

# IEC 61850 MMS Client Driver

© 2018 PTC Inc. All Rights Reserved.

# Table of Contents

<b>IEC 61850 MMS Client Driver</b> .....	<b>1</b>
<b>Table of Contents</b> .....	<b>2</b>
IEC 61850 MMS Client Driver .....	4
Overview .....	4
<b>Setup</b> .....	<b>5</b>
Channel Properties — General .....	5
Channel Properties — Ethernet Communications .....	6
Channel Properties — Write Optimizations .....	6
Channel Properties — Advanced .....	7
Device Properties — General .....	8
Device Properties — Scan Mode .....	9
Device Properties — Timing .....	10
Device Properties — Auto-Demotion .....	11
Device Properties — Tag Generation .....	11
Device Properties — Automatic Configuration .....	13
Device Properties — Connection .....	15
Device Properties — Communication .....	17
Device Properties — Control .....	18
Device Properties — Tag Database Settings .....	19
Device Properties — Redundancy .....	20
<b>Data Types Description</b> .....	<b>22</b>
<b>Address Descriptions</b> .....	<b>23</b>
Supported Functional Constraints .....	24
<b>Error Descriptions</b> .....	<b>26</b>
Duplicate address <tag address> produced from SCL file. Tag will not be created. ....	27
Error: Device <device name>, Address <MMS address>, Invalid bit string data <bit string>. Please enter a string of characters '0' and '1'. ....	27
Error: Device <device name>, Address <MMS address>, Invalid data type <data type> for data attribute. ....	27
Error: Device <device name>, Address <MMS address>, Invalid octet string data <octet string>. Please enter an even number of hexadecimal digits (for example, "00 01 C3"). ....	27
Error: Device <device name>, Cannot write to structured data attribute <MMS address>. ....	28
Error: Device <device name>, Invalid object <MMS address>. ....	28
Error: Device <device name>, Structured data attribute <MMS address> cannot be applied to tag. ....	28
Failed to open SCL file <file path>. ....	29

---

Invalid address <tag address> produced from SCL file. Tag will not be created. ....	29
No ConnectedAP of SubNetwork <SubNetwork> with IED <IED> and AccessPoint <AccessPoint> found in file <file path>. ....	29
Selected AccessPoint <AccessPoint > not found in file <file path>. ....	29
Selected IED <IED name> not found in file <file path>. ....	29
Selected SubNetwork <SubNetwork> not found in file <file path>. ....	30
<b>Resources</b> .....	<b>31</b>
<b>Index</b> .....	<b>32</b>

## IEC 61850 MMS Client Driver

---

Help version 1.028

### CONTENTS

#### Overview

What is the IEC 61850 MMS Client Driver?

#### Device Setup

How do I configure devices for use with this driver?

#### Data Types Description

What data types does the IEC 61850 MMS Client Driver support?

#### Address Descriptions

How do I address a data location on a device?

#### Error Descriptions

What error messages are produced by the IEC 61850 MMS Client Driver?

### Overview

---

IEC 61850 is a modern electrical substation communication protocol designed with the goal of decreasing data management effort. It uses symbolic addresses to reduce the time-intensive data mapping of numerically-addressed protocols, supports several self-description services for online data configuration and validation, and supports meta data in addition to real, measured data.

The IEC 61850 MMS Client Driver supports the following:

- Solicited data access through Manufacturing Message Specification (MMS) Read and Write requests.
- Unsolicited data through MMS Information Reports.
- Special handling of Reads and Writes as they relate to IEC 61850 control operations.

## Setup

### Maximum Number of Channels and Devices

The maximum number of supported channels is 256. The maximum number of supported devices is 256.

### Channel Properties — General

This server supports the use of simultaneous multiple communications drivers. Each protocol or driver used in a server project is called a channel. A server project may consist of many channels with the same communications driver or with unique communications drivers. A channel acts as the basic building block of an OPC link. This group is used to specify general channel properties, such as the identification attributes and operating mode.

Property Groups	[-] <b>Identification</b>	
General	Name	
Write Optimizations	Description	
Advanced	Driver	
	[-] <b>Diagnostics</b>	
	Diagnostics Capture	Disable

### Identification

**Name:** User-defined identity of this channel. In each server project, each channel name must be unique. Although names can be up to 256 characters, some client applications have a limited display window when browsing the OPC server's tag space. The channel name is part of the OPC browser information.

• For information on reserved characters, refer to "How To... Properly Name a Channel, Device, Tag, and Tag Group" in the server help.

**Description:** User-defined information about this channel.

• Many of these properties, including Description, have an associated system tag.

**Driver:** Selected protocol / driver for this channel. This property specifies the device driver that was selected during channel creation. It is a disabled setting in the channel properties.

• **Note:** With the server's online full-time operation, these properties can be changed at any time. This includes changing the channel name to prevent clients from registering data with the server. If a client has already acquired an item from the server before the channel name is changed, the items are unaffected. If, after the channel name has been changed, the client application releases the item and attempts to re-acquire using the old channel name, the item is not accepted. With this in mind, changes to the properties should not be made once a large client application has been developed. Utilize the User Manager to prevent operators from changing properties and restrict access rights to server features.

### Diagnostics

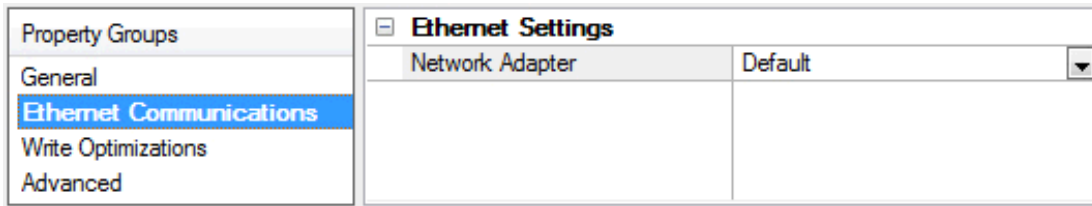
**Diagnostics Capture:** When enabled, this option makes the channel's diagnostic information available to OPC applications. Because the server's diagnostic features require a minimal amount of overhead processing, it is recommended that they be utilized when needed and disabled when not. The default is disabled.

• **Note:** This property is disabled if the driver does not support diagnostics.

• For more information, refer to "Communication Diagnostics" in the server help.

## Channel Properties — Ethernet Communications

Ethernet Communication can be used to communicate with devices.

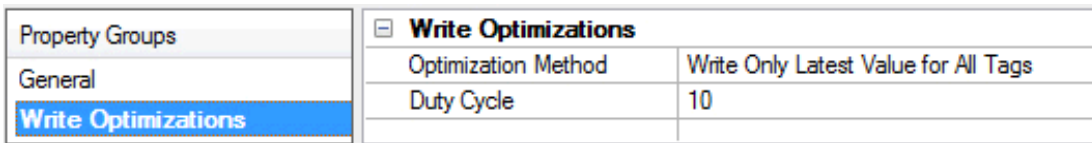


### Ethernet Settings

**Network Adapter:** Specify the network adapter to bind. When Default is selected, the operating system selects the default adapter.

## Channel Properties — Write Optimizations

As with any OPC server, writing data to the device may be the application's most important aspect. The server intends to ensure that the data written from the client application gets to the device on time. Given this goal, the server provides optimization properties that can be used to meet specific needs or improve application responsiveness.



### Write Optimizations

**Optimization Method:** controls how write data is passed to the underlying communications driver. The options are:

- **Write All Values for All Tags:** This option forces the server to attempt to write every value to the controller. In this mode, the server continues to gather write requests and add them to the server's internal write queue. The server processes the write queue and attempts to empty it by writing data to the device as quickly as possible. This mode ensures that everything written from the client applications is sent to the target device. This mode should be selected if the write operation order or the write item's content must uniquely be seen at the target device.
- **Write Only Latest Value for Non-Boolean Tags:** Many consecutive writes to the same value can accumulate in the write queue due to the time required to actually send the data to the device. If the server updates a write value that has already been placed in the write queue, far fewer writes are needed to reach the same final output value. In this way, no extra writes accumulate in the server's queue. When the user stops moving the slide switch, the value in the device is at the correct value at virtually the same time. As the mode states, any value that is not a Boolean value is updated in the server's internal write queue and sent to the device at the next possible opportunity. This can greatly improve the application performance.

● **Note:** This option does not attempt to optimize writes to Boolean values. It allows users to optimize the operation of HMI data without causing problems with Boolean operations, such as a momentary push button.

- **Write Only Latest Value for All Tags:** This option takes the theory behind the second optimization mode and applies it to all tags. It is especially useful if the application only needs to send the latest value to the device. This mode optimizes all writes by updating the tags currently in the write queue before they are sent. This is the default mode.

**Duty Cycle:** is used to control the ratio of write to read operations. The ratio is always based on one read for every one to ten writes. The duty cycle is set to ten by default, meaning that ten writes occur for each read operation. Although the application is performing a large number of continuous writes, it must be ensured that read data is still given time to process. A setting of one results in one read operation for every write operation. If there are no write operations to perform, reads are processed continuously. This allows optimization for applications with continuous writes versus a more balanced back and forth data flow.

● **Note:** It is recommended that the application be characterized for compatibility with the write optimization enhancements before being used in a production environment.

## Channel Properties — Advanced

This group is used to specify advanced channel properties. Not all drivers support all properties; so the Advanced group does not appear for those devices.

Property Groups	[-] <b>Non-Normalized Float Handling</b>	
General	Floating-Point Values	Replace with Zero
Write Optimizations	[-] <b>Inter-Device Delay</b>	
<b>Advanced</b>	Inter-Device Delay (ms)	0

**Non-Normalized Float Handling:** A non-normalized value is defined as Infinity, Not-a-Number (NaN), or as a Denormalized Number. The default is Replace with Zero. Drivers that have native float handling may default to Unmodified. Non-normalized float handling allows users to specify how a driver handles non-normalized IEEE-754 floating point data. Descriptions of the options are as follows:

- **Replace with Zero:** This option allows a driver to replace non-normalized IEEE-754 floating point values with zero before being transferred to clients.
- **Unmodified:** This option allows a driver to transfer IEEE-754 denormalized, normalized, non-number, and infinity values to clients without any conversion or changes.

● **Note:** This property is disabled if the driver does not support floating point values or if it only supports the option that is displayed. According to the channel's float normalization setting, only real-time driver tags (such as values and arrays) are subject to float normalization. For example, EFM data is not affected by this setting.

● *For more information on the floating point values, refer to "How To ... Work with Non-Normalized Floating Point Values" in the server help.*

**Inter-Device Delay:** Specify the amount of time the communications channel waits to send new requests to the next device after data is received from the current device on the same channel. Zero (0) disables the delay.

● **Note:** This property is not available for all drivers, models, and dependent settings.

## Device Properties — General

A device represents a single target on a communications channel. If the driver supports multiple controllers, users must enter a device ID for each controller.

Property Groups	Identification	
General	Name	
Scan Mode	Description	
	Channel Assignment	
	Driver	
	Model	
	ID Format	Decimal
	ID	2
	Operating Mode	
	Data Collection	Enable
	Simulated	No

### Identification

**Name:** This property specifies the name of the device. It is a logical user-defined name that can be up to 256 characters long, and may be used on multiple channels.

**Note:** Although descriptive names are generally a good idea, some OPC client applications may have a limited display window when browsing the OPC server's tag space. The device name and channel name become part of the browse tree information as well. Within an OPC client, the combination of channel name and device name would appear as "ChannelName.DeviceName".

**For more information, refer to "How To... Properly Name a Channel, Device, Tag, and Tag Group" in server help.**

**Description:** User-defined information about this device.

Many of these properties, including Description, have an associated system tag.

**Channel Assignment:** User-defined name of the channel to which this device currently belongs.

**Driver:** Selected protocol driver for this device. This property specifies the driver selected during channel creation. It is disabled in the channel properties.

**Model:** This property specifies the specific type of device that is associated with this ID. The contents of the drop-down menu depends on the type of communications driver being used. Models that are not supported by a driver are disabled. If the communications driver supports multiple device models, the model selection can only be changed when there are no client applications connected to the device.

**Note:** If the communication driver supports multiple models, users should try to match the model selection to the physical device. If the device is not represented in the drop-down menu, select a model that conforms closest to the target device. Some drivers support a model selection called "Open," which allows users to communicate without knowing the specific details of the target device. For more information, refer to the driver help documentation.

**ID:** This property specifies the device's station / node / identity / address. The type of ID entered depends on the communications driver being used. For many drivers, the ID is a numeric value. Drivers that support a Numeric ID provide users with the option to enter a numeric value whose format can be changed to suit the needs of the application or the characteristics of the selected communications driver. The ID format can be Decimal, Octal, and Hexadecimal. If the driver is Ethernet-based or supports an unconventional station or



node name, the device's TCP/IP address may be used as the device ID. TCP/IP addresses consist of four values that are separated by periods, with each value in the range of 0 to 255. Some device IDs are string based. There may be additional properties to configure within the ID field, depending on the driver.

## Operating Mode

**Data Collection:** This property controls the device's active state. Although device communications are enabled by default, this property can be used to disable a physical device. Communications are not attempted when a device is disabled. From a client standpoint, the data is marked as invalid and write operations are not accepted. This property can be changed at any time through this property or the device system tags.

**Simulated:** This option places the device into Simulation Mode. In this mode, the driver does not attempt to communicate with the physical device, but the server continues to return valid OPC data. Simulated stops physical communications with the device, but allows OPC data to be returned to the OPC client as valid data. While in Simulation Mode, the server treats all device data as reflective: whatever is written to the simulated device is read back and each OPC item is treated individually. The item's memory map is based on the group Update Rate. The data is not saved if the server removes the item (such as when the server is reinitialized). The default is No.

### Notes:

1. This System tag (\_Simulated) is read only and cannot be written to for runtime protection. The System tag allows this property to be monitored from the client.
2. In Simulation mode, the item's memory map is based on client update rate(s) (Group Update Rate for OPC clients or Scan Rate for native and DDE interfaces). This means that two clients that reference the same item with different update rates return different data.

Simulation Mode is for test and simulation purposes only. It should never be used in a production environment.

## Device Properties — Scan Mode

The Scan Mode specifies the subscribed-client requested scan rate for tags that require device communications. Synchronous and asynchronous device reads and writes are processed as soon as possible; unaffected by the Scan Mode properties.

Property Groups	<input checked="" type="checkbox"/> <b>Scan Mode</b>	
General	Scan Mode	Respect Client-Specified Scan Rate ▼
Scan Mode	Initial Updates from Cache	Disable

**Scan Mode:** specifies how tags in the device are scanned for updates sent to subscribed clients.

Descriptions of the options are:

- **Respect Client-Specified Scan Rate:** This mode uses the scan rate requested by the client.
- **Request Data No Faster than Scan Rate:** This mode specifies the maximum scan rate to be used. The valid range is 10 to 99999990 milliseconds. The default is 1000 milliseconds.
  - **Note:** When the server has an active client and items for the device and the scan rate value is increased, the changes take effect immediately. When the scan rate value is decreased, the changes do not take effect until all client applications have been disconnected.
- **Request All Data at Scan Rate:** This mode forces tags to be scanned at the specified rate for subscribed clients. The valid range is 10 to 99999990 milliseconds. The default is 1000 milliseconds.

- **Do Not Scan, Demand Poll Only:** This mode does not periodically poll tags that belong to the device nor perform a read to get an item's initial value once it becomes active. It is the client's responsibility to poll for updates, either by writing to the `_DemandPoll` tag or by issuing explicit device reads for individual items. *For more information, refer to "Device Demand Poll" in server help.*
- **Respect Tag-Specified Scan Rate:** This mode forces static tags to be scanned at the rate specified in their static configuration tag properties. Dynamic tags are scanned at the client-specified scan rate.

**Initial Updates from Cache:** When enabled, this option allows the server to provide the first updates for newly activated tag references from stored (cached) data. Cache updates can only be provided when the new item reference shares the same address, scan rate, data type, client access, and scaling properties. A device read is used for the initial update for the first client reference only. The default is disabled; any time a client activates a tag reference the server attempts to read the initial value from the device.

## Device Properties — Timing

The device Timing properties allow the driver's response to error conditions to be tailored to fit the application's needs. In many cases, the environment requires changes to these properties for optimum performance. Factors such as electrically generated noise, modem delays, and poor physical connections can influence how many errors or timeouts a communications driver encounters. Timing properties are specific to each configured device.

Property Groups	[-] <b>Communication Timeouts</b>	
General	Connect Timeout (s)	3
Scan Mode	Request Timeout (ms)	5000
<b>Timing</b>	Retry Attempts	3
Auto-Demotion	[-] <b>Timing</b>	
	Inter-Request Delay (ms)	0

### Communications Timeouts

**Connect Timeout:** This property (which is used primarily by Ethernet based drivers) controls the amount of time required to establish a socket connection to a remote device. The device's connection time often takes longer than normal communications requests to that same device. The valid range is 1 to 30 seconds. The default is typically 3 seconds, but can vary depending on the driver's specific nature. If this setting is not supported by the driver, it is disabled.

● **Note:** Due to the nature of UDP connections, the connection timeout setting is not applicable when communicating via UDP.

**Request Timeout:** This property specifies an interval used by all drivers to determine how long the driver waits for a response from the target device to complete. The valid range is 50 to 9,999,999 milliseconds (167.6667 minutes). The default is usually 1000 milliseconds, but can vary depending on the driver. The default timeout for most serial drivers is based on a baud rate of 9600 baud or better. When using a driver at lower baud rates, increase the timeout to compensate for the increased time required to acquire data.

**Attempts Before Timeout:** This property specifies how many times the driver issues a communications request before considering the request to have failed and the device to be in error. The valid range is 1 to 10. The default is typically 3, but can vary depending on the driver's specific nature. The number of attempts configured for an application depends largely on the communications environment. This property applies to both connection attempts and request attempts.

## Timing

**Inter-Request Delay:** This property specifies how long the driver waits before sending the next request to the target device. It overrides the normal polling frequency of tags associated with the device, as well as one-time reads and writes. This delay can be useful when dealing with devices with slow turnaround times and in cases where network load is a concern. Configuring a delay for a device affects communications with all other devices on the channel. It is recommended that users separate any device that requires an inter-request delay to a separate channel if possible. Other communications properties (such as communication serialization) can extend this delay. The valid range is 0 to 300,000 milliseconds; however, some drivers may limit the maximum value due to a function of their particular design. The default is 0, which indicates no delay between requests with the target device.

● **Note:** Not all drivers support Inter-Request Delay. This setting does not appear if it is not available.

## Device Properties — Auto-Demotion

The Auto-Demotion properties can temporarily place a device off-scan in the event that a device is not responding. By placing a non-responsive device offline for a specific time period, the driver can continue to optimize its communications with other devices on the same channel. After the time period has been reached, the driver re-attempts to communicate with the non-responsive device. If the device is responsive, the device is placed on-scan; otherwise, it restarts its off-scan time period.

Property Groups	Auto-Demotion	
General	Demote on Failure	Enable
Scan Mode	Timeouts to Demote	3
Timing	Demotion Period (ms)	10000
Auto-Demotion	Discard Requests when Demoted	Disable

**Demote on Failure:** When enabled, the device is automatically taken off-scan until it is responding again.

● **Tip:** Determine when a device is off-scan by monitoring its demoted state using the `_AutoDemoted` system tag.

**Timeouts to Demote:** Specify how many successive cycles of request timeouts and retries occur before the device is placed off-scan. The valid range is 1 to 30 successive failures. The default is 3.

**Demotion Period:** Indicate how long the device should be placed off-scan when the timeouts value is reached. During this period, no read requests are sent to the device and all data associated with the read requests are set to bad quality. When this period expires, the driver places the device on-scan and allows for another attempt at communications. The valid range is 100 to 3600000 milliseconds. The default is 10000 milliseconds.

**Discard Requests when Demoted:** Select whether or not write requests should be attempted during the off-scan period. Disable to always send write requests regardless of the demotion period. Enable to discard writes; the server automatically fails any write request received from a client and does not post a message to the Event Log.

## Device Properties — Tag Generation

The automatic tag database generation features make setting up an application a plug-and-play operation. Select communications drivers can be configured to automatically build a list of tags that correspond to

device-specific data. These automatically generated tags (which depend on the nature of the supporting driver) can be browsed from the clients.

If the target device supports its own local tag database, the driver reads the device's tag information and uses the data to generate tags within the server. If the device does not natively support named tags, the driver creates a list of tags based on driver-specific information. An example of these two conditions is as follows:

1. If a data acquisition system supports its own local tag database, the communications driver uses the tag names found in the device to build the server's tags.
2. If an Ethernet I/O system supports detection of its own available I/O module types, the communications driver automatically generates tags in the server that are based on the types of I/O modules plugged into the Ethernet I/O rack.

● **Note:** Automatic tag database generation's mode of operation is completely configurable. For more information, refer to the property descriptions below.

Property Groups	☐ <b>Tag Generation</b>	
General	On Property Change	Yes
Scan Mode	On Device Startup	Do Not Generate on Startup
Timing	On Duplicate Tag	Delete on Create
Auto-Demotion	Parent Group	
<b>Tag Generation</b>	Allow Automatically Generated Subgroups	Enable
Tag Import	Create	Create tags
Redundancy		

**On Property Change:** If the device supports automatic tag generation when certain properties change, the **On Property Change** option is shown. It is set to **Yes** by default, but it can be set to **No** to control over when tag generation is performed. In this case, the **Create tags** action must be manually invoked to perform tag generation.

**On Device Startup:** This property specifies when OPC tags are automatically generated. Descriptions of the options are as follows:

- **Do Not Generate on Startup:** This option prevents the driver from adding any OPC tags to the tag space of the server. This is the default setting.
- **Always Generate on Startup:** This option causes the driver to evaluate the device for tag information. It also adds tags to the tag space of the server every time the server is launched.
- **Generate on First Startup:** This option causes the driver to evaluate the target device for tag information the first time the project is run. It also adds any OPC tags to the server tag space as needed.

● **Note:** When the option to automatically generate OPC tags is selected, any tags that are added to the server's tag space must be saved with the project. Users can configure the project to automatically save from the **Tools | Options** menu.

**On Duplicate Tag:** When automatic tag database generation is enabled, the server needs to know what to do with the tags that it may have previously added or with tags that have been added or modified after the communications driver since their original creation. This setting controls how the server handles OPC tags that were automatically generated and currently exist in the project. It also prevents automatically generated tags from accumulating in the server.

For example, if a user changes the I/O modules in the rack with the server configured to **Always Generate on Startup**, new tags would be added to the server every time the communications driver detected a new I/O module. If the old tags were not removed, many unused tags could accumulate in the server's tag space. The options are:

- **Delete on Create:** This option deletes any tags that were previously added to the tag space before any new tags are added. This is the default setting.
- **Overwrite as Necessary:** This option instructs the server to only remove the tags that the communications driver is replacing with new tags. Any tags that are not being overwritten remain in the server's tag space.
- **Do not Overwrite:** This option prevents the server from removing any tags that were previously generated or already existed in the server. The communications driver can only add tags that are completely new.
- **Do not Overwrite, Log Error:** This option has the same effect as the prior option, and also posts an error message to the server's Event Log when a tag overwrite would have occurred.

● **Note:** Removing OPC tags affects tags that have been automatically generated by the communications driver as well as any tags that have been added using names that match generated tags. Users should avoid adding tags to the server using names that may match tags that are automatically generated by the driver.

**Parent Group:** This property keeps automatically generated tags from mixing with tags that have been entered manually by specifying a group to be used for automatically generated tags. The name of the group can be up to 256 characters. This parent group provides a root branch to which all automatically generated tags are added.

**Allow Automatically Generated Subgroups:** This property controls whether the server automatically creates subgroups for the automatically generated tags. This is the default setting. If disabled, the server generates the device's tags in a flat list without any grouping. In the server project, the resulting tags are named with the address value. For example, the tag names are not retained during the generation process.

● **Note:** If, as the server is generating tags, a tag is assigned the same name as an existing tag, the system automatically increments to the next highest number so that the tag name is not duplicated. For example, if the generation process creates a tag named "AI22" that already exists, it creates the tag as "AI23" instead.

**Create:** Initiates the creation of automatically generated OPC tags. If the device's configuration has been modified, **Create tags** forces the driver to reevaluate the device for possible tag changes. Its ability to be accessed from the System tags allows a client application to initiate tag database creation.

● **Note:** **Create tags** is disabled if the Configuration edits a project offline.

---

## Device Properties — Automatic Configuration

---

Property Groups	<input type="checkbox"/> <b>Automatic Configuration</b>	
General	Automatic Configuration Source	Device
Scan Mode	<input type="checkbox"/> <b>Configuration from SCL File</b>	
Timing	File	
Auto-Demotion	SubNetwork	
Tag Generation	IED	
<b>Automatic Configuration</b>	AccessPoint	
Connection	Parameter Import	Select from File...

**Automatic Configuration Source:** This field specifies the source for automatic device configuration. Options include Device and SCL File. When Device is selected, tags will be created using the online device self-description services. When SCL File is selected, tags will be created from the configured SCL file, and the Connection properties will be imported. The default setting is Device.

● **Note:** For more information on the Connection properties, refer to [Connection](#).

**File:** This property is set automatically after the configuration file is specified. To locate the file, click the browse (...) button. The selected file can have an .icd, .cid, or .scl extension. This property is only available when the Automatic Configuration Source is set to **SCL File**.

**SubNetwork:** This property is set automatically after the SubNetwork is confirmed in SCL Parameter Import. It is only available when the Automatic Configuration Source is set to **SCL File**. For more information, refer to [Parameter Import](#).

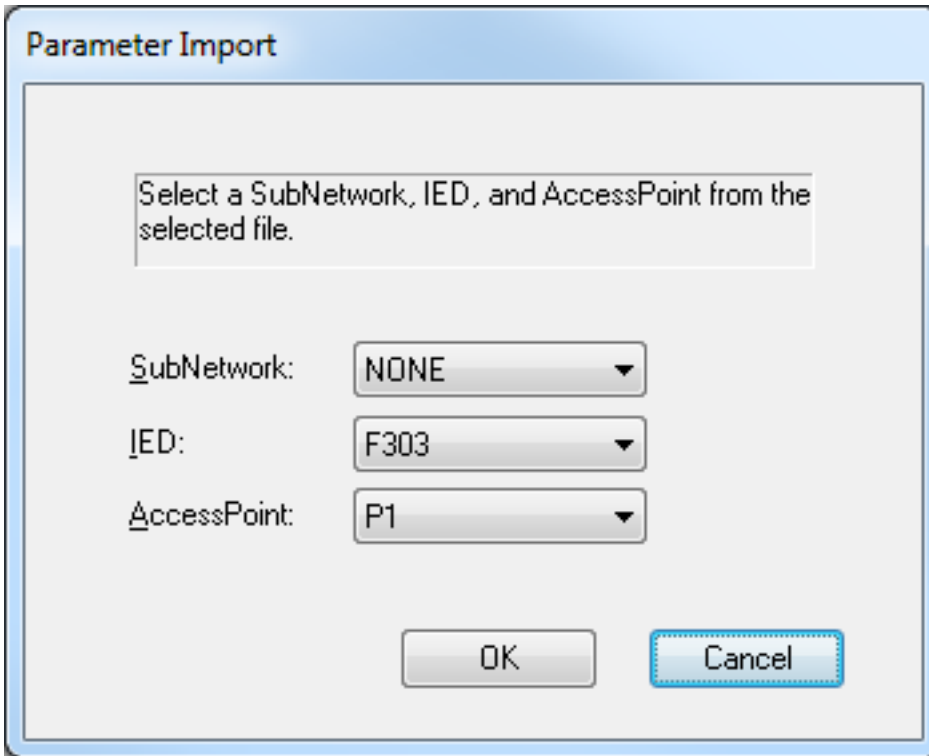
**IED:** This property will be set automatically after the Intelligent Electronic Device (IED) is confirmed in SCL Parameter Import. It is only available when the Automatic Configuration Source is set to SCL File. For more information, refer to [Parameter Import](#).

**AccessPoint:** This property will be set automatically after the AccessPoint is confirmed in Parameter Import. It is only available when the Automatic Configuration Source is set to SCL File. For more information, refer to [Parameter Import](#).

**Parameter Import:** When clicked, this button launches the SCL Parameter Import dialog. It is only available when the Automatic Configuration Source is set to **SCL File**. For more information, refer to [Parameter Import](#).

## Parameter Import

This dialog displays the SubNetworks, IEDs, and AccessPoints available in the file specified in the Automatic Configuration property group. Once confirmed, the selections will be set in the Automatic Configuration property group.

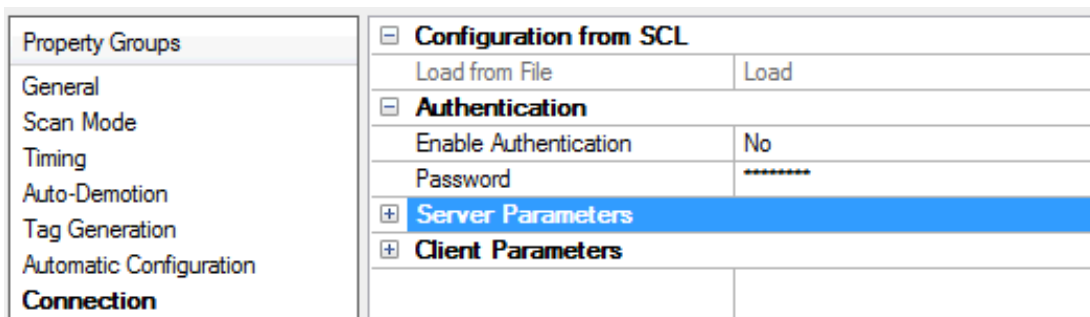


**SubNetwork:** This property specifies a SubNetwork within the specified file. If multiple SubNetworks exist in the file, the first one encountered is selected by default.

**IED:** This property specifies an IED within the file. If multiple IEDs exist in the file, the first one encountered is selected by default.

**AccessPoint:** This property specifies an AccessPoint within the file. If multiple AccessPoints exist in the file, the first one encountered is selected by default.

## Device Properties — Connection



### Configuration from SCL

- **Load from File:** When enabled, the properties of the selected file are used to populate the properties listed beneath the Authentication and Server Parameters sections. This option is only available if the specified Automatic Configuration Source is **SCL File** (and all properties have been configured). For more information, refer to [Automatic Configuration](#).

- **Enable Authentication:** This property specifies whether Association Control Service Element (ACSE) authentication is enabled.
- **Password:** This property specifies the password for ACSE authentication.

**Note:** For information on the properties listed beneath the Server Parameters and Client Parameters sections, refer to the subtopics below.

## Server Parameters

This section is used to configure the server-specific connection properties.

Property Groups	Server Parameters	
General	Use Optional Server Paramet...	Yes
Scan Mode	Server AE Invoke ID	10
Timing	Server AE Qualifier	12
Auto-Demotion	Server AP Invoke ID	10
Tag Generation	Server Application ID	1,1,1,999,1
Automatic Configuration	Server Transport Selector	00 01
<b>Connection</b>	Server Session Selector	00 01
Communication	Server Presentation Selector	00 00 00 01

Descriptions of the properties are as follows:

- **Use Optional Server Parameters:** When enabled, the Server Parameters will be available for editing and will be included when initiating a connection with a device. The default setting is Yes.
- **Server AE Invoke ID:** This property specifies the ACSE AE Invoke ID. The valid range is 0 to 65535. The default setting is 10.
- **Server AE Qualifier:** This property specifies the ACSE AE Qualifier. The valid range is 0 to 65535. The default setting is 12.
- **Server AP Invoke ID:** This property specifies the ACSE AP Invoke ID. The valid range is 0 to 65535. The default setting is 10.
- **Server Application ID:** This property specifies the ACSE Application ID. It must be 10 or less integers, and delimited by commas. The default setting is "1,1,1,999,1".
- **Server Transport Selector:** This property configures the server OSI-TSEL and is used in establishing a remote server connection. It specifies a byte array, which is expressed as pairs of hexadecimal digits separated by zero or more spaces. The maximum size is 50 bytes (or 100 hex chars). The default setting is "00 01".
- **Server Session Selector:** This property configures the server OSI-SSEL and is used in establishing a remote server connection. It specifies a byte array, which is expressed as pairs of hexadecimal digits separated by zero or more spaces. The maximum size is 50 bytes (or 100 hex chars). The default setting is "00 01".
- **Server Presentation Selector:** This property configures the server OSI-PSEL and is used in establishing a remote server connection. It specifies a byte array, which is expressed as pairs of hexadecimal digits separated by zero or more spaces. The maximum size is 50 bytes (or 100 hex chars). The default setting is "00 00 00 01".

## Client Parameters

This section is used to configure the client-specific connection properties.



Property Groups	<input checked="" type="checkbox"/> <b>Configuration from SCL</b>	
General	<input checked="" type="checkbox"/> <b>Authentication</b>	
Scan Mode	<input checked="" type="checkbox"/> <b>Server Parameters</b>	
Timing	<input type="checkbox"/> <b>Client Parameters</b>	
Auto-Demotion	Use Optional Client Parameters	Yes
Tag Generation	Client AE Invoke ID	10
Automatic Configuration	Client AE Qualifier	12
<b>Connection</b>	Client AP Invoke ID	10
Communication	Client Application ID	1,1,1,999,1
Control	Client Transport Selector	00 01
Tag Database Settings	Client Session Selector	00 01
Redundancy	Client Presentation Selector	00 00 00 01

Descriptions of the properties are as follows:

- **Use Optional Client Parameters:** When enabled, the Client Parameters will be available for editing and will be included when initiating a connection with a device. The default setting is Yes.
- **Client AE Invoke ID:** This property specifies the ACSE AE Invoke ID. The valid range is 0 to 65535. The default setting is 10.
- **Client AE Qualifier:** This property specifies the ACSE AE Qualifier. The valid range is 0 to 65535. The default setting is 12.
- **Client AP Invoke ID:** This property specifies the ACSE AP Invoke ID. The valid range is 0 to 65535. The default setting is 10.
- **Client Application ID:** This property specifies the ACSE Application ID. It must be 10 or less integers, and delimited by commas. The default setting is "1,1,1,999,1".
- **Client Transport Selector:** This property configures the server OSI-TSEL and is used in establishing a remote server connection. It specifies a byte array, which is expressed as pairs of hexadecimal digits separated by zero or more spaces. The maximum size is 50 bytes (or 100 hex chars). The default setting is "00 01".
- **Client Session Selector:** This property configures the server OSI-SSEL and is used in establishing a remote server connection. It specifies a byte array, which is expressed as pairs of hexadecimal digits separated by zero or more spaces. The maximum size is 50 bytes (or 100 hex chars). The default setting is "00 01".
- **Client Presentation Selector:** This property configures the server OSI-PSEL and is used in establishing a remote server connection. It specifies a byte array, which is expressed as pairs of hexadecimal digits separated by zero or more spaces. The maximum size is 50 bytes (or 100 hex chars). The default setting is "00 00 00 01".

## Device Properties — Communication

Property Groups	<input type="checkbox"/> <b>Reporting</b>	
General	Buffer Size	100
Automatic Configuration	Playback Rate (ms)	2000
Connection	Integrity Poll Rate (ms)	5000
<b>Communication</b>	<input type="checkbox"/> <b>Polling</b>	
Control	Polling Level	Logical Node

**Buffer Size:** This property specifies the data buffer size. If the length of the data buffer exceeds the maximum, the oldest value on the buffer is discarded. The valid range is 1 to 10000. The default setting is 100.

**Playback Rate (ms):** This property specifies the amount of time before a value is removed from the data buffer after it is assigned to a tag. The valid range is 50 to 999999999 milliseconds. The default setting is 2000 milliseconds.

● **Notes:**

1. If multiple clients reference the same buffered data, the playback rate should be greater than the longest scan rate. If only one client references the buffered data, a playback rate of 0 is sufficient.
2. Enabling the OPC DA setting **Return initial updates for items in a single callback** may result in loss of buffered data when using drivers that support Event Playback for unsolicited device protocols. The compliance setting should be disabled if loss of buffered data is a concern.

**Integrity Poll Rate (ms):** An integrity poll is used to verify connectivity to the IED at a minimum of every x milliseconds. It is recommended when the IED is only sending reports as a way to know that connectivity is lost. It can also be used to verify connectivity faster than the poll rate when polling the IED slowly (such as in minutes). This property specifies the amount of time that can elapse between either receiving a report or receiving a solicited response before the driver must check the integrity of its connection with the IED. The valid range is 0 to 999999999 milliseconds. The default setting is 5000 milliseconds. To disable integrity polling, specify a value of 0 milliseconds.

● **Note:** When the integrity poll rate has elapsed without any communication, the channel sends a read for an RCB attribute to the IED. If the read fails, the connection to the IED is considered lost. At that point, the device is placed into an error state and all tags are set to bad quality. If the read succeeds, no action is taken.

**Polling Level:** This property specifies the level at which data is grouped and polled. Options include Logical Node, Functional Constraint, and Data Object. The default setting is Logical Node.

## Device Properties — Control

Property Groups	Default Control Values	
General	orCat	bay-control
Scan Mode	orIdent	
Timing	ctlNum	0
Auto-Demotion	Test	Disable
Tag Generation	Check	00
Automatic Configuration		
Connection		
Communication		
<b>Control</b>		
Tag Database Settings		

**orCat:** This property specifies the value of orCat when making a structured write to a control object. The default setting is bay-control. Options include the following:

- not-supported
- bay-control

- station-control
- remote-control
- automatic-bay
- automatic-station
- automatic-remote
- maintenance
- process

**orIdent:** This property specifies the value of the \_orIdent Tag. The value must be a hex byte array (such as "01 7A F0"). It is blank by default.

**ctlNum:** This property specifies the value of the \_ctlNum Tag. The value must be an 8 bit unsigned integer. The default is 0.

**Test:** This property specifies the Boolean value assigned to the \_Test Tag. When enabled, the value is 1. When disabled, the value is 0. The default is disabled.

**Check:** This property specifies the value of the \_Check Tag. The value must be a 2 bit string. The default is 00.

## Device Properties — Tag Database Settings

Property Groups	<input type="checkbox"/> <b>Tag Generation Settings</b>	
General	Display Descriptions	Yes
Scan Mode	Generate Reported Data Sets	Yes
Timing	<input type="checkbox"/> <b>Tag Generation Functional Constraints</b>	
Auto-Demotion	Select All	<a href="#">Select All</a>
Tag Generation	Deselect All	<a href="#">Deselect All</a>
Automatic Configuration	ST	Yes
Connection	MX	Yes
Communication	CO	Yes
Control	SP	Yes
<b>Tag Database Settings</b>	SV	Yes
Redundancy	CF	Yes
	DC	Yes
	SG	Yes
	SE	Yes
	EX	Yes
	BR	Yes
	RP	Yes
	LG	Yes
	GO	Yes
	GS	Yes
	MS	Yes
	US	Yes

### Tag Generation Settings

**Display Descriptions:** If enabled, this option will apply the SCL file data attribute descriptions to the tag descriptions. It is only available when tags are created from SCL. The default is Yes.

**Generate Reported Data Sets:** If enabled, this option generates tags for data sets referenced by report control blocks. The default is Yes.

## Tag Generation Functional Constraints

**Select All:** When clicked, this option will select all Functional Constraints.

**Deselect All:** When clicked, this option will deselect all Functional Constraints.

**ST:** If enabled, this option generates tags with the ST Functional Constraint. The default is Yes.

**MX:** If enabled, this option generates tags with the MX Functional Constraint. The default is Yes.

**CO:** If enabled, this option generates tags with the CO Functional Constraint. The default is Yes.

**SP:** If enabled, this option generates tags with the SP Functional Constraint. The default is Yes.

**SV:** If enabled, this option generates tags with the SV Functional Constraint. The default is Yes.

**CF:** If enabled, this option generates tags with the CF Functional Constraint. The default is Yes.

**DC:** If enabled, this option generates tags with the DC Functional Constraint. The default is Yes.

**SG:** If enabled, this option generates tags with the SG Functional Constraint. The default is Yes.

**SE:** If enabled, this option generates tags with the SE Functional Constraint. The default is Yes.

**EX:** If enabled, this option generates tags with the EX Functional Constraint. The default is Yes.

**BR:** If enabled, this option generates tags with the BR Functional Constraint. The default is Yes.

**RP:** If enabled, this option generates tags with the RP Functional Constraint. The default is Yes.

**LG:** If enabled, this option generates tags with the LG Functional Constraint. The default is Yes.

**GO:** If enabled, this option generates tags with the GO Functional Constraint. The default is Yes.

**GS:** If enabled, this option generates tags with the GS Functional Constraint. The default is Yes.

**MS:** If enabled, this option generates tags with the MS Functional Constraint. The default is Yes.

**US:** If enabled, this option generates tags with the US Functional Constraint. The default is Yes.

 For more information, refer to [Supported Functional Constraints](#).

## Device Properties — Redundancy

---

Property Groups	<input type="checkbox"/> <b>Redundancy</b>	
General	Secondary Path	...
Scan Mode	Operating Mode	Switch On Failure
Timing	Monitor Item	
<b>Redundancy</b>	Monitor Interval (s)	300
	Return to Primary ASAP	Yes

Redundancy is available with the Media-Level Redundancy Plug-In.

● *Consult the website, a sales representative, or the user manual for more information.*

## Data Types Description

OPC Data Type	MMS Data Type	Description
Bool	Boolean	Single bit
Char	Integer (8 bit)	Signed 8 bit value
Byte	Unsigned Integer (8 bit)	Unsigned 8 bit value
Short	Integer (16 bit)	Signed 16 bit value
Word	Unsigned Integer (16 bit)	Unsigned 16 bit value
Long	Integer (32 bit)	Signed 32 bit value
DWord	Unsigned Integer (32 bit)	Unsigned 32 bit value
Float	Floating-point (32 bit)	32 bit floating point value
Byte	Bit string (length <= 8)	*
Word	Bit string (8 < length <= 16)	*
DWord	Bit string (16 < length <= 32)	*
String	Bit string (length > 32)	*
String	Octet-string	An array of bytes (octets).
String	Visible-string	ANSI characters up to 255 characters in length.
String	MMS String	UTF-8 encoded string up to 255 characters in length.
Date	Binary Time	6 byte structure containing days since Jan 1, 1984 and milliseconds since midnight. It uses the format "MM/DD/YYYY_HH:MM:SS.mmm".
Date	UTC Time	8 byte structure containing seconds of the century, a fraction of a second, and a time quality. It uses the format "MM/DD/YYYY_HH:MM:SS.mmm".

\*MMS supports bit strings up to 256 bits in length. Bit strings less than 32 bits long can be assigned to unsigned integers, whereas larger bit strings can only be applied to strings. They will be assigned to unsigned integers with the first bit in the bit string corresponding to the integer's lowest order bit.

## Address Descriptions

---

### Polled Tag Addressing

The IEC 61850 MMS Client Driver syntax for Polled Tag Addressing is

*LDName/LNName\$FC\$DataName\$DataAttrName[\$DataAttrComponent[\$ ...]],* where:

- *LDName* indicates the Logical Device Name. It is limited to 32 characters.
- *LNName\$FC\$DataName\$DataAttrName[\$DataAttrComponent[\$ ...]]* indicates the Object Name. It is limited to 64 characters.
- *[\$ ]* indicates an option.
- *[\$ ...]* indicates additional names of recursively nested definitions.
- *FC* indicates the Functional Constraint (FC) that describes the services that can be performed on the data.
- *\$* is a separator.

Examples of the address syntax include "Rly1/LLN0\$ST\$Mod\$stVal" and "Rly2/LLN0\$BR\$brcb1\$RptEna".

**Note:** A primitive Data Attribute's object reference syntax is *LDName/LNName.DataName.DataAttrName [.DataAttrComponent[ ...]]*. The '.' separator used in IEC 61850-7-2 is replaced by '\$' in MMS addressing.

### Reported Tag Addressing

The IEC 61850 MMS Client Driver syntax for Reported Tag Addressing is

*LDName/LNName\$FC\$RCBName:LDName/LNName\$FC\$DataName\$DataAttrName[\$DataAttrComponent[\$ ...]],* where:

- *LDName/LNName\$FC\$RCBName* indicates the Report Control Block (RCB) name. It is limited to 64 characters.
- *:* is a separator between the RCB name and the name of the reported object.
- *LNName\$FC\$DataName\$DataAttrName[\$DataAttrComponent[\$ ...]]* indicates the Object Name. It is limited to 64 characters.

An example of the address syntax is "Rly1/LLN0\$BR\$brcb1:Rly1/LLN0\$ST\$Mod\$stVal". Valid functional constraints for the RCB name include BR (which indicates a buffered report control block) and RP (which indicates an unbuffered report control block). The Object Name must indicate a primitive attribute that is included in a data set member referenced by the RCB.

### Automatic Subscription

For reported tags like "Rly1/LLN0\$BR\$brcb1:Rly1/LLN0\$ST\$Mod\$stVal," the driver writes a value of '1' to "Rly1/LLN0\$BR\$brcb1\$RptEna" on each scan of the tag until the Report Control Block is enabled. Attempts to enable the RCB fail if it is already enabled by another client. If all tags referencing the RCB "Rly1/LLN0\$BR\$brcb1" are removed, the driver writes a value of '0' to "RptEna" to unsubscribe from receiving reports.

### Initial Update

Once a Report Control Block has been successfully enabled, the driver must request an initial value for the data monitored by the RCB. How the driver accomplishes this depends on the services that are supported by the RCB. If the driver supports general interrogation, it writes a value of '1' to the RCB's GI attribute to request a general interrogation report. Support is indicated by the general interrogation bit of the RCB's TrgOps attribute.

- If an unbuffered RCB does not support general interrogation, the driver reads the data set referenced by the RCB's DataSet attribute for an initial update.
- If a buffered RCB does not support general interrogation, the initial update logic depends on whether the RCB supports including EntryID values in reports. Support is indicated by the EntryID bit of the RCB's OptFlds attribute.
  - An EntryID bit of '0' indicates that it is not supported. In this case, the driver gives the device one scan period to send all buffered reports that it may contain. If values have not been reported for all members of the RCB's referenced data set after that time, the driver reads the referenced data set.
  - An EntryID bit of '1' indicates that it is supported. In this case, the driver gives the device as many scan periods as necessary for the EntryID of the last received report to equal the current value of the RCB's EntryID attribute. When these two values are equal (or if zero buffered reports are received within one scan period), the implication is that all buffered data has been sent to the driver. If values have not been reported for all members of the RCB's referenced data set at this time, the driver reads the referenced data set.

## Supported Functional Constraints

Each Data Attribute and object reference are associated with a Functional Constraint. It must be included in order to fully describe a Data Attribute. The IEC 61850 MMS Client Driver inserts an FC NamedVariable object between the logical node level and the data level so that Data Attributes can be described by their address only. For more information on the supported Functional Constraints, refer to the table below.

Constraint	Definition	Description	Access
ST	Status Information	This represents a status information whose value can be read, substituted, reported, and logged but not written.	Read Only
MX	Measurands (Analog Values)	This represents a measurand information whose value can be read, substituted, reported, and logged but not written.	Read Only
CO	Control	This represents a control information whose value may be operated and read.	Read/Write
SP	Set Point	This represents a set point information whose value may be controlled and read. Values that are controlled become effective immediately.	Read/Write
SV	Substitution	This represents a substitution information whose value can be written to substitute the value attribute and read.	Read/Write
CF	Configuration	This represents a configuration information whose value may be written and read. Values that are controlled become effective immediately or are deferred.	Read/Write
DC	Description	This represents a description information whose value can be written and read.	Read/Write
SG	Setting Group	Logical Devices that implement the SGCB class maintain multiple grouped values of all instances of DataAttributes with the Functional Constraint SG. Each group contains one value for each DataAttribute with Functional Constraint SG, which will be the current active value. Values of the DataAttribute	Read Only



<b>Constraint</b>	<b>Definition</b>	<b>Description</b>	<b>Access</b>
		with FC=SG cannot be written.	
SE	Setting Group Editable	This represents a DataAttribute that can be edited by SGCB services.	Read/Write
EX	Extended Definition	This represents an extension information that provides a reference to a name space. Extensions are used in conjunction with extended definitions of LNs, DATA, and DataAttributes. Values of the DataAttribute with FC=EX cannot be written.	Read Only
BR	Buffered Report	This represents a report control information of a BRCB that can be written and read.	Read/Write
RP	Unbuffered Report	This represents a report control information of a URCB that can be written and read.	Read/Write
LG	Logging	This represents a log control information of a LCB that can be written and read.	Read/Write
GO	Goose Control*	This represents a goose control information of a GoCB that can be written and read.	Read/Write
GS	Gsse Control*	This represents a goose control information of a GsCB that can be written and read.	Read/Write
MS	Multi-cast Sampled Value Control	This represents a sampled value control information of an MSVCB that can be written and read.	Read/Write
US	Uni-cast Sampled Value Control	This represents a sampled value control information of an instance of a UNICAST-SVC that can be written and read.	Read/Write

\* Reserved for control classes.

## Error Descriptions

---

The following error/warning messages may be generated. Click on the link for a description of the message.

[Duplicate address <tag address> produced from SCL file. Tag will not be created.](#)

[Error: Device <device name>, Address <MMS address>, Invalid bit string data <bit string>. Please enter a string of characters '0' and '1'.](#)

[Error: Device <device name>, Address <MMS address>, Invalid data type <data type> for data attribute.](#)

[Error: Device <device name>, Address <MMS address>, Invalid octet string data <octet string>. Please enter an even number of hexadecimal digits \(for example, "00 01 C3"\).](#)

[Error: Device <device name>, Cannot write to structured data attribute <MMS address>.](#)

[Error: Device <device name>, Invalid object <MMS address>.](#)

[Error: Device <device name>, Structured data attribute <MMS address> cannot be applied to tag.](#)

[Failed to open SCL file <file path>.](#)

[Invalid address <tag address> produced from SCL file. Tag will not be created](#)

[No ConnectedAP of SubNetwork <SubNetwork> with IED <IED> and AccessPoint <AccessPoint> found in file <file path>.](#)

[Selected AccessPoint <AccessPoint > not found in file <file path>.](#)

[Selected IED <IED name> not found in file <file path>.](#)

[Selected SubNetwork <SubNetwork > not found in file <file path>.](#)

---

**Duplicate address <tag address> produced from SCL file. Tag will not be created.**

---

**Error Type:**

Warning

**Possible Cause:**

1. A structured data object in the selected SCL file has two file attributes with the same name.
2. Two control blocks are configured with the same name.

**Solution:**

Verify that all data object and control block names are unique.

---

**Error: Device <device name>, Address <MMS address>, Invalid bit string data <bit string>. Please enter a string of characters '0' and '1'.**

---

**Error Type:**

Warning

**Possible Cause:**

Data that was written to a tag representing an MMS bit string includes a character besides '0' or '1'.

**Solution:**

Verify that the data being written to bit string tags only includes the '0' and '1' characters. For example, "001101".

---

**Error: Device <device name>, Address <MMS address>, Invalid data type <data type> for data attribute.**

---

**Error Type:**

Warning

**Possible Cause:**

The tag's data type is not valid for the data attribute referenced by the tag's address.

**Solution:**

Verify that the tag has an appropriate data type for the data attribute to which it refers.

**Note:**

For more information on the OPC data types' corresponding MMS data types, refer to [Data Types Description](#).

---

**Error: Device <device name>, Address <MMS address>, Invalid octet string data <octet string>. Please enter an even number of hexadecimal digits (for example, "00 01 C3").**

---

**Error Type:**

Warning

**Possible Cause:**

Data was written to a tag representing an MMS octet string that is not an even number of hexadecimal digits.

**Solution:**

Verify that the data written being to octet string tags includes an even number of hexadecimal digits. For example, "00 01 C3".

**Error: Device <device name>, Cannot write to structured data attribute <MMS address>.**

---

**Error Type:**

Warning

**Possible Cause:**

A tag has been configured with the address of a structured data attribute.

**Solution:**

Verify that all tags only refer to data attributes with primitive types.

**Note:**

Tags must refer to data attributes with primitive types.

**Error: Device <device name>, Invalid object <MMS address>.**

---

**Error Type:**

Warning

**Possible Cause:**

A tag has been configured with an address that is not valid for the connected device.

**Solution:**

Verify that the tags (including the given address) are configured correctly.

**Error: Device <device name>, Structured data attribute <MMS address> cannot be applied to tag.**

---

**Error Type:**

Warning

**Possible Cause:**

A tag has been configured with the address of a structured data attribute.

**Solution:**

Verify that all tags only refer to data attributes with primitive types.

**Note:**

Tags must refer to data attributes with primitive types.

---

**Failed to open SCL file <file path>.**

---

**Error Type:**

Warning

**Possible Cause:**

The selected SCL File is not a valid UNC path.

**Solution:**

Verify that the selected SCL file path is valid.

---

**Invalid address <tag address> produced from SCL file. Tag will not be created.**

---

**Error Type:**

Warning

**Possible Cause:**

A data object in the specified SCL File has a character that is not valid for IEC 61850 addresses.

**Solution:**

Verify that the DataTypeTemplates section of the specified SCL File only includes valid characters.

---

**No ConnectedAP of SubNetwork <SubNetwork> with IED <IED> and AccessPoint <AccessPoint> found in file <file path>.**

---

**Error Type:**

Warning

**Possible Cause:**

The SubNetwork and IED selected in the Automatic Configuration group (located in **Device Properties**) do not form a ConnectedAP in the selected SCL file.

**Solution:**

Verify that the selected SubNetwork, IED, and SCL File paths are valid.

---

**Selected AccessPoint <AccessPoint > not found in file <file path>.**

---

**Error Type:**

Warning

**Possible Cause:**

The AccessPoint property is not present in the file as specified in the SCL File field (located in the **Automatic Configuration** group of **Device Properties**).

**Solution:**

Verify that both the AccessPoint and SCL file properties are configured correctly.

---

**Selected IED <IED name> not found in file <file path>.**

---

**Error Type:**

Warning

**Possible Cause:**

The IED is not present in the file as specified in the SCL File field (located in the **Automatic Configuration** group of **Device Properties**).

**Solution:**

Verify that both the IED and SCL File properties are configured correctly.

**Selected SubNetwork <SubNetwork> not found in file <file path>.**

---

**Error Type:**

Warning

**Possible Cause:**

The SubNetwork property is not present in the file as specified in the SCL File field (located in the **Automatic Configuration** group of **Device Properties**).

**Solution:**

Verify that both the SubNetwork and SCL File properties are configured correctly.

# Resources

In addition to this user manual, there are a variety of resources available to assist customers, answer questions, provide more detail about specific implementations, or help with troubleshooting specific issues.

[Knowledge Base](#)

[Whitepapers](#)

[Connectivity Guides](#)

[Technical Notes](#)

[Training Programs](#)

[Training Videos](#)

[Kepware Technical Support](#)

[PTC Technical Support](#)

# Index

## A

Address Descriptions 23  
Advanced Channel Properties 7  
Allow Sub Groups 13  
Attempts Before Timeout 10  
Automatic Configuration 13

## C

Channel Assignment 8  
Channel Properties - General 5  
Channel Properties — Ethernet Communications 6  
Channel Properties — Write Optimizations 6  
Communication 17  
Communications Timeouts 10-11  
Connect Timeout 10  
Connection 15  
Control 18  
Create 13

## D

Data Collection 9  
Data Types Description 22  
Delete 13  
Demote on Failure 11  
Demotion Period 11  
Description 8  
Device Properties — Auto-Demotion 11  
Device Properties — General 8  
Device Properties — Tag Generation 11  
Diagnostics 5  
Discard Requests when Demoted 11  
Do Not Scan, Demand Poll Only 10  
Driver 5, 8



Duplicate address <tag address> produced from SCL file. Tag will not be created. 27

Duty Cycle 7

## E

Error Descriptions 26

Error: Device <device name>, Address <MMS address>, Invalid bit string data <bit string>. Please enter a string of characters '0' and '1'. 27

Error: Device <device name>, Address <MMS address>, Invalid data type <data type> for data attribute. 27

Error: Device <device name>, Address <MMS address>, Invalid octet string data <octet string>. Please enter an even number of hexadecimal digits (for example, 00 01 C3). 27

Error: Device <device name>, Cannot write to structured data attribute <MMS address>. 28

Error: Device <device name>, Invalid object <MMS address>. 28

Error: Device <device name>, Structured data attribute <MMS address> cannot be applied to tag. 28

## F

Failed to open SCL file <file path>. 29

## G

Generate 12

## H

Help Contents 4

## I

ID 8

IEEE-754 floating point 7

Initial Updates from Cache 10

Inter-Request Delay 11

Invalid address <tag address> produced from SCL file. Tag will not be created. 29

**M**

Model 8

**N**

Name 8

Network Adapter 6

No ConnectedAP of SubNetwork <SubNetwork> with IED <IED> and AccessPoint <AccessPoint> found in file <file path>. 29

Non-Normalized Float Handling 7

**O**

On Device Startup 12

On Duplicate Tag 12

On Property Change 12

Optimization Method 6

Overview 4

Overwrite 13

**P**

Parent Group 13

**R**

Redundancy 20

Request All Data at Scan Rate 9

Request Data No Faster than Scan Rate 9

Request Timeout 10

Resources 31

Respect Client-Specified Scan Rate 9

Respect Tag-Specified Scan Rate 10

**S**

Scan Mode 9

Selected AccessPoint <AccessPoint > not found in file <file path>. 29

Selected IED <IED name> not found in file <file path>. 29

Selected SubNetwork <SubNetwork> not found in file <file path>. 30

Setup 5

Simulated 9

Supported Functional Constraints 24

**T**

Tag Database Settings 19

Tag Generation 11

Timeouts to Demote 11

**W**

Write All Values for All Tags 6

Write Only Latest Value for All Tags 7

Write Only Latest Value for Non-Boolean Tags 6

Write Optimizations 6