DDE Client Driver

Table of Contents

DDE Client Driver

Help version 1.022

CONTENTS

Overview

The DDE Client Driver provides a reliable way to connect DDE Clients to OPC Client applications, including HMI, SCADA, Historian, MES, ERP, and countless custom applications.

Features

- The server provides single-point management of multiple DDE servers through OPC.
- The DDE Client Driver can communicate with any server that supports the standard CF TEXT DDE data format. It automatically reconnects when DDE servers are brought back online.
- Clients may reside on remote machines and communicate with this driver via OPC or NetDDE; however, the current version of this driver cannot access data from DDE servers running on remote machines.

Note: A ReinitializeDDE Tag is provided for the resolution of DDE communication failures. For more information, refer to **ReinitializeDDE Tag**.

Driver Setup

In the DDE Client Driver, the device in the OPC server project does not necessarily correspond to a physical device on a network. The device is actually a DDE client program that collects data provided by the various DDE servers on the system.

Important: The DDE Client Driver only supports one channel with one device (or one DDE client program). That device may be used to communicate with any number of physical devices through the DDE server applications, however. Furthermore, the OPC server project can have additional channels that use other drivers.

Important: For the DDE Client Driver to connect to DDE servers, the runtime must be allowed to interact with the desktop. For more information, refer to the "How To... Allow Desktop Interactions" section of the main server help file.

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		Identification	
General		Name	
Write Optimizations		Description	
Advanced		Driver	
		Diagnostics	
		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 — 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		□ **Non-Normalized Float Handling**	
General		Floating-Point Values	Replace with Zero
Write Optimizations		□ **Inter-Device Delay**	
Advanced		Inter-Device Delay (ms)	0

Non-Normalized Float Handling: Non-normalized float handling allows users to specify how a driver handles non-normalized IEEE-754 floating point data. 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. 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	
Auto-Demotion	Channel Assignment	
Redundancy	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	Scan Mode	
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 — DDE Server Communications

Users may change the DDE Server Communication properties at any time by right-clicking on the DDE Client Driver device and then selecting **Properties | DDE Server Communications**.

Property Groups	DDE Server Communications	
General	Initial Update Request Delay (s)	5
Scan Mode	DDE Server Check Interval (s)	15
DDE Server Communicati…		

Initial Update Request Delay: Specify the amount of time that the driver should wait for a link's initial update before ensuring it an explicit request. This property is useful for DDE servers that do not automatically send their clients an item's current value once a link to that item is made.

Tip: Some DDE servers perform poorly when updates are requested while a large number of links are being created. It is recommended that a larger value be specified for these servers to allow time for all links to be created before the driver begins issuing requests for initial updates. Servers that automatically send initial updates when links are created should work well with any value.

DDE Server Check Interval: Specify the amount of time that the driver waits before checking whether all DDE links have received a data update. The valid range is 0 to 300 seconds. A value of zero disables the link check. The default setting is 15 seconds.

Notes:

1. This driver uses hot links, where a DDE server only passes data to the driver when the data value has changed. This provides a significant performance advantage over the alternative request / response loop approach because DDE transactions can be time consuming. The disadvantage of this method is that the driver may not receive notice that a DDE server has become unavailable (such as in an abnormal shutdown). Because of this possibility, users should configure the driver to check the DDE links. When checking a link, the driver makes an explicit request for data from the DDE server. If the DDE server does not respond to this request, the driver attempts to reconnect. If the attempt fails, a message is placed in the OPC server's Event Log and the driver continues attempting until the problem is resolved.

2. If a value of zero is chosen, the ReinitializeDDE Tag may need to be used to rebuild all links after a DDE server is brought back up. *For more information, refer to ReinitializeDDE Tag.*

Defining Tags

A tag must be created for each DDE item that are available to OPC clients.

🔹 *For information on specifying the link between a tag and an item provided by a DDE server, refer to **Address Descriptions**.*

ReinitializeDDE Tag

The DDE Client Driver is usually able to automatically establish a hot link whenever a DDE server makes a properly addressed item available. It can also detect when that item becomes unavailable. In rare cases, it may be necessary to reinitialize the driver in order to properly link with DDE items after a communications breakdown. This operation is initiated by writing True (any non-zero value) to the ReinitializeDDE Tag. Success or failure will be reported in the OPC server's Event Log. Common causes of failure include one or more tags with an invalid address, and DDE servers that are not currently running.

🔹 **Note:** The ReinitializeDDE Tag is automatically created by the driver, and is located through the client's tag browser in the "<device name>.InternalTags" group. If the OPC Quick Client is launched from the OPC server, the ReinitializeDDE Tag will be placed at the device level.

Data Types Description

Data Type	Description
Boolean	Single bit
Byte	Unsigned 8-bit value bit 0 is the low bit bit 7 is the high bit
Char	Signed 8-bit value bit 0 is the low bit bit 6 is the high bit bit 7 is the sign bit
Word	Unsigned 16-bit value bit 0 is the low bit bit 15 is the high bit
Short	Signed 16-bit value bit 0 is the low bit bit 14 is the high bit bit 15 is the sign bit
DWord	Unsigned 32-bit value bit 0 is the low bit bit 31 is the high bit
Long	Signed 32-bit value bit 0 is the low bit bit 30 is the high bit bit 31 is the sign bit
Float	32-bit Floating point value. The driver interprets two consecutive 16-bit registers as a Floating point value by making the second register the high word and the first register the low word.
Double	64-bit Floating point value
String	Zero terminated character array. This driver limits string lengths to 256 characters.

● **Note:** The default data type is DWord.

Address Descriptions

The DDE Client Driver supports standard DDE addressing. An example of the syntax is as follows:

<DDE Service Name>|<Topic Name>!<Item Name>

Descriptions of the elements are as follows:

- **DDE Service Name:** The name that an application exposes for DDE communication. It is often the same as the application name.
- **Topic Name:** The name that an application gives to a logical grouping of addressable items.
- **Item Name:** The name of the specific data item associated with a topic.

Notes:

1. An example of the syntax is "Excel|Sheet3!R1C1". This example demonstrates how to access a value from a Microsoft Excel spreadsheet. The DDE service name is "Excel," the topic name is the name of the sheet that will be accessed ("Sheet3"), and the item name is the spreadsheet cell that will be accessed ("R1C1" for row 1 and column 1). These same principles can be applied to any DDE server.

2. For a specific DDE server application's service, topic and item names, refer to its documentation.

Addressing Notes

1. The address is not case sensitive.

2. An equal sign (=) may precede an address, but will be ignored by the driver.

3. Single quotation marks will not be removed before an address is passed to the DDE server application. For example, "Excel|'Sheet3'!R1C1" will not work because there are no single quotes in the worksheet name. Instead, the address should be "Excel|Sheet3!R1C1".

4. Spaces are valid and will be passed to the DDE server application. For example, "Excel|Test Sheet!R1C1" is valid. There is a space between Test and Sheet.

5. This driver cannot address data on remote machines using NetDDE; thus, any address containing the string "ndde$" is invalid. The client application can access the data collected by this driver via NetDDE and OPC.

6. The DDE Client Driver will accept any address that has valid syntax, but it will not tell if the specified DDE server, topic, and item are available until a tag is made active by a client application. If a link for that tag cannot be made at that time, the tag's data quality will be set to bad and an appropriate error message will be placed in the OPC server's Event Log. *For more information, refer to **Error Descriptions**. service name; however, users may experience difficulties establishing links to these items.*

7. Although the OPC server can also act as a DDE server, users should not attempt to access its items with this driver. Any address using the default service name of the OPC server will not be accepted. This should not present a problem because the purpose of this driver is to provide OPC connectivity to other DDE servers via the OPC server and this driver. This driver allows users to connect to items in the OPC server in order to change its default DDE

Error Descriptions

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

Address Validation

Address <address> is out of range for the specified device or register.

Array support is not available for the specified address: <address>.

Data Type <type> is not valid for device address <address>.

Device address <address> contains a syntax error.

Device address <address> is not supported by model <model name>.

Device address <address> is read only.

Missing address.

Device Status Messages

Device <device name> is not responding.

Unable to write to <address> on device <device name>. Error code <error code>.

DDE Client Driver Specific Messages

DDE data error on link <server name>, <topic name>, <item name> detected.

DDEML failed to initialize.

DDEML reports low memory error.

Failed to connect server <server name> topic <topic name>.

Failed to establish hot link to <server name>, <topic name>, <item name>.

Reinitialization failed.

Server <server name> topic <topic name> disconnected.

Address <address> is out of range for the specified device or register.

Error Type:
Warning

Possible Cause:
A tag address that contains the DDE service name of the OPC server has been entered.

Solution:
The OPC server should not use the DDE Client Driver to serve data to itself.

Array support is not available for the specified address: <address>.

Error Type:
Warning

Possible Cause:
A tag address that has been specified statically contains an array reference for an address type that doesn't support arrays.

Solution:
Re-enter the address in the client application to remove the array reference or correct the address type.

Data Type <type> is not valid for device address <address>.

Error Type:
Warning

Possible Cause:
A tag address that has been specified statically has been assigned an invalid data type.

Solution:
Modify the requested data type in the client application.

Device address <address> contains a syntax error.

Error Type:
Warning

Possible Cause:
A tag address that has been specified statically contains one or more invalid characters.

Solution:
Re-enter the address in the client application.

Device address <address> is not supported by model <model name>.

Error Type:
Warning

Possible Cause:

A tag address that has been specified statically references a location that is valid for the communications protocol but not supported by the target device.

Solution:
Verify that the address is correct; if it is not, re-enter it in the client application. Also verify that the selected model name for the device is correct.

Device address <address> is read only.

Error Type:
Warning

Possible Cause:
A tag address that has been specified statically has a requested access mode that is not compatible with what the device supports for that address.

Solution:
Change the access mode in the client application.

Missing address.

Error Type:
Warning

Possible Cause:
A tag address that has been specified statically has no length.

Solution:
Re-enter the address in the client application.

Device <device name> is not responding.

Error Type:
Serious

Possible Cause:

1. The serial connection between the device and the host PC is broken.

2. The communication properties for the serial connection are incorrect.

3. The named device may have been assigned an incorrect Network ID.

4. The response from the device took longer to receive than the amount of time specified in the "Request Timeout" device property.

Solution:

1. Verify the cabling between the PC and the device.

2. Verify that the specified communication properties match those of the device.

3. Verify that the Network ID given to the named device matches that of the actual device.

4. Increase the Request Timeout property so that the entire response can be handled.

Unable to write to <address> on device <device name>. Error code <error code>.

Error Type:
Serious

Possible Cause:

1. The serial connection between the device and the host PC is broken.

2. The communication properties for the serial connection are incorrect.

3. The named device may have been assigned an incorrect Network ID.

4. For more information, refer to the specific error code in the table below.

Solution:

1. Verify the cabling between the PC and the device.

2. Verify that the specified communication properties match those of the device.

3. Verify that the Network ID given to the named device matches that of the actual device.

4. For more information, refer to the specific error code in the table below.

Error Code	Description
0x0000	DMLERR_NO_ERROR
0x4000	DMLERR_ADVACKTIMEOUT
0x4001	DMLERR_BUSY
0x4002	DMLERR_DATAACKTIMEOUT
0x4003	DMLERR_DLL_NOT_INITIALIZED
0x4005	DMLERR_EXECACKTIMEOUT
0x4006	DMLERR_INVALIDPARAMETER
0x4008	DMLERR_MEMORY_ERROR
0x4009	DMLERR_NOTPROCESSED
0x400A	DMLERR_NO_CONV_ESTABLISHED
0x400B	DMLERR_POKEACKTIMEOUT
0x400C	DMLERR_POSTMSG_FAILED
0x400D	DMLERR_REENTRANCY
0x400E	DMLERR_SERVER_DIED
0x4010	DMLERR_UNADVACKTIMEOUT

DDE data error on link <server name>, <topic name>, <item name> detected.

Error Type:
Serious

Possible Cause:

1. The driver was unable to convert CF_TEXT data from a DDE server to a numerical type.

2. An attempt was made to write a string that is too long.

Solution:

1. Ensure that the DDE server is configured to send numerical data (in CF_TEXT format) if the client is expecting numerical data.

2. Ensure that strings are within length requirements.

See Also:
Data Types Description

DDEML failed to initialize.

Error Type:
Fatal

Possible Cause:
The Microsoft Windows DDE Management Library .dll failed to register the DDE Client Driver properly.

Solution:
Shut down all unnecessary applications and restart the OPC server project.

DDEML reports low memory error.

Error Type:
Serious

Possible Cause:
The Microsoft Windows DDE Management Library dll has exceeded its resource limit due to a prolonged race condition. This occurs when the server outruns the client.

Solution:

1. If possible, decrease the update rate of the DDE server.

2. Reduce the size of the OPC server project.

Failed to connect server <server name> topic <topic name>.

Error Type:
Serious

Possible Cause:

1. The address is incorrect.

2. The DDE server is not running.

3. The DDE server does not support the specified topic.

Solution:

1. Ensure that the address is correct.

2. Ensure that the DDE server is running.

3. Ensure that the DDE server supports specified topic.

4. If a zero has been specified as the server check interval, use the ReinitializeDDE Tag.

☀ See Also:
Driver Setup
ReinitalizeDDE Tag

Failed to establish hot link to <server name>, <topic name>, <item name>.

Error Type:
Serious

Possible Cause:

1. The address is incorrect.

2. The DDE server is not running.

3. The DDE server does not support the specified item.

Solution:

1. Ensure that the address is correct.

2. Ensure that the DDE server is running.

3. Ensure that the DDE server supports specified item.

4. If a zero has been specified as the server check period, use the ReinitializeDDE Tag.

☀ See Also:
Driver Setup
ReinitalizeDDE Tag

Reinitialization failed.

Error Type:
Serious

Possible Cause:

One or more links could not be made.

Solution:

Check to see if the server in question is running and supports the specified topic and item.

Server <server name> topic <topic name> disconnected

Error Type:

Serious

Possible Cause:

The specified DDE server has stopped running.

Solution:

1. Restart the DDE server in question.

2. If a zero has been specified for the server check period, use the ReinitializeDDE Tag.

See Also:

Driver Setup

ReinitalizeDDE Tag

Index

A

B

C

D