Logix 5000 Controllers EDS AOP Guidelines for Logix Designer

1756 ControlLogix, 1756 GuardLogix, 1769 CompactLogix, 1769 Compact GuardLogix, 1789 SoftLogix, 5069 CompactLogix, Emulate 5570

Rockwell Automation

Programming Manual

Original Instructions

Important User Information

Throughout this manual, when necessary, we use notes to make you aware of safety considerations.

WARNING: Identifies information about practices or circumstances that can cause an explosion in a hazardous environment, which may lead to personal injury or death, property damage, or economic loss.

ATTENTION: Identifies information about practices or circumstances that can lead to personal injury or death, property damage, or economic loss. Attentions help you identify a hazard, avoid a hazard, and recognize the consequence.

IMPORTANT: Identifies information that is critical for successful application and understanding of the product.

These labels may also be on or inside the equipment to provide specific precautions.

SHOCK HAZARD: Labels may be on or inside the equipment, for example, a drive or motor, to alert people that dangerous voltage may be present.

BURN HAZARD: Labels may be on or inside the equipment, for example, a drive or motor, to alert people that surfaces may reach dangerous temperatures.

ARC FLASH HAZARD: Labels may be on or inside the equipment, for example, a motor control center, to alert people to potential Arc Flash. Arc Flash will cause severe injury or death. Wear proper Personal Protective Equipment (PPE). Follow ALL Regulatory requirements for safe work practices and for Personal Protective Equipment (PPE).

The following icon may appear in the text of this document.

Identifies information that is useful and can help to make a process easier to do or easier to understand.

Rockwell Automation recognizes that some of the terms that are currently used in our industry and in this publication are not in alignment with the movement toward inclusive language in technology. We are proactively collaborating with industry peers to find alternatives to such terms and making changes to our products and content. Please excuse the use of such terms in our content while we implement these changes.

Summary of changes

This manual includes new and updated information. Use these reference tables to locate changed information.

Grammatical and editorial style changes are not included in this summary.

Global changes

This table identifies changes that apply to all information about a subject in the manual and the reason for the change. For example, the addition of new supported hardware, a software design change, or additional reference material would result in changes to all of the topics that deal with that subject.

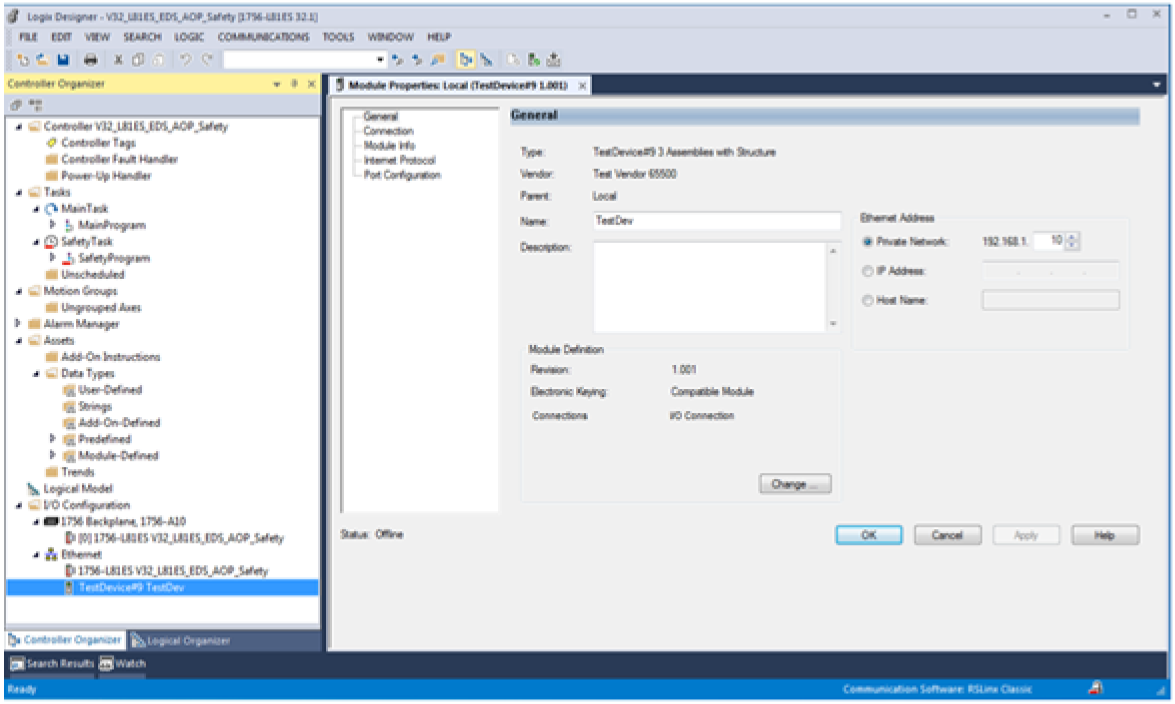
No global changes for this release.

New or enhanced features

Change	Topic
Added the `DataExchangeCategories` keyword.	DataExchangeCategories keyword on page 37
Added the `DataExchangeCatalogNumbers` keyword.	DataExchangeCatalogNumbers keyword on page 37
Added the `DataExchangeAliasTargetsN` keyword.	DataExchangeAliasTargetsN keyword on page 38

Preface

The document provides an overview of the Logix Designer application EDS Add-On Profile (AOP) feature, how Electronic Data Sheet (EDS) content is used by the feature, and guidelines for creating EDS content that integrate well within the Logix Designer application development environment. The EDS file format for the Common Industrial Protocol (CIP) networks is defined by ODVA.



Type: TestDevice#9 3 Assemblies with Structure
Vendor: Test Vendor 65500
Parent: Local
Name: TestDev
Description:

Ethernet Address
[✓ Private Network:] 192.168.1. 10
IP Address:
Host Name:

Module Definition
Revision: 1.001
Electronic Keying: Compatible Module
Connections: I/O Connection

Status: Offline

Rockwell Automation recognizes that some of the terms that are currently used in our industry and in this publication are not in alignment with the movement toward inclusive language in technology. We are proactively collaborating with industry peers to find alternatives to such terms and making changes to our products and content. Please excuse the use of such terms in our content while we implement these changes.

Studio 5000 environment

The Studio 5000 Automation Engineering & Design Environment® combines engineering and design elements into a common environment. The first element is the Studio 5000 Logix Designer® application. The Logix Designer application is the rebranding of RSLogix 5000®

software and will continue to be the product to program Logix 5000™ controllers for discrete, process, batch, motion, safety, and drive-based solutions.

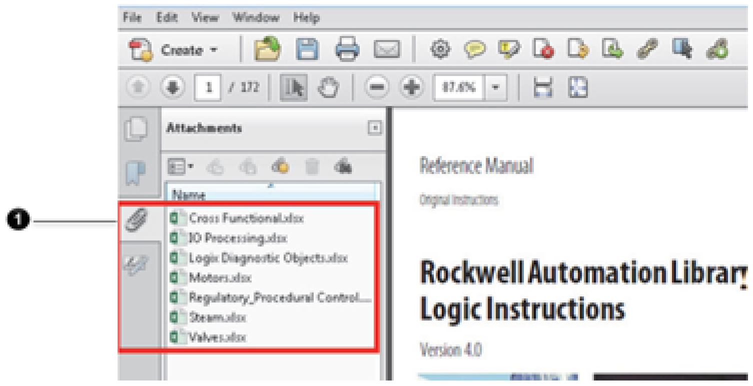
Copyright © 2022 FactoryTalk by Rockwell Automation, Inc.
All Rights Reserved.
This program is protected by U.S. and international copyright laws as described in the about box.

The Studio 5000® environment is the foundation for the future of Rockwell Automation® engineering design tools and capabilities. The Studio 5000 environment is the one place for design engineers to develop all elements of their control system.

Access the attachments

The attachments in this PDF file contain EDS files to assist with configuration.

To use an EDS file, select the attachments link shown in the left ribbon of the PDF and open the applicable EDS file. If the PDF file opens in a browser and the attachment link does not show, save the PDF file to a computer and then reopen the PDF file to view the attachment symbol.



Cross Functional.xlsx
IO Processing.xlsx
Logix Diagnostic Objects.xlsx
Motors.xlsx
Regulatory_Procedural Control...
Steam.xlsx
Valves.xlsx

Reference Manual
Original Instructions
Rockwell Automation Library
Logic Instructions
Version 4.0

Table 1.

Number	Description
1	Attachment symbol.

Additional resources

These documents contain additional information concerning related Rockwell Automation products.

Resource	Description
Industrial Automation Wiring and Grounding Guidelines, publication, 1770-4.1	Provides general guidelines for installing a Rockwell Automation industrial system.
Logix 5000™ Controllers Design Considerations, publication 1756-RM094.	Provides information to help design and plan Logix 5000 systems.
Rockwell Automation product certifications	Provides declarations of conformity, certificates, and other certification details.

View or download publications at https://www.rockwellautomation.com/en-us/support/documentation/literature-library.html. To order paper copies of technical documentation, contact a local Rockwell Automation distributor or sales representative.

Legal notices

Software and Cloud Services Agreement

Open Source Software Licenses

Introduction

Creating an EDS is an efficient way to enable a module to operate within the Rockwell Automation integrated architecture, and gives the module developer a better alternative to using the Generic Device AOP.

- The Generic Device AOP requires minimal development time because it uses the generic device user interface, but only specifies the basic connection parameters (assembly instance and the size of the Configuration, Input, and Output arrays). The module developer still creates documentation describing the instance values and tag details. The Generic Device AOP requires the user to manually enter often cryptic configuration parameters.

This document describes the specific parts of the EDS file that affect module configuration, I/O, and the user interface in the Logix Designer application. This document is intended for module developers creating products that integrate with the Logix Designer application who want their module configured by the EDS AOP using EDS content. This document is not intended for end-users applying products. EDS files should only be modified by the product vendor.

Module developers should:

- Be very familiar with the CIP EDS definition; see Chapter 7 in *THE CIP NETWORKS LIBRARY Volume 1, Common Industrial Protocol (CIP™)*, and *THE CIP NETWORKS LIBRARY Volume 2, EtherNet/ IP Adaptation of CIP*.
- Be familiar with the Logix Designer application dialog boxes. Dialog boxes are described in *Module Configuration dialog boxes*.
- Be familiar with Module-Defined data types generated by AOPs.

The Logix Designer application EDS AOP feature is limited to devices with a single CIP port that attach to the Ethernet bus.

The EDS AOP feature is supported in the Logix Designer application in v20.00.00 and later. Support for safety devices is in v32.00.00 and later. If v32.00.00 is installed on a PC with other Logix Designer application versions, then safety devices are supported in v24.00.00 and later. Support for v34.00.00 data types is in v36.00.00 or later. If v36.00.00 is installed on a PC with other Logix Designer application versions, then support for Logix Designer v34.00.00 data types are supported in v34.00.00 and later.

Unsupported devices:

- Devices with one or more CIP ports and a non-CIP backplane
- Devices with more than one CIP port that are routable

References

Documents referenced in or related to this document.

- *THE CIP NETWORKS LIBRARY Volume 1, Common Industrial Protocol (CIP™)*
- *THE CIP NETWORKS LIBRARY Volume 2, EtherNet/IP Adaptation of CIP*

Definitions, acronyms, and abbreviations

Definitions, acronyms, and abbreviations used in this document.

Terms	Definition
EDS	Electronic Data Sheet. A text file that contains configuration data for specific device types. The device vendor provides the EDS for a device. The EDS is required for compliance with ODVA standards.
AOP	Add-On Profile. Logix Designer application component separately installed and used for configuring one or more modules.
EDS AOP	EDS Add-On Profile. The Add-On Profile that uses EDS content to construct the profile to support various module types.
Profile	A subsystem of the Logix Designer application. Each supported module type has an associated profile. The profile provides information needed to establish topology and module-defined data types, as well as graphical user interface for configuration (in some cases).

General EDS file information

This section provides general information about EDS tools, EDS installation, and EDS icon files.

EDS file authoring tools

Create EDS files using any ASCII text editor that is capable of handling the character set specified in *THE CIP NETWORKS LIBRARY Volume 1, Common Industrial Protocol (CIP™)*.

Rockwell Automation recommends the use of a specialized EDS editing tool, such as the ODVA EZ- EDS tool.

Developers who need to create large families of EDS files of nearly identical modules may choose to use or develop tools that automatically create the EDS files for a set of modules within the product family. If EZ-EDS is not used to construct the file(s), use it to verify the EDS file(s) syntax (ODVA uses the latest version during conformance testing).

EDS file installation

The module vendor provides EDS files. The EDS files deploy in one or more ways:

- Embedded in the module as described in Section 5A-42 (File Object) of *THE CIP NETWORKS LIBRARY Volume 1, Common Industrial Protocol (CIP™)*.
- Provided on separate media, such as a CD-ROM.
- Downloaded from the module vendor website.

Use the EDS Hardware Installation Tool in the Logix Designer application version 20.00.00 or later to register EDS files. There is no need to shut down the Logix Designer application to register a new EDS, and the EDS is immediately available. The Logix Designer application does not allow the registering of an older version EDS file over a newer version EDS file.

The EDS Hardware Installation Tool issues warnings when it detects problems in the EDS (for example, about discarded entries).

IMPORTANT: Consider warnings when registering an EDS. For example, connections may be discarded because of a missing assembly referenced in the connection.

If an EDS file is not available for a module that exists in the Logix Designer application I/O tree, Logix Designer application displays a message that the module is not registered. This can happen when Logix Designer application detects a new module, or when copying a Logix Designer application project ACD file to a different computer that does not have the correct EDS files registered. Logix Designer application also displays a message if the device configuration has a newer version of the EDS than the one registered on the computer. In both cases, the Logix Designer application cannot display the module's **Module Properties** dialog box until the correct EDS file is installed. The Logix Designer application can still download the project to the controller.

Icon files

If there is no module icon file, Logix Designer application assigns a default icon. The icon allows a graphical representation of the module in the FactoryTalk® Linx™ and Logix Designer application environment. Embed icons in modules using the method described in Section 5A-42 (File Object) of *THE CIP NETWORKS LIBRARY Volume 1, Common Industrial Protocol (CIP™).*

Module icon files are highly recommended.

The EZ-EDS tool version 3.9, or later, supports embedding icons inside the EDS file. If icons are embedded inside the EDS using the method described in Chapter 7 of *THE CIP NETWORKS LIBRARY Volume 1, Common Industrial Protocol (CIP™)*, they are extracted automatically from the EDS.

Logix Designer application automatically assigns icons provided as separate files if their file names match the ones specified by the Icon keyword in the Device Description section, *[Device] Section*, of the EDS. The icon files must reside in the same directory as the EDS. If these conditions are not met, the EDS Hardware Installation Tool allows for manually assignment of an icon file when registering the EDS.

To create icon files, use these guidelines:

- At a minimum, create a 32x32 pixel, 16-color icon in the Microsoft format.

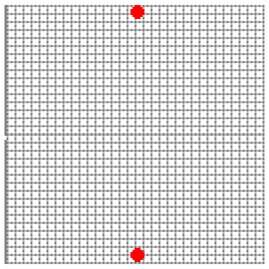
  For best results, define a:

  - 16x16 pixel, 16 color icon
  - 32x32 pixel, 16 color icon
  - 48x48 pixel, 256 color icon

    Store all three icons in one icon file.

- Center the icon in both the horizontal and vertical directions.
- Use transparent pixels for the pixels that are not part of the product. Do not use solid colors for backgrounds.

The red dots in the graphic below indicate the points at which the icon should be centered horizontally. This insures that the icon appears correctly.



In addition, make sure the Logix Designer application overlays (for example, inhibited device, error indication) are visible and evident in the Logix Designer application I/O tree.

License key

The standard form of the EDS AOP does not create named I/O tags. As with the generic EtherNet/IP module profile, the inputs, outputs, and configuration tags are displayed in the form of an array.

For the Logix Designer application to display named tags, the EDS file needs to include a license key. These license keys are issued by RA Technologies Inc. on a case-by-case basis. To obtain an EDS file with a license key, please email RA Technologies Inc. at RATech@RA.Rockwell.com.

Once the license key has been included in the file, there are three further requirements - specific to the I/O structure - that need to be met in order for named tags to be created correctly. These are:

1. The assemblies must have a structure that is built of parameters, as defined in this publication.
2. The parameter data types in the EDS file, and their respective minimum and maximum values need to match the data types that are supported in the Logix Designer application.
3. The assembly structure must follow alignment rules. Parameters that are a 2-byte data type must start on a 2-byte boundary and those that are a 4-byte data type must start on a 4-byte boundary. Parameters must be rearranged, or pad parameters added to meet the alignment rules. These changes must be made in both the EDS and the firmware of the device.

If these requirements are not met, the resulting tag structure will be an array - same as if no EDS license key is present.

A common problem is the use of unsigned data types as the data type for the EDS parameter. An unsigned parameter is omitted from the resulting tag structure if its maximum value is greater than the maximum value of its signed counterpart. For example, a USINT will be omitted in the resulting tag structure if its maximum value is 128 or greater. However in v36 (and in v34 and v35 with AOP Core v30 installed), unsigned parameters are now converted to unsigned tag members with support for the full range of the unsigned data type; therefore, this constraint does not apply.

Create an EDS file

This section gives an overview of the process of creating an EDS file for a module.

Connections

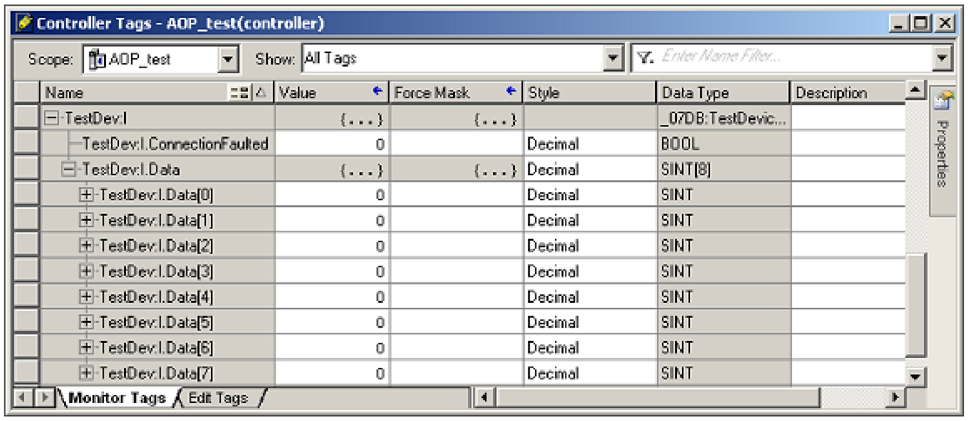
A connection identifies a CIP connection supported by the module and allows for setting some connection parameters that define controller-to-module behavior. Different connections may have different purposes. For example, optimize one connection for speed, and another connection for carrying more data. The connection also defines its Input, Output, and Configuration assemblies. See the sections in *Input and Output data structure and parameters* and *Configuration data structures and parameters*.

Connections are defined in the *[Connection Manager] section* of the EDS file. The connection information appears in the *Module Definition dialog box* and the *Connection tab.*

Input and Output data structures and parameters

Input tags

The values for Input tags are sent from the target module to the controller.

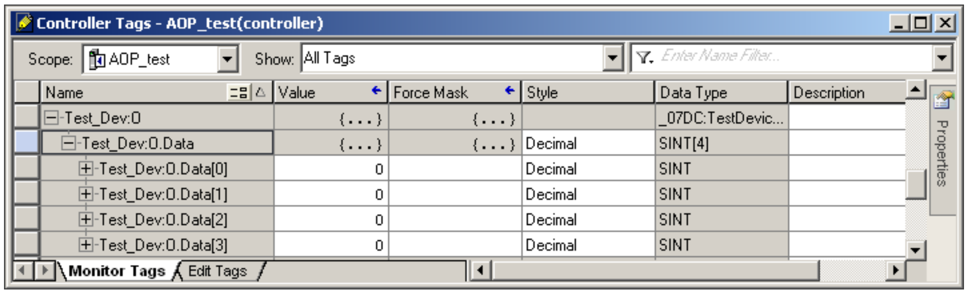

Controller Tags - AOP_test(controller)

Scope: AOP_test Show: All Tags

Name	Value	Force Mask	Style	Data Type	Description
TestDev:I	{...}	{...}		_07DB:TestDevic...	
TestDev:I.ConnectionFaulted	0		Decimal	BOOL	
TestDev:I.Data	{...}	{...}	Decimal	SINT[8]	
TestDev:I.Data[0]	0		Decimal	SINT	
TestDev:I.Data[1]	0		Decimal	SINT	
TestDev:I.Data[2]	0		Decimal	SINT	
TestDev:I.Data[3]	0		Decimal	SINT	
TestDev:I.Data[4]	0		Decimal	SINT	
TestDev:I.Data[5]	0		Decimal	SINT	
TestDev:I.Data[6]	0		Decimal	SINT	
TestDev:I.Data[7]	0		Decimal	SINT	

Output tags

The values for Output tags are sent from the controller to the target module.

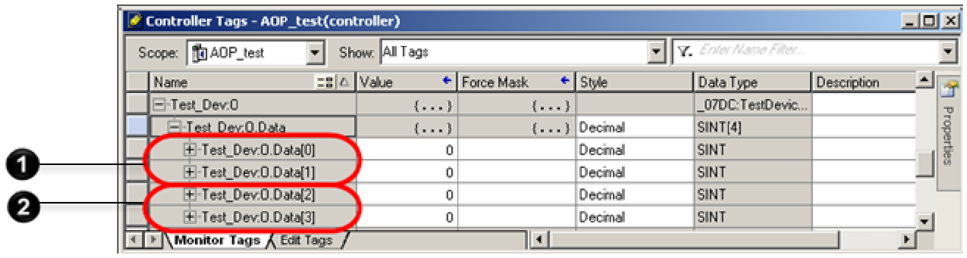

Controller Tags - AOP_test(controller)

Scope: AOP_test Show: All Tags

Name	Value	Force Mask	Style	Data Type	Description
Test_Dev:O	{...}	{...}		_07DC:TestDevic...	
Test_Dev:O.Data	{...}	{...}	Decimal	SINT[4]	
Test_Dev:O.Data[0]	0		Decimal	SINT	
Test_Dev:O.Data[1]	0		Decimal	SINT	
Test_Dev:O.Data[2]	0		Decimal	SINT	
Test_Dev:O.Data[3]	0		Decimal	SINT	

I/O parameter definition

Define I/O parameters in the *[Params] section* of the EDS file. This example shows two circled Output parameters (the first bit- enumerated).



Name	Value	Force Mask	Style	Data Type	Description
Test_Dev:0	{...}	{...}		_07DC:TestDevic...	
Test_Dev:0.Data	{...}	{...}	Decimal	SINT[4]	
Test_Dev:0.Data[0]	0		Decimal	SINT	
Test_Dev:0.Data[1]	0		Decimal	SINT	
Test_Dev:0.Data[2]	0		Decimal	SINT	
Test_Dev:0.Data[3]	0		Decimal	SINT	

Number	Description
1	Param2 corresponds to Data[0] and Data[1]
2	Param3 corresponds to Data[2] and Data[3]

The parameters are included in an assembly in the *[Assembly] section*.

```
Assem101 =
  "Output Data",
  ,
  ,
  0x0000,
  ,,
  ,Param2, $ Output Commands parameter
  ,Param3; $ Speed Reference parameter
```

The parameters are defined in the *[Params] section*.

```
Param2 =
  0,  $ reserved, shall equal 0
  ,,  $ Link Path Size, Link Path
  0x0000, $ Descriptor
  0xD2, $ Data Type
  2,  $ Data Size in bytes
  "Output Commands", $ name
  "",  $ units
  "",  $ help string
  ,,0, $ min, max, default data values
  ,,,, $ mult, div, base, offset scaling
  ,,,, $ mult, div, base, offset links
  ;  $ decimal places
```

```
Enum2 =
  0,"Stop",
  1,"Run",
  2,"Reverse";
```

```
Param3 =
  0,  $ reserved, shall equal 0
```

```
,,  $ Link Path Size, Link Path
0x0000, $ Descriptor
0xC7, $ Data Type
2,  $ Data Size in bytes
"Speed Reference", $ name
"rpm", $ units
"",  $ help string
0,32767,0, $ min, max, default data values
,,,, $ mult, div, base, offset scaling
,,,, $ mult, div, base, offset links
;  $ decimal places
```

Configuration data structures and parameters

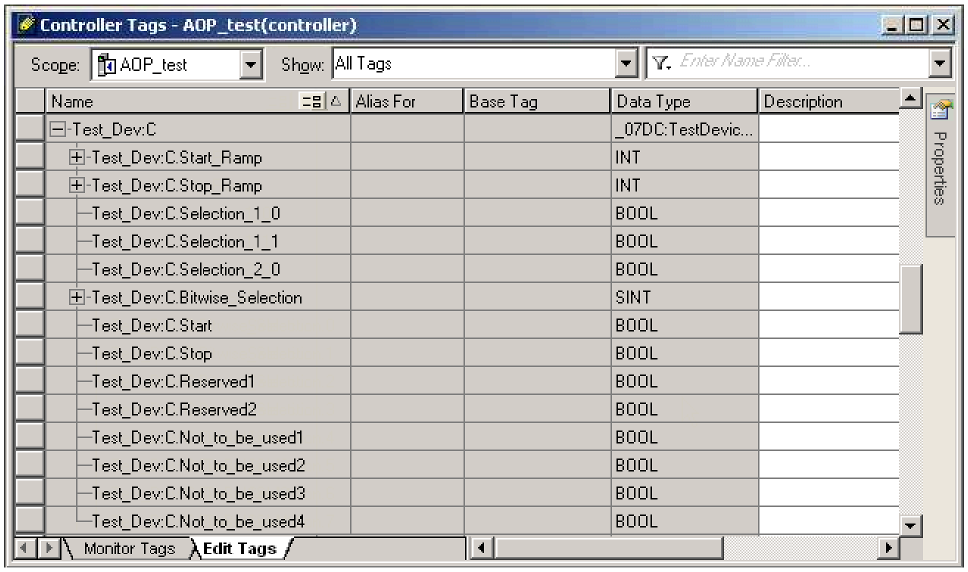
This section covers defining the module configuration data structures and parameters, and how they display in the Logix Designer application. In the EDS file, the *[Assembly] section* defines the data structures. The EDS file details are described later in this document, beginning with the *EDS File sections*.

Configuration tags

Configuration tags are parameters that are sent from a controller to a target module when an I/O connection is established. The tags configure and verify how a target module is configured.

The configuration data structure defines the configurable parameters for a module.

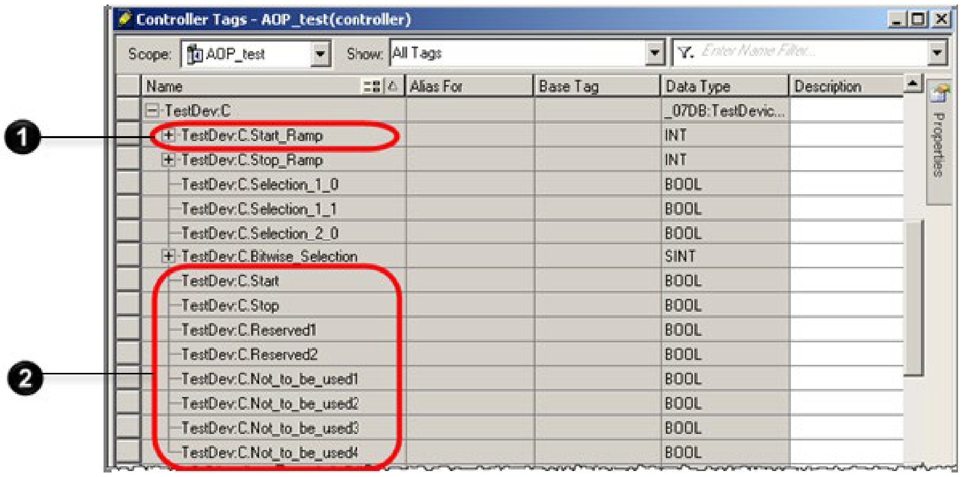
- Configuration tags may include descriptive tag names, as well as individual tag data types that define individual configuration parameters.
- In the EDS file, define a detailed assembly or a non-detailed assembly.
  - The detailed assembly results in a detailed data type. (This is highly recommended.)
  - The non-detailed assembly only defines the size of the assembly, and results in a non-detailed data type. See *[Assembly]* in *[Assembly section]* and *[Params]* in *[Params section]*.
- Type of configuration: The configuration is delivered using the Forward_Open service in the Data Segment. The configuration data is described by the ConnectionN entry.



Name	Alias For	Base Tag	Data Type	Description
Test_Dev:C			_07DC:TestDevic...	
Test_Dev:C.Start_Ramp			INT	
Test_Dev:C.Stop_Ramp			INT	
Test_Dev:C.Selection_1_0			BOOL	
Test_Dev:C.Selection_1_1			BOOL	
Test_Dev:C.Selection_2_0			BOOL	
Test_Dev:C.Bitwise_Selection			SINT	
Test_Dev:C.Start			BOOL	
Test_Dev:C.Stop			BOOL	
Test_Dev:C.Reserved1			BOOL	
Test_Dev:C.Reserved2			BOOL	
Test_Dev:C.Not_to_be_used1			BOOL	
Test_Dev:C.Not_to_be_used2			BOOL	
Test_Dev:C.Not_to_be_used3			BOOL	
Test_Dev:C.Not_to_be_used4			BOOL	

Configuration parameter definition

Define configuration parameters in the *[Params] section* of the EDS file. The example shows two circled configuration items. Safety configuration parameters do not show as tags in the Logix Designer application.



Name	Alias For	Base Tag	Data Type	Description
TestDev:C			_07DB:TestDevic...	
TestDev:C.Start_Ramp			INT	
TestDev:C.Stop_Ramp			INT	
TestDev:C.Selection_1_0			BOOL	
TestDev:C.Selection_1_1			BOOL	
TestDev:C.Selection_2_0			BOOL	
TestDev:C.Bitwise_Selection			SINT	
TestDev:C.Start			BOOL	
TestDev:C.Stop			BOOL	
TestDev:C.Reserved1			BOOL	
TestDev:C.Reserved2			BOOL	
TestDev:C.Not_to_be_used1			BOOL	
TestDev:C.Not_to_be_used2			BOOL	
TestDev:C.Not_to_be_used3			BOOL	
TestDev:C.Not_to_be_used4			BOOL	

Number	Description
❶	Corresponds to Param7
❷	Corresponds to Param11

The parameters are included in an assembly in the *[Assembly] section*:

```
Assem100 =
  "Configuration Assembly",
  ,
  ,
  0x0000,
  ,,
  16,Param7,  $ Start Ramp parameter
  16,Param8,
  2,Param9,  $ Selection1 parameter
  1,Param10,
  5,,
  8,Param11;
```

The parameters are defined in the *[Params] section:*

```
Param7 =
  0,   $ reserved, shall equal 0
  ,,   $ Link Path Size, Link Path
  0x0000,  $ Descriptor
  0xC7,  $ Data Type
  2,   $ Data Size in bytes
  "Start Ramp", $ name
  "ms",  $ units
  "This is the value for the start ramp", $ help string
  1,10000,1000 $ min, max, default data values
  ,,,,  $ mult, div, base, offset scaling
  ,,,,  $ mult, div, base, offset links
  ;   $ decimal places
```

Parameters can include bit-enumeration:

```
Param11 =
  0,   $ reserved, shall equal 0
  ,,   $ Link Path Size, Link Path
  0x0000,  $ Descriptor
  0xD1,  $ Data Type
  1,   $ Data Size in bytes
  "Bitwise Selection",  $ name
  "",    $ units
  "New Help String", $ help string
  ,,0,  $ min, max, default data values
  ,,,,  $ mult, div, base, offset scaling
  ,,,,  $ mult, div, base, offset links
  ;   $ decimal places
```

```
Enum11 =
  0,"Start",
  1,"Stop",
  2,"Reserved",
  3,"Reserved",
  4,"Not to be used",
  5,"Not to be used",
  6,"Not to be used",
  7,"Not to be used";
```

Parameters used in a Configuration Assembly should not include a link path. A link path makes it available to be written to separately by the Logix Designer application, which can result in configuration inconsistencies between the device and the Logix Designer application.

Multiple language support

Including support for multiple languages in the EDS file is strongly recommended. The Logix Designer application is translated into English, French, German, Spanish, Portuguese, Italian, Korean, Japanese, and Chinese. Names to translate include connections, configuration parameters, and assemblies. Tag member names in data structures always use the default string from the original definition in the EDS file.

When the Logix Designer application displays the **Module Properties** dialog box for a device with an EDS file, these rules control which text shows:

- For non-English languages, the Logix Designer application automatically uses the translated strings from the [Internationalization] section of the EDS file to match the current language setting.
- For English, or if there is no matching translated string, the Logix Designer application uses the English ("eng") string from the [Internationalization] section.
- If there is no English string in the [Internationalization] section, the Logix Designer application uses the default string from the original definition in the EDS file.

```
[Internationalization]      Param7 =
{
2,
{"eng",0xD0,4,"Start Ramp"},
{"deu",0xD0,4,"Start-Rampe"}
},
{
2,
{"eng",0xD0,4,"ms"},
{"deu",0xD0,4,"ms"}
},
{
2,
{"eng",0xD0,4,"The value for the start ramp"},
{"deu",0xD0,4,"Der Wert für die Start-Rampe"}
};
```

The first group of EDS keywords support internationalization. Always translate:

- ProdName
- ConnectionN

The second group of EDS keywords support internationalization. Only translate if they represent items that appear in the **Module Definition** dialog box:

- ParamN
- EnumN
- GroupN
- AssemN, AssemExaN

See *[Internationalization] section* for more information.

Module Configuration dialog boxes

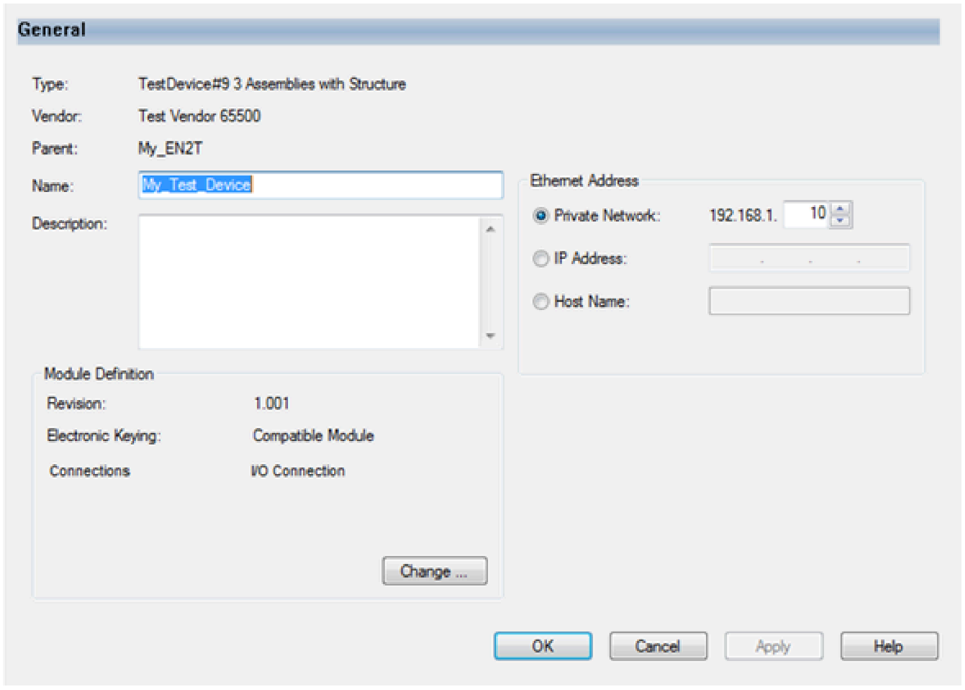
The Logix Designer application uses tabbed dialog boxes with a common look and feel to display the information from an EDS file and from the corresponding module when online. This simplifies device configuration in the integrated architecture. This section lists the dialog boxes, describes their functions, and references the EDS file sections that affect the dialog boxes in the Logix Designer application. Use this section to understand how the EDS AOP displays the information from the EDS in the Logix Designer user interface.

General tab

Use the **General** tab to view and configure module properties.

- View module type and vendor name. See *ProdTypeStr keyword* and *VendName keyword*.
- View the name of the parent module in the Logix Designer I/O tree.
- Enter the name and description of the device as it appears in the Logix Designer application.
- Configure network addressing. See *[Device Classification] section* and *[Modular] section*.
- View revision, electronic keying, and connection type information. See *Module Definition dialog box*.
- If a safety connection is enabled, then the **Safety Network Number** displays and can be edited.
- If a safety connection is enabled, an **Advanced** button displays, which allows the configuration of the IP addresses when the module and controller communicate through a Network Address Translation (NAT) device.

Select the module in the Logix Designer I/O tree and press **Enter**, or right-click the module and select **Properties** to open the **Module Properties** dialog box.

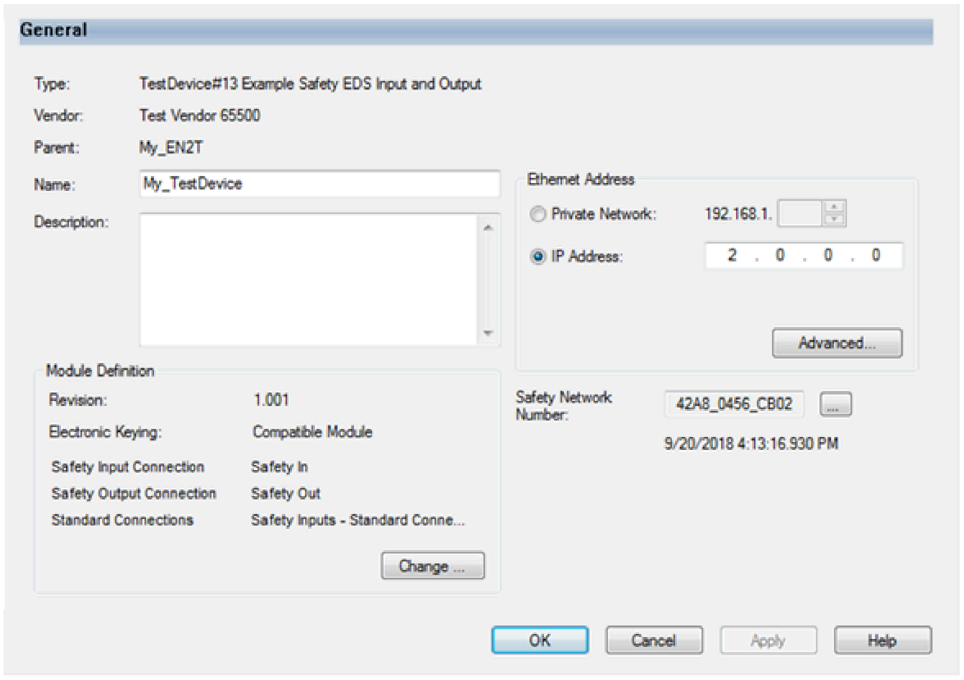

General

Type: TestDevice#9 3 Assemblies with Structure
Vendor: Test Vendor 65500
Parent: My_EN2T
Name: My_Test_Device
Description:

Ethernet Address
Private Network: 192.168.1. 10
IP Address:
Host Name:

Module Definition
Revision: 1.001
Electronic Keying: Compatible Module
Connections I/O Connection

The **Module Properties** dialog box, **General** tab when safety is enabled:



Type: TestDevice#13 Example Safety EDS Input and Output
Vendor: Test Vendor 65500
Parent: My_EN2T
Name: My_TestDevice
Description:

Private Network: 192.168.1.
IP Address: 2 . 0 . 0 . 0

Revision: 1.001
Electronic Keying: Compatible Module
Safety Input Connection: Safety In
Safety Output Connection: Safety Out
Standard Connections: Safety Inputs - Standard Conne...

Safety Network Number: 42A8_0456_CB02
9/20/2018 4:13:16.930 PM

About Module Profile dialog box

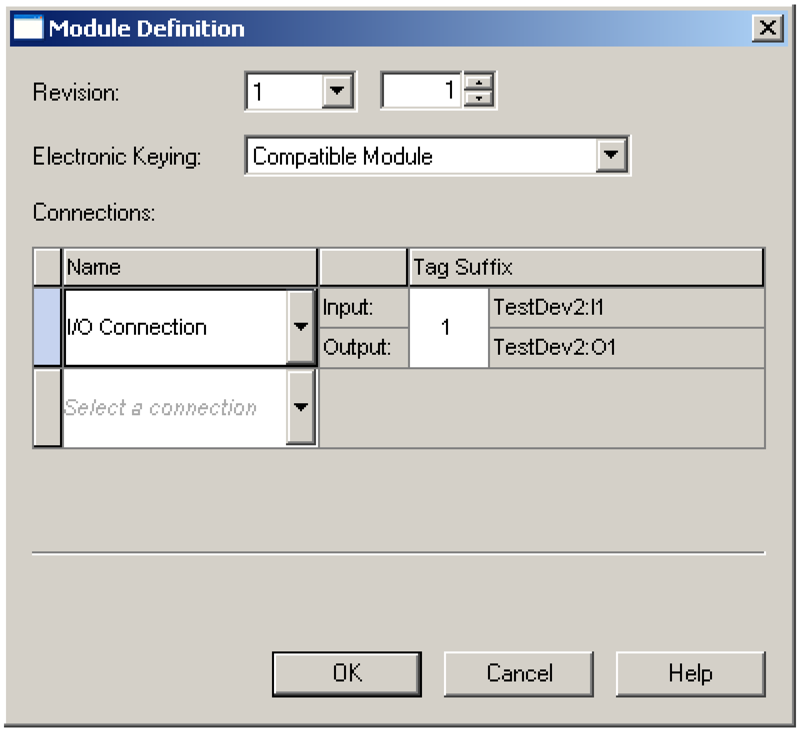
Module Definition dialog box

Configure the module details and the Input/Output data to transfer using the **Module Definition** dialog box. Typically, when defining the properties for a standard module, select:

- The module revision (major and minor). See *MajRev Keywords* and *MinRev keywords*. Note that the module revision number is not the same as the EDS file version number.
- The electronic keying option.
- Connections that the controller opens (see *ConnectionN keywords*).

Optionally, for modules that support configuring module definition properties, (as defined by the module developer) also:

- Configure connection settings, for example, remote data, selectable assemblies, configurable sizes, or custom data types.
- Select multiple, simultaneous connections, including CIP Safety connections (for those modules that support them).


Module Definition

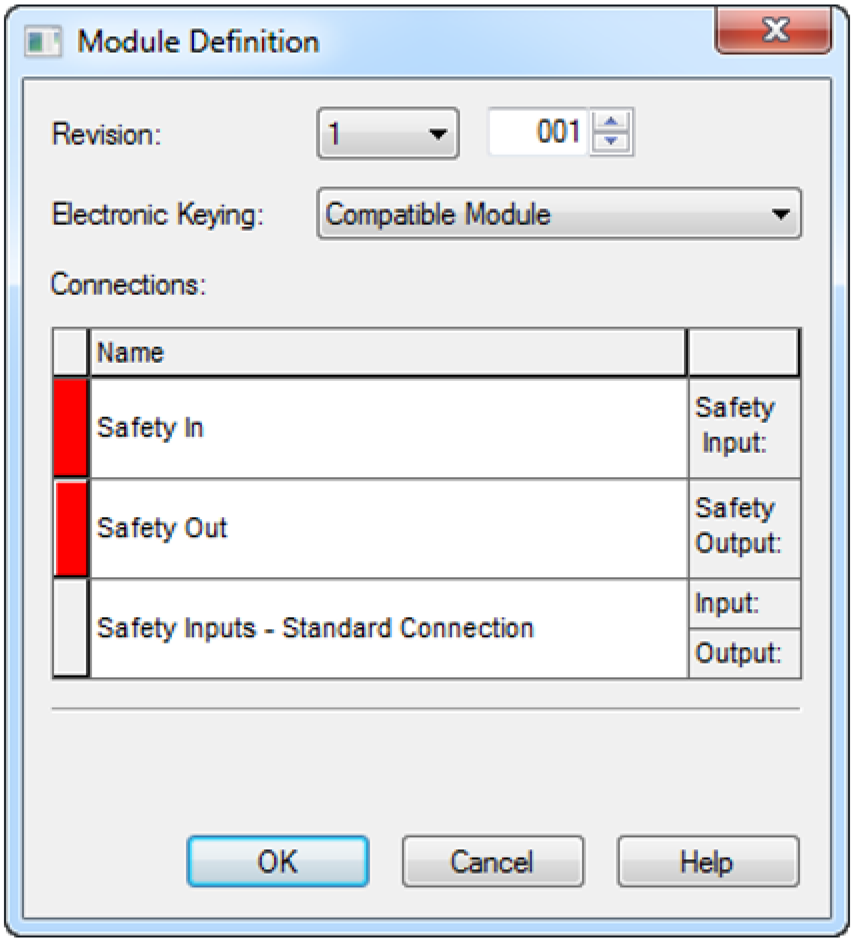
Revision: 1 1

Electronic Keying: Compatible Module

Connections:

	Name			Tag Suffix
	I/O Connection	Input:	1	TestDev2:I1
		Output:		TestDev2:O1
	Select a connection			

The **Module Definition** dialog box when a safety connection is enabled:


Module Definition

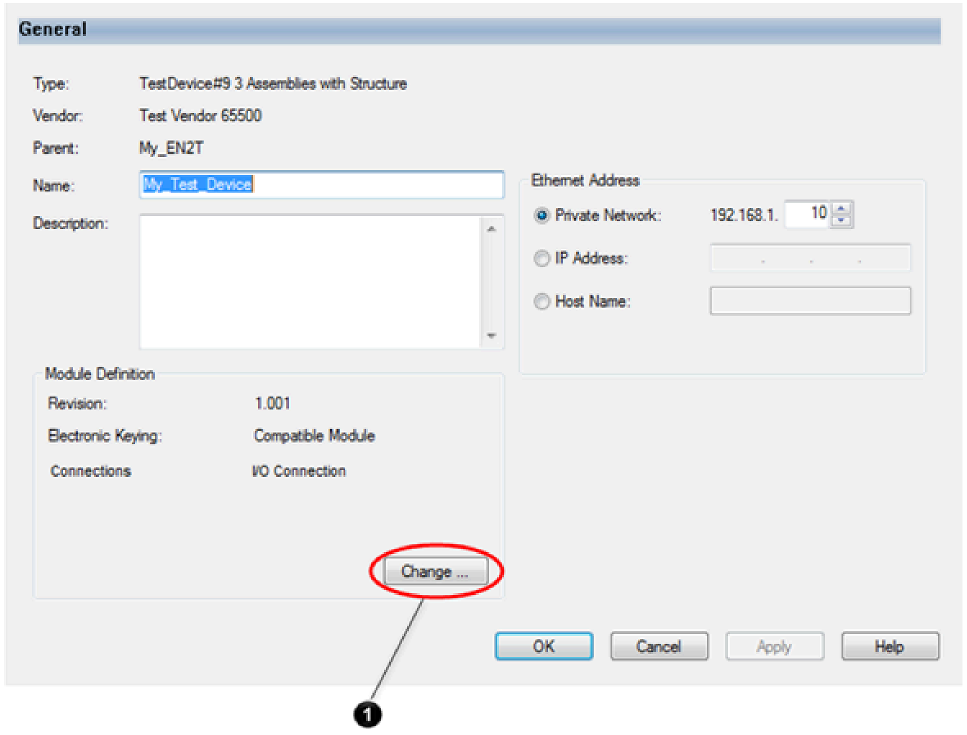
Revision: 1 001

Electronic Keying: Compatible Module

Connections:

	Name	
🟥	Safety In	Safety Input:
🟥	Safety Out	Safety Output:
	Safety Inputs - Standard Connection	Input:
		Output:

In the **Module Properties** dialog box, select **Change** to display the **Module Definition** dialog box.



General

Type: TestDevice#9 3 Assemblies with Structure

Vendor: Test Vendor 65500

Parent: My_EN2T

Name: My_Test_Device

Description:

Ethernet Address

Private Network: 192.168.1. 10

IP Address:

Host Name:

Module Definition

Revision: 1.001

Electronic Keying: Compatible Module

Connections I/O Connection

NOTE: From the **Module Properties** dialog box, select **Change** to open the **Module Definition** dialog box.

Custom Data Type dialog box

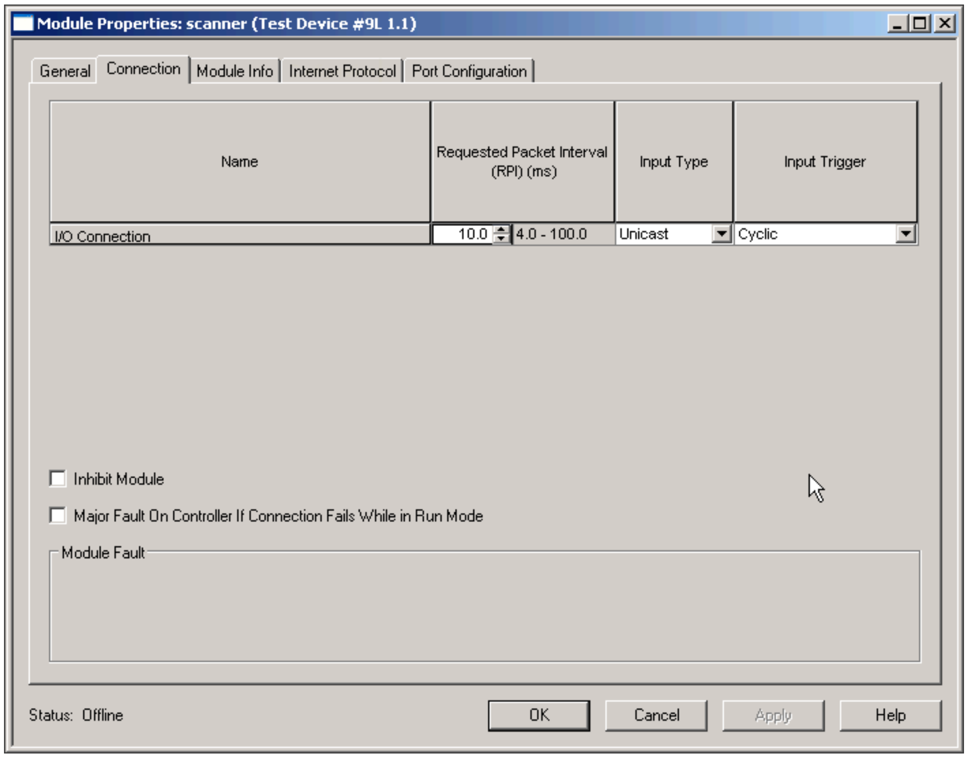
Connection tab

The **Connection** tab displays the set of configured connections (see *ConnectionN keyword*) and allows setting some connection parameters to define controller-to-module behavior. The **Connection** tab also shows the status of the connection(s) between the controller and the target module.

The data on the **Connection** tab comes directly from the controller.

n the **Connection** tab:

- Select a **Requested Packet Interval**.
- Specify an **Input Type**.
- Select an **Input Trigger**.
- Select to inhibit the module.
- Configure fault handling and view module faults.



Name	Requested Packet Interval (RPI) (ms)		Input Type	Input Trigger
I/O Connection	10.0	4.0 - 100.0	Unicast	Cyclic

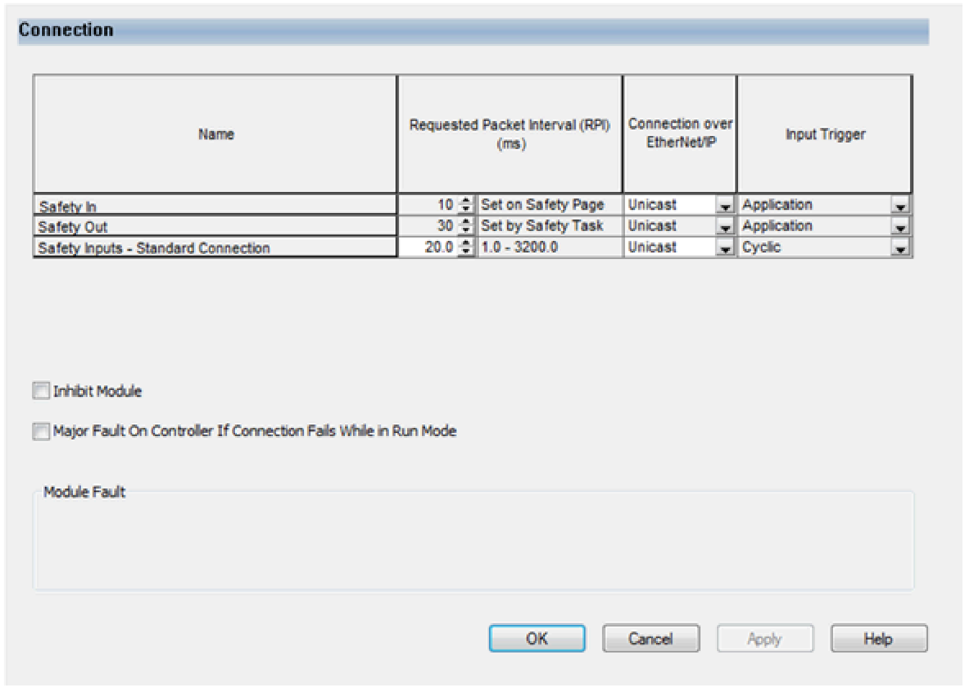
☐ Inhibit Module

☐ Major Fault On Controller If Connection Fails While in Run Mode

Module Fault

Status: Offline

The **Connection** tab when a safety connection is enabled.



Name	Requested Packet Interval (RPI) (ms)		Connection over EtherNet/IP	Input Trigger
Safety In	10	Set on Safety Page	Unicast	Application
Safety Out	30	Set by Safety Task	Unicast	Application
Safety Inputs - Standard Connection	20.0	1.0 - 3200.0	Unicast	Cyclic

☐ Inhibit Module

☐ Major Fault On Controller If Connection Fails While in Run Mode

Module Fault

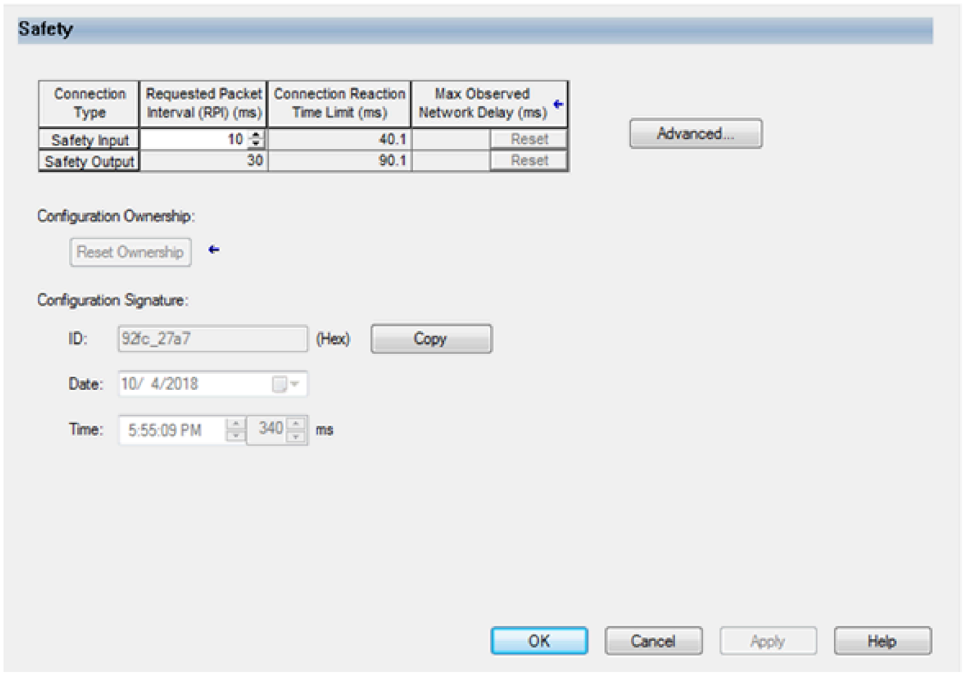
Safety tab

The **Safety** tab only displays if a safety connection is enabled. Use the **Safety** tab to:

- Select a requested packet interval.
- View reaction time.

- View and reset configuration ownership.
- View and copy the configuration signature.

The **Safety** tab if the EDS contains a safety configuration.



Safety

Connection Type	Requested Packet Interval (RPI) (ms)	Connection Reaction Time Limit (ms)	Max Observed Network Delay (ms)	
Safety Input	10	40.1		Reset
Safety Output	30	90.1		Reset

Configuration Ownership:

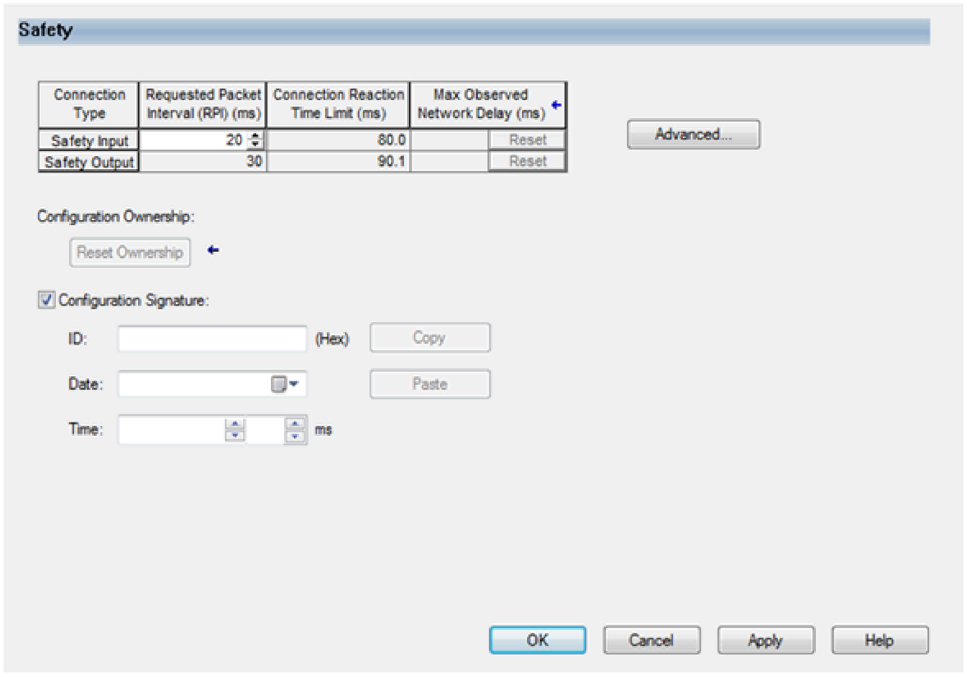
Configuration Signature:

ID: 92fc_27a7 (Hex)

Date: 10/ 4/2018

Time: 5:55:09 PM 340 ms

The Safety tab if the EDS does not contain a safety configuration.



Safety

Connection Type	Requested Packet Interval (RPI) (ms)	Connection Reaction Time Limit (ms)	Max Observed Network Delay (ms)	
Safety Input	20	80.0		Reset
Safety Output	30	90.1		Reset

Configuration Ownership:

☑ Configuration Signature:

ID: (Hex)

Date:

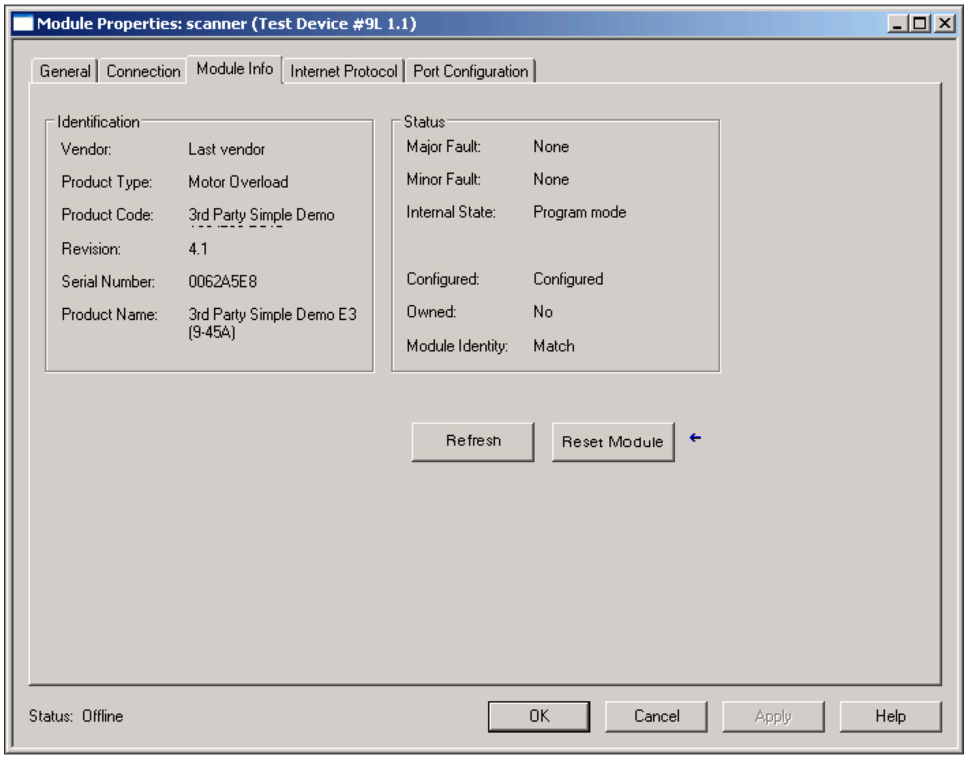
Time: ms

Module Info tab

The **Module Info** tab displays module and status information. Use this tab to:

- Display the module identity. See *[Device] section*.
- Display the module status.
- Reset the module to its power-up state.

The information on the **Module Info** tab comes directly from the module, and is not displayed if Logix Designer application is offline, or if currently creating a module. When online, the Logix Designer application indicates if there is a mismatch between the configured module in Logix Designer application and the online module.



Module Properties: scanner (Test Device #9L 1.1)

Identification

Vendor:	Last vendor
Product Type:	Motor Overload
Product Code:	3rd Party Simple Demo
Revision:	4.1
Serial Number:	0062A5E8
Product Name:	3rd Party Simple Demo E3 (9-45A)

Status

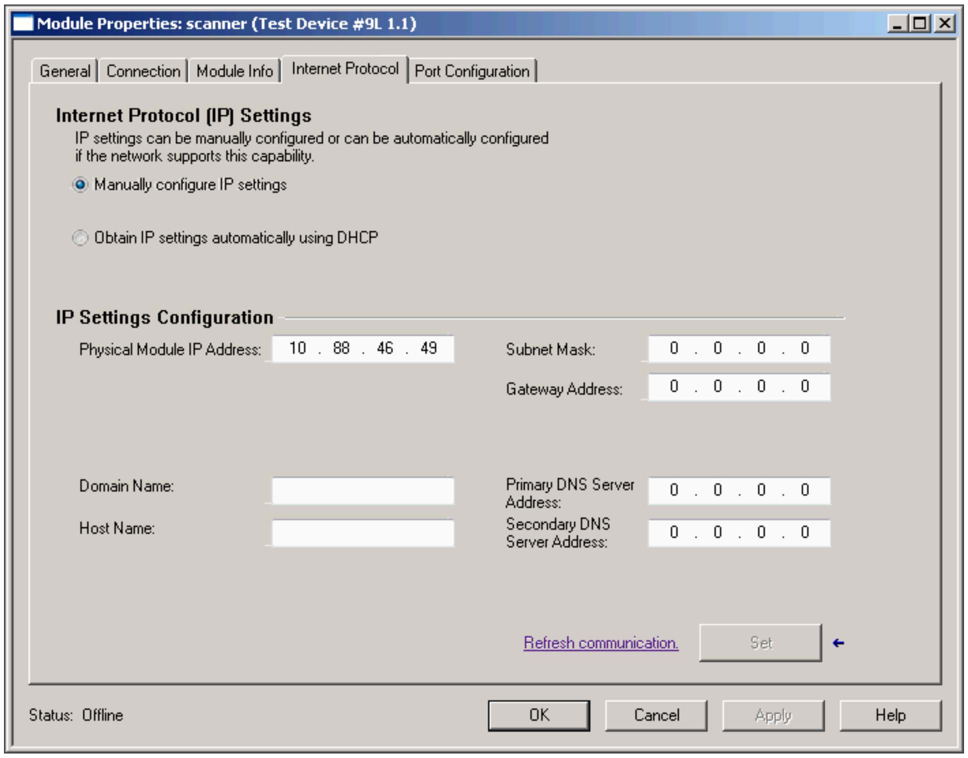
Major Fault:	None
Minor Fault:	None
Internal State:	Program mode
Configured:	Configured
Owned:	No
Module Identity:	Match

Status: Offline

Safety Configuration tab

Internet Protocol tab

Use the **Internet Protocol** tab to view the TCP/IP object attribute values in the online module and make changes to the online module. See *[TCP/IP Interface Class] section*.


Module Properties: scanner (Test Device #9L 1.1)

Internet Protocol (IP) Settings

IP settings can be manually configured or can be automatically configured if the network supports this capability.

[✓ Manually configure IP settings]

Obtain IP settings automatically using DHCP

IP Settings Configuration

Physical Module IP Address: 10 . 88 . 46 . 49

Subnet Mask: 0 . 0 . 0 . 0

Gateway Address: 0 . 0 . 0 . 0

Domain Name:

Host Name:

Primary DNS Server Address: 0 . 0 . 0 . 0

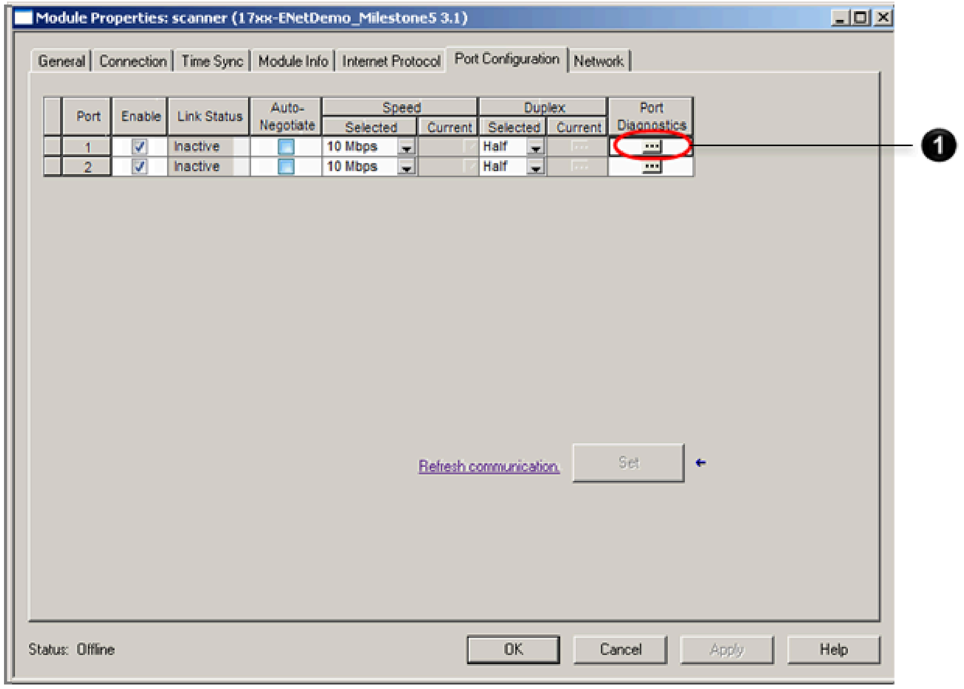
Secondary DNS Server Address: 0 . 0 . 0 . 0

Status: Offline

Port Configuration tab

Use the **Port Configuration** tab to view the Ethernet Link object attribute values in the online module and to change the online module. See *[Ethernet Link Class]* section.

In the **Port Configuration** tab, select the browse icon to open the **Port Diagnostics** dialog box.



Module Properties: scanner (17xx-ENetDemo_Milestone5 3.1)

Port	Enable	Link Status	Auto-Negotiate	Speed Selected	Speed Current	Duplex Selected	Duplex Current	Port Diagnostics
1	✓	Inactive		10 Mbps		Half		...
2	✓	Inactive		10 Mbps		Half		...

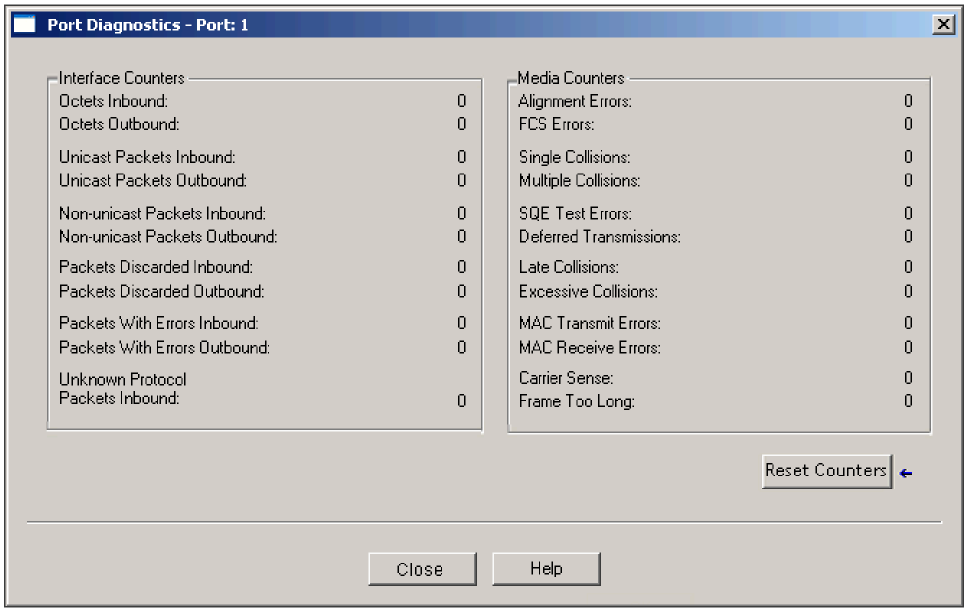
Status: Offline

NOTE: Select the browse icon to open the **Port Diagnostics** dialog box.

Port Diagnostics dialog box

Use the **Port Diagnostics** dialog box to view diagnostic information in the online module for the selected port and to reset the diagnostic counters. See *Port Configuration tab*.

Select **[...]** on the **Port Configuration** tab to display the **Port Diagnostics** dialog box.



Port Diagnostics - Port: 1

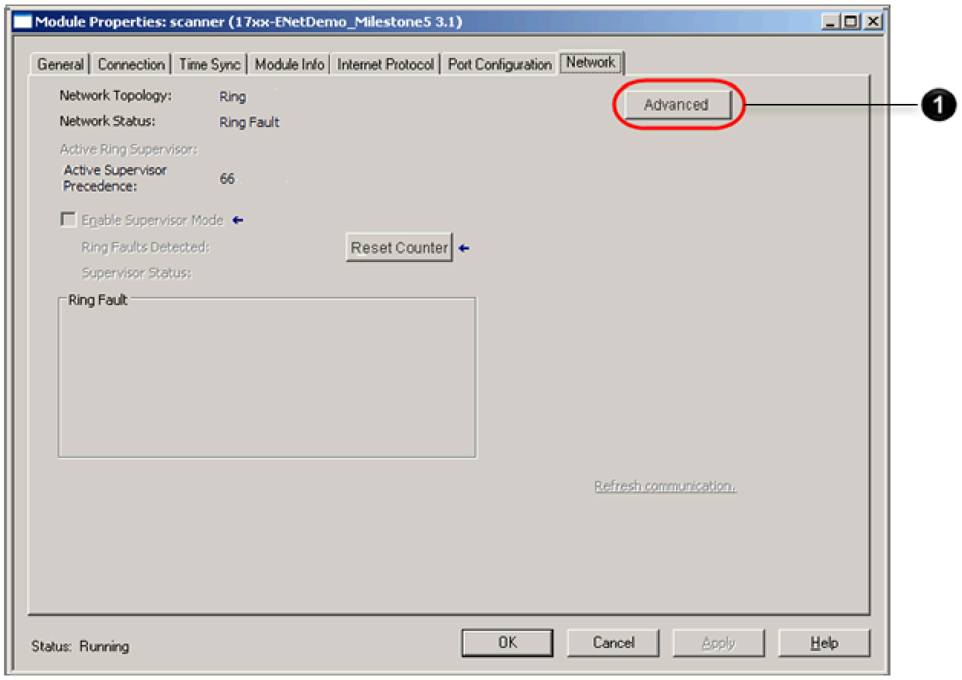
Interface Counters		Media Counters	
Octets Inbound:	0	Alignment Errors:	0
Octets Outbound:	0	FCS Errors:	0
Unicast Packets Inbound:	0	Single Collisions:	0
Unicast Packets Outbound:	0	Multiple Collisions:	0
Non-unicast Packets Inbound:	0	SQE Test Errors:	0
Non-unicast Packets Outbound:	0	Deferred Transmissions:	0
Packets Discarded Inbound:	0	Late Collisions:	0
Packets Discarded Outbound:	0	Excessive Collisions:	0
Packets With Errors Inbound:	0	MAC Transmit Errors:	0
Packets With Errors Outbound:	0	MAC Receive Errors:	0
		Carrier Sense:	0
Unknown Protocol Packets Inbound:	0	Frame Too Long:	0

Network tab

Use the **Network** tab to view the Device Level Ring object attribute values in the online module and to make changes to the online module. If the module supports the **Supervisor Mode** function, the **Network** tab includes a control to enable **Supervisor Mode**.

The **Network** tab displays only for EtherNet/IP modules that support the Device Level Ring object. See *[DLR Class] section.*

In the **Module Properties** dialog box, **Network** tab, select **Advanced** to display the **Advanced Network Configuration** dialog box.

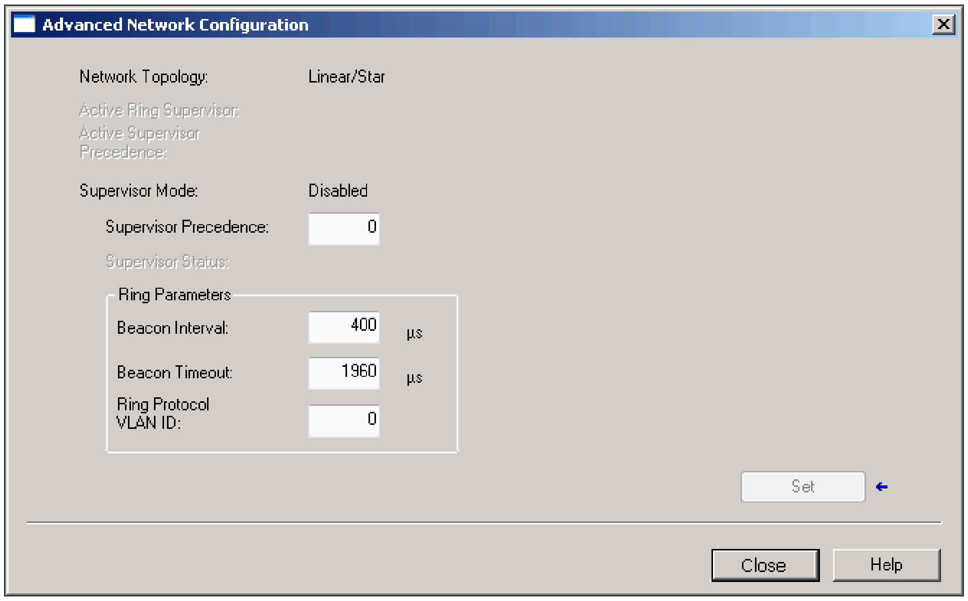


Network Topology: Ring
Network Status: Ring Fault
Active Ring Supervisor:
Active Supervisor Precedence: 66
Ring Faults Detected:
Supervisor Status:
Ring Fault
Status: Running

NOTE: In the **Module Properties** dialog box, **Network** tab, select **Advanced** to display the **Advanced Network Configuration** dialog box.

Advanced Network Configuration dialog box

The **Advanced Network Configuration** dialog box displays the network **Supervisor Mode** settings when the module is a Network Supervisor.



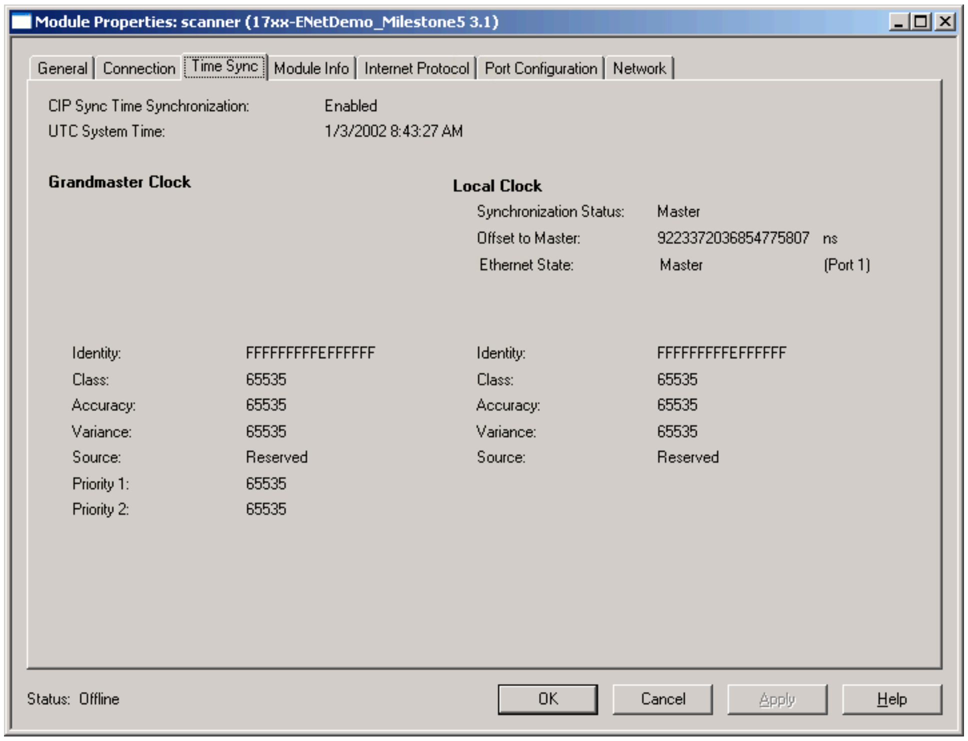
Network Topology: Linear/Star
Active Ring Supervisor:
Active Supervisor Precedence:
Supervisor Mode: Disabled
Supervisor Precedence: 0
Supervisor Status:
Ring Parameters
Beacon Interval: 400 µs
Beacon Timeout: 1960 µs
Ring Protocol VLAN ID: 0

Select **Advanced** on the **Network** tab to display the **Advanced Network Configuration** dialog box.

Time Sync tab

Use the **Time Sync** tab to display the Time Sync object attribute values in the online module and to make changes to the online module. See *[Time Sync Class] section.*

The **Time Sync** tab is only displayed for modules that support the Time Sync object.



Module Properties: scanner (17xx-ENetDemo_Milestone5 3.1)

CIP Sync Time Synchronization: Enabled
UTC System Time: 1/3/2002 8:43:27 AM

Grandmaster Clock

Identity:	FFFFFFFFFEFFFFFF
Class:	65535
Accuracy:	65535
Variance:	65535
Source:	Reserved
Priority 1:	65535
Priority 2:	65535

Local Clock

Synchronization Status: Master
Offset to Master: 9223372036854775807 ns
Ethernet State: Master (Port 1)

Identity:	FFFFFFFFFEFFFFFF
Class:	65535
Accuracy:	65535
Variance:	65535
Source:	Reserved

Status: Offline

EDS file sections

This section describes the EDS sections and entry keywords used by the Logix Designer EDS AOP feature. This section covers how the keyword affects a CIP service (for example, the Forward_Open service) or affects the **Module Properties** dialog box. The *Example EDS Files* section describe several examples of EDS files.

EDS sections and keywords that are not mentioned in this document are not supported in the Logix Designer EDS AOP and are ignored.

The sections are:

[File] section

The [File] section lists the File entry keywords and the keyword value.

```
[File]
  DescText = "Test EDS Description";
  CreateDate = 01-14-2011;
  CreateTime = 14:26:47;
  ModDate = 12-23-2011;
  ModTime = 11:55:27;
  Revision = 1.0;
  EDSFileCRC = 0xnnnnnnnn; $ number is different for every EDS
 file
```

Revision, CreateDate, CreateTime, ModDate and ModTime keywords

The revision, creation, and modification information displays on the **About Module Profile** dialog box. The EDSFileCRC is a value calculated by the product developer and inserted in this entry. The value is unique to each EDS file. See CIP NETWORKS LIBRARY, Volume 5, Chapter 7 for details about this keyword.

[Device] section

The [Device] section lists the Device entry keywords and the keyword value.

```
[Device]
  VendCode = 65500;
  VendName = "Test Vendor 65500";
  ProdType = oxFFFF;
  ProdTypeStr = "Generic Device";
  ProdCode = 9;
  MajRev = 1;
  MinRev = 1;
  ProdName = "3 Assemblies with Structure";
  Catalog = "TestDevice#9";
```

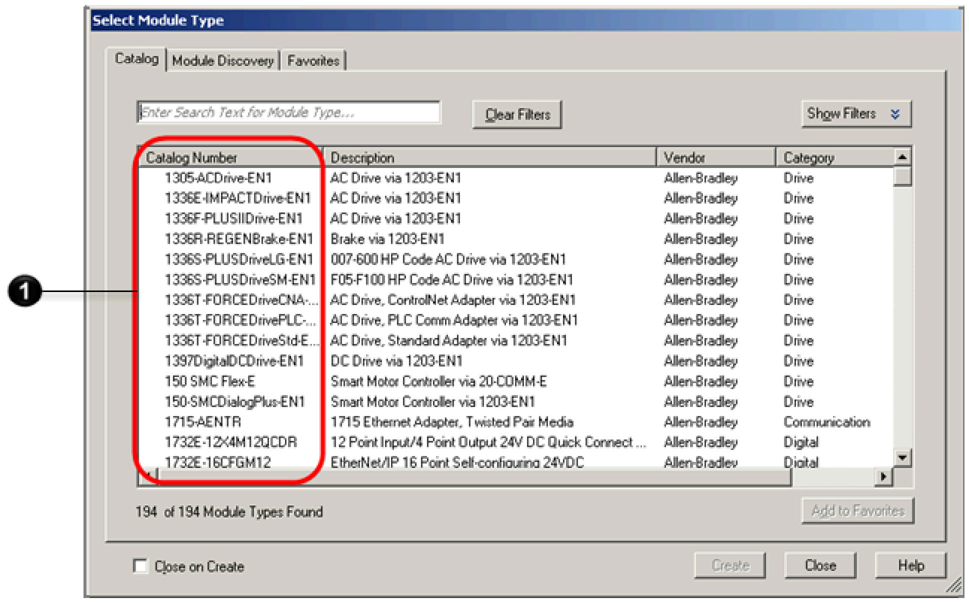
Catalog and ProdName keywords

The Catalog string, followed by the ProdName string, are concatenated for display in the **Module Properties** dialog box, **General** tab, **Type** box. If the optional Catalog keyword is not present, only the value of the ProdName keyword shows.

Use a unique combination strings for Catalog and ProdName for all modules from a vendor to distinguish them from each other.

Catalog keyword

The Catalog keyword value displays in the **Select Module Type** dialog box, **Catalog** and **Favorites** tabs, **Catalog Number** column. If the optional Catalog keyword is omitted, a string of the form xxxx_yyyy_zzzz (xxxx is hex VendCode, yyyy is hex ProdType, zzzz is hex ProdCode) displays instead.



Catalog Number	Description	Vendor	Category
1305-ACDrive-EN1	AC Drive via 1203-EN1	Allen-Bradley	Drive
1336E-IMPACTDrive-EN1	AC Drive via 1203-EN1	Allen-Bradley	Drive
1336F-PLUSIIDrive-EN1	AC Drive via 1203-EN1	Allen-Bradley	Drive
1336R-REGENBrake-EN1	Brake via 1203-EN1	Allen-Bradley	Drive
1336S-PLUSDriveLG-EN1	007-600 HP Code AC Drive via 1203-EN1	Allen-Bradley	Drive
1336S-PLUSDriveSM-EN1	F05-F100 HP Code AC Drive via 1203-EN1	Allen-Bradley	Drive
1336T-FORCEDriveCNA-...	AC Drive, ControlNet Adapter via 1203-EN1	Allen-Bradley	Drive
1336T-FORCEDrivePLC-...	AC Drive, PLC Comm Adapter via 1203-EN1	Allen-Bradley	Drive
1336T-FORCEDriveStd-E...	AC Drive, Standard Adapter via 1203-EN1	Allen-Bradley	Drive
1397DigitalDCDrive-EN1	DC Drive via 1203-EN1	Allen-Bradley	Drive
150 SMC Flex-E	Smart Motor Controller via 20-COMM-E	Allen-Bradley	Drive
150-SMCDialogPlus-EN1	Smart Motor Controller via 1203-EN1	Allen-Bradley	Drive
1715-AENTR	1715 Ethernet Adapter, Twisted Pair Media	Allen-Bradley	Communication
1732E-12X4M12QCDR	12 Point Input/4 Point Output 24V DC Quick Connect ...	Allen-Bradley	Digital
1732E-16CFGM12	EtherNet/IP 16 Point Self-configuring 24VDC	Allen-Bradley	Digital

194 of 194 Module Types Found

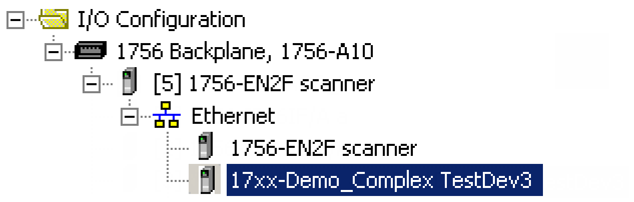
NOTE: Select Module Type dialog box, **Catalog Number** column contains a list of catalog keyword values.

IMPORTANT: Module developers should include the Catalog keyword.

Do not use spaces in the Catalog keyword. Space characters cause problems with the **Path** control found in the **Message Configuration** dialog box, **Communication** tab (the **Message Configuration** dialog box is invoked from the MSG instruction).

The Catalog keyword is included in Catalog Organizer, Data Types tree, Module- Defined data type name. Use alpha-numeric characters and the underscore (_) character. Other characters are removed, except dash (-), which is converted to underscore (_).

The Catalog keyword value displays in the **Controller Organizer, I/O Configuration** tree module.



ProdName keyword

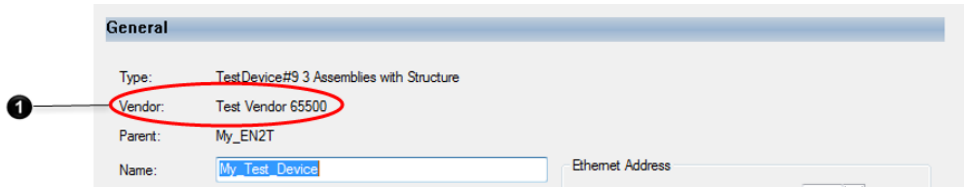
If this ProdName is defined, it is used instead of the ProdName specified in the [Device] section.

VendName keyword

If an ODVA-defined vendor string is not known for the VendCode, the VendName from the first EDS file registered for the VendCode is used; otherwise, the ODVA-defined vendor string is used.

IMPORTANT: Once the VendName is set by registering the first EDS file with that VendName, there is no mechanism to change the VendName.

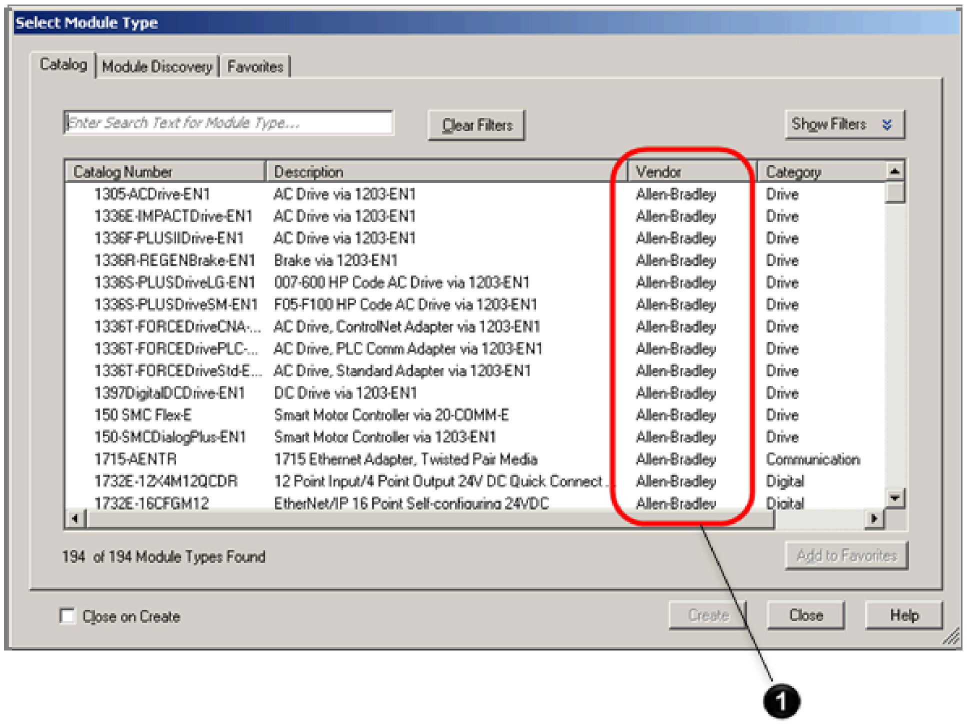
The Rockwell Automation-defined VendName keyword value displays in the **Module Properties** dialog box, **General** tab, **Vendor** area.


General

Type: TestDevice#9 3 Assemblies with Structure
Vendor: Test Vendor 65500
Parent: My_EN2T
Name: My_Test_Device

NOTE: Module Properties dialog box, **General** tab, **Vendor** area lists the VendName keyword value.

The Rockwell Automation-defined VendName keyword value also displays in the **Select Module Type** dialog box, **Catalog** and **Favorites** tabs, **Vendor** column.

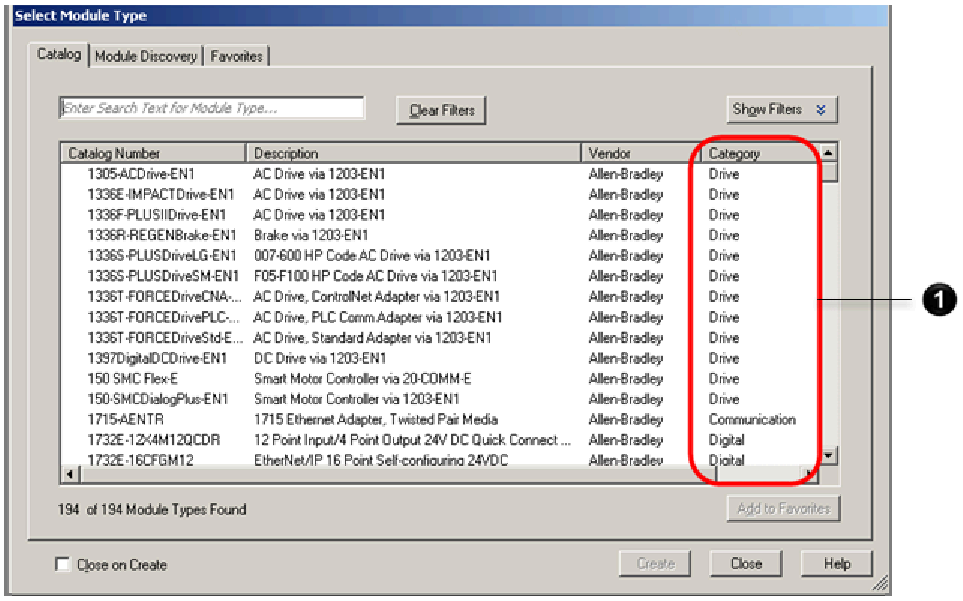


Catalog Number	Description	Vendor	Category
1305-ACDrive-EN1	AC Drive via 1203-EN1	Allen-Bradley	Drive
1336E-IMPACTDrive-EN1	AC Drive via 1203-EN1	Allen-Bradley	Drive
1336F-PLUSIIDrive-EN1	AC Drive via 1203-EN1	Allen-Bradley	Drive
1336R-REGENBrake-EN1	Brake via 1203-EN1	Allen-Bradley	Drive
1336S-PLUSDriveLG-EN1	007-600 HP Code AC Drive via 1203-EN1	Allen-Bradley	Drive
1336S-PLUSDriveSM-EN1	F05-F100 HP Code AC Drive via 1203-EN1	Allen-Bradley	Drive
1336T-FORCEDriveCNA-...	AC Drive, ControlNet Adapter via 1203-EN1	Allen-Bradley	Drive
1336T-FORCEDrivePLC-...	AC Drive, PLC Comm Adapter via 1203-EN1	Allen-Bradley	Drive
1336T-FORCEDriveStd-E...	AC Drive, Standard Adapter via 1203-EN1	Allen-Bradley	Drive
1397DigitalDCDrive-EN1	DC Drive via 1203-EN1	Allen-Bradley	Drive
150 SMC Flex-E	Smart Motor Controller via 20-COMM-E	Allen-Bradley	Drive
150-SMCDialogPlus-EN1	Smart Motor Controller via 1203-EN1	Allen-Bradley	Drive
1715-AENTR	1715 Ethernet Adapter, Twisted Pair Media	Allen-Bradley	Communication
1732E-12X4M12QCDR	12 Point Input/4 Point Output 24V DC Quick Connect ...	Allen-Bradley	Digital
1732E-16CFGM12	EtherNet/IP 16 Point Self-configuring 24VDC	Allen-Bradley	Digital

NOTE: Select Module Type dialog box, **Catalog** and **Favorites** tabs, **Vendor** column also lists the VendName keyword values.

ProdTypeStr keyword

The ProdTypeStr keyword value displays in the **Select Module Type** dialog box, **Catalog** and **Favorites** tabs, **Category** column.



Catalog Number	Description	Vendor	Category
1305-ACDrive-EN1	AC Drive via 1203-EN1	Allen-Bradley	Drive
1336E-IMPACTDrive-EN1	AC Drive via 1203-EN1	Allen-Bradley	Drive
1336F-PLUSIIDrive-EN1	AC Drive via 1203-EN1	Allen-Bradley	Drive
1336R-REGENBrake-EN1	Brake via 1203-EN1	Allen-Bradley	Drive
1336S-PLUSDriveLG-EN1	007-600 HP Code AC Drive via 1203-EN1	Allen-Bradley	Drive
1336S-PLUSDriveSM-EN1	F05-F100 HP Code AC Drive via 1203-EN1	Allen-Bradley	Drive
1336T-FORCEDriveCNA-...	AC Drive, ControlNet Adapter via 1203-EN1	Allen-Bradley	Drive
1336T-FORCEDrivePLC-...	AC Drive, PLC Comm Adapter via 1203-EN1	Allen-Bradley	Drive
1336T-FORCEDriveStd-E...	AC Drive, Standard Adapter via 1203-EN1	Allen-Bradley	Drive
1397DigitalDCDrive-EN1	DC Drive via 1203-EN1	Allen-Bradley	Drive
150 SMC Flex-E	Smart Motor Controller via 20-COMM-E	Allen-Bradley	Drive
150-SMCDialogPlus-EN1	Smart Motor Controller via 1203-EN1	Allen-Bradley	Drive
1715-AENTR	1715 Ethernet Adapter, Twisted Pair Media	Allen-Bradley	Communication
1732E-12X4M12QCDR	12 Point Input/4 Point Output 24V DC Quick Connect ...	Allen-Bradley	Digital
1732E-16CFGM12	EtherNet/IP 16 Point Self-configuring 24VDC	Allen-Bradley	Digital

NOTE: Select Module Type dialog box, **Catalog** and **Favorites** tabs, **Category** column lists the ProdTypeStr keyword value.

If an ODVA-defined profile string is not known for publicly defined ProdType values, the ProdTypeStr from the first EDS registered for the ProdType is used; otherwise, the ODVA-defined profile name is used. See *THE CIP NETWORKS LIBRARY Volume 1, Common Industrial Protocol (CIP™)*, table 6- 7.1

For vendor-specific ProdType values, the ProdTypeStr from the first EDS registered for the VendCode/ProdType is used.

IMPORTANT: Once ProdTypeStr is set by registering the first EDS with that ProdTypeStr, there is no mechanism to change ProdTypeStr.

MajRev keyword

The MajRev keyword value is available for selection in the **Module Definition** dialog box, **Major Revision** control.

MinRev keyword

The MinRev value is used as the default value of **Minor Revision** in the **Module Definition** dialog box.

DataExchangeCategories keyword

The DataExchangeCategories keyword declares the Data Exchange Toolkit (DET) categories that are supported by an EDS device. To allow the Logix Designer application to access this information, copy it into the Catalog Services description of the device.

This is the keyword definition:

```
[Device Classification]
1_DataExchangeCategories =
    CategoryName, CategoryIsRequired
```

Keyword values

- `CategoryName` is a string specifying the supported DET category name.
- `CategoryIsRequired` determines whether the category is `required` or `optional`.

Example use:

```
[Device Classification]
1_DataExchangeCategories =
    "ControlLogixFamily", Required,
    "DigitalModule", Required,
    "InputModule", Required;
```

DataExchangeCatalogNumbers keyword

The DataExchangeCatalogNumbers keyword defines the AML catalog number mapping for EDS-based Add-On Profiles. To allow the Logix Designer application to access this information, merge it into Catalog Services.

For EPLAN Electric P8, use AML data files (.aml) and associated PLC data import/export workflows to exchange data between the Logix Designer application, versions 32.00 and later, and EPLAN Electric P8, versions 2.9 and later.

This is the keyword definition:

```
[Device]
1_DataExchangeCatalogNumbers = "20G11GC072JA0NNNNN", …;
```

DataExchangeAliasTargetsN keyword

Use the DataExchangeAliasTargetsN keyword to explicitly declare the alias targets within the EDS file when you develop an EDS-based Add-On Profile.

This is the keyword definition:

```
[Assembly]
1_DataExchangeAliasTargetsN =
    Index, MemberSize, MemberReference, PrimitiveDataType
    ...;
```

Keyword values

- The integer `N` associates this keyword with the `AssemN` or `AssemExaN` that has the same N.

When you update an existing EDS file, if a `ConnectionN` specifies either the input or output as a single `ParamN`, you need to replace that `ParamN` with an `AssemN` or `AssemExaN` that contains the single `ParamN`. This allows it to be associated with the `1_DataExchangeAliasTargetsN` using the `N`.

- The `Index` value is a positive integer, including zero. It represents the point or channel number when the module is a discrete or analog module.
- `MemberSize` specifies the size (in bits) of the data to be used within `MemberReference`. This value is optional. If omitted, the natural size of `MemberReference` will be used. Because Logix Designer can only create alias tags that target primitive data types or a bit, this value should always be 1 or left empty (to indicate the member's natural size should be used).
- `MemberReference` specifies the target of the alias. This reference can specify an enitre assembly if the alias should refer to an entire assembly. When referring to an entire assembly, you must specify the following `Submember`. This member reference can end with an optional bit index (as demonstrated in the following example).
- The `PrimitiveDataType` specification is required. It must specify one of the following CIP data types: BOOL, SINT, INT, DINT, REAL.

The `1_DataExchangeAliasTargetsN` keyword allows for more granularity in specifying the alias target than the Logix Designer application allows. For example, a `MemberSize` of 3 and a `MemberReference` of Param12:2 would resolve to bits 2, 3 and 4 of Param12. Logix can only create aliases that resolve to its primitive data types and cannot create an alias to only these bits. If such an alias target is specified, the API used to get this information will return an error.

When the API is invoked to get the alias tag target for a device, use the `1_DataExchangeAliasTargetsN` to determine the alias target in the associated `AssemN` or `AssemExaN`. Then, the member reference should be resolved to a relative Logix data type path. For example, given the following EDS definitions, when the API to get the alias target for input point 2 is invoked, the string `Data.2` will be returned.

```
[Params]
    Param12 =
        0,                                  $ reserved
        , "21 28 03 24 01 30 05",           $ path size, path
        0x00,                               $ descriptor
        0xC2, 1,                            $ data type and size
 - (SINT)
        "Data",                             $ name
        "",                                 $ unit string
        "",                                 $ help string
        , , ,                               $ min, max, default
 values
        , , , ,                             $ mult, div, base,
 offset scaling (Not Used)
        , , , ,                             $ mult, div, base,
 offset links (Not Used)
        ;                                   $ decimal precision

[Assembly]

    AssemExa1 =
        "In Data",                          $ name
        ,                                   $ path
        ,                                   $ size
        0x0800,                             $ descriptor
        ,,                                  $ reserved, reserved
        ,Param12;                           $ member size,
 reference

    1_DataExchangeAliasTargets1 =
        0, 1, Param12:0, BOOL,                    $ point, alias
 target
        1, 1, Param12:1, BOOL,                    $ point, alias
 target
        2, 1, Param12:2, BOOL,                    $ point, alias
 target
        3, 1, Param12:3, BOOL,                    $ point, alias
 target
        4, 1, Param12:4, BOOL,                    $ point, alias
 target
        5, 1, Param12:5, BOOL,                    $ point, alias
 target
        6, 1, Param12:6, BOOL,                    $ point, alias
 target
        7, 1, Param12:7, BOOL;                    $ point, alias
 target

[Connection Manager]
    Connection1 =
        0x04020002,                         $ trigger & transport
        0x66240405,                         $ point/multicast &
 priority & realtime format
```

```
       Param400,,AssemExa2,                $ O=>T (output) rpi,
size, format
       Param400,,AssemExa1,                $ T=>O (input) rpi,
size, format
       ,,                                  $ proxy cfg size,
format
       ,Assem190,                          $ target cfg size,
format
       "Control and Status",               $ connection name
       "",                                 $ help string
       "20 04 24 BE 2C C0 2C BF";          $ path
```

The member reference may end with an indirect parameter. If so, when the API is called to get the alias target, the indirect parameter must be de-referenced, and the referenced parameter's name must be used in the returned string. If the value of the indirect parameter is zero (indicating termination, skip, or pad), an error will be returned and the alias target will not be created.

Alias Targets for EDS Devices Containing Top Level VariantN, VariantExaN or BitStringVariantN

The EDS AOP feature allows variants to exist at the top level of an input or output data type. The variant EDS constructs (`VariantN` , `VariantExaN` , and`BitStringVariantN`) allow a selector (whose value is typically set in the configuration tag) to dynamically determine the input or output data type. This means that the `1_DataExchangeAliasTargetsN` keyword must also afford some flexibility when specifying alias targets.

To accommodate this, each index can be specified multiple times with unique `MemberSize` and `MemberReference` values. When the API is called to get the alias target, the AML synchronization feature will iterate over the `MemberSize` and `MemberReferences` specified for the requested index, returning the first one that exists for the device's current configuration. This example shows:

- Connection1's input assembly is AssemExa2.
- AssemExa2 contains a top level VariantN that uses Param1 as a selector to determine whether the input is a discrete or analog input.
- AssemExa2 is associated with 1_DataExchangeAliasTargets2 (via N = 2), which defines two different possible alias targets. The first alias target will exist when the device is configured for discrete input, and the second will exist when the device is configured for analog input.
- The AML Synchronization feature will evaluate each target alias to determine which is valid for the current configuration, and then resolve the member reference to a relative Logix alias target path. For example, when the device is configured for analog input, the returned alias target will be **Ch0Data**.

```
[Params]

    Param1 =
        0,                          $ reserved
        ,,                          $ path size, path
        0x0002,                     $ descriptor
        0xC6                        $ data type : USINT
        ,                           $ data size (bytes)
        "Device Mode",                $ name
        "",                           $ units
        "discrete/analog configuration",    $ help string
        0,1,0,                        $ min, max, default data
 values
        0,0,0,0,                      $ mult, dev, base, offset
 scaling not used
        0,0,0,0,                      $ mult, dev, base, offset
 link not used
        0;                            $ decimal places - not used

    Enum1 =
        0, "Discrete Mode",
        1, "Analog Mode 4-20mA";

    Param2 =
        0,                            $ reserved
        ,,                            $ path size (bytes), path
        0x0002,                       $ descriptor
        0xC1,                         $ data type - BOOL
        ,                             $ data size (bytes)
        "Pt0Data",                    $ name
        "",                           $ units
        "",                           $ help string
        0,1,0,                        $ min, max, default
        0,0,0,0,                      $ base: mult, div, base,
 offset
        0,0,0,0,                      $ link: mult, div, base,
 offset
        0;                            $ decimal places

    Param3 =
        0,                            $ reserved
        ,,                            $ path size (bytes), path
        0x0002,                       $ descriptor
        0xC3,                         $ data type - INT
        ,                             $ data size (bytes)
        "Ch0Data",                    $ name
        "mA",                         $ units
        "",                           $ help string
        4,20,0,                       $ min, max, default
        0,0,0,0,                      $ base: mult, div, base,
 offset
        0,0,0,0,                      $ link: mult, div, base,
 offset
        0;                            $ decimal places

[Assembly]

    AssemExa1 =
        "configuration",        $ name
        ,                       $ path
        ,                       $ size (bytes)
        ,                       $ descriptor
        ,,                      $ reserved
        1, Param1,              $ member size (bits), reference -
 discrete/analog configuration
        7,;                     $ member size (bits), reference -
 pad

    AssemExa2 =
        "input data",           $ name
        "",                     $ path
        ,                       $ size (bytes)
        ,                       $ descriptor
        ,,                      $ reserved
        , Variant1;             $ member size (bits), reference

    Variant1 =
        "input mode",           $ name
        ,,,,                    $ reserved
        Param1,                 $ selector
        0x00, AssemExa3,        $ if 0, discrete mode
        0x01, AssemExa4;        $ if 1, analog 4-20mA mode

    1_DataExchangeAliasTargets2 =
        0, , AssemExa3:Param2:0, BOOL,
        0, , AssemExa4:Param3, INT;

    AssemExa3 =
        "discrete input",       $ name
        "",                     $ path
        ,                       $ size (bytes)
        ,                       $ descriptor
        ,,                      $ reserved
        , Param2;               $ member size (bits), member
 reference

    AssemExa4 =
        "analog input",         $ name
        "",                     $ path
        ,                       $ size (bytes)
        ,                       $ descriptor
        ,,                      $ reserved
        , Param3;               $ member size (bits), member
 reference

[Connection Manager]
    Connection1 =
        0x04020002,                         $ trigger & transport
        0x66240405,                         $ point/multicast,
 priority, realtime format
        ,,,                                 $ O=>T (output) rpi,
 size, format
        ,,AssemExa2,                        $ T=>O (input) rpi,
 size, format
```

```
       ,,                                   $ proxy cfg size,
format
       ,AssemExa1,                          $ target cfg size,
format
       "I/O Connection",                    $ connection name
       "",                                  $ help string
       "20 04 24 BE 2C C0 2C BF";           $ path
```

[Device Classification] section

The [Device Classification] section includes the ClassN keyword.

```
[Device Classification]
  Class1 = EtherNetIP;
  Class2 = Safety;
```

ClassN keyword

If the module is an EtherNet/IP module, a ClassN keyword must exist in the [Device Classification] section. A keyword example is:

```
Class1 = EtherNetIP;
```

[Connection Manager] section

The *CIP NETWORKS LIBRARY, Volume 5, Chapter 7*, has restrictions and requirements for some of the values used in this section when the connection entry is describing a CIP Safety connection. Please refer to the specification for these restrictions and requirements. Any further restrictions and requirements imposed by EDS AOP are noted where appropriate.

```
[Connection Manager]
  Connection1 =
  0x04030002,
  0x44640405,
  Param1,,Assem101,  $ O->T RPI, size, format
  Param1,,Assem102,  $ T->O RPI, size, format
  ,,     $ proxy config size, format
  ,Assem100,    $ target config size, format
  "I/O Connection",  $ Connection Name
  "",     $ help string
  "20 04 24 64 2C 65 2C 66";  $ Path
  MaxSafetyConnections = 4;
  MaxSafetyInputCnxns = 4;
  MaxSafetyOutputCnxns = 1;
  DefaultSafetyConnections = 3,6;
```

Extra comment lines have been removed.

ConnectionN keyword

If the Internationalized Connection Name String is defined, it is used instead of the name defined in the *[Connection Manager] section* in all cases.

The Internationalized Connection Help String is not used.

ConnectionN fields

The ConnectionN fields include:

- Trigger and Transport
- Connection Parameters
- T->O and O->T RPI,
- T->O and O->T Size
- T->O and O->T Format
- Proxy Config Size and Format
- Target Config Size
- Target Config Format
- Connection Name String
- Help String
- Path

Trigger and Transport field

The Trigger and Transport field consist of Transport class, Trigger, Application type, and Direction.

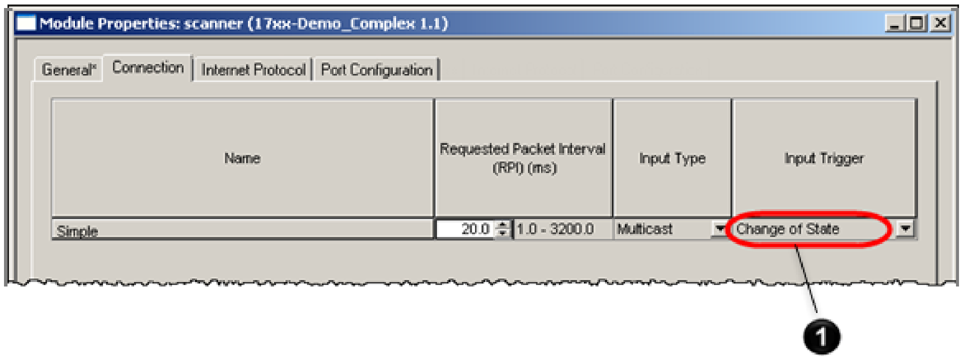
- Transport class

  Must indicate Class 1 support (may indicate others).

  The Transport Class field value of the Transport Class and Trigger parameter sent in the Forward_Open request is always set to 1.

- Trigger

  The Triggerdisplays on the **Modules Properties** dialog box, **Connection** tab, **Input Trigger** column. Select the appropriate trigger (**Cyclic**, **Change of State**, or **Application**).



Name	Requested Packet Interval (RPI) (ms)		Input Type	Input Trigger
Simple	20.0	1.0 - 3200.0	Multicast	Change of State

NOTE: Module Properties dialog box, **Connection** tab, **Input Trigger** column lists the Trigger keyword values to choose.

The user-selected Input Trigger value is the Forward_Open Transport Class and Trigger parameter.

- Application type

  Must indicate one of **Listen-Only**, **Input-Only** or **Exclusive-Owner**. Redundant-Owner is not supported.
- Direction

  The Direction value is the Forward_Open Transport Class and Trigger parameter, Dir field value. Set this to Client.

Connection Parameters field

The Connection Parameters field consist of T®O and O®T fields, Proxy Config and Format field, Target Config Size field, Target Config Format field, Connection Name String field, Help String field, and Path field.

T->O and O->T fixed/variable size supported bits

Only supports fixed size. The T®O and O®T fixed size supported bits must be set. The T®O and O®T variable size supported bits are ignored.

O->T Real time transfer format

Must indicate one:

- Connection is pure data and modeless – the controller does not include a 32 bit run/idle header.
- Heartbeat – no application data.
- 32 bit run/idle header – the controller includes a 32 bit run/idle header.
- Safety (for CIP Safety connections)

T->O Real time transfer format

Must indicate one:

- Connection is pure data and modeless – a 32 bit run/idle header is not included in the Input Module-Defined data type/tag definition.
- Heartbeat – no application data.
- 32 bit run/idle header – a 32 bit run/idle header is included in the Input Module-Defined data type or tag definition.
- Safety (for CIP Safety connections)

O->T Connection type

Must indicate POINT2POINT. May also specify NULL. For more information, see below for *T®O and O®T Connection types with NULL type selected.*

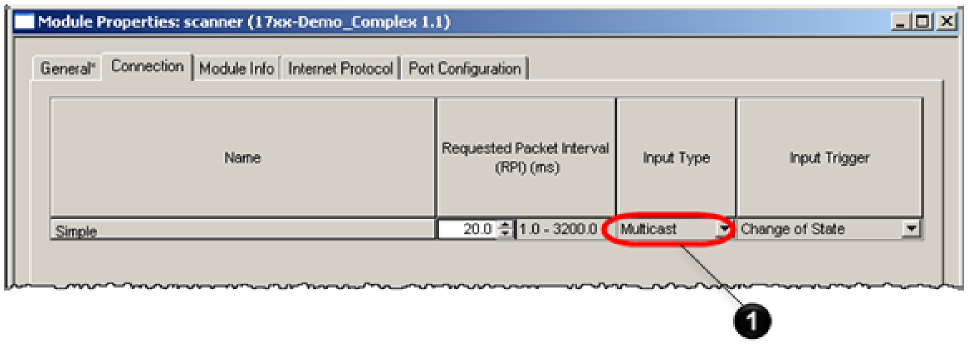
Point to Point is the Forward_Open O®T Network Connection parameter, Connection Type field value.

T->O Connection type

The T®O Connection type must indicate POINT2POINT, MULTICAST, or both; may also specify NULL. For more information, see below for *T®O and O®T Connection types with NULL type selected*.

The T®O Connection type displays in the **Input Type** column of the **Connection** tab. Users select **Unicast** or **Multicast** as the T®O Connection type.

The user selection of Point to Point or Multicast is the Forward_Open T®O Network Connection parameter, Connection Type field value.



Name	Requested Packet Interval (RPI) (ms)		Input Type	Input Trigger
Simple	20.0	1.0 - 3200.0	Multicast	Change of State

NOTE: Module Properties dialog box, **Connection** tab, **Input Type** column contain the T®O Connection type selections **Unicast** and **Multicast**.

T->O and O->T Connection types with NULL type selected

The module can be reconfigured by a NULL Forward_Open service (while the I/O connection is open) if these conditions are true:

- Both T→O and O→T NULL connection types are indicated.
- The Write Allowed bit group in the configuration assembly descriptor has a value of 0 (assembly may or may not be written when I/O connection is open) or the Write Allowed bit group in the configuration assembly descriptor has a value of 1 (assembly can be written when any I/O connection is open).

A NULL Forward_Open service is sent when a MSG instruction, Module Reconfiguration Message Type is executed.

T->O and O->T Priority

The T®O and O®T Priority must indicate either both T®O and O®T HIGH priority or both T®O and O®T SCHEDULED priority. Both HIGH and SCHEDULED can be indicated.

High or Scheduled (Scheduled if both are specified) is the Forward_Open T®O and O®T Network Connection parameters, Priority field values.

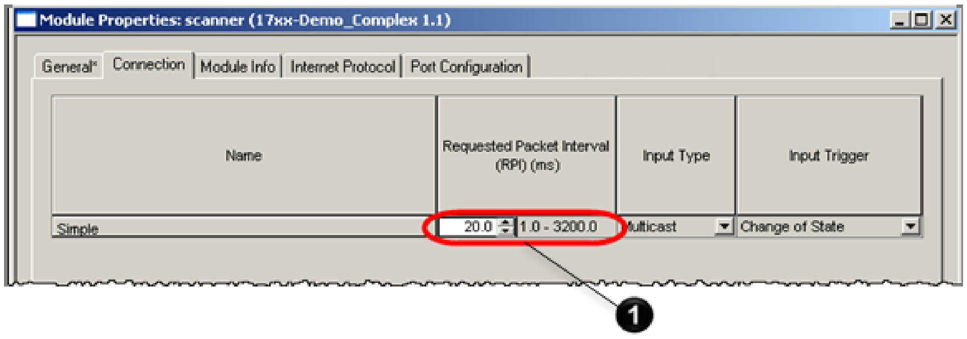
T->O and O->T RPI fields

If both fields are empty, the default RPI for the bus type will be selected and the default range for the bus type will be enabled.

For Ethernet bus – minimum RPI is 1.0ms, maximum RPI is 3200.0ms, and default RPI is 20.0ms. If a UDINT value or ParamN is specified, the intersection of the T®O and O®T value(s) and bus minimum and maximum values are used to generate the minimum and maximum RPI values. If no intersection exists, cannot choose an RPI value and as a consequence, cannot add the connection via EDS AOP.

Important:	Enumerated ParamNs are not supported for RPIs.

The RPI displays in the **Modules Properties** dialog box, **Connection** tab, **Requested Packet Interval (RPI) (ms)** column.



Name	Requested Packet Interval (RPI) (ms)		Input Type	Input Trigger
Simple	20.0	1.0 - 3200.0	Multicast	Change of State

NOTE: Module Properties dialog box, **Connection** tab, **Requested Packet Interval (RPI) (ms)** column contains the RPI value.

The user selected RPI value is the Forward_Open T®O and O®T RPI parameter values.

If the O®T production is heartbeat, then a longer O®T RPI value is calculated and used to reduce unneeded traffic.

See also

Connection tab on page 24

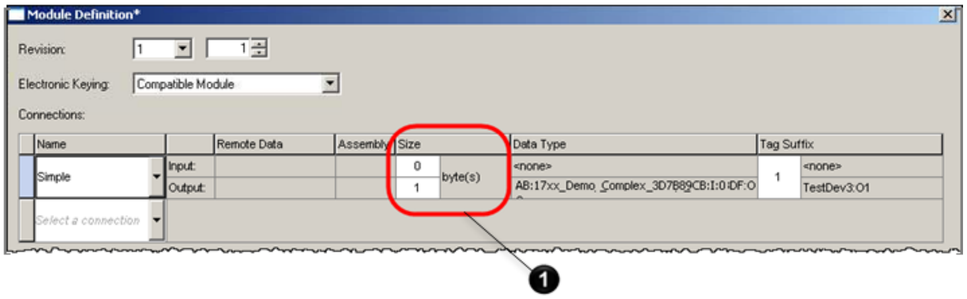
T->O and O->T Size fields

The T®O and O®T Size field must be either UINT or empty.

If the size field is empty, the size of the item specified in the corresponding format field size plus sequence count size (2) and optional 32 bit run/idle header size (4) is the Forward_Open T®O and O®T Network Connection parameter, Connection Size values.

If the size field is a UINT, then the size field value plus sequence count size (2) and optional 32 bit run/idle header size (4) is the Forward_Open T®O and O®T Network Connection parameter, Connection Size values.

The T®O and O®T Size displays in the **Module Definition** dialog box, **Size** column.



Module Definition*

Revision: 1 1

Electronic Keying: Compatible Module

Connections:

Name		Remote Data	Assembly	Size		Data Type	Tag Suffix	
Simple	Input:			0	byte(s)	<none>	1	<none>
	Output:			1		AB:17xx_Demo_Complex_3D7B89CB:I:0:DF:O		TestDev3:O1
Select a connection								

NOTE: Module Definition dialog box, **Size** column contains T®O and O®T Size value.

If the size field is a ParamN, then the user selected size plus sequence count size (2) and optional 32 bit run/idle header size (4) are the Forward_Open T®O and O®T Network Connection parameter, Connection Size values. If the size field is a parameter, either range or enumeration can be supported, but not both.

T->O and O->T Format fields

If the format field is empty, no Input or Output Module-Defined data type or tag is created.

IMPORTANT: To create Input/Output Module-Defined data types and tags, specify the format field even when the data format is dynamic.

In this case:

- An AssemN must be specified as the format with the Allow Value Edit bit of the Descriptor of this AssemN set, and
- The size of the AssemN must be specified by the AssemN Size field, or
- A single Member Size/Member Reference pair with an empty Member Reference field.

If the format specifies one or more ParamN entries, a Module-Defined data type and tag is created if the format results in one or more ParamNs.

IMPORTANT: The Input Module-Defined data types created for T®O includes a BOOL member named ConnectionFaulted in the first 32 bits.

The Input Module-Defined data type created for T®O includes a BOOL member named RunMode in the first 32 bits if the T®O Real Time transfer format is 32-bit run/idle header.

The Module-Defined data types created have a data member of type SINT, INT, DINT or REAL array as chosen by the user in the **Module Definition** dialog box, **Size** column.

- The type choice of INT is only allowed when the I/O sizes (in bytes) is divisible by 2.
- The type choices of DINT and REAL are allowed when the I/O sizes (in bytes) are divisible by 4.
- If the size fields are ParamNs, then the maximum values of the ParamN are used to determine if INT, DINT and REAL choices are available.
- If the maximum values are divisible by 2, the choice of SINT and INT are available.
- If the maximum values are divisible by 4 then the choices of SINT, INT, DINT and REAL are available.

Proxy Config Size and Format fields

If not empty, the connection is ignored.

Target Config Size field

If the size field is empty, the size of the item specified in the corresponding format field size is the Forward_Open Connection_Path parameter data segment size value (converted from bytes to words and rounded up).

If the size field is a UINT, then the size field is the Forward_Open Connection_Path parameter data segment size value (converted from bytes to words and rounded up).

If the size field is a ParamN, the default value of the ParamN is the Forward_Open Connection_Path parameter data segment size value (converted from bytes to words and rounded up).

Target Config Format field

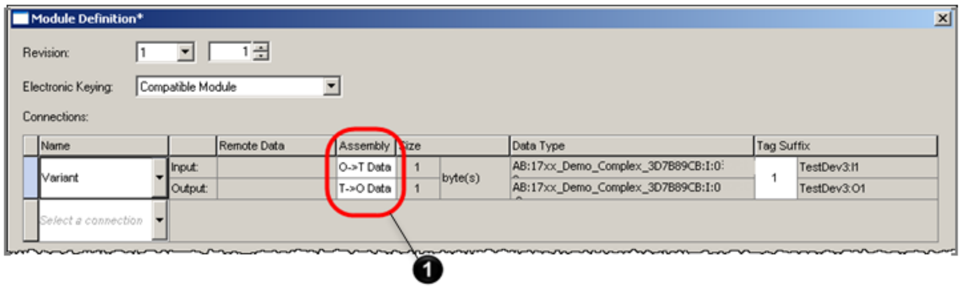
If the format field is empty and the Target Config Size field contains a non-null size, the Forward_Open Connection_Path parameter data segment is filled with zeros.

If the format field is a ParamN or AssemN, the value of the ParamN or AssemN is the Forward_Open Connection_Path parameter data segment data values.

If the format specifies one or more ParamN entries, a Module-Defined data type and tag is created if the format results in one or more ParamNs.

Some configuration parameters are not included in the configuration Module- Defined data type and tag. The parameters not included are:

- I/O assembly variant selectors – if a configuration parameter is an I/O variant selector, the variant selections are displayed on the **Module Definition** dialog box, **Assembly** column.



Revision: 1 1

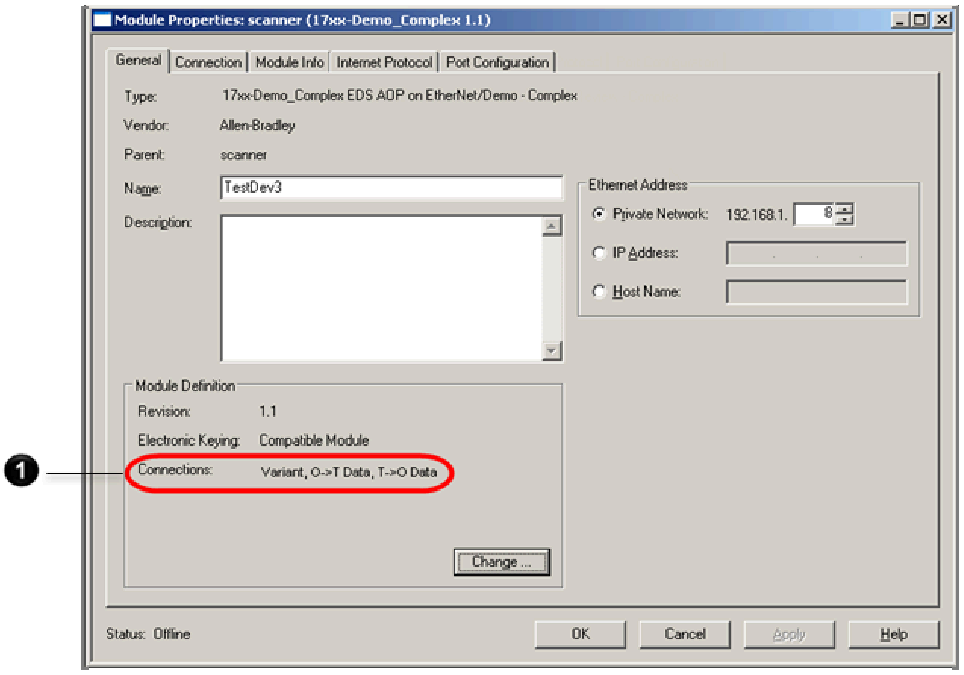
Electronic Keying: Compatible Module

Connections:

Name		Remote Data	Assembly	Size		Data Type	Tag Suffix	
Variant	Input:		O->T Data	1	byte(s)	AB:17xx_Demo_Complex_3D7B89CB:I:0	1	TestDev3:I1
	Output:		T->O Data	1		AB:17xx_Demo_Complex_3D7B89CB:I:0		TestDev3:O1
Select a connection								

NOTE: If a configuration parameter is an I/O variant selector, the variant selections are shown in the **Module Definition** dialog box, **Assembly** column.

- I/O assembly variant selectors – if a configuration parameter is an I/O variant selector, the variant selections show in the **Module Properties** dialog box, **General** tab, **Module Definition** group, **Connections** area.


Module Properties: scanner (17xx-Demo_Complex 1.1)

Type: 17xx-Demo_Complex EDS AOP on EtherNet/Demo - Complex
Vendor: Allen-Bradley
Parent: scanner
Name: TestDev3
Description:

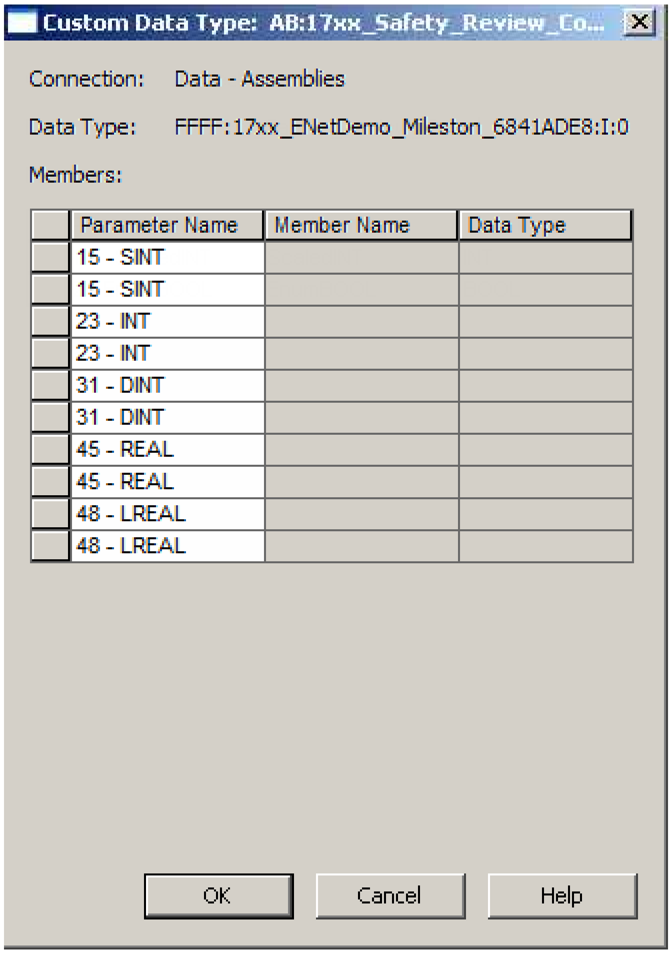
Ethernet Address
Private Network: 192.168.1. 8
IP Address:
Host Name:

Module Definition
Revision: 1.1
Electronic Keying: Compatible Module
Connections: Variant, O->T Data, T->O Data

Status: Offline

NOTE: If a configuration parameter is an I/O variant selector, the variant selections show in the **Module Properties** dialog box, **General** tab, **Module Definition** group, **Connections** area.

- Indirect I/O parameter – if a configuration parameter is an indirect I/O parameter, the pointed to name displays on the **Custom Data Type** dialog box (launched from **Module Definition** dialog box).



Custom Data Type: AB:17xx_Safety_Review_Co...

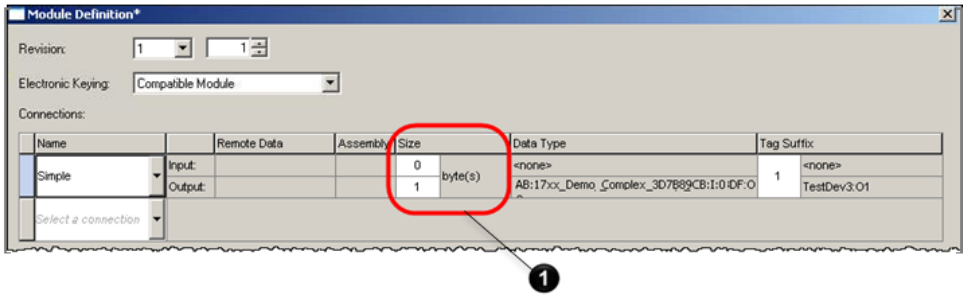
Connection: Data - Assemblies

Data Type: FFFF:17xx_ENetDemo_Mileston_6841ADE8:I:0

Members:

Parameter Name	Member Name	Data Type
15 - SINT		
15 - SINT		
23 - INT		
23 - INT		
31 - DINT		
31 - DINT		
45 - REAL		
45 - REAL		
48 - LREAL		
48 - LREAL		

- I/O size parameter – if a configuration parameter is used to specify a variable I/O size, it displays on the **Module Definition** dialog box, **Size** column.



Module Definition*

Revision: 1 1

Electronic Keying: Compatible Module

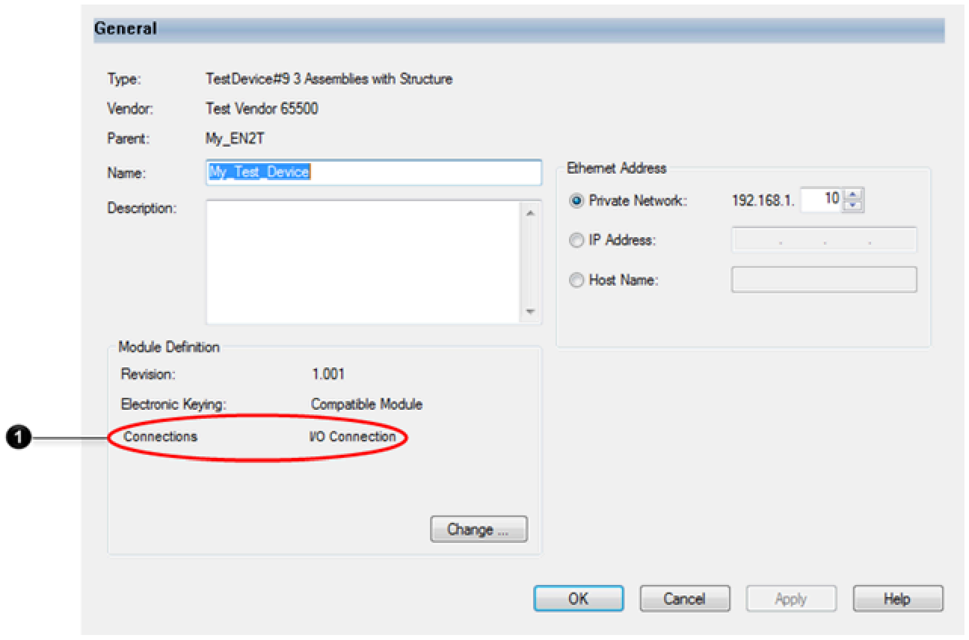
Connections:

Name		Remote Data	Assembly	Size		Data Type	Tag Suffix	
Simple	Input:			0	byte(s)	<none>	1	<none>
	Output:			1		AB:17xx_Demo_Complex_3D7B89CB:I:0:DF:O		TestDev3:O1
Select a connection								

NOTE: If a configuration parameter is used to specify a variable I/O size, it displays on the **Module Definition** dialog box, **Size** column.

Connection Name String field

The Connection Name String displays in the **Module Properties** dialog box, **General** tab, **Module Definition** group, **Connections** area.



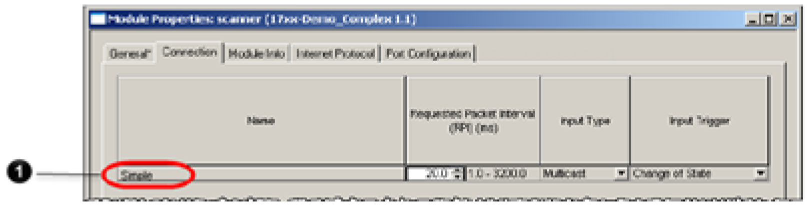
Type: TestDevice#9 3 Assemblies with Structure
Vendor: Test Vendor 65500
Parent: My_EN2T
Name: My_Test_Device
Description:

Ethernet Address
(●) Private Network: 192.168.1. 10
( ) IP Address:
( ) Host Name:

Module Definition
Revision: 1.001
Electronic Keying: Compatible Module
Connections I/O Connection

NOTE: The Connection Name String displays in the **Module Properties** dialog box, **General** tab, **Module Definition** group, **Connections** area.

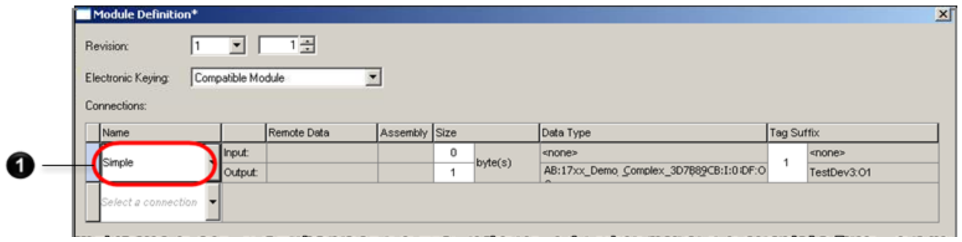
The Connection Name string displays in the **Module Properties** dialog box, **Connection** tab, **Connections** grid.



Name	Requested Packet Interval (RPI) (ms)		Input Type	Input Trigger
Simple	20.0	1.0 - 3200.0	Multicast	Change of State

NOTE: The Connection Name string displays in the **Module Properties** dialog box, **Connection** tab, **Connections** grid.

The Connection Name string displays in the **Module Definition** dialog box, **Connections** grid.



Revision: 1 1
Electronic Keying: Compatible Module
Connections:

Name		Remote Data	Assembly	Size		Data Type	Tag Suffix	
Simple	Input:			0	byte(s)	<none>	1	<none>
	Output:			1		AB:17xx_Demo_Complex_3D7B89CB:I:0:DF:O		TestDev3:O1
Select a connection								

NOTE: The Connection Name string displays in the **Module Definition** dialog box, **Connections** grid.

Limiting the name strings to 22 characters is recommended. This allows displaying the chosen connection name without resizing the dialog box.

Help String field

Ignored.

Path field

Ignored.

[ParamClass] section

The [ParamClass] section includes the CfgAssembly, CfgAssemblyExa, Descriptor, and SafetyCfgAssembly keywords.

CfgAssembly keyword

Ignored.

CfgAssemblyExa keyword

Ignored.

Descriptor keyword

Ignored.

SafetyCfgAssembly keyword

Required.

See *CIP NETWORKS LIBRARY*, *Volume 5, Chapter 7*.

[Assembly] section

The [Assembly] section includes the AssemN/AssemExaN and VariantN/VariantExaN keywords.

AssemN/AssemExaN keyword

Used only for assemblies selected by I/O variants. If the Internationalized Assembly Name String is defined, it is used instead of the name defined in the *[Assembly] section* in all cases.

Supported uses

AssemN/AssemExaN keywords have these supported uses:

- ConnectionN O®T Format
- ConnectionN T®O Format
- ConnectionN Target Config Format

For more information about these supported uses, See *ConnectionN keyword*.

Nested assemblies

Assemblies can be nested. The result of nested assemblies is nested structures.

AssemN/AssemExaN fields

The AssemN/AssemExaN fields include:

- Name
- Path
- Size
- Descriptor
- Member Size
- Member Reference

Name field

If this AssemN/AssemExaN is specified as a VariantExaN Selection Entry for a variant specified for a ConnectionN T®O or O®T Format, then the name displays on the **Module Definition** dialog box, **Assembly** column.

If this AssemN/AssemExaN is specified as a VariantExaN Selection Entry for a variant specified for a ConnectionN T®O or O®T Format, then the name displays on the **Module Properties** dialog box, **General** tab, **Module Definition** group, **Connections** area.

Path field

Ignored.

Size field

The Size field defines the Assembly size. The Size field is empty if the Member Size/Member Reference is provided for all members.

Rockwell Automation recommends leaving this field blank when the Member Size/Member Reference is provided.

Descriptor field

The Descriptor field:

- Supports settable path

  Ignored.
- Get_Enumerated_String Supported

  Ignored.

  Enumerations are supported as described in other parts of this document without this bit being set.
- Supports Scaling

Ignored.

- Supports Scaling Links

  Ignored.
- Read Only Parameter

  If the parameter described by the ParamN entry is included in the module configuration, it is hidden in the Configuration module-defined data type and will not be included in an output Module-Defined tag. It will also not appear in the output data. The output reverts to an array of SINT.
- Monitor Parameter

  Ignored.
- Supports Extended Precision

  Ignored.
- Supports non-consecutive enumerated strings
- Ignored.
- Allows both enumeration and min/max range values

  Ignored.
- Non-displayed parameter

  If set, ParamN is not displayed.
- Indirect Parameter reference

  If set and the ParamN is in a Configuration assembly, it shows in the **Custom Data Type** dialog box, **Module Definition**. If the ParamN is in an Input or Output assembly, the selected ParamN in the **Custom Data Type** dialog box, **Module Definition**, is used as the Module-Defined data type member.
- Not Addressable

  Ignored.
- Save Supported

  Ignored.
- Apply Supported

  Ignored.
- Write Only Parameter

  Ignored.

Member Size field

No EDS AOP specific handling.

Member Reference field

If this AssemN/AssemExaN is specified as the ConnectionN Target Config Format, then this member is included in the configuration Module-Defined data type.

If this AssemN/AssemExaN is specified as the ConnectionN T®O or O®T Format, then the size of this member is included in the Input/ Output Module-Defined data type array size.

VariantN/VariantExaN keyword

The Variant/VariantExaN keywords are supported as defined in the CIP Common specification, see *THE CIP NETWORKS LIBRARY Volume 1, Common Industrial Protocol (CIP™), Edition 3.11*.

The supported uses of the Variant/VariantExaN keywords are listed in the description of the referencing keyword. See *[Connection Manager] section*, *ConnectionN keyword* and *Target Config Format field* descriptions, and *[Assembly section]*, *AssemN/AssemExaN keyword*, *Name field* description.

[Params] section

The [Params] section includes support for these keywords:

- ParamN keyword
- EnumN keywords

ParamN keyword

Ignored

International Parameter Name field

Ignored

International Engineering Units field

Ignored

International Help String field

Ignored

Input/Output/Configuration details uses

Input/Output/Configuration details uses includes:

- ConnectionN O®T Format.
- ConnectionN T®O Format.
- ConnectionN Target Config Format.

  See the *ConnectionN keyword* for more information.

  The ParamN name should use Module-Defined data type member name naming conventions. See *Parameter Name field* for more information.

ParamN fields

The ParamN keyword includes these fields:

- Link Path
- Descriptor

- Data Type
- Data Size
- Parameter Name
- Units String
- Help String
- Minimum Value
- Maximum Value
- Default Value
- Link (Multiplier, Divider, Base, and Offset)
- Scaling (Multiplier, Divider, Base, and Offset)
- Decimal Precision
- International Parameter Name
- International Engineering Units
- International Help String

Link Path field

Ignored.

Descriptor field

The Descriptor field:

- Supports settable path

  Ignored.

- Get_Enumerated_String Supported

  Ignored.

  Enumerations are supported as described in other parts of this document without this bit being set.

- Supports Scaling

  Ignored.

- Supports Scaling Links

  Ignored.

- Read Only Parameter

  If the parameter described by the ParamN entry is included in the module configuration, it is hidden in the Configuration module-defined data type and will not be included in an output Module-Defined tag. It will also not appear in the output data. The output reverts to an array of SINT.

- Monitor Parameter

Ignored.

- Supports Extended Precision

  Ignored.

- Supports non-consecutive enumerated strings
- Ignored.
- Allows both enumeration and min/max range values

  Ignored.

- Non-displayed parameter

  If set, ParamN is not displayed.

- Indirect Parameter reference

  If set and the ParamN is in a Configuration assembly, it shows in the **Custom Data Type** dialog box, **Module Definition**. If the ParamN is in an Input or Output assembly, the selected ParamN in the **Custom Data Type** dialog box, **Module Definition**, is used as the Module-Defined data type member.

- Not Addressable

  Ignored.

- Save Supported

  Ignored.

- Apply Supported

  Ignored.

- Write Only Parameter

  Ignored.

Data Type field

STRING2, STRINGN, EPATH, and STRINGI types are not supported. The ParamN entry is discarded and any keyword that specifies the ParamN entry as a field value is discarded.

If ParamN is in a Configuration assembly, the data type determines the Module- Defined data type member type:

- CIP BOOL type results in controller BOOL type.
- CIP SINT type results in controller SINT type.
- CIP INT type results in controller INT type.
- CIP DINT type results in controller DINT type.
- CIP LINT type results in controller LINT type.
- CIP USINT type results in controller USINT type.

  If the maximum value of the unsigned 8-bit integer is 128 or larger, the entire Module-Defined data type will be a SINT array.

- CIP UINT type results in controller UINT type.

If the maximum value of the unsigned 16-bit integer is 32768 or larger, the entire Module-Defined data type will be a SINT array.

- CIP UDINT type results in controller UDINT type.

  If the maximum value of the unsigned 32-bit integer is 2147483648 or larger, the entire Module-Defined data type will be a SINT array.

- CIP ULINT type results in controller ULINT type.

  If the maximum unsigned ULINT value would result in a negative LINT value, the entire Module-Defined data type becomes a SINT array.

- CIP REAL type results in controller REAL type.
- CIP LREAL type results in controller LREAL type.
- CIP DATE type results in controller UINT type.
- CIP TIME_OF_DAY type results in controller UDINT type.
- CIP STRING type results in array of controller USINT type.
- CIP BYTE type results in controller USINT type.
- CIP WORD type results in controller UINT type.
- CIP DWORD type results in controller UDINT type.
- CIP LWORD type results in controller ULINT types.
- CIP FTIME type results in controller TIME32 type.
- CIP ITIME type results in controller INT type.
- CIP SHORT_STRING type results in array of controller USINT type.
- CIP NTIME type results in controller LTIME type
- CIP LTIME type results in controller TIME type
- CIPT TIME type results in controller DINT type
- CIP UTIME type results in controller DT type
- CIP STIME type results in controller LDT type
- CIP DATE_AND_TIME type results in controller UINT (DATE) and UDINT (TIME)
- CIP ENGUNIT type results in controller UINT type

Data Size field

The Data Size field follows the CIP specification. See *THE CIP NETWORKS LIBRARY Volume 1, Common Industrial Protocol (CIP™).*

Parameter Name field

If the ParamN is a member of a Configuration assembly, the Parameter Name is used as the Module-Defined data type member name. The Parameter Name is converted to conform to the IEC 61131 name standard of alpha-numeric characters or underscore (_) characters. All unsupported characters, including spaces, are replaced with an underscore (_) character during the conversion, consecutive underscore (_) characters and trailing underscore (_) characters are removed and leading numbers are prefixed with an underscore character. Do not use underscore and space characters in Parameter Names.

Follow these guidelines for Parameter Names that are included in Configuration assemblies:

- Use short Member names when practical. Users do not want to type ConfigDataBits when Data suffices. The user can infer much from the context. Data appearing in an input data type is obviously input data. Data from a discrete module is almost always represented as a bit. The goal for Module-Defined data type member names is 15 characters or less.
- Capitalize the first letter in each word making up the parameter name. This allows the words to stand out when the data type name is compressed after spaces are removed.
- Use consistent naming. Refer to existing data types as a guide.
- Do not use redundant naming.
- Do not use the data type in the parameter name.
- Naming of BOOLEAN members should be for the positive sense of the member. Note that this requires the proper design of the device. Also note that the on state is not necessarily the most interesting state.
- Prefix (not post fixed) channel data with Ch followed by the channel number. It is not spelled out since it has become a common convention.
- In cases when using individual homogeneous bits named (versus indexed through a Data array), prefixed (not post fixed) the individual bits with Pt followed by the 2-digit point number.

Units String field

Ignored.

Help String field

Ignored.

Minimum Value field

Ignored if the parameter is part of a Configuration assembly. If used in the **Module Definition** dialog box, this field represents the minimum value that the user can choose.

Maximum Value field

If in a Configuration assembly, used to determine if a parameter can be properly represented. If used in the **Module Definition** dialog box, this field represents the maximum value that can be chosen.

Default Value field

Used as defined in the CIP specification. See *THE CIP NETWORKS LIBRARY Volume 1, Common Industrial Protocol (CIP™), Edition 3.11.*

Link (Multiplier, Divider, Base and Offset) fields

Ignored.

Scaling (Multiplier, Divider, Base and Offset) fields

Ignored.

Decimal Precision field

Ignored.

International Parameter Name field

Ignored.

International Engineering Units field

Ignored.

International Help String field

Ignored.

ParamN enumeration, EnumN

If the EnumN is for a bit string (BYTE, WORD, DWORD, LWORD) ParamN that is a member of a configuration assembly, then the bit name is used as the Module-Defined data type member name. The bit name is converted to conform to the IEC 61131 name standard. All unsupported characters, including spaces, are replaced with an underscore (_) character during the conversion, consecutive underscore (_) characters and trailing underscore (_) characters are removed and leading numbers are prefixed with an underscore (_) character.

Follow these guidelines for bit names that are included in Configuration assemblies:

- Use short Bit names when practical.
- Capitalize the first letter in each word making up the bit name. This allows the words to stand out when the data type name is compressed after spaces are removed.
- Use consistent naming (for example, Data and Status). If two different modules return status, name both Status. Refer to existing data types as a guide as opposed to getting creative.
- Do not use redundant naming. For example, O.On instead of O.OutputOn.
- Do not use the data type in the name. For example, On, not OnBit.
- Naming of bits should be for the positive sense of the member. FuseBlown is 1 when the fuse is blown, 0 when it is not. It is generally preferable to name a bit member in a

manner that reflects the interesting state rather than the steady state value. In most cases, this is the on, enabled, or energized state versus the off state. For example, FuseBlown is preferable to FuseOK or FuseNotBlown. Note that this requires the proper design of the device. Note also that the on state is not necessarily the most interesting state.

[Groups] section

Ignored.

[Internationalization] section

The language the Logix Designer application displays determines which string from the EDS file is used for the items below. If a string for the Logix Designer application language is not specified, then the string from the non-Internationalization section version of the keyword is used.

The keywords listed below should have English, French, German, Spanish, Portuguese, Italian, Korean, Japanese, and Chinese strings because Logix Designer application is translated to each of these languages.

ProdName keyword

If this ProdName is defined, it is used instead of the ProdName specified in the [Device] section.

ParamN keyword

Ignored

International Parameter Name field

Ignored

International Engineering Units field

Ignored

International Help String field

Ignored

EnumN keyword

Ignored.

GroupN keyword

Ignored

AssemN/AssemExaN keyword

Used only for assemblies selected by I/O variants. If the Internationalized Assembly Name String is defined, it is used instead of the name defined in the *[Assembly] section* in all cases.

ConnectionN keyword

If the Internationalized Connection Name String is defined, it is used instead of the name defined in the *[Connection Manager] section* in all cases.

The Internationalized Connection Help String is not used.

[Modular] section

Ignored.

[Ethernet Link Class] section

According to the ODVA specifications, devices must implement one instance of the Ethernet Link object per external EtherNet/IP port. It is recommended that the EDS file should also contain the [Ethernet Link Class] section.

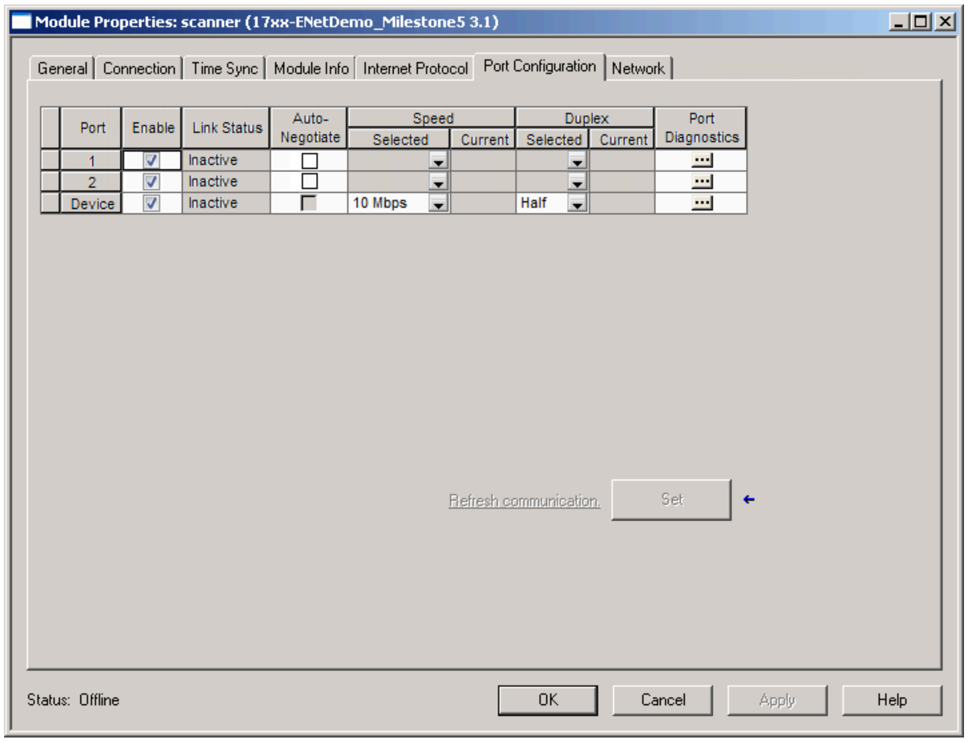
IMPORTANT: If the module has more than one Ethernet port, this is reflected in the EDS as specified in the ODVA specifications.

Define these keywords:

- Number_Of_Static_Instances
- MaxInst
- Either
  - InterfaceLabelN
  - In the [Params] section, create one or more ParamN entries with Path value of 20 F6 24 NN 30 0A where NN is the Ethernet Link object instance. The default value of these ParamN entries should be the interface label string.

    If no interface label is specified, a default label is displayed, the instance value.

The **Port Configuration** tab display is affected by these items. It is also recommended that the device implement the Ethernet Link object Interface Counters and Media Counters attributes.



Port	Enable	Link Status	Auto-Negotiate	Speed Selected	Speed Current	Duplex Selected	Duplex Current	Port Diagnostics
1	✓	Inactive	☐					...
2	✓	Inactive	☐					...
Device	✓	Inactive	☐	10 Mbps		Half		...

Status: Offline

[TCP/IP Interface Class] section

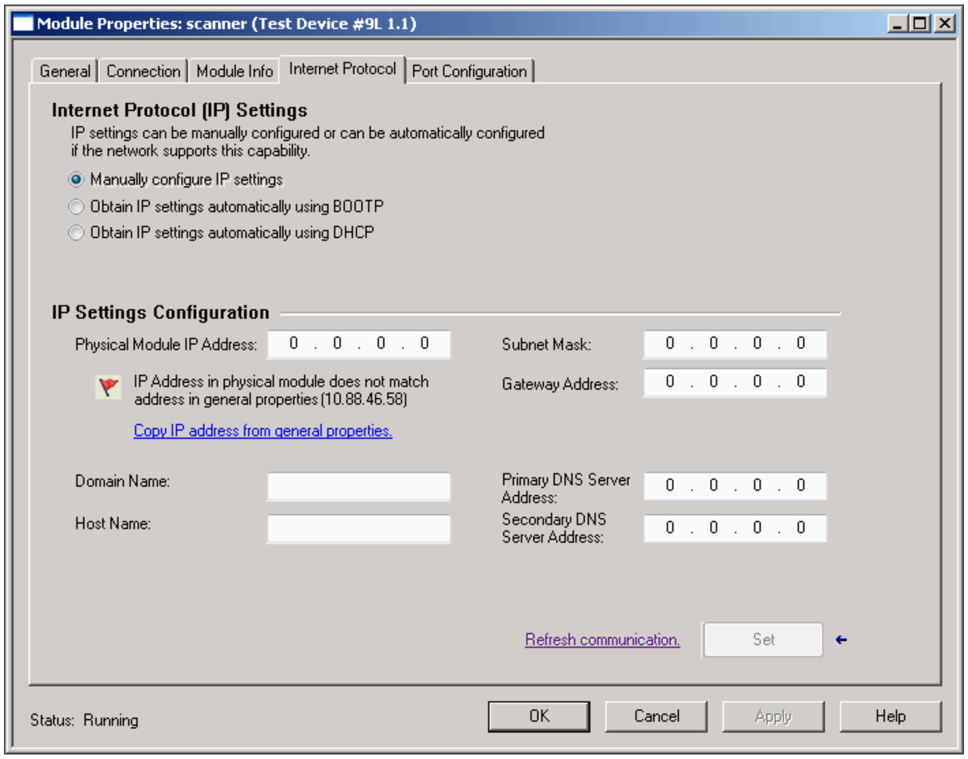
The TCP/IP Interface object is a required object for EtherNet/IP devices. As such, the EDS file should include the [TCP/IP Interface Class] section.

Define these keywords:

- In the [Params] section, create a ParamN entry with a Path value of 20 F5 24 01 30 02. The default value of this ParamN should represent the TCP/IP object capabilities for this module. This is necessary for the **Internet Protocol** tab to properly reflect the device's capabilities.

If no ParamN entry exists, the value of 0 is used for the TCP/ IP capabilities attribute.

The **Internet Protocol** tab is affected by this item.



Module Properties: scanner (Test Device #9L 1.1)

General | Connection | Module Info | Internet Protocol | Port Configuration

Internet Protocol (IP) Settings

IP settings can be manually configured or can be automatically configured if the network supports this capability.

[✓ Manually configure IP settings]

Obtain IP settings automatically using BOOTP

Obtain IP settings automatically using DHCP

IP Settings Configuration

Physical Module IP Address: 0 . 0 . 0 . 0

Subnet Mask: 0 . 0 . 0 . 0

IP Address in physical module does not match address in general properties (10.88.46.58)

Gateway Address: 0 . 0 . 0 . 0

Copy IP address from general properties.

Domain Name:

Primary DNS Server Address: 0 . 0 . 0 . 0

Host Name:

Secondary DNS Server Address: 0 . 0 . 0 . 0

Refresh communication. Set

Status: Running

OK Cancel Apply Help

[DLR Class] section

If the device is capable of being a ring supervisor, this keyword assignment enables ring supervisor functionality on the **Network** tab.

Ring_Supervisor_Capable = Yes

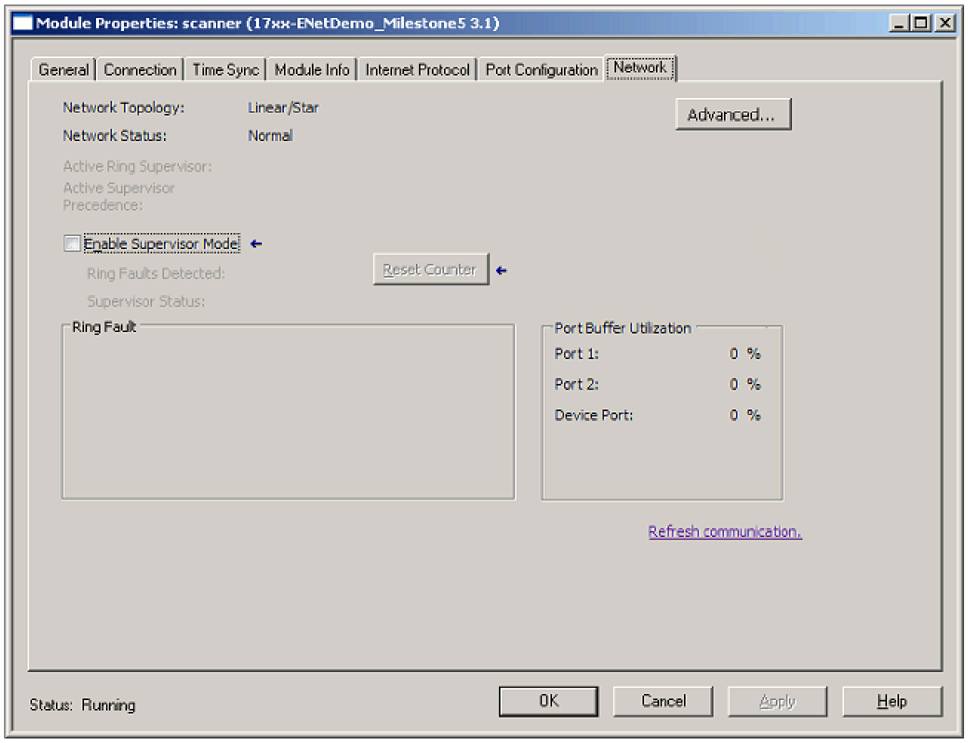
If the device cannot be a ring supervisor, use the value **No** for this keyword.

If the keyword is omitted, the device is assumed to not have supervisor capability.

If the device is capable of being a ring supervisor, also define these:

- In the [Assembly] section, create an AssemN entry with Path value of 20 47 24 01 30 04.
  - The AssemN entry should include the Member References for each member of the Ring Supervisor Config attribute. The ParamN that represents the Beacon Interval member should specify the minimum and maximum values supported for this module's Beacon Interval.
  - If no Ring Supervisor Config AssemN entry exists, 400 microseconds is used as the minimum Beacon Interval and 100,000 microseconds is used as the maximum Beacon Interval.

The **Network** tab displays if this section exists.



Network Topology: Linear/Star
Network Status: Normal
Active Ring Supervisor:
Active Supervisor Precedence:
Ring Faults Detected:
Supervisor Status:
Ring Fault
Port Buffer Utilization
Port 1: 0 %
Port 2: 0 %
Device Port: 0 %
Status: Running

[Time Sync Class] section

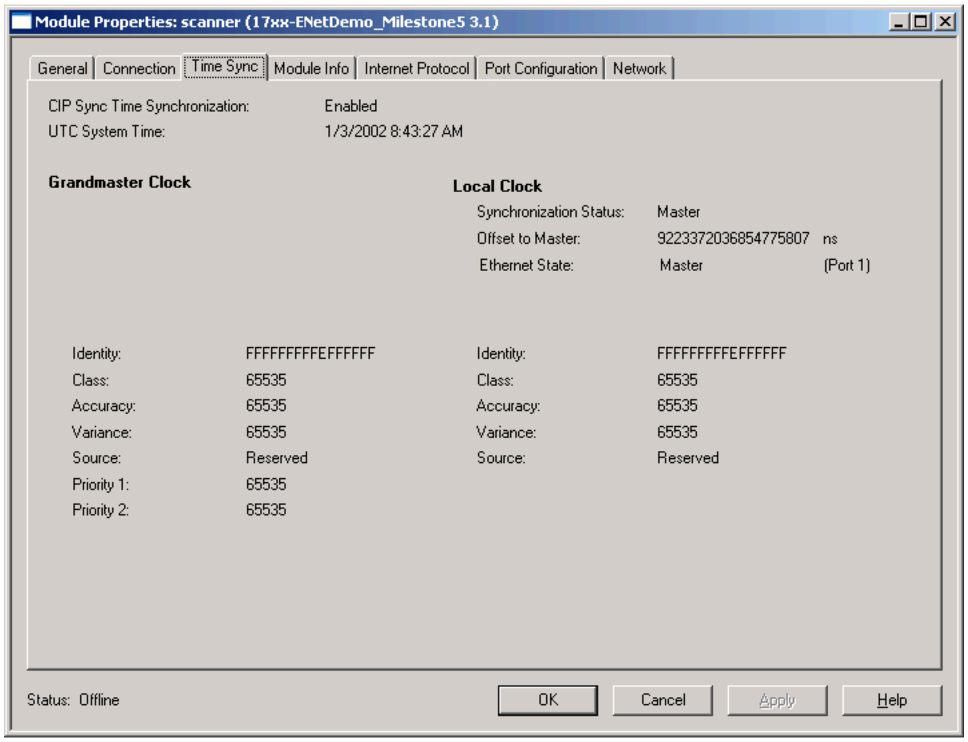
If the module supports the Time Sync object, this section should be included in the EDS file. The information found in this dialog box comes from the required attributes of the object class.

Define the keyword:

- In the [Params] section, create a ParamN entry with a Path value of 20 43 24 01 30 0B. This ParamN is used to initialize the **Time Sync** dialog box with the number of time sync ports. If this does not exist, then the value of 1 is used. This is necessary for the **Time Sync** dialog box to properly reflect the device's current time sync information.

If the product has a Rockwell Automation vendor ID, a check is made to see if vendor specific service 0x35 of the Time Sync object is supported, or if the 1_Time_Sync_Request_Clock_Description_Extension keyword is defined in the Time Sync section. This is used to initialize the **Time Sync** dialog box, indicating the ReqClockDescription service is supported.

The **Time Sync** tab displays if the [Time Sync Class] section exists.



CIP Sync Time Synchronization: Enabled
UTC System Time: 1/3/2002 8:43:27 AM

Grandmaster Clock

Identity:	FFFFFFFFFFFFFFFF
Class:	65535
Accuracy:	65535
Variance:	65535
Source:	Reserved
Priority 1:	65535
Priority 2:	65535

Local Clock

Synchronization Status: Master
Offset to Master: 9223372036854775807 ns
Ethernet State: Master (Port 1)

Identity:	FFFFFFFFFFFFFFFF
Class:	65535
Accuracy:	65535
Variance:	65535
Source:	Reserved

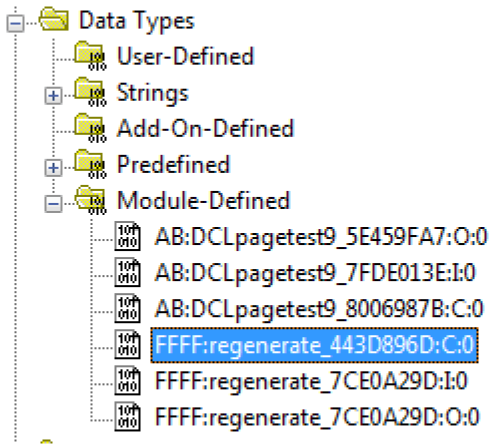
Status: Offline

Description of data types and tags

This section describes the rules the EDS AOP follows for creating Module- Defined Configuration, Input, and Output data type names, tag names, and Module-Defined data type member names.

Module-Defined data type naming

Module-defined data types define the structure of the data used by the module for its Input, Output, and Configuration Data. These tags provide access to this data via the controller's program. This section describes the rules Logix Designer application uses for naming module-defined data types. The names display in the **Controller Organizer,** under **Data Types>Module-Defined**.


Data Types
User-Defined
Strings
Add-On-Defined
Predefined
Module-Defined
AB:DCLpagetest9_5E459FA7:O:0
AB:DCLpagetest9_7FDE013E:I:0
AB:DCLpagetest9_8006987B:C:0
FFFF:regenerate_443D896D:C:0
FFFF:regenerate_7CE0A29D:I:0
FFFF:regenerate_7CE0A29D:O:0

A C/I/O Module Defined data type name is constructed from:
<Vendor>:<Catalog_Number>_<CRC>:<Tag_Type>:<Version>

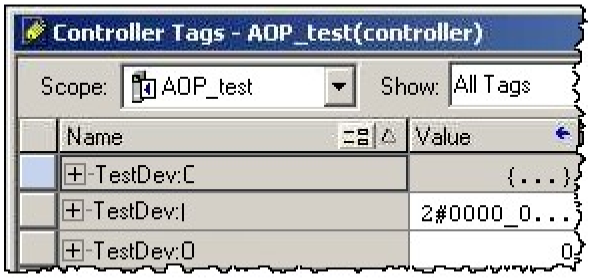
- Vendor code: the 4-digit hex value assigned by ODVA.
- Catalog_Number string value:

  The Catalog_Number String comes from the Catalog string in the EDS file. Dash (-) characters are converted to underscore (_) characters and all non-IEC-61131 characters are removed. If no Catalog string exists in the EDS file, the 4 digit hex ProdType and 4 digit hex ProdCode are used, separated by an underscore (_) character (for example: 0002_0006). The Catalog_Number string is truncated to limit the Module-Defined data type name length to 40 characters.
- CRC string value: Generated from the Module-Defined data type contents, the details of this algorithm are not provided.
- Tag_Type string value:
  - C
  - I
  - O
  - SI
  - SO
- Version string value: Always 0.

Tag naming

Tag naming follows these principles:

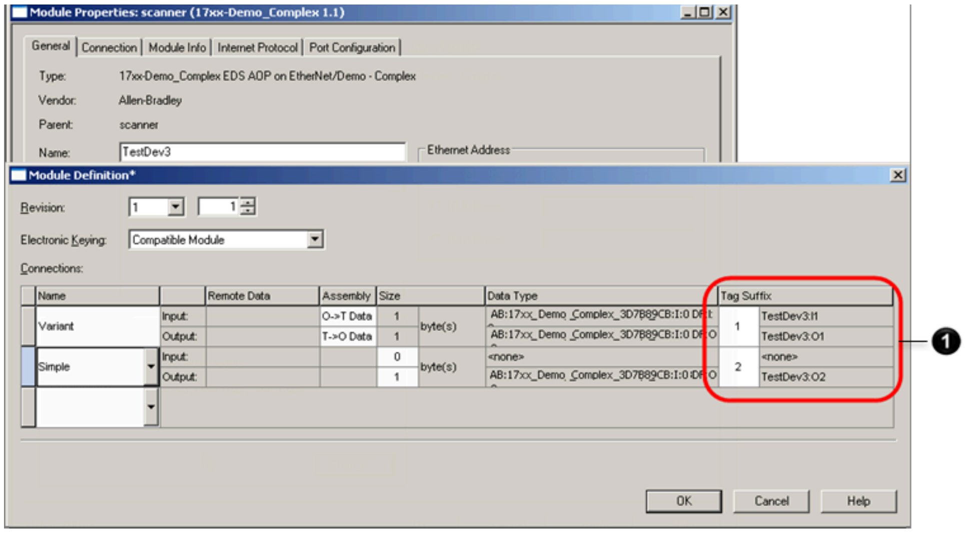
- Use the Name on the **Module Properties** dialog box, **General** tab:
  - <Name>:<I, O, SI, or SO>[<Ordinal>]
  - <Name>:C


Controller Tags - AOP_test(controller)

Scope: AOP_test Show: All Tags

Name	Value
+-TestDev:C	{...}
+-TestDev:I	2#0000_0...
+-TestDev:O	0

- The ordinal name comes from the **Module Definition** dialog box, **Tag Suffix** column. Ordinal only exists when the user can configure more than one connection.


Module Properties: scanner (17xx-Demo_Complex 1.1)

Type: 17xx-Demo_Complex EDS AOP on EtherNet/Demo - Complex
Vendor: Allen-Bradley
Parent: scanner
Name: TestDev3
Ethernet Address

Module Definition*

Revision: 1 1
Electronic Keying: Compatible Module
Connections:

Name		Remote Data	Assembly	Size		Data Type	Tag Suffix	
Variant	Input:		O->T Data	1	byte(s)	AB:17xx_Demo_Complex_3D7B89CB:I:0 DI	1	TestDev3:I1
	Output:		T->O Data	1		AB:17xx_Demo_Complex_3D7B89CB:I:0 DO		TestDev3:O1
Simple	Input:			0	byte(s)	<none>	2	<none>
	Output:			1		AB:17xx_Demo_Complex_3D7B89CB:I:0 DO		TestDev3:O2

NOTE: The ordinal name comes from the **Module Definition** dialog box, **Tag Suffix** column.

Module-Defined data type member naming

Member naming for module-defined data types follow these principles:

- Member names are truncated to 40 characters.
- Duplicate member names are made unique by appending an ordinal suffix.

Example EDS files

These sections describe examples of EDS files and how these EDS files translate into details in Logix Designer application.

The example EDS files are attached to this PDF file. In Adobe Acrobat. or Adobe Reader, you display the Attachments panel by:

- Selecting the small paperclip icon in the bottom left corner of the Acrobat or Reader window.
- Selecting the **View** > **Navigation Panels** > **Attachments** command to display the Attachments panel.

See *Access the attachments* for more details.

Example EDS files are discussed in these section:

- Minimum EDS on page 70
- Enhanced EDS, single connection on page 71
- Enhanced EDS, single connection, configuration data on page 72
- Enhanced EDS, multiple choice of connections and multiple simultaneous connections on page 72
- Enhanced EDS, safety and standard connections on page 74

Minimum EDS

This section describes the minimum that is required to make an EDS usable in a Logix Designer application environment, plus a number of extensions that improve its usability.

The Minimal EDS file described in this section makes use of the Allow Value Edit bit while the more complex EDS files described later use parameters to structure the assemblies. Rockwell Automation strongly recommends adding more details to an EDS as outlined in the more detailed EDS files shown later. The Minimal EDS file is shown here, mainly for reference and to break the evolution of an EDS into several steps.

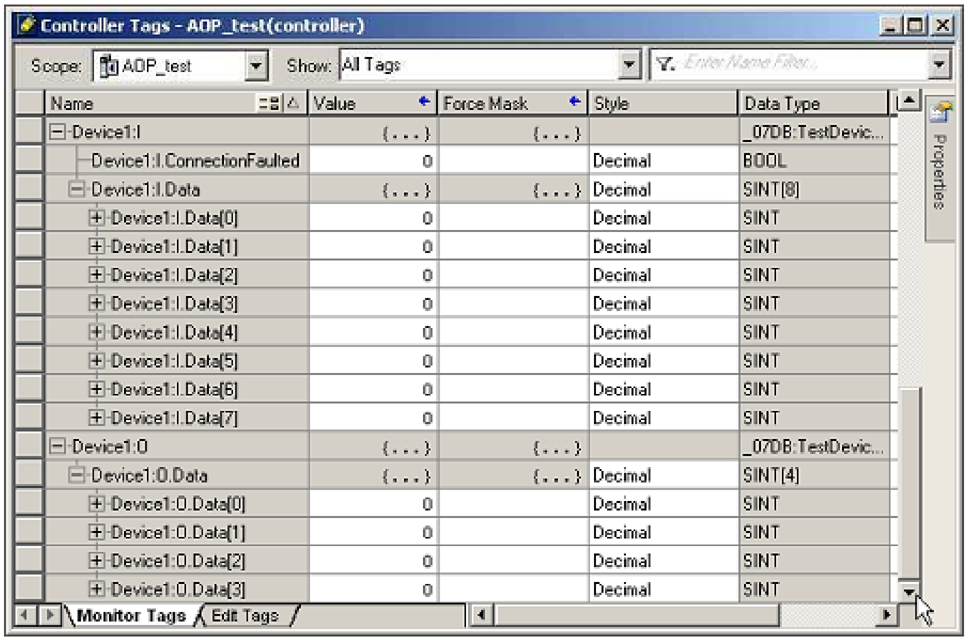
With the Minimal EDS, the Logix Designer application EDS AOP offers only these choices and structures.

- Choice of IP address.
- Selection of Unicast and Multicast data production where appropriate and offered by the EDS.
- Selection of Cyclic or Change-of-State trigger if offered by the EDS.
- Creation of an Input tag (an Exclusive-Owner Connection, an Input-Only Connection, or a Listen-Only Connection) and possibly an Output tag (only for an Exclusive-Owner Connection), depending on what is offered in the EDS.
- Choice of SINT, INT, DINT, or REAL data structure for the I/O tags.

This sample EDS actually contains only an entry for an Exclusive-Owner Connection. See the attached *Minimal EDS.eds* file for details.

The Minimal EDS file creates only an Input data tag and an Output data tag of the defined sizes. No structure or meaning of the I/O data is visible. Logix Designer application adds a status bit to the Input tag that reports when the connection fails.

The resulting tags appear with data type SINT:



Controller Tags - AOP_test(controller)

Scope: AOP_test Show: All Tags Enter Name Filter...

Name	Value	Force Mask	Style	Data Type
Device1:I	{...}	{...}		_07DB:TestDevic...
Device1:I.ConnectionFaulted	0		Decimal	BOOL
Device1:I.Data	{...}	{...}	Decimal	SINT[8]
Device1:I.Data[0]	0		Decimal	SINT
Device1:I.Data[1]	0		Decimal	SINT
Device1:I.Data[2]	0		Decimal	SINT
Device1:I.Data[3]	0		Decimal	SINT
Device1:I.Data[4]	0		Decimal	SINT
Device1:I.Data[5]	0		Decimal	SINT
Device1:I.Data[6]	0		Decimal	SINT
Device1:I.Data[7]	0		Decimal	SINT
Device1:O	{...}	{...}		_07DB:TestDevic...
Device1:O.Data	{...}	{...}	Decimal	SINT[4]
Device1:O.Data[0]	0		Decimal	SINT
Device1:O.Data[1]	0		Decimal	SINT
Device1:O.Data[2]	0		Decimal	SINT
Device1:O.Data[3]	0		Decimal	SINT

Monitor Tags / Edit Tags

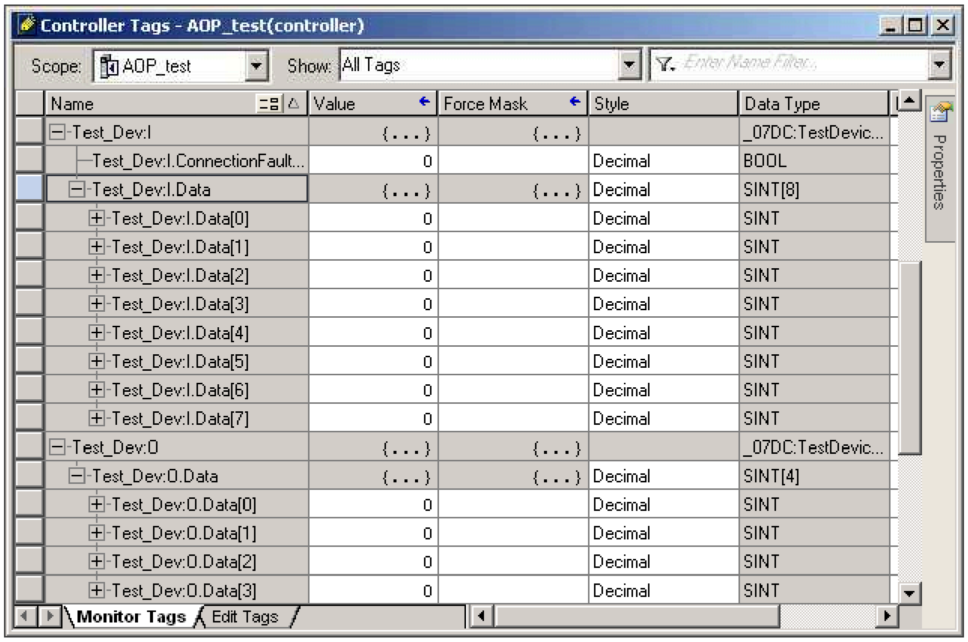
Enhanced EDS, single connection

For a better user experience, add a range limitation for Requested Packet Interval (RPI).

See the attached *One Connection, IO Assemblies with Structures EDS.eds* file for details. The EDS file includes six parameters and two assemblies.

- Param1 limits the RPI range to values between 4.0 and 100.0ms with a default value of 10.0ms.

This module appears in Logix Designer application:



Controller Tags - AOP_test(controller)

Scope: AOP_test Show: All Tags Enter Name Filter...

Name	Value	Force Mask	Style	Data Type
Test_Dev:I	{...}	{...}		_07DC:TestDevic...
Test_Dev:I.ConnectionFault...	0		Decimal	BOOL
Test_Dev:I.Data	{...}	{...}	Decimal	SINT[8]
Test_Dev:I.Data[0]	0		Decimal	SINT
Test_Dev:I.Data[1]	0		Decimal	SINT
Test_Dev:I.Data[2]	0		Decimal	SINT
Test_Dev:I.Data[3]	0		Decimal	SINT
Test_Dev:I.Data[4]	0		Decimal	SINT
Test_Dev:I.Data[5]	0		Decimal	SINT
Test_Dev:I.Data[6]	0		Decimal	SINT
Test_Dev:I.Data[7]	0		Decimal	SINT
Test_Dev:O	{...}	{...}		_07DC:TestDevic...
Test_Dev:O.Data	{...}	{...}	Decimal	SINT[4]
Test_Dev:O.Data[0]	0		Decimal	SINT
Test_Dev:O.Data[1]	0		Decimal	SINT
Test_Dev:O.Data[2]	0		Decimal	SINT
Test_Dev:O.Data[3]	0		Decimal	SINT

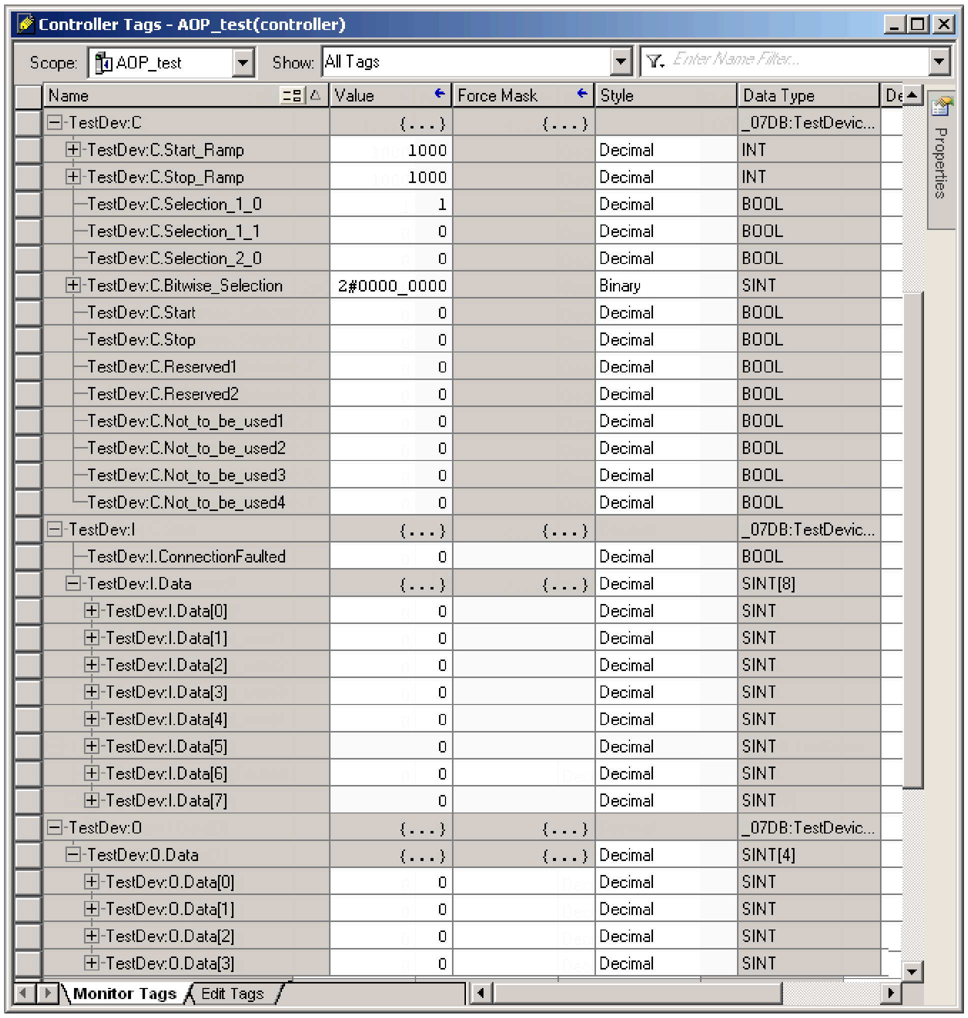
Monitor Tags / Edit Tags

Enhanced EDS, single connection, configuration data

The *One Connection, 3 Assemblies with Structures EDS.eds* file includes a configuration assembly structured into individual parameters, in addition to the features of the *One Connection, IO Assemblies with Structures EDS.eds* file.

There are now a total of 11 parameters in the EDS file. The additional parameters, 7 through 11, describe the structure of the Configuration Assembly.

This module appears in the Logix Designer application tag structure:



Controller Tags - AOP_test(controller)

Scope: AOP_test Show: All Tags

Name	Value	Force Mask	Style	Data Type
TestDev:C	{...}	{...}		_07DB:TestDevic...
TestDev:C.Start_Ramp	1000		Decimal	INT
TestDev:C.Stop_Ramp	1000		Decimal	INT
TestDev:C.Selection_1_0	1		Decimal	BOOL
TestDev:C.Selection_1_1	0		Decimal	BOOL
TestDev:C.Selection_2_0	0		Decimal	BOOL
TestDev:C.Bitwise_Selection	2#0000_0000		Binary	SINT
TestDev:C.Start	0		Decimal	BOOL
TestDev:C.Stop	0		Decimal	BOOL
TestDev:C.Reserved1	0		Decimal	BOOL
TestDev:C.Reserved2	0		Decimal	BOOL
TestDev:C.Not_to_be_used1	0		Decimal	BOOL
TestDev:C.Not_to_be_used2	0		Decimal	BOOL
TestDev:C.Not_to_be_used3	0		Decimal	BOOL
TestDev:C.Not_to_be_used4	0		Decimal	BOOL
TestDev:I	{...}	{...}		_07DB:TestDevic...
TestDev:I.ConnectionFaulted	0		Decimal	BOOL
TestDev:I.Data	{...}	{...}	Decimal	SINT[8]
TestDev:I.Data[0]	0		Decimal	SINT
TestDev:I.Data[1]	0		Decimal	SINT
TestDev:I.Data[2]	0		Decimal	SINT
TestDev:I.Data[3]	0		Decimal	SINT
TestDev:I.Data[4]	0		Decimal	SINT
TestDev:I.Data[5]	0		Decimal	SINT
TestDev:I.Data[6]	0		Decimal	SINT
TestDev:I.Data[7]	0		Decimal	SINT
TestDev:O	{...}	{...}		_07DB:TestDevic...
TestDev:O.Data	{...}	{...}	Decimal	SINT[4]
TestDev:O.Data[0]	0		Decimal	SINT
TestDev:O.Data[1]	0		Decimal	SINT
TestDev:O.Data[2]	0		Decimal	SINT
TestDev:O.Data[3]	0		Decimal	SINT

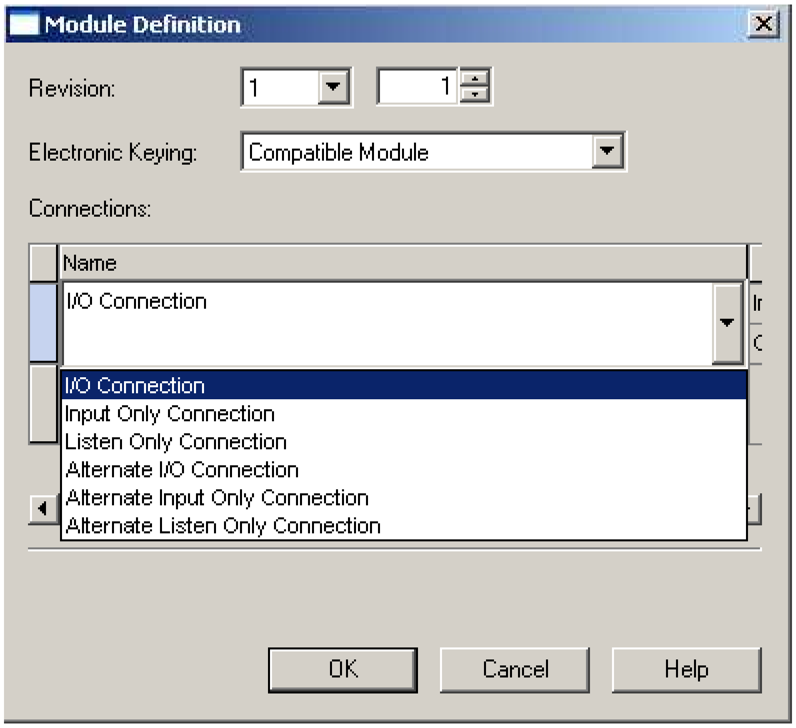
Enhanced EDS, multiple choice of connections and multiple simultaneous connections

The attached *Multiple Connections 3 Assemblies with Structures EDS.eds* file defines multiple ConnectionN entries for a module. There are typically a set of up to three ConnectionN entries describing one or more connections:

- An Exclusive-Owner Connection (if the module has Output data).
- An Input-Only Connection.
- A Listen-Only Connection.

Such a set of ConnectionN entries specify the same Input Connection Point and (if included) the same configuration data. Multiple sets like this may exist that describe connections using different I/O Assemblies.

With the *Multiple Connections 3 Assemblies with Structures EDS.eds* file, select one of the connections in the **Module Definition** dialog box. The first Exclusive-Owner Connection in the EDS displays as the default connection.



Module Definition

Revision: 1 1

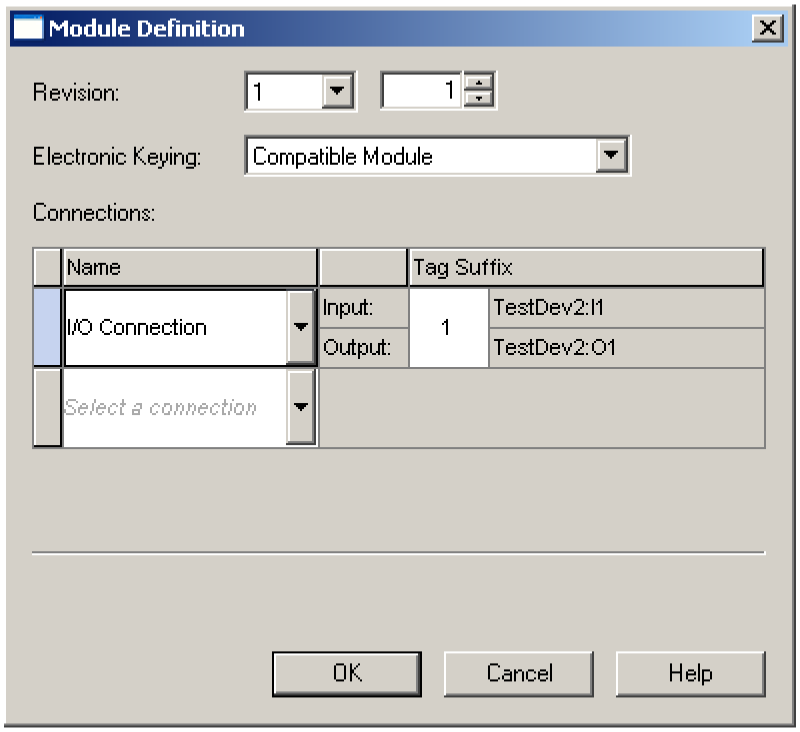
Electronic Keying: Compatible Module

Connections:

Name
I/O Connection

- I/O Connection
- Input Only Connection
- Listen Only Connection
- Alternate I/O Connection
- Alternate Input Only Connection
- Alternate Listen Only Connection

If Logix Designer application determines that there are multiple sets of ConnectionN entries, a group of connections is created.



Module Definition

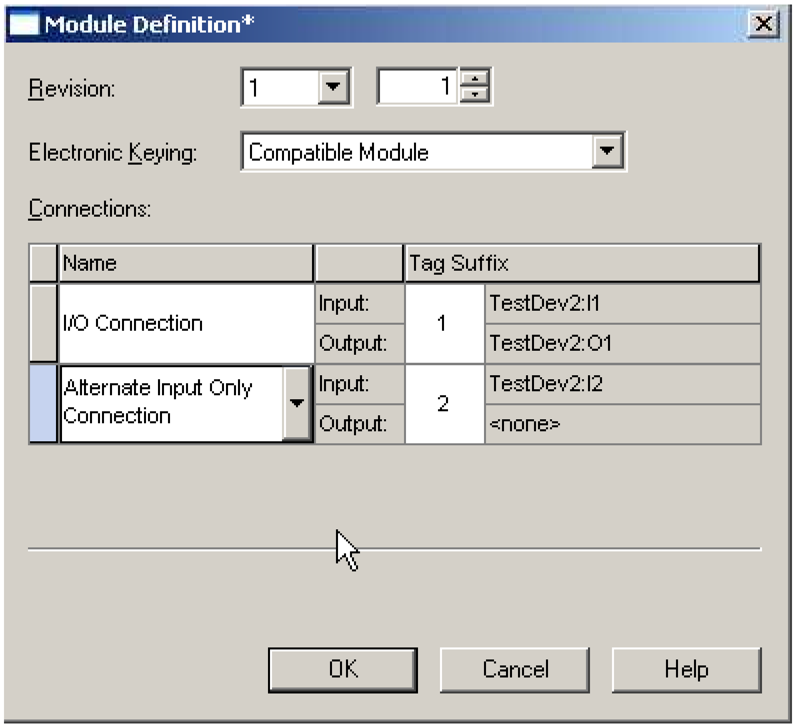
Revision: 1 1

Electronic Keying: Compatible Module

Connections:

Name			Tag Suffix
I/O Connection	Input:	1	TestDev2:I1
	Output:		TestDev2:O1
Select a connection			

Once a member of a group of connections has been chosen, further connections are only selected from ConnectionN entries outside that group.



Revision: 1 1

Electronic Keying: Compatible Module

Connections:

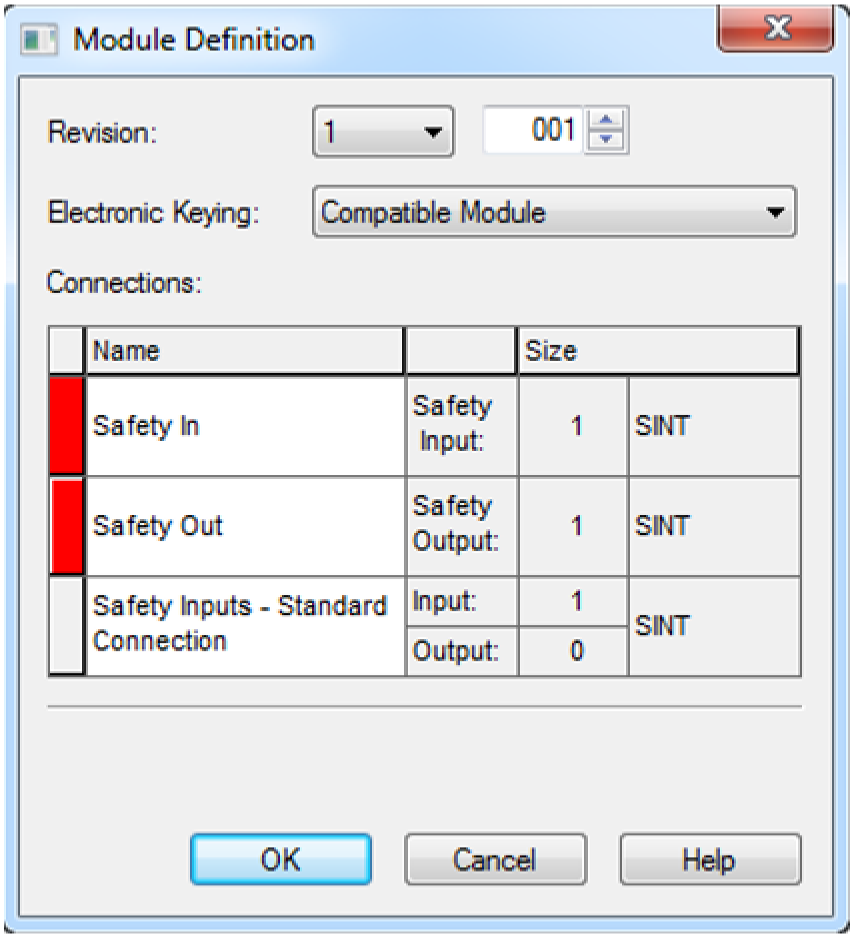
	Name			Tag Suffix
	I/O Connection	Input:	1	TestDev2:I1
		Output:		TestDev2:O1
	Alternate Input Only Connection	Input:	2	TestDev2:I2
		Output:		<none>

The tags created by multiple connections have numbers appended to make them unambiguous.

Enhanced EDS, safety and standard connections

An EDS file may define both safety and standard connections. See the attached *Std +SafetyInOutExample.eds* file for details. This example also shows the use of ParamN entries that are used by the **Connection Properties** and **Internet Protocol** tabs. See Params500 through Param510 in the EDS file.

With this EDS, the **Module Definition** dialog box lets users enable the safety input connection, safety output connection, standard connection, or a combination of these connections.



Module Definition

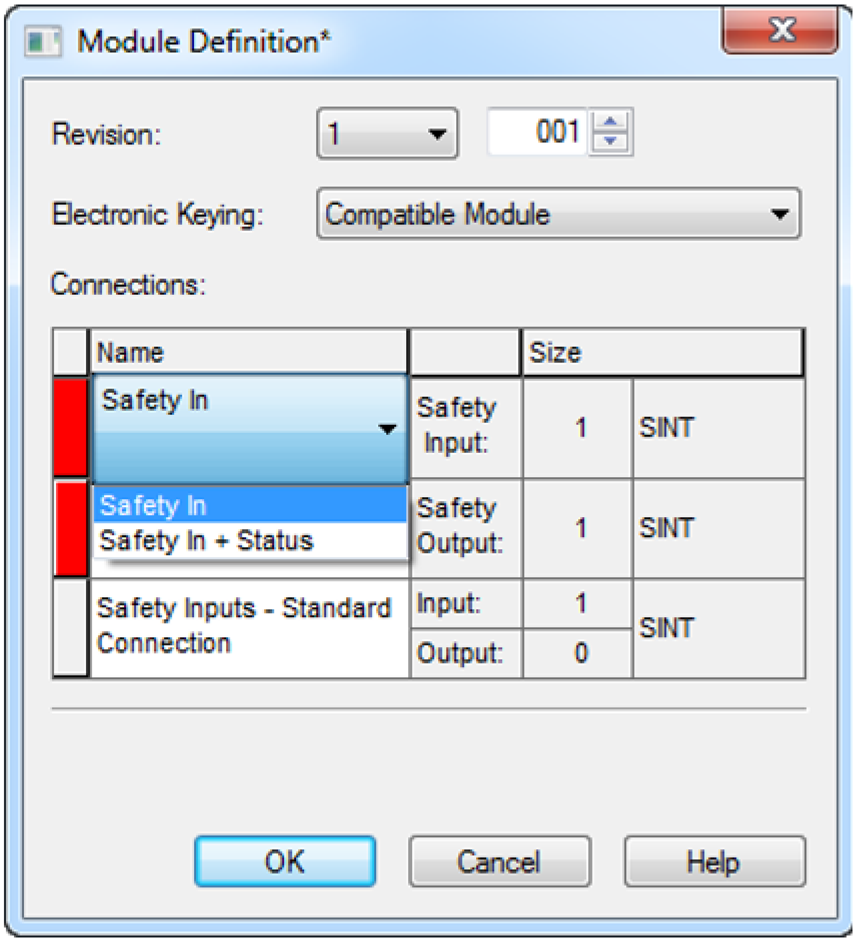
Revision: 1 001

Electronic Keying: Compatible Module

Connections:

	Name		Size	
🟥	Safety In	Safety Input:	1	SINT
🟥	Safety Out	Safety Output:	1	SINT
	Safety Inputs - Standard Connection	Input:	1	SINT
		Output:	0	

To select a connection, select the drop-down arrow next to the connection name.


Module Definition*

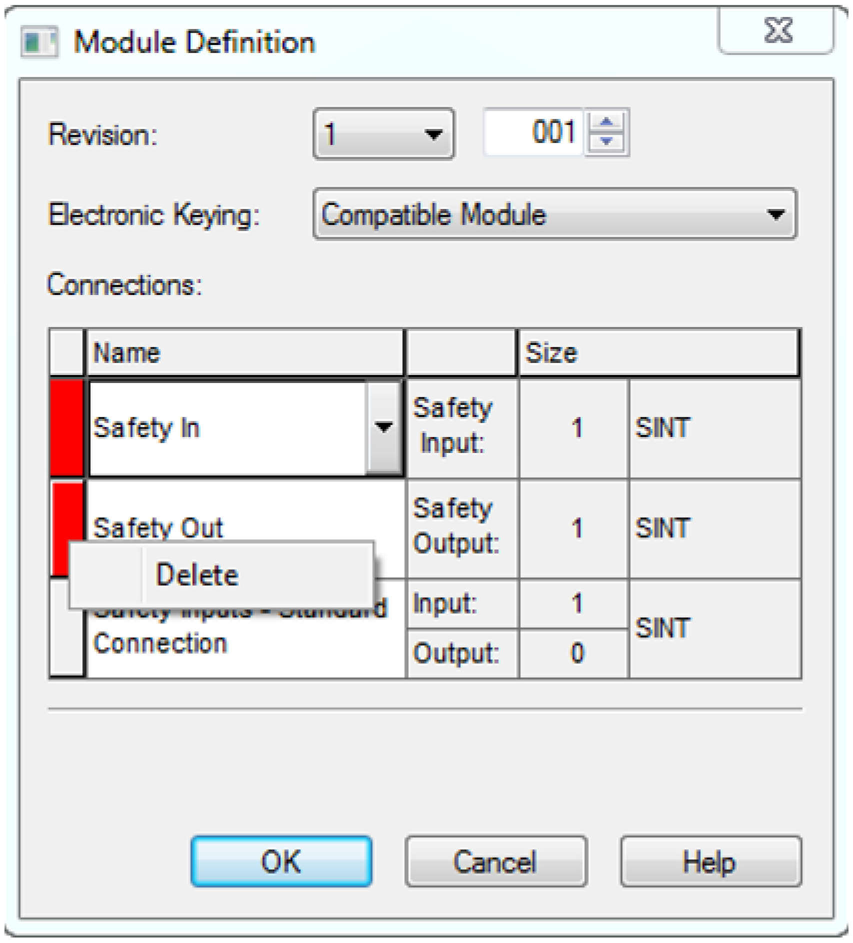
Revision: 1 001

Electronic Keying: Compatible Module

Connections:

	Name		Size	
🟥	Safety In	Safety Input:	1	SINT
🟥	Safety In / Safety In + Status	Safety Output:	1	SINT
	Safety Inputs - Standard Connection	Input:	1	SINT
		Output:	0	

To remove a connection, select the row's left bar and choose **Delete**.



Module Definition

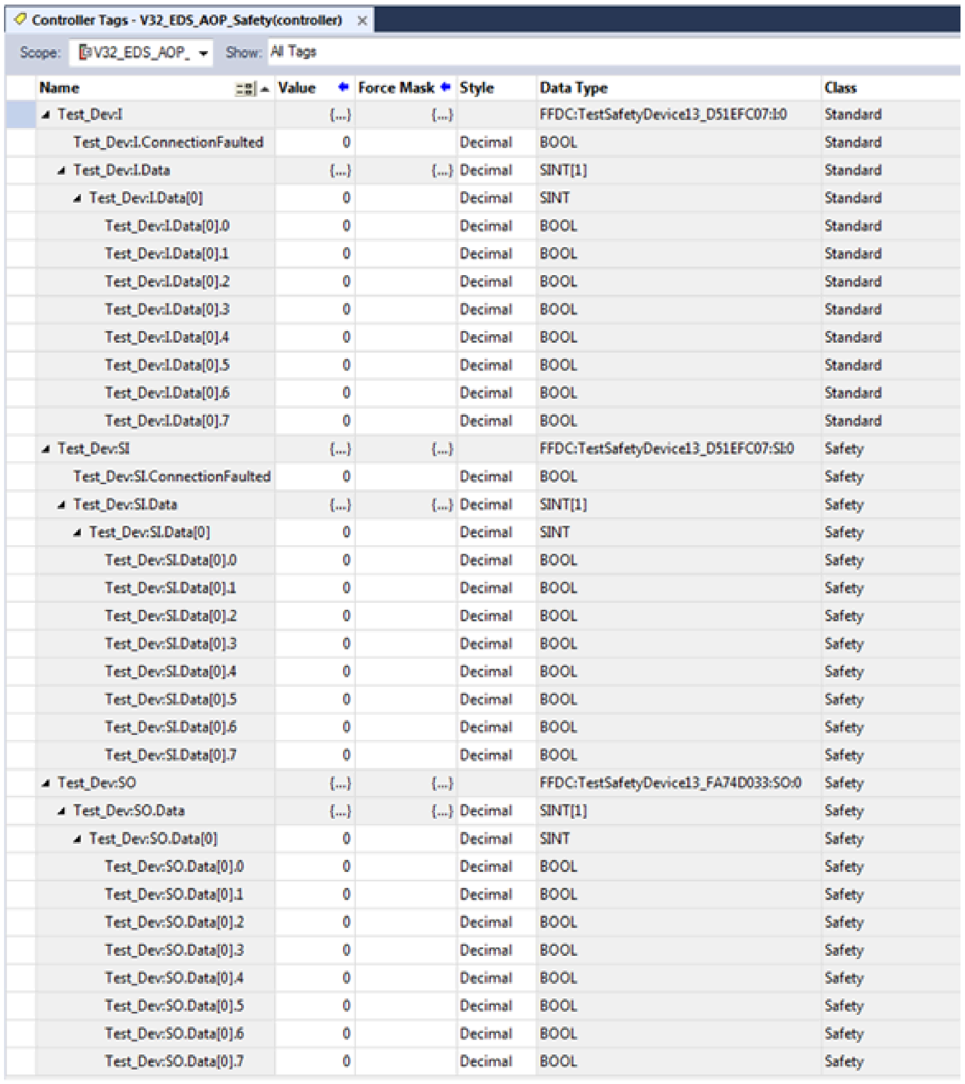
Revision: 1 001

Electronic Keying: Compatible Module

Connections:

	Name		Size	
🟥	Safety In	Safety Input:	1	SINT
🟥	Safety Out	Safety Output:	1	SINT
	Safety Inputs - Standard Connection	Input:	1	SINT
		Output:	0	

In the Logix Designer application, the standard connection is designated with the I tag type. The safety connection tags are designated with the SI and SO tag types.



Controller Tags - V32_EDS_AOP_Safety(controller)

Scope: V32_EDS_AOP_ Show: All Tags

Name	Value	Force Mask	Style	Data Type	Class
Test_Dev:I	{...}	{...}		FFDC:TestSafetyDevice13_D51EFC07:I:0	Standard
Test_Dev:I.ConnectionFaulted	0		Decimal	BOOL	Standard
Test_Dev:I.Data	{...}	{...}	Decimal	SINT[1]	Standard
Test_Dev:I.Data[0]	0		Decimal	SINT	Standard
Test_Dev:I.Data[0].0	0		Decimal	BOOL	Standard
Test_Dev:I.Data[0].1	0		Decimal	BOOL	Standard
Test_Dev:I.Data[0].2	0		Decimal	BOOL	Standard
Test_Dev:I.Data[0].3	0		Decimal	BOOL	Standard
Test_Dev:I.Data[0].4	0		Decimal	BOOL	Standard
Test_Dev:I.Data[0].5	0		Decimal	BOOL	Standard
Test_Dev:I.Data[0].6	0		Decimal	BOOL	Standard
Test_Dev:I.Data[0].7	0		Decimal	BOOL	Standard
Test_Dev:SI	{...}	{...}		FFDC:TestSafetyDevice13_D51EFC07:SI:0	Safety
Test_Dev:SI.ConnectionFaulted	0		Decimal	BOOL	Safety
Test_Dev:SI.Data	{...}	{...}	Decimal	SINT[1]	Safety
Test_Dev:SI.Data[0]	0		Decimal	SINT	Safety
Test_Dev:SI.Data[0].0	0		Decimal	BOOL	Safety
Test_Dev:SI.Data[0].1	0		Decimal	BOOL	Safety
Test_Dev:SI.Data[0].2	0		Decimal	BOOL	Safety
Test_Dev:SI.Data[0].3	0		Decimal	BOOL	Safety
Test_Dev:SI.Data[0].4	0		Decimal	BOOL	Safety
Test_Dev:SI.Data[0].5	0		Decimal	BOOL	Safety
Test_Dev:SI.Data[0].6	0		Decimal	BOOL	Safety
Test_Dev:SI.Data[0].7	0		Decimal	BOOL	Safety
Test_Dev:SO	{...}	{...}		FFDC:TestSafetyDevice13_FA74D033:SO:0	Safety
Test_Dev:SO.Data	{...}	{...}	Decimal	SINT[1]	Safety
Test_Dev:SO.Data[0]	0		Decimal	SINT	Safety
Test_Dev:SO.Data[0].0	0		Decimal	BOOL	Safety
Test_Dev:SO.Data[0].1	0		Decimal	BOOL	Safety
Test_Dev:SO.Data[0].2	0		Decimal	BOOL	Safety
Test_Dev:SO.Data[0].3	0		Decimal	BOOL	Safety
Test_Dev:SO.Data[0].4	0		Decimal	BOOL	Safety
Test_Dev:SO.Data[0].5	0		Decimal	BOOL	Safety
Test_Dev:SO.Data[0].6	0		Decimal	BOOL	Safety
Test_Dev:SO.Data[0].7	0		Decimal	BOOL	Safety

Rockwell Automation Support

Use these resources to access support information.

Technical Support Center	Find help with how-to videos, FAQs, chat, user forums, and product notification updates.	rok.auto/support
Local Technical Support Phone Numbers	Locate the telephone number for your country.	rok.auto/phonesupport
Technical Documentation Center	Quickly access and download technical specifications, installation instructions, and user manuals.	rok.auto/techdocs
Literature Library	Find installation instructions, manuals, brochures, and technical data publications.	rok.auto/literature
Product Compatibility and Download Center (PCDC)	Get help determining how products interact, check features and capabilities, and find associated firmware.	rok.auto/pcdc

Documentation Feedback

Your comments help us serve your documentation needs better. If you have any suggestions on how to improve our content, complete the form at rok.auto/docfeedback.

Waste Electrical and Electronic Equipment (WEEE)

At the end of life, this equipment should be collected separately from any unsorted municipal waste.

Rockwell Automation maintains current product environmental information on its website at rok.auto/pec.

Rockwell Otomasyon Ticaret A.Ş. Kar Plaza İş Merkezi E Blok Kat:6 34752 İçerenköy, İstanbul, Tel: +90 (216) 5698400 EEE Yönetmeliğine Uygundur

Connect with us.

rockwellautomation.com — expanding **human possibility**®

AMERICAS: Rockwell Automation, 1201 South Second Street, Milwaukee, WI 53204-2496 USA, Tel: (1) 414.382.2000
EUROPE/MIDDLE EAST/AFRICA: Rockwell Automation NV, Pegasus Park, De Kleetlaan 12a, 1831 Diegem, Belgium, Tel: (32) 2663 0600
ASIA PACIFIC: Rockwell Automation SEA Pte Ltd, 2 Corporation Road, #04-05, Main Lobby, Corporation Place, Singapore 618494, Tel: (65) 6510 6608
UNITED KINGDOM: Rockwell Automation Ltd., Pitfield, Kiln Farm, Milton Keynes, MK11 3DR, United Kingdom, Tel: (44)(1908) 838-800

Publication 1756-PM002G-EN-P - September 2025
Supersedes Publication 1756-PM002F-EN-D - November 2023