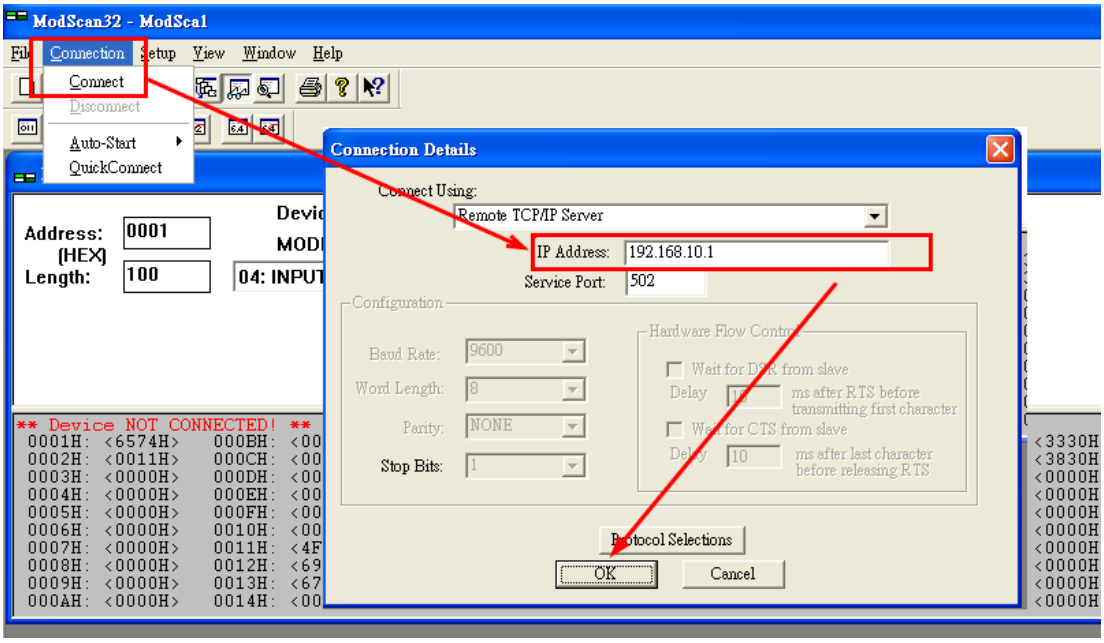


# 1.4 Monitoring Modbus TCP information

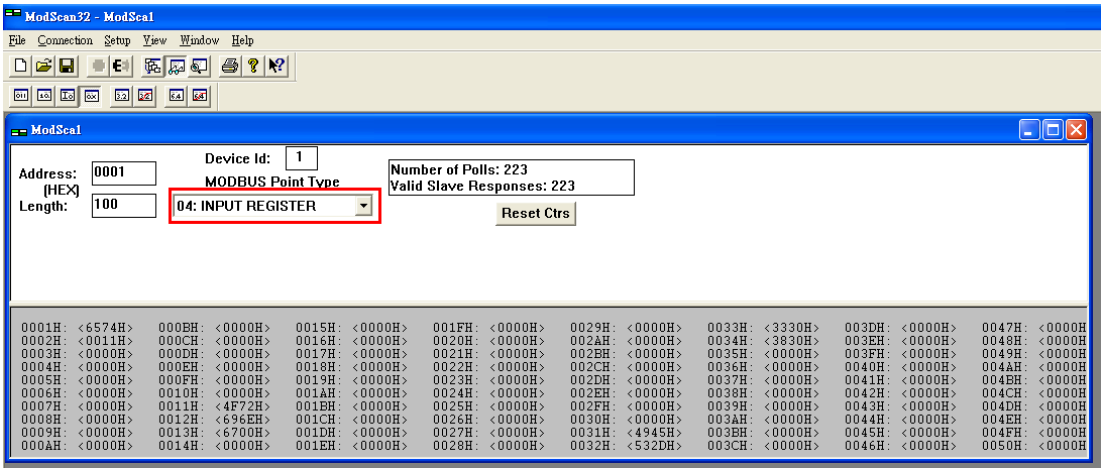
After configuring Modbus TCP on Oring Switches, users can read the information through any Modbus TCP based software and monitor the status of the switch easily.

Step 1: Start a Modbus TCP read software like “Modscan 32”.

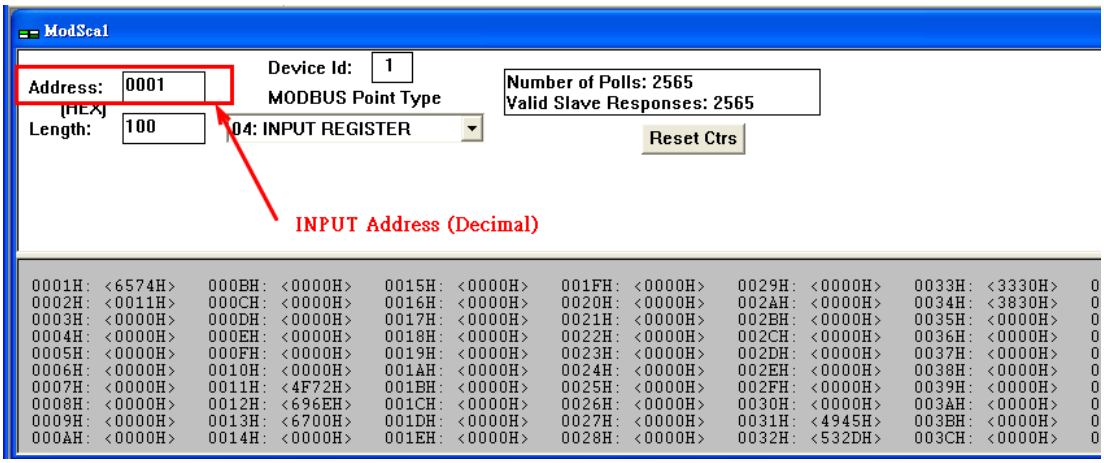
Step 2: Connect the IP Address of the switch




Step 3: Select Modbus Point Type = 04 :INPUT REGSTER



Step 4: Input the Modbus Address (Decimal) which you want to see in the data display area.




# 1.5 Modbus TCP Mapping Table

## 1.5.1 System Information

| Tag Name | Address Offset(HEX) | Modbus Address(DEC) | Data Type | Data Length (Words) | Description |
|---|---|---|---|---|---|
| Vender | 0x0000 | 1 | Word | 1 | 0x6574 |
| UnitID | 0x0001 | 2 | Word | 1 | Unit ID (Ethernet = 1) |
| Product Code | 0x0002 | 3 | Word | 1 | Product Code |
| Switch Port Number | 0x0008 | 9 | Word | 1 | Port Numbers |
| VenderName | 0x0010 | 17 | String | 16 | ORing |
| ProductName | 0x0030 | 49 | String | 16 | Product Name |
| Version | 0x0051 | 82 | Word | 2 | firmware & kernel version |
| Firmware Release Date | 0x0053 | 84 | Word | 2 | For example:<br>2017-05-06 at 09 o'clock<br>Word 0 = 0 x 0609<br>Word 1 = 0 x 1705 |
| MacAddress | 0x0055 | 86 | Word | 3 | For example:<br>0x001e 0x9411 0x2233 |
| Power 1 | 0x0058 | 89 | Word | 1 | 0x0000: off<br>0x0001: on |
| Power 2 | 0x0059 | 90 | Word | 1 | 0x0000: off<br>0x0001: on |
| Fault LED Status | 0x005a | 91 | Word | 1 | 0x0000: off<br>0x0001: on |
| IPAddress | 0x0090 | 145 | String | 16 | For example:<br>192.168.10.1 |
| SysName | 0x0100 | 257 | String | 128 | System Name |
| SysDescription | 0x0200 | 513 | String | 128 | System Description |
| SysLocation | 0x0300 | 769 | String | 128 | System Location |
| SysContact | 0x0400 | 1025 | String | 128 | System Contact |

## 1.5.2 Port Information

| Tag Name | Address Offset(HEX) | Modbus Address(DEC) | Data Type | Data Length (Words) | Description |
|---|---|---|---|---|---|
| PortStatus Port 1~N | 0x1000 ~0x100N | 4097 | Word | 1 | 0x0000: link down<br>0x0001: link up<br>0x0002: disable |
| PortSpeed Port 1~N | 0x1100 ~0x110N | 4353 | Word | 1 | 0x0000: 10M-Half<br>0x0001: 10M-Full<br>0x0002: 100M-Half<br>0x0003: 100M-Full<br>0x0000: 10M-Half<br>0x0001: 10M-Full<br>0x0002: 100M-Half<br>0x0003: 100M-Full<br>0x0004: 1G-Half<br>0x0005: 1G-Full<br>0x0006: 2.5G-Half<br>0x0007: 2.5G-Full<br>0x0008: 10G-Half<br>0x0009: 10G-Full |
| PortFlowCtrl Port 1~N | 0x1200 ~0x120N | 4609 | Word | 1 | 0x0000: off<br>0x0001: on |
| PortDescription Port 1~N | 0x1400 ~0x140N | 5121 | String | 16 | Port Type, eg. "100M", "1G", "10G", "SFP1G", "1GCOMBO" ... |

## 1.5.3 PoE Information

| Tag Name | Address Offset(HEX) | Modbus Address(DEC) | Data Type | Data Length (Words) | Description |
|---|---|---|---|---|---|
| Port PoE Voltage | 0x1800~ | 6145 | Word | 1 | For example:<br>0x0005 = 5V |
| Port PoE Current | 0x1830~ | 6193 | Word | 1 | For example:<br>0x000D = 13A |
| Port PoE Power | 0x1860~ | 6241 | Word | 1 | For example:<br>0x000A = 10W |

## 1.5.4 Packets Information

| Tag Name | Address Offset(HEX) | Modbus Address(DEC) | Data Type | Data Length (Words) | Description |
|---|---|---|---|---|---|
| Port Tx Packets | 0x2000~ | 8193 | Word | 2 | For example:<br>Amount = 44332211<br>Word 0 = 4433<br>Word 1 = 2211 |
| Port Rx Packets | 0x2100~ | 8449 | Word | 2 | For example:<br>Amount = 44332211<br>Word 0 = 4433<br>Word 1 = 2211 |
| port Tx Error Packets | 0x2200~ | 8705 | Word | 2 | For example:<br>Amount = 44332211<br>Word 0 = 4433<br>Word 1 = 2211 |
| port Rx Error Packets | 0x2300~ | 8961 | Word | 2 | For example:<br>Amount = 44332211<br>Word 0 = 4433<br>Word 1 = 2211 |

### 1.5.5 Redundant Information

| Tag Name | Address Offset(HEX) | Modbus Address(DEC) | Data Type | Data Length (Words) | Description |
|---|---|---|---|---|---|
| Redundancy Protocol | 0x3000 | 12289 | Word | 1 | 0x0000: none<br>0x0001: RSTP<br>0x0002: O-Ring<br>0x0003: O-Chain |
| RSTP Root | 0x3100 | 12545 | Word | 1 | 0x0000: not root bridge<br>0x0001: root bridge<br>0xffff: protocol not enabled |
| RSTP Port Status Port 1~N | 0x3200~ | 12801 | Word | 1 | 0x0000: disabled<br>0x0001: not RSTP port<br>0x0002: link down<br>0x0003: discarding<br>0x0004: learning<br>0x0005: forwarding<br>0xffff: protocol not enabled |
| O-Ring Master/Slave | 0x3300 | 13057 | Word | 1 | 0x0000: slave<br>0x0001: master<br>0xffff: protocol not enabled |
| O-Ring Port1 Status | 0x3301 | 13058 | Word | 1 | 0x0002: link down<br>0x0003: blocked<br>0x0005: forwarding<br>0xffff: protocol not enabled |
| O-Ring Port2 Status | 0x3302 | 13059 | Word | 1 | 0x0002: link down<br>0x0003: blocked<br>0x0005: forwarding<br>0xffff: protocol not enabled |
| Couple Ring Enable | 0x3303 | 13060 | Word | 1 | 0x0000: off<br>0x0001: on |
| Couple Port Status | 0x3304 | 13061 | Word | 1 | 0x0002: link down<br>0x0003: blocked<br>0x0005: forwarding<br>0xffff: protocol not enabled |
| O-Chain Edge Switch | 0x3700 | 14081 | Word | 1 | 0x0000: not edge switch<br>0x0001: edge switch<br>0xffff: protocol not enabled |
| O-Chain Port1 Status | 0x3701 | 14082 | Word | 1 | 0x0002: link down<br>0x0003: blocked<br>0x0005: forwarding<br>0xffff: protocol not enabled |
| O-Chain Port2 Status | 0x3702 | 14083 | Word | 1 | 0x0002: link down<br>0x0003: blocked<br>0x0005: forwarding<br>0xffff: protocol not enabled |

# EtherNet/IP

## 2.1 Introduction

EtherNet/IP is an industrial network protocol that adapts the Common Industrial Protocol (CIP) to standard Ethernet. The EtherNet/IP and CIP are managed by ODVA, Inc., which founded in 1995 with over 300 corporate members. CIP uses its object-oriented design to provide EtherNet/IP with the services and device profiles needed for real-time control applications and to promote consistent implementation of automation functions across products. With EtherNet/IP, users can integrate I/O control, device configuration and data collection across multiple networks like Rockwell systems or other EtherNet/IP systems. For more information, please visit https://www.odva.org

## 2.2 Messaging Types

EtherNet/IP supports two types of communications: Explicit and Implicit Messaging.

- Explicit messaging – a client/server communications which uses TCP/IP over Ethernet.
- Implicit messaging – a time critical, scheduled producer/consumer communication which uses UDP/IP over Ethernet.

For large data communication, Explicit Messaging is the preferred choice because bandwidth is saved, as data is only requested when necessary.

For high speed, real-time applications, Implicit Messaging is the preferred choice. Implicit messaging is often referred to as I/O messaging because it's frequently used for communication between a controller and remote I/O.

## 2.3 Configuring EtherNet/IP on Oring Switches

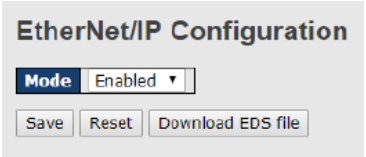
Follow the steps below to configure EtherNet/IP on Oring Switches.

Step 1: Login the web page of the switch

Step 2: Find EtherNet/IP Configuration Page on Basic Setting→ EtherNet/IP

Step 3: Select Enable and click Save to enable the EtherNet/IP.

Step 4: Click Download EDS File.




Note: Since Ethernet/IP devices can generate a lot of multicast traffic, users are recommended to enable IGMP snooping to avoid overloading.

# 2.4 Monitoring EtherNet/IP Information

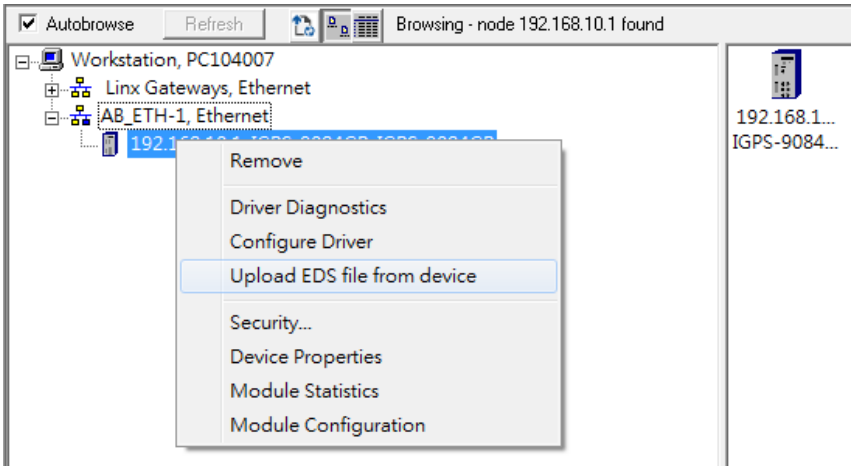
After configuring EtherNet/IP on Oring Switches, users can read the information through any EtherNet/IP based software and monitor the status of the switch easily.

**Create Device on EtherNet/IP based software**

Step 1: Start an EtherNet/IP management software like “Rockwell Studio 5000”.

Step 2: Upload the EDS File of the switch. You can right click the device to upload EDS file or download the EDS file from web page and upload it using other configuration tool.

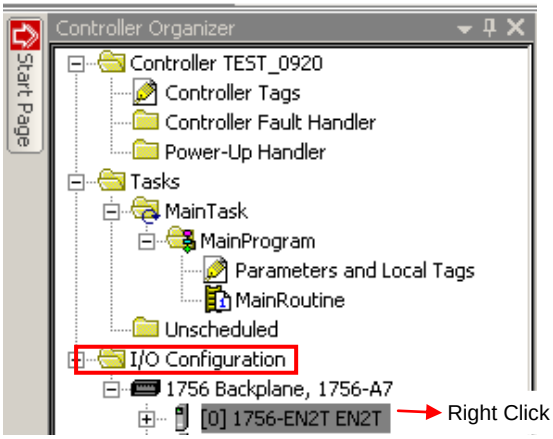
EDS files are simple text files used by network configuration tools to help to identify products and easily commission them on a network.

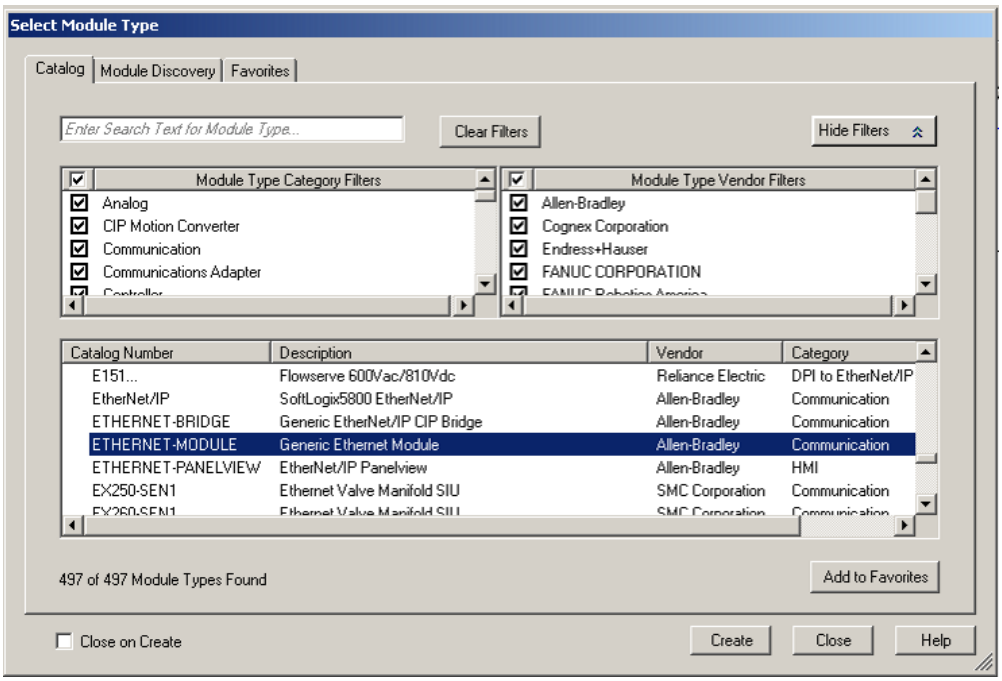



**Monitor Implicit Messaging**

Implicit messaging is often referred to as I/O messaging because it’s frequently used for communication between a controller and remote I/O.

Step 1: Find I/O configuration→Find your PLC device→ Right click to add a module→ Choose “Generic Ethernet Module” and click “Create”





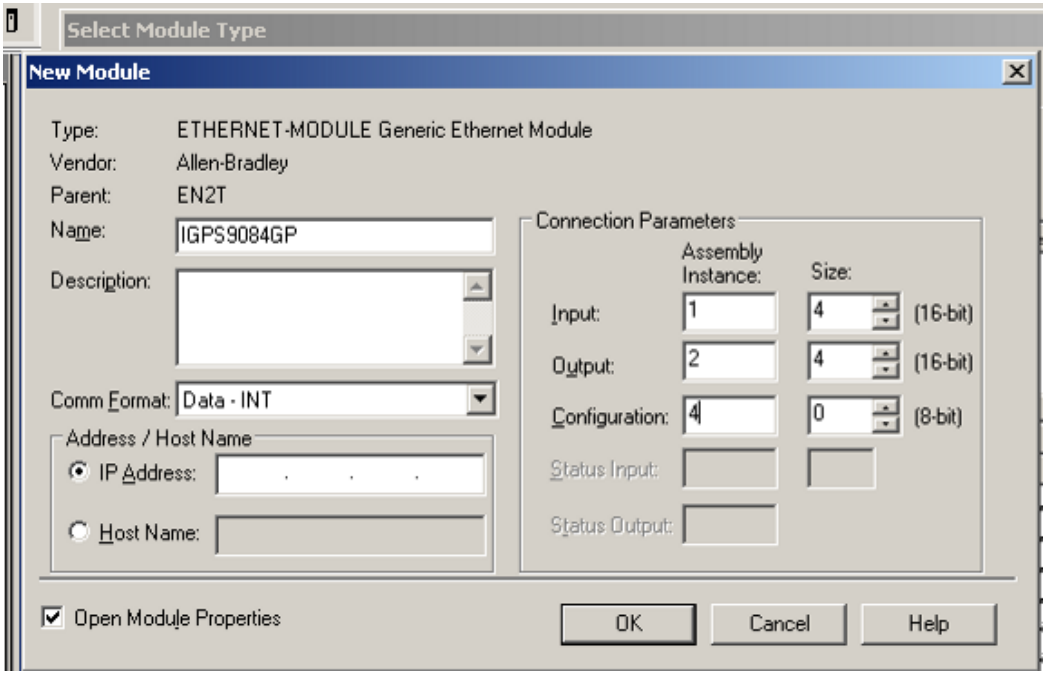

Step 2: Set the ORing switch on this module.

Set the following value and click OK.

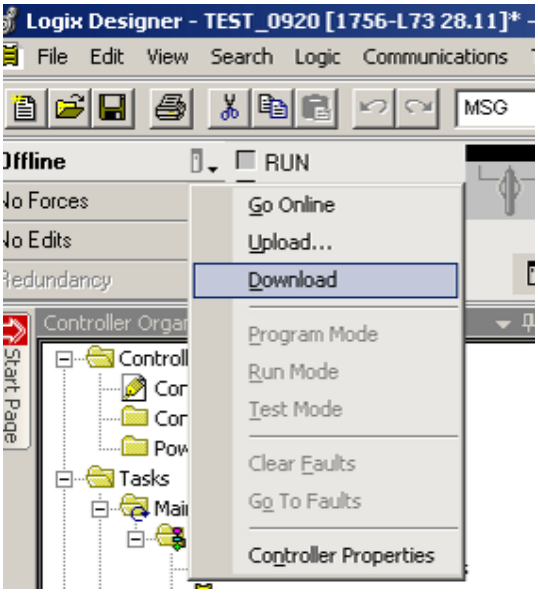
- Name: ORing switch model name
- Comm Format: Data-INT
- IP Address: the default IP address of ORing switch is (192.168.10.1)
- Input: Assembly Instance=1, Size=4

  Output: Assembly Instance=2, Size=4

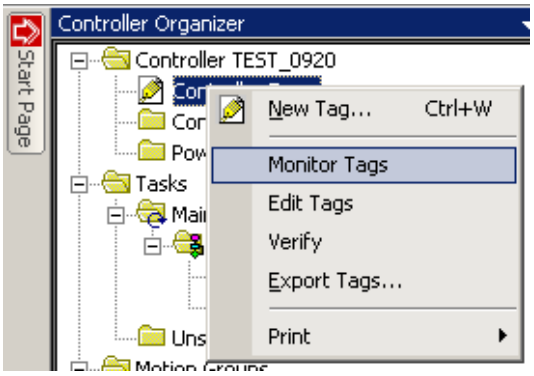
  Configuration: Assembly Instance=4, Size=0



Step 3: Download the configuration to save the setting.

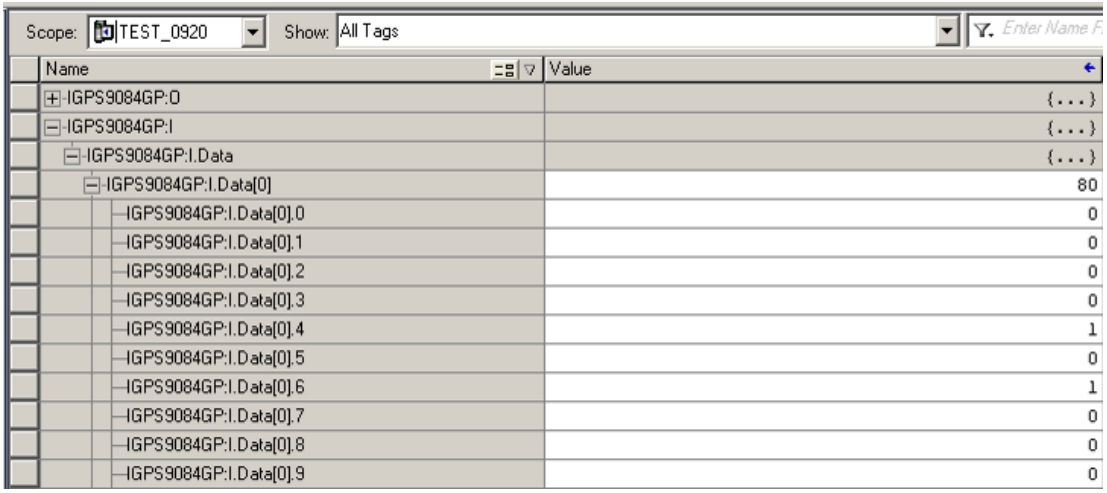



Step 4: Right click “Controller Tags” to Monitor Tags.




Step 5: the Implicit messaging of the ORing Switch will be shown as below.

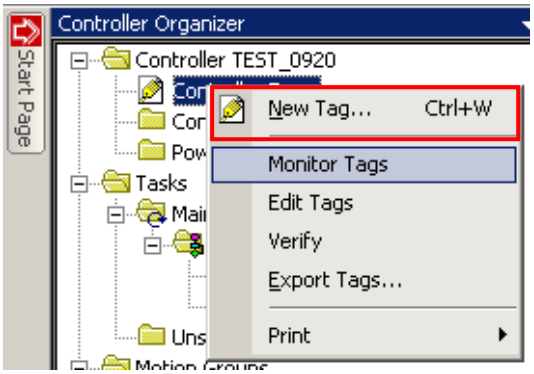
Check the Chapter 2.5 (CIP Objects of EtherNet/IP) to get the details of the values.

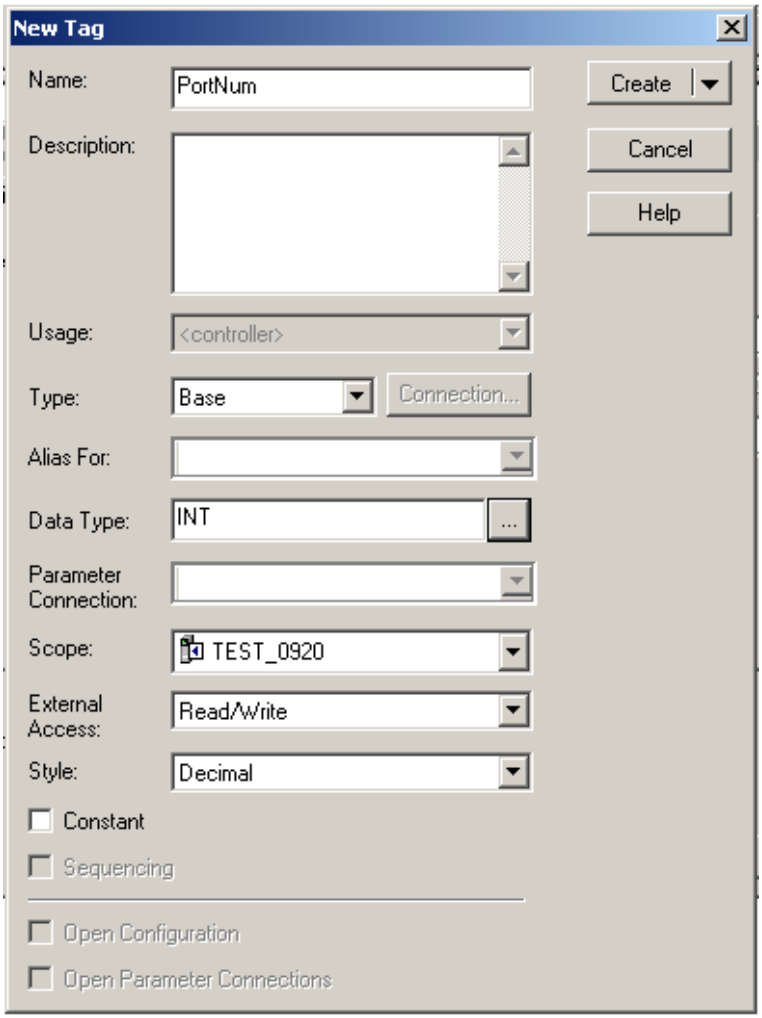


**Monitor Explicit Messaging**

The messages ORing supported are defined in Chapter 2.5 (CIP Objects of EtherNet/IP).

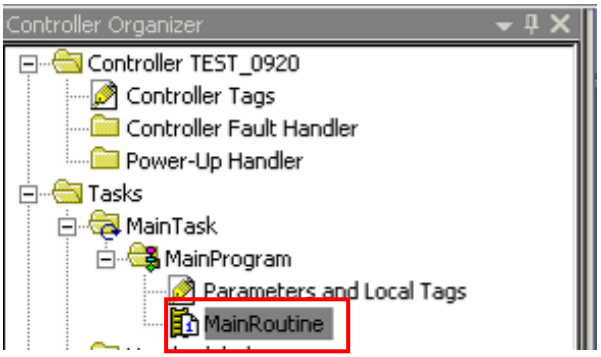
Step 1: Right clicks “Controller Tags” and click “New Tag” to create a new tag for messaging.

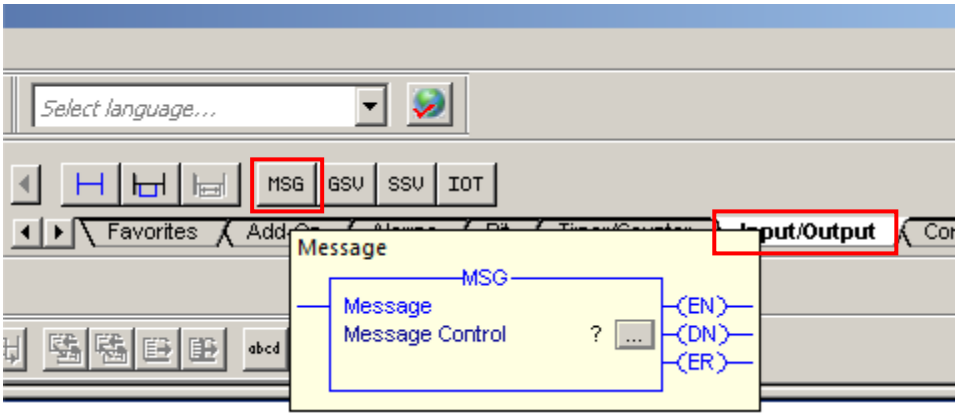


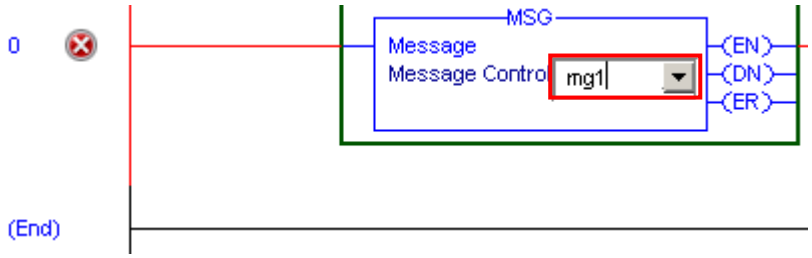



Step 2: Add a Message.

Click “MainRoutine” → click “Input/Output” Page on the top right side→ click “MSG” on the top side→ Enter the Message Control name.



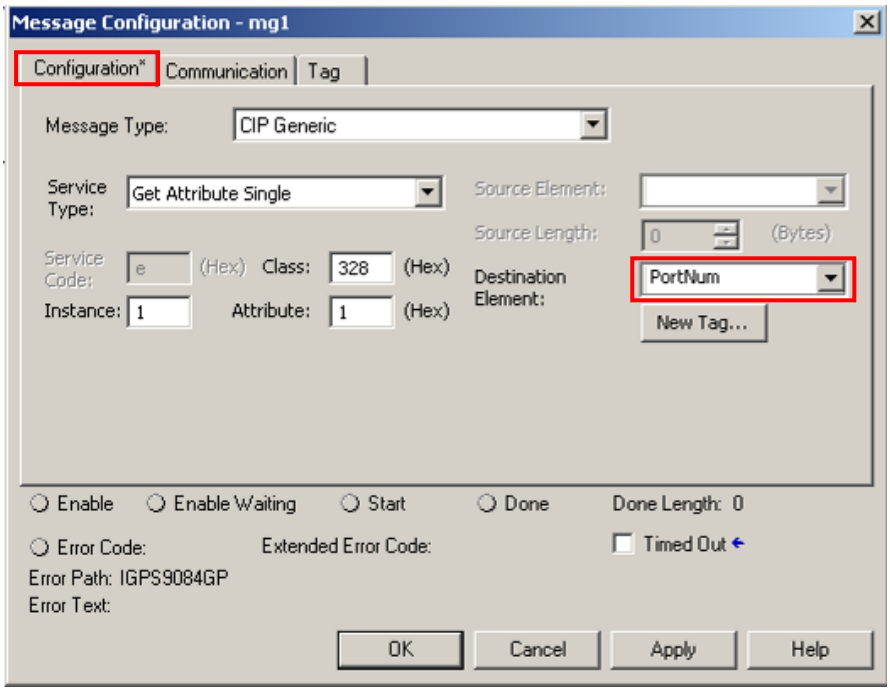



Step 3: Set the Message Configuration based on the value defined in Chapter 2.5 (CIP Objects of EtherNet/IP).

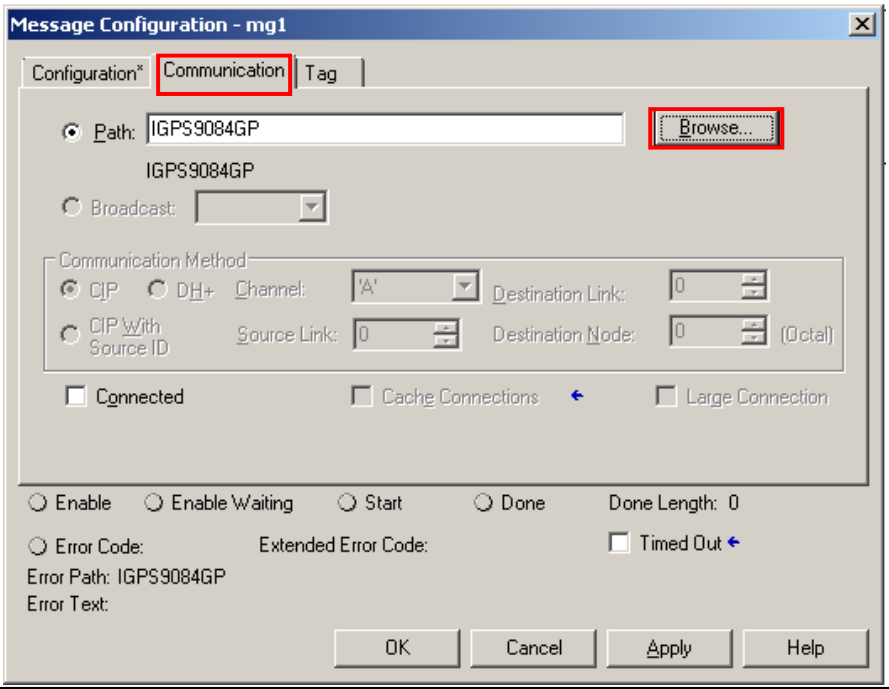
For example, Switch Port Number is defined in ORing Networking Object (class=328), and the other information show as below.

| Attr ID | Access Rule | Name | Data Type | Description |
|---|---|---|---|---|
| 1 | Get | Switch Port Number | USINT (8) | Switch max port number |

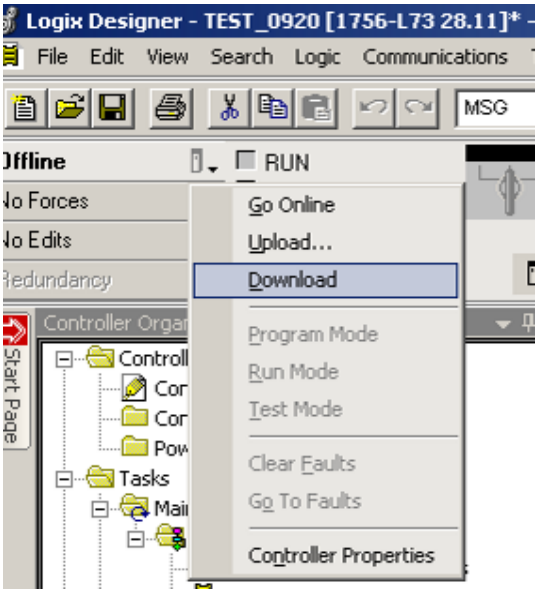
Enter the value defined in Chapter 2.5 and choose the Destination Tag created on step 1.



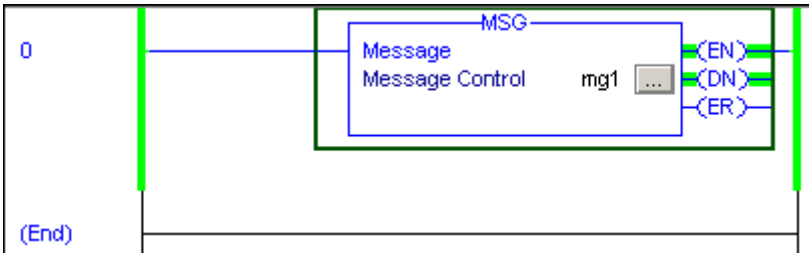

Select the ORing switch on Communication Page, and then click OK.

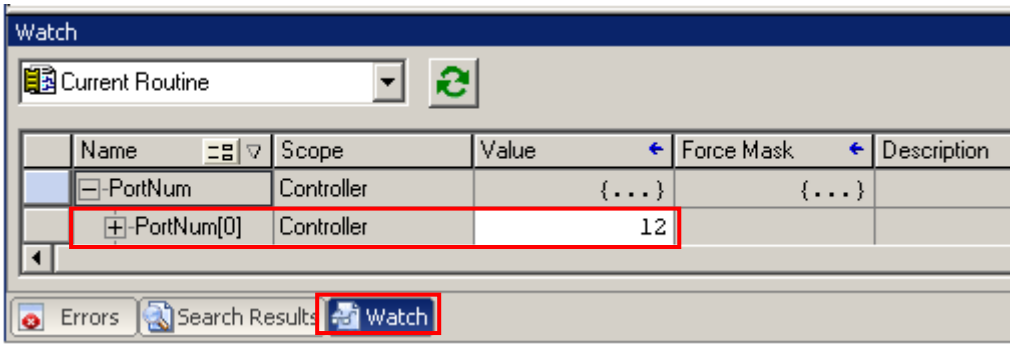


Step 4: Download the configuration to save the setting.




Step 5: The message mg1 is working. Click the Watch page on the bottom side to see the port number information.



## 2.5 CIP Objects of EtherNet/IP

ORing switches support the following objects for PLCs and SCADA systems to monitor:

- Identity Object
- TCP/IP Interface Object
- Ethernet Link Object
- Assembly Object
- Port Object
- ORing Networking Object (Vendor Specific)

The supported attributes and services of the above objects are introduced in the table below, including the access rules for each attribute. To understand the details of each attribute of the standard objects, refer to the official documents of CIP introduction (Vol. 1) and the EtherNet/IP Adaptation of CIP (Vol. 2).

### 2.5.1 Identity Object

The Class code of Identity object is 0x01 (Defined in CIP Vol1, 5A-2).
There is one instance of this object in our product. It stores the information of the production and the device. The following tables summarize the class attributes and the instance attributes.

#### 2.5.1.1 Class Attribute List

| Attribute ID | Access Rule | Name | Data Type | Description |
|---|---|---|---|---|
| 1 | Get | Revision | UINT (16) | Revision of this object |
| 2 | Get | Max Instance | UINT (16) | Maximum instance number of an object currently created in this class level of the device |

#### 2.5.1.2 Instance Attribute List

| Attribute ID | Access Rule | Name | Struct | Data Type | Description |
|---|---|---|---|---|---|
| 1 | Get | Vendor ID | | UINT (16) | 1405, the vendor ID of ORing. |
| 2 | Get | Device Type | | UINT (16) | 0x38F, ORing EtherNet/IP Device |
| 3 | Get | Product Code | | UINT (16) | Product Code |
| 4 | Get | Revision | | (Struct) | The version of the Identity object |

| | | | | | |
|---|---|---|---|---|---|
| | | | Major | USINT (8) | The structure member, major |
| | | | Minor | USINT (8) | The structure member, minor. |
| 5 | Get | Status | | WORD (16) | Not used |
| 6 | Get | Serial Number | | UDINT (32) | The serial number of each device |
| 7 | Get | Product Name | | SHORT STRING | The product name in human-readable format |

### 2.5.1.3 Common Service List

| Service Code | Implementation | | Service Name | Description |
|---|---|---|---|---|
| | Class | Instance | | |
| 0x01 | | ✓ | Get_Attributes_All | Returns the contents of all attributes of the class |
| 0x05 | | ✓ | Reset | Invokes the reset service for the device |
| 0x0E | ✓ | ✓ | Get_Attribute_Single | Used to read an object instance attribute. |

## 2.5.2 Assembly Object

The ORing switch support static assembly object for CIP I/O messaging.

The Class code is 0x04 (Defined in CIP Vol 1, 5A-5).

There are three instances of this object as the following.

| | Instance Number | Size (16 bit) |
|---|---|---|
| Input | 1 | 4 |
| Output | 2 | 4 |
| Configuration | 4 | 0 |

The Input means the data is produced by switch which includes the information and status report to the originator for monitoring. The Output means the data is generated by the originator (remote host) and is consumed by switch.

### 2.5.2.1 Class Attribute List

| Attr ID | Access Rule | Name | Data Type | Description |
|---|---|---|---|---|
| 1 | Get | Revision | UINT (16) | Revision of this object |

### 2.5.2.2 Instance Attribute List

| Attr ID | Access Rule | Name | (Struct.) | Data Type | Description |
|---|---|---|---|---|---|
| 3 | Get/Set | Data | | Array of BYTE | The implicit messaging content |
| 4 | Get | Size | | UINT (16) | Number of bytes in Attr. 3 |

### 2.5.2.3 Common Service List

| Service Code | Implementation | | Service Name | Description |
|---|---|---|---|---|
| | Class | Instance | | |
| 0x0E | ✓ | ✓ | Get_Attribute_Single | Used to read an object instance attribute. |
| 0x10 | | ✓ | Set_Attribute_Single | Used to write an object instance attribute |

### 2.5.2.4 I/O Messaging Content

| Direction | I/O data | Size | Value & Description |
|---|---|---|---|
| Input | Port Link Status | ULINT (64) | Please refer to ORing Networking Object Attr ID 7. |
| Output | Port Enable | ULINT (64) | Please refer to ORing Networking Object Attr ID 6. |

## 2.5.3 TCP/IP Interface Object

The Class code of TCP/IP Interface object is 0xf5 (Defined in CIP Vol2, 5-4). There is one instance of this object. The following tables summarize the attributes of this object.

### 2.5.3.1 Class Attribute List

| Attr ID | Access Rule | Name | Data Type | Description |
|---|---|---|---|---|
| 1 | Get | Revision | UINT (16) | Revision of this object. |

### 2.5.3.2 Instance Attribute List

| Attr ID | Access Rule | Name | (Struct.) | Data Type | Description |
|---|---|---|---|---|---|
| 1 | Get | Status | | DWORD (32) | Interface status<br>0 = The Interface Configuration attribute has not been configured.<br>1 = The Interface Configuration attribute contains valid configuration obtained from BOOTP, DHCP or non-volatile storage. |
| 2 | Get | Configuration Capability | | DWORD (32) | Interface capability flags<br>Bit map of capability flags:<br>Bit 0: BOOTP Client<br>Bit 1: DNS Client<br>Bit 2: DHCP Client<br>Bit 3: DHCP-DNS Updat<br>Bit 4: Configuration Settable |

| | | | | | |
|---|---|---|---|---|---|
| 3 | Get/Set | Configuration Control | | DWORD (32) | Interface control flags<br>Bit map of control flags:<br>Bit 0 to 3: Startup Configuration<br>0 = The device shall use the interface configuration values previously stored (for example, in non-volatile memory or via hardware witches).<br>1 = The device shall obtain its interface configuration values via BOOTP.<br>2 = The device shall obtain its interface configuration values via DHCP upon start-up.<br>3 to15 = Reserved. |
| 4 | Get | Physical Link Object | | (Struct.) | Path to physical link object |
| | | | Path Size | UINT (16) | Size of Path |
| | | | Path | Padded EPATH | Logical segments identifying the physical link object |
| 5 | Get/Set | Interface Configuration | | (Struct.) | TCP/IP network interface configuration |
| | | | IP Address | UDINT (32) | The device's IP address |
| | | | Network Mask | UDINT (32) | The device's network mask |
| | | | Gateway Address | UDINT (32) | Default gateway address |
| | | | Name Server | UDINT (32) | Primary name server |
| | | | Name Server2 | UDINT (32) | Secondary name server |
| | | | Domain Name | STRING | Default domain name |
| 6 | Get/Set | Host Name | | STRING | Host name |
| 13 | Get/Set | Encapsulation Inactivity Timeout | | UINT(16) | Number of seconds of inactivity before TCP connection or DTLS session is closed |

### 2.5.3.3 Common Service List

| Service Code | Implementation | | Service Name | Description |
|---|---|---|---|---|
| | Class | Instance | | |
| 0x0E | ✓ | ✓ | Get_Attribute_Single | Used to read an object instance attribute. |
| 0x10 | | ✓ | Set_Attribute_Single | Used to write an object instance attribute |

## 2.5.4 Ethernet Link Object

The Class code of Ethernet Link object is 0xf6 (Defined in CIP Vol2, 5-5). For each switch port, there is an instance of this class. The following table shows the mapping of instance number and the switch port number.

| Instance Number | Mapping to |
|---|---|
| 0 | Ethernet Link class |
| 1 | 1st switch port |
| 2 | 2nd switch port |
| … | … |

### 2.5.4.1 Class Attribute List

| Attr ID | Access Rule | Name | Data Type | Description |
|---|---|---|---|---|
| 1 | Get | Revision | UINT (16) | Revision of this object |
| 2 | Get | Max Instance | UINT (16) | Maximum instance number of an object currently created in this class level of the device |
| 3 | Get | Number of Instances | UINT (16) | Number of object instances currently created in this class level of the device |

### 2.5.4.2 Instance attribute list

The following tables summarize the attributes of the Ethernet Link object.

There are some vendor specific attributes in the table (Starting from attribute Id 100).

| Attr ID | Access Rule | Name | (Struct.) | Data Type | Description |
|---|---|---|---|---|---|
| 1 | Get | Interface Speed | | UDINT (32) | Interface speed currently in use (Speed in Mbps, e.g., 0, 10, 100, 1000, etc.) |
| 2 | Get | Interface Flags | | DWORD (32) | Refer to the Interface Flags table. |
| 3 | Get | Physical Address | | ARRAY of 6 USINT(8) | MAC layer address (The System MAC address). |
| 4 | Get | Interface Counters | | (Struct.) | Counters relevant to the receipt of packets. |
| | | | In Octets | UDINT (32) | Octets received on the interface. |
| | | | In Ucast Packets | UDINT (32) | Unicast packets received on the interface. |
| | | | In NUcast Packets | UDINT (32) | Non-unicast packets received on the interface. |
| | | | In Discards | UDINT (32) | Inbound packets received on the interface but are discarded. |
| | | | In Errors | UDINT (32) | Inbound packets that contain Errors (does not include In Discards). |
| | | | In Unknown Protos | UDINT (32 | Inbound packets with unknown protocol |
| | | | Out Octets | UDINT (32) | Octets sent on the interface. |
| | | | Out Ucast Packets | UDINT (32) | Unicast packets sent on the interface. |
| | | | Out NUcast Packets | UDINT (32) | Non-unicast packets sent on the interface. |
| | | | Out Discards | UDINT (32) | Discarded outbound packets. |
| | | | Out Errors | UDINT (32) | Outbound packets that contain errors. |
| 5 | Get | Media Counters | | (Struct.) | |
| | | | Alignment Errors | UDINT (32) | Received frames that are not an integral number of octets in length. |
| | | | FCS Errors | UDINT (32) | Received frames that do not pass the FCS check. |
| | | | Single Collisions | UDINT (32) | Successfully transmitted frames which experienced exactly one collision. |
| | | | Multiple Collisions | UDINT (32) | Successfully transmitted frames which experienced more than one collision. |
| | | | SQE Test Errors | UDINT (32) | Number of times the SQE test error message is generated. |
| | | | Deferred Transmissions | UDINT (32) | Frames for which first transmission attempt is delayed because the medium is busy. |

| | | | | | |
|---|---|---|---|---|---|
| | | | Late Collisions | UDINT (32) | Number of times a collision is detected later than 512 bit times into the transmission of a packet. |
| | | | Excessive Collisions | UDINT (32) | Frames for which transmission fails due to excessive collisions. |
| | | | MAC Transmit Errors | UDINT (32) | Frames for which transmission fails due to an internal MAC sublayer transmit error. |
| | | | Carrier Sense Errors | UDINT (32) | Times that the carrier sense condition was lost or never asserted when attempting to transmit a frame. |
| | | | Frame Too Long | UDINT (32) | Received frames that exceed the maximum permitted frame size. |
| | | | MAC Receive Errors | UDINT (32) | Frames for which reception on an interface fails due to an internal MAC sublayer receive error. |
| 6 | Get/Set | Interface Control | | (Struct.) | Configuration for physical interface. |
| | | | Control Bits | WORD (16) | Bit 0: Auto-Negotiate<br>Value 0: Force<br>Value 1: Auto-Nego<br>Bit 1: Half/Full Duplex<br>Value 0: half duplex<br>Value 1: full duplex<br>Bit 2 to 15: Reserved, all zero |
| | | | Forced Interface Speed | UINT (16) | Speed at which the interface shall be forced to operate. |
| 10 | Get | Interface Label | | SHORT_STRING | Human readable identification |
| 11 | Get | Interface Capability | | (Struct.) | Indication of capabilities of the interface |
| | | | Capability Bits | DWORD | Interface capabilities, other than speed/duplex |
| | | | Speed/Duplex Options | (Struct.) | Indicates speed/duplex pairs supported in the Interface Control attribute |
| | | | | USINT | Speed/Duplex Array Count |
| | | | | (Array Struct.): | Speed/Duplex Array |
| | | | | UINT | Interface Speed |
| | | | | USINT | Interface Duplex Mode |
| 12 | Get | HC Interface Counters | | (Struct.) | High Capacity Interface Counters. |
| | | | HCInOctets | ULINT | The total number of octets received on the interface. |
| | | | HCInUcastPkts | ULINT | Unicast packets received on the interface. |
| | | | HCInMcastPkts | ULINT | Multicast packets received on the interface. |
| | | | HCInBcastPkts | ULINT | Broadcast packets received on the interface. |
| | | | HCOutOctets | ULINT | Octets sent on the interface. |
| | | | HCOutUcastPkts | ULINT | Unicast packets sent on the interface. |
| | | | HCOutMcastPkts | ULINT | Multicast packets sent on the interface. |
| | | | HCOutBcastPkts | ULINT | Broadcast packets sent on the interface. |
| 13 | Get | HC Media Counters | | (Struct.) | High Capacity Media Counters. |
| | | | HCStatsAlignmentErrors | ULINT | Frames received that are not an integral number of octets in length and do not pass the FCS check. |
| | | | HCStatsFCSErrors | ULINT | Frames received that are an integral number of octets in length but do not pass the FCS check. |

| | | | | | |
|---|---|---|---|---|---|
| | | | HCStatsInternalMacTransmitErrors | ULINT | Frames for which transmission fails due to an internal MAC sublayer transmit error. |
| | | | HCStatsFrameTooLongs | ULINT | Frames received that exceed the maximum permitted frame size. |
| | | | HCStatsInternalMacReceiveErrors | ULINT | Frames for which reception on an interface fails due to an internal MAC sublayer receive error. |
| | | | HCStatsSymbolErrors | ULINT | Number of times there was an invalid data symbol on the media when a valid carrier was present. |

### 2.5.4.3 Interface Flags

| Bit(s) | Called | Definition |
|---|---|---|
| 0 | Link Status | 0 indicates an inactive link;<br>1 indicates an active link. |
| 1 | Half/Full Duplex | 0 indicates half duplex;<br>1 indicates full duplex. |
| 2-4 | Negotiation Status | Indicates the status of link auto-negotiation<br>0 = Auto-negotiation in progress.<br>1 = Auto-negotiation and speed detection failed. Using default values for speed and duplex. Default values are product-dependent; recommended defaults are 10Mbps and half duplex.<br>2 = Auto negotiation failed but detected speed. Duplex was defaulted. Default value is product-dependent; recommended default is half duplex.<br>3 = Successfully negotiated speed and duplex.<br>4 = Auto-negotiation not attempted. Forced speed and duplex. |
| 5 | Manual Setting Requires Reset | 0 indicates the interface can activate changes to link parameters (auto-negotiate, duplex mode, interface speed) automatically. 1 indicates the device requires a Reset service be issued to its Identity Object in order for the 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. The meaning of this is product-specific. For example, an AUI/MII interface might detect no transceiver attached, or a radio modem might detect no antenna attached. In contrast to the soft, possibly self-correcting nature of the Link Status being inactive, this is assumed a hard-fault requiring user intervention. |
| 7~31 | Reserved. | Shall be set to zero |

### 2.5.4.4 Common Service List

| Service Code | Implementation | | Service Name | Description |
|---|---|---|---|---|
| | Class | Instance | | |
| 0x0E | ✓ | ✓ | Get_Attribute_Single | Used to read an object instance attribute. |
| 0x10 | | ✓ | Set_Attribute_Single | Used to write an object instance attribute |
| 0x4C | | ✓ | Get_and_Clear | Gets then clears the specified attribute. Supported by Attr 4, 5, 12, and 13. |

## 2.5.5 Port Object

The port object represents the underlying interface of CIP which is EtherNet/IP.

The class code is 0xf4 (Defined in CIP Vol 1, 3-7). There is one instance of this object.

The instance attribute “Port Type” identifies the CIP adaptation.

### 2.5.5.1 Class Attribute List

| Attr ID | Access Rule | Name | (Struct.) | Data Type | Description |
|---|---|---|---|---|---|
| 1 | Get | Revision | | UINT (16) | Revision of this object |
| 2 | Get | Max Instance | | UINT (16) | Maximum instance number of an object currently created in this class level of the device |
| 3 | Get | Number of Instances | | UINT (16) | Number of object instances currently created at this class level of the device. |
| 8 | Get | Entry Port | | UINT (16) | The attribute ID number of the last class attribute of the class definition implemented in the device |
| 9 | Get | Port Instance Info | | (Array of Struct.) | |
| | | | Port Type | UINT (16) | Enumerates the type of port |
| | | | Port Number | UINT (16) | CIP port number associated with this port |

### 2.5.5.2 Instance Attribute List

| Attr ID | Access Rule | Name | (Struct.) | Data Type | Description |
|---|---|---|---|---|---|
| 1 | Get | Port Type | | UINT (16) | Enumerates the type of port.<br>4 = EtherNet/IP. |
| 2 | Get | Port Number | | UINT (16) | CIP port number associated with this port. (Value 1 is reserved for internal product use) |
| 3 | Get | Link Object | | Struct.) | |
| | | | Path Length | UINT (16) | Number of 16 bit words in the following path. |
| | | | Link Path | Padded EPATH | Logical path segments that identify the object for this port. |
| 4 | Get | Port Name | | SHORT_STR ING | String which names the physical network port. The maximum number of characters in the string is 64. |
| 7 | Get | Port Number and Node Address | | Padded EPATH | This is a single Port Segment containing the Port Number of this port and the Link Address of this device on this port. |
| 10 | Get | Port Routing Capabilities | | DWORD | Bit string that defines the routing capabilities of this port |

### 2.5.5.3 Common Service List

| Service Code | Implementation | | Service Name | Description |
|---|---|---|---|---|
| | Class | Instance | | |
| 0x0E | ✓ | ✓ | Get_Attribute_Single | Used to read an object instance attribute. |
| 0x10 | | ✓ | Set_Attribute_Single | Used to write an object instance attribute |

## 2.5.6 ORing Networking Object (Vendor Specific)

The ORing Networking object includes system information and status. It can also be used to do the device diagnostic & configuration through Explicit Messaging. The class code is 0x328.

### 2.5.6.1 Class Attribute List

| Attr. ID | Access Rule | Name | Data Type | Description |
|---|---|---|---|---|
| 1 | Get | Revision | UINT (16) | Revision of this object |

### 2.5.6.2 Instance Attribute List

| Attr ID | Access Rule | Name | Data Type | Description |
|---|---|---|---|---|
| 1 | Get | Switch Port Number | USINT (8) | Switch max port number |
| 2 | Get | Firmware Version | SINT(4) | Firmware Version + Kernel Version |
| 3 | Get | Power status | USINT (8) | Bit 0: power status<br>Value 0: Off<br>Value 1: On<br>Bit 1: power 1 status<br>Value 0: Off<br>Value 1: On<br>Bit 2: power 2 status<br>Value 0: Off<br>Value 1: On<br>Bit 3: power 3 status<br>Value 0: Off<br>Value 1: On<br>Bit 3~7: Reserved |
| 4 | Get | Fault LED Status | SINT(8) | Value 0: Off<br>Value 1: On |
| 5 | Get | Port Exist | ULINT (64) | switch per port exist<br>Bit mask, the LSB indicates the first port.<br>Value 0: Not exist<br>Value 1: Exist |
| 6 | Get/Set | Port Enable | ULINT (64) | Switch per port enable<br>Bit mask, the LSB indicates the first port.<br>Value 0: Disable<br>Value 1: Enable |
| 7 | Get | Port Link Status | ULINT (64) | Switch per port link status<br>Bit mask, the LSB indicates the first port.<br>Value 0: Link down<br>Value 1: Link up |
| 8 | Get | Redundancy Protocol | USINT (8) | Bit 0: RSTP status<br>Value 0: Disable<br>Value 1: Enable<br>Bit 1: O-Ring status<br>Value 0: Disable<br>Value 1: Enable<br>Bit 2: O-Chain status<br>Value 0: Disable<br>Value 1: Enable<br>Bit 3~7: Reserved |
| 9 | Get | RSTP Root | SINT(8) | Value 0: not root bridge<br>Value 1: root bridge |
| 10 | Get | RSTP Port Status | SINT(port num) | Character 1 stands for port 1<br>Value 0x0: disable<br>Value 0x1: not RSTP port<br>Value 0x2: link down<br>Value 0x3: discarding<br>Value 0x4: learning<br>Value 0x5: forwarding<br>Value 0xf: protocol not enabled |

| | | | | |
|---|---|---|---|---|
| 11 | Get | O-Ring Master/Slave | SINT(8) | Value 0: slave<br>Value 1: master<br>Value 0xf: protocol not enabled |
| 12 | Get | O-Ring Port1 Status | USINT (8) | Value 0x2: link down<br>Value 0x3: discarding<br>Value 0x5: forwarding<br>Value 0xf: protocol not enabled |
| 13 | Get | O-Ring Port2 Status | USINT (8) | Value 0x2: link down<br>Value 0x3: discarding<br>Value 0x5: forwarding<br>Value 0xf: protocol not enabled |
| 14 | Get | Couple Ring Enable | SINT(8) | Value 0: Off<br>Value 1: On |
| 15 | Get | Couple Ring Port Status | USINT (8) | Value 0x2: link down<br>Value 0x3: discarding<br>Value 0x5: forwarding<br>Value 0xf: protocol not enabled |
| 16 | Get | Dual homing Ring Enable | SINT(8) | Value 0: Off<br>Value 1: On |
| 17 | Get | Dual homing Ring Port Status | USINT (8) | Value 0x2: link down<br>Value 0x3: discarding<br>Value 0x5: forwarding<br>Value 0xf: protocol not enabled |
| 18 | Get | O-Chain Edge Switch | SINT(8) | Value 0: not edge switch<br>Value 1: edge switch<br>Value 0xf: protocol not enabled |
| 19 | Get | O-Chain Port1 Status | USINT (8) | Value 0x2: link down<br>Value 0x3: blocked<br>Value 0x5: forwarding<br>Value 0xf: protocol not enabled |
| 20 | Get | O-Chain Port2 Status | USINT (8) | Value 0x2: link down<br>Value 0x3: blocked<br>Value 0x5: forwarding<br>Value 0xf: protocol not enabled |
| 21 | Get/Set | IGMP Snooping Enable | USINT (8) | IGMP snooping enable<br>Value 0: Disable<br>Value 1: Enable |
| 22 | Get/Set | Power 1 Relay | USINT (8) | Power input 1 failure warning<br>Value 0: Disable(default)<br>Value 1: Enable |
| 23 | Get/Set | Power 2 Relay | USINT (8) | Power input 2 failure warning<br>Value 0: Disable(default)<br>Value 1: Enable |
| 24 | Get/Set | Power 3 Relay | USINT (8) | Power input 3 failure warning<br>Value 0: Disable(default)<br>Value 1: Enable |
| 25 | Get | Device Up Time | UDINT (32) | Number of seconds since the device was powered up |
| 26 | Set | Reset Device | USINT (8) | Reboot and reset to default<br>Value 1: Reboot the device<br>Value 2: Reboot and Reset to default (EtherNet/IP will be disable) |

### 2.5.6.3 Common Service List

| Service Code | Implementation | | Service Name | Description |
|---|---|---|---|---|
| | Class | Instance | | |
| 0x0E | ✓ | ✓ | Get_Attribute_Single | Used to read an object instance attribute. |
| 0x10 | | ✓ | Set_Attribute_Single | Used to write an object instance attribute |