OMNI Flow Computer Driver

Table of Contents

OMNI Flow Computer Driver

Help version 1.045

CONTENTS

Overview

The OMNI Flow Computer Driver provides real-time and EFM data access. In addition to archive, time zone, and device password settings, the driver configuration also maps data in the device to the server's EFM Model (which consists of various EFM attributes such as pressure, temperature, and so forth).

Note: EFM functionality is not available in all server versions. To determine whether support is available, refer to the "Server Summary Information" topic located in the server help file.

Setup

Supported Communication Properties

Baud Rate: 1200, 2400, 9600, and 19200.
Parity: Odd, Even, and None.
Data Bits: 8.
Stop Bits: 1 and 2.

Note: Not all of the listed configurations may be supported in every device.

Supported Firmware Versions

20.xx
21.xx
22.xx
23.xx
24.xx
26.xx
27.xx

Maximum Number of Channels and Devices

The maximum number of supported channels is 256. The maximum number of devices supported per channel is 255.

Ethernet Encapsulation

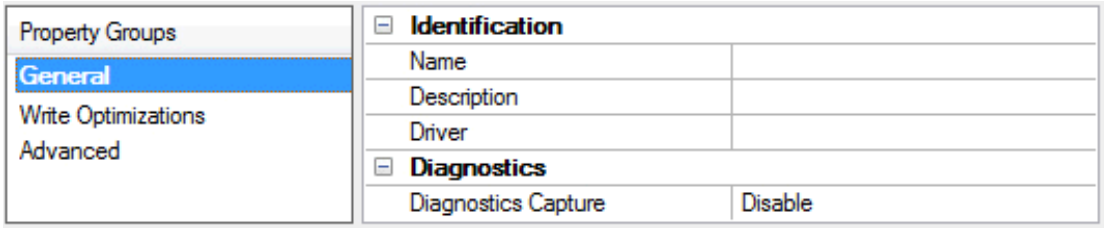
This driver supports Ethernet Encapsulation, which allows the driver to communicate with serial devices attached to an Ethernet network using a terminal server. It may be enabled through the Communications group in Channel Properties. For more information, refer to the main server's help file.

Communication Serialization

The OMNI Flow Computer Driver supports Communication Serialization, which specifies whether data transmissions should be limited to one channel at a time. For more information, refer to "Channel Properties - Advanced" in the 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.



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.

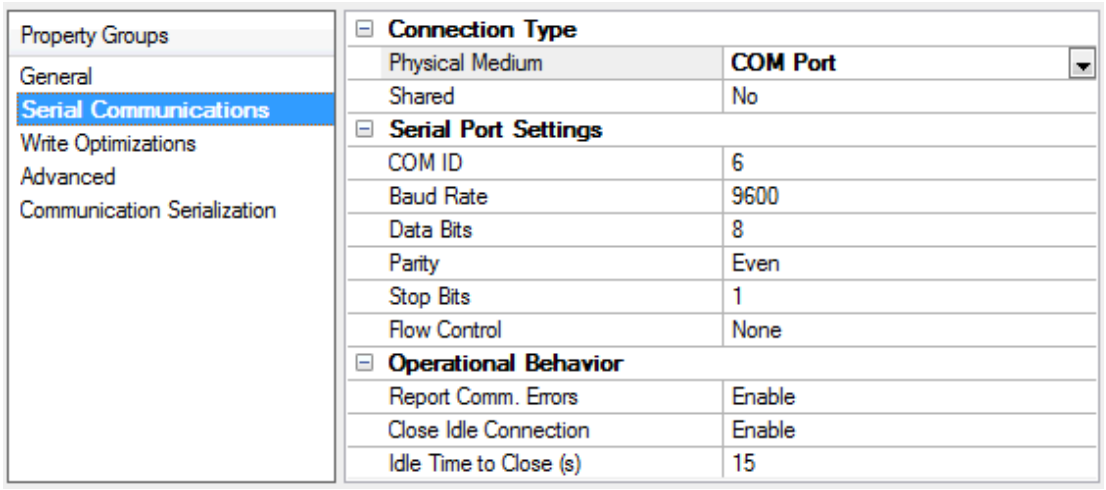
Diagnostics

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 — Serial Communications

Serial communication properties are available to serial drivers and vary depending on the driver, connection type, and options selected. Below is a superset of the possible properties.
Click to jump to one of the sections: **Connection Type**, **Serial Port Settings** or **Ethernet Settings**, and **Operational Behavior**.

Note: With the server's online full-time operation, these properties can be changed at any time. Utilize the User Manager to restrict access rights to server features, as changes made to these properties can temporarily disrupt communications.



Connection Type

Physical Medium: Choose the type of hardware device for data communications. Options include COM Port, None, Modem, and Ethernet Encapsulation. The default is COM Port.

- **None**: Select None to indicate there is no physical connection, which displays the **Operation with no Communications** section.
- **COM Port**: Select Com Port to display and configure the **Serial Port Settings** section.
- **Modem**: Select Modem if phone lines are used for communications, which are configured in the **Modem Settings** section.
- **Ethernet Encap.**: Select if Ethernet Encapsulation is used for communications, which displays the **Ethernet Settings** section.
- **Shared**: Verify the connection is correctly identified as sharing the current configuration with another channel. This is a read-only property.

Serial Port Settings

COM ID: Specify the Communications ID to be used when communicating with devices assigned to the channel. The valid range is 1 to 9991 to 16. The default is 1.

Baud Rate: Specify the baud rate to be used to configure the selected communications port.

Data Bits: Specify the number of data bits per data word. Options include 5, 6, 7, or 8.

Parity: Specify the type of parity for the data. Options include Odd, Even, or None.

Stop Bits: Specify the number of stop bits per data word. Options include 1 or 2.

Flow Control: Select how the RTS and DTR control lines are utilized. Flow control is required to communicate with some serial devices. Options are:

- **None**: This option does not toggle or assert control lines.
- **DTR**: This option asserts the DTR line when the communications port is opened and remains on.

- **RTS**: This option specifies that the RTS line is high if bytes are available for transmission. After all buffered bytes have been sent, the RTS line is low. This is normally used with RS232/RS485 converter hardware.
- **RTS, DTR**: This option is a combination of DTR and RTS.
- **RTS Always**: This option asserts the RTS line when the communication port is opened and remains on.
- **RTS Manual**: This option asserts the RTS line based on the timing properties entered for RTS Line Control. It is only available when the driver supports manual RTS line control (or when the properties are shared and at least one of the channels belongs to a driver that provides this support).
  RTS Manual adds an **RTS Line Control** property with options as follows:
  - **Raise**: This property specifies the amount of time that the RTS line is raised prior to data transmission. The valid range is 0 to 9999 milliseconds. The default is 10 milliseconds.
  - **Drop**: This property specifies the amount of time that the RTS line remains high after data transmission. The valid range is 0 to 9999 milliseconds. The default is 10 milliseconds.
  - **Poll Delay**: This property specifies the amount of time that polling for communications is delayed. The valid range is 0 to 9999. The default is 10 milliseconds.

Tip: When using two-wire RS-485, "echoes" may occur on the communication lines. Since this communication does not support echo suppression, it is recommended that echoes be disabled or a RS-485 converter be used.

Operational Behavior

- **Report Comm. Errors**: Enable or disable reporting of low-level communications errors. When enabled, low-level errors are posted to the Event Log as they occur. When disabled, these same errors are not posted even though normal request failures are. The default is Enable.
- **Close Idle Connection**: Choose to close the connection when there are no longer any tags being referenced by a client on the channel. The default is Enable.
- **Idle Time to Close**: Specify the amount of time that the server waits once all tags have been removed before closing the COM port. The default is 15 seconds.

Ethernet Settings

Note: Not all serial drivers support Ethernet Encapsulation. If this group does not appear, the functionality is not supported.

Ethernet Encapsulation provides communication with serial devices connected to terminal servers on the Ethernet network. A terminal server is essentially a virtual serial port that converts TCP/IP messages on the Ethernet network to serial data. Once the message has been converted, users can connect standard devices that support serial communications to the terminal server. The terminal server's serial port must be properly configured to match the requirements of the serial device to which it is attached. *For more information, refer to "How To... Use Ethernet Encapsulation" in the server help.*

- **Network Adapter**: Indicate a network adapter to bind for Ethernet devices in this channel. Choose a network adapter to bind to or allow the OS to select the default.
  Specific drivers may display additional Ethernet Encapsulation properties. For more information, refer to Channel Properties - Ethernet Encapsulation.

Modem Settings

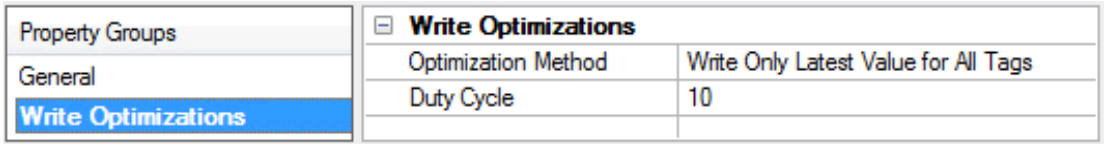
- **Modem**: Specify the installed modem to be used for communications.
- **Connect Timeout**: Specify the amount of time to wait for connections to be established before failing a read or write. The default is 60 seconds.
- **Modem Properties**: Configure the modem hardware. When clicked, it opens vendor-specific modem properties.
- **Auto-Dial**: Enables the automatic dialing of entries in the Phonebook. The default is Disable. *For more information, refer to "Modem Auto-Dial" in the server help.*
- **Report Comm. Errors**: Enable or disable reporting of low-level communications errors. When enabled, low-level errors are posted to the Event Log as they occur. When disabled, these same errors are not posted even though normal request failures are. The default is Enable.
- **Close Idle Connection**: Choose to close the modem connection when there are no longer any tags being referenced by a client on the channel. The default is Enable.
- **Idle Time to Close**: Specify the amount of time that the server waits once all tags have been removed before closing the modem connection. The default is 15 seconds.

Operation with no Communications

- **Read Processing**: Select the action to be taken when an explicit device read is requested. Options include Ignore and Fail. Ignore does nothing; Fail provides the client with an update that indicates failure. The default setting is Ignore.

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.

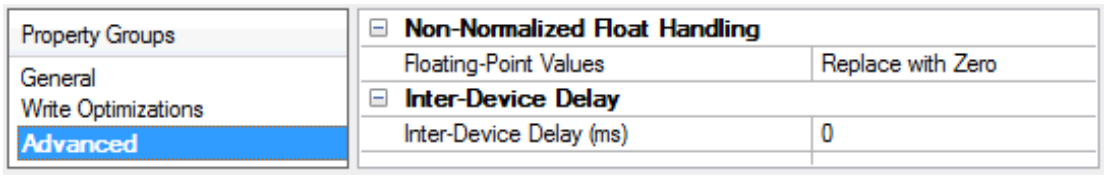
- **Write Only Latest Value for All Tags**: This option takes the theory behind the second optimization mode and applies it to all tags. It is especially useful if the application only needs to send the latest value to the device. This mode optimizes all writes by updating the tags currently in the write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



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.

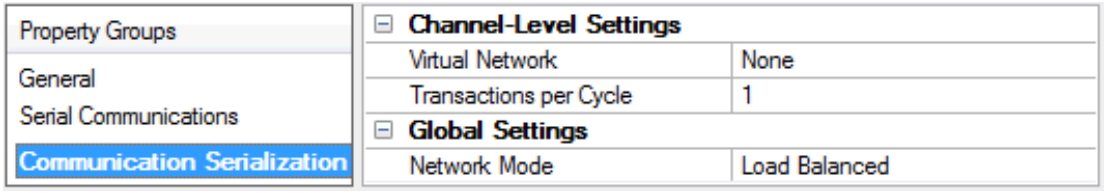
Inter-Device Delay: Specify the amount of time the communications channel waits to send new requests to the next device after data is received from the current device on the same channel. Zero (0) disables the delay.

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

Channel Properties — Communication Serialization

The server's multi-threading architecture allows channels to communicate with devices in parallel. Although this is efficient, communication can be serialized in cases with physical network restrictions (such as Ethernet radios). Communication serialization limits communication to one channel at a time within a virtual network.

The term "virtual network" describes a collection of channels and associated devices that use the same pipeline for communications. For example, the pipeline of an Ethernet radio is the master radio. All channels using the same master radio associate with the same virtual network. Channels are allowed to communicate each in turn, in a “round-robin” manner. By default, a channel can process one transaction before handing communications off to another channel. A transaction can include one or more tags. If the controlling channel contains a device that is not responding to a request, the channel cannot release control until the transaction times out. This results in data update delays for the other channels in the virtual network.



Channel-Level Settings

Virtual Network This property specifies the channel's mode of communication serialization. Options include None and Network 1 - Network 50. The default is None. Descriptions of the options are as follows:

- **None**: This option disables communication serialization for the channel.
- **Network 1 - Network 50**: This option specifies the virtual network to which the channel is assigned.

Transactions per Cycle This property specifies the number of single blocked/non-blocked read/write transactions that can occur on the channel. When a channel is given the opportunity to communicate, this number of transactions attempted. The valid range is 1 to 99. The default is 1.

Global Settings

- **Network Mode**: This property is used to control how channel communication is delegated. In **Load Balanced** mode, each channel is given the opportunity to communicate in turn, one at a time. In **Priority** mode, channels are given the opportunity to communicate according to the following rules (highest to lowest priority):
    - Channels with pending writes have the highest priority.
    - Channels with pending explicit reads (through internal plug-ins or external client interfaces)

are prioritized based on the read's priority.

- Scanned reads and other periodic events (driver specific).

The default is Load Balanced and affects *all* virtual networks and channels.

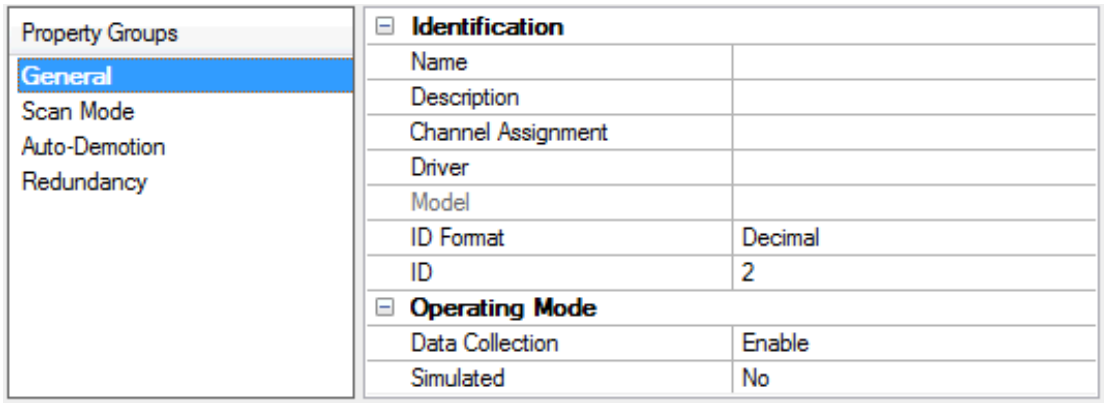
Devices that rely on unsolicited responses should not be placed in a virtual network. In situations where communications must be serialized, it is recommended that Auto-Demotion be enabled.

Due to differences in the way that drivers read and write data (such as in single, blocked, or non-blocked transactions); the application's Transactions per cycle property may need to be adjusted. When doing so, consider the following factors:

- How many tags must be read from each channel?
- How often is data written to each channel?
- Is the channel using a serial or Ethernet driver?
- Does the driver read tags in separate requests, or are multiple tags read in a block?
- Have the device's Timing properties (such as Request timeout and Fail after *x* successive timeouts) been optimized for the virtual network's communication medium?

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.



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.

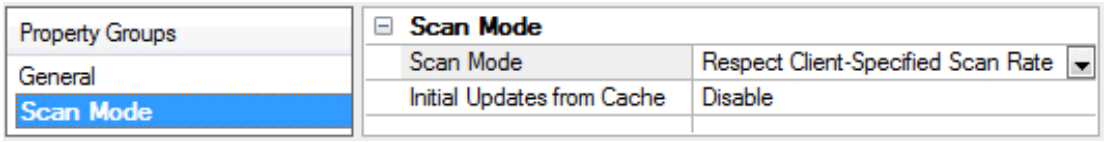
Notes:

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.



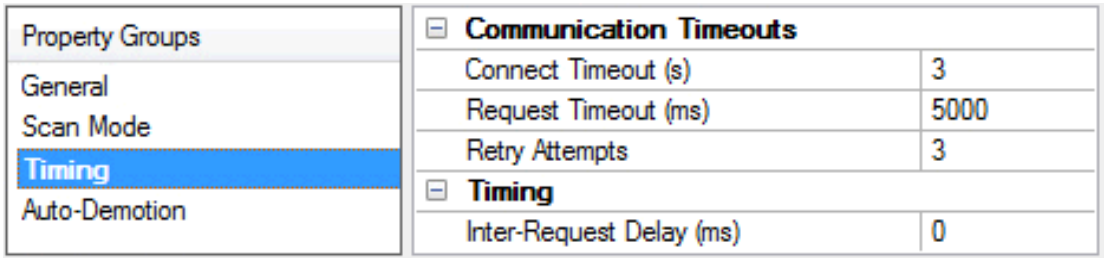
Scan Mode: specifies how tags in the device are scanned for updates sent to subscribed clients. Descriptions of the options are:

- **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.



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.

Retry Attempts: This property specifies how many times the driver retries 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 retries configured for an application depends largely on the communications environment. This property applies to both connection attempts and request attempts.

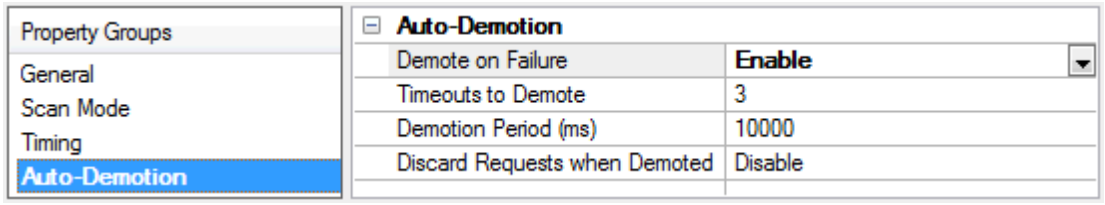
Timing

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.



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.

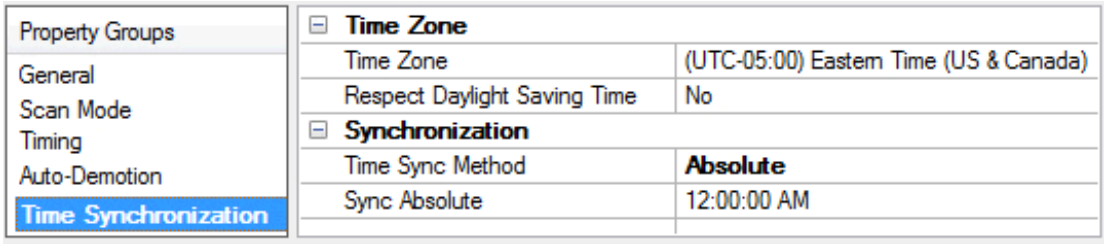
Timeouts to Demote: Specify how many successive cycles of request timeouts and retries occur before the device is placed off-scan. The valid range is 1 to 30 successive failures. The default is 3.

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 — Time Synchronization

This group is used to specify the device's time zone and time synchronization properties. It primarily applies to time stamped data or information from battery-powered devices at remote locations where the device time may deviate (causing issues with the time-stamped data). To prevent this problem from occurring, users can specify that the server synchronize the device time.



Note: Not all drivers and models support all options.

Time Zone: This property specifies the device's time zone. To ignore the time zone, select one of the first four options in the list (which do not have an offset). The default is the time zone of the local system.
Note: The driver uses this property both when synching the device time and when converting EFM timestamps from the device to UTC time.

Respect Daylight Saving Time: Select Yes to follow Daylight Saving Time offset when synching the device time. Select No to ignore Daylight Saving Time. Only time zones that observe Daylight Saving Time will be affected. The default is No (disabled).
Note: When enabled, the time of the device is adjusted by +1 hour for Daylight Saving Time (in the spring), and adjusted by -1 hour after Daylight Saving Time (in the fall).

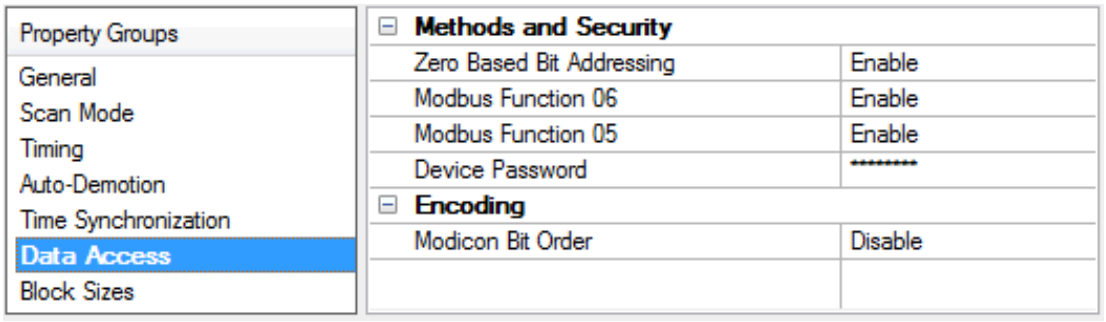
Method: This property specifies the method of synchronization. Options include Disabled, Absolute, and Interval. The default is Disabled. Descriptions of the options are as follows:

- **Disabled**: No synchronization.
- **Absolute**: Synchronizes to an absolute time of day specified through the Time property (appears only when Absolute is selected).
- **Interval**: Synchronizes on startup and every number of minutes specified through the Sync Interval

property (appears only when Interval is selected). The default is 60 minutes.

- **OnPoll**: Synchronizes when poll is completed (applicable only to EFM devices).

Device Properties - Data Access



Descriptions of the properties are as follows:

- **Zero-Based Bit Addressing:** When enabled, this option will use zero based bit addressing within registers and will start the first bit at 0. The default is enabled. For more information, refer to the "Zero vs. One Based Bit Addressing Within Registers" subtopic below.
- **Modbus Function 06:** When enabled, this option will use Modbus function 06 for single 16 bit register writes. The default is enabled. For more information, refer to the "Modbus Function 06" subtopic below.
- **Modbus Function 05:** When enabled, this option will use Modbus function 05 for single Boolean writes. This allows the driver to operate as it has historically, switching between 05 and 15 as needed. When disabled, all writes will be done using only Modbus function 15. The default is enabled. For more information, refer to the "Modbus Function 05" subtopic below.
- **Device Password:** Specify a password for the device.
- **Modicon Bit Order:** When enabled, the driver will reverse the bit order on reads and writes to registers to follow the convention of the Modicon Modsoft programming software. For example, when enabled, a write to address 40001.0/1 will affect bit 15/16 in the device. The default is disabled. For more information, refer to the "Modicon Bit Ordering" subtopic below.

Zero vs. One Based Bit Addressing Within Registers

Memory types that allow bits within Words can be referenced as a Boolean. The addressing notation for this is <*address*>.<*bit*>, where <*bit*> represents the bit number within the word. Bit level addressing within registers provides two ways of addressing a bit within a given word: Zero Based and One Based. Zero Based Bit addressing within registers simply means the first bit begins at 0. One Based Bit addressing means that the first bit begins at 1. Descriptions are as follows:

- **Zero Based:** For the Word data type, the bit range is 0 to 15.
- **One Based:** For the Word data type, the bit range is 1 to 16.

Modbus Function 06

Although all OMNI Firmware revisions support Function Code 06 when writing a single 16 bit register, revisions older than xx.44 will not support Function Code 06 when writing a single 32 bit register. As such, Function Code 16 must be used when writing one or more 32 bit registers on the older Firmware revisions. Even though newer Firmware revisions support Function Code 06 for single 32 bit register writes, Function Code 16 will always be used for single 32 bit register writes in order to support legacy installations.

Modbus Function 05

The OMNI Flow Computer Driver can use two Modbus protocol functions to write output coil data to the target device. In most cases, it will switch between these two functions based on the number of coils being written. When writing a single coil, the driver will use the Modbus function 05. When writing an array of coils, the driver will use Modbus function 15. The standard Modicon PLC can use either of these functions. There are many Third-Party devices that have implemented the Modbus protocol, however, and only support the use of Modbus function 15 to write to output coils (regardless of the number of coils being written).

Modicon Bit Ordering

For the following example, the 1st through 16th bit signifies either 0 to 15 bits or 1 to 16 bits, depending on whether the driver is set at zero based addressing within registers. In the tables below, MSB is the Most Significant Bit and LSB is the Least Significant Bit.

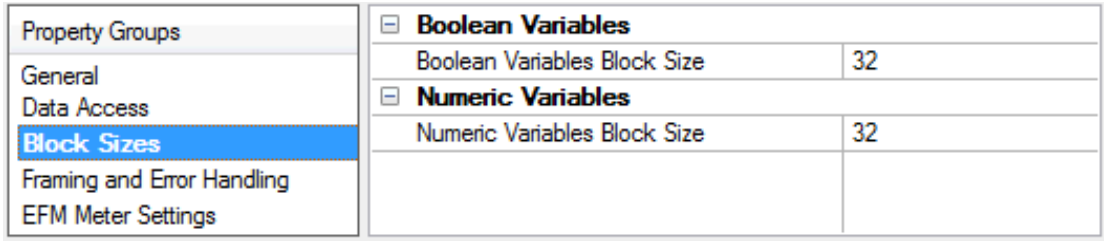
Modicon Bit Order Enabled

MSB															LSB
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

Modicon Bit Order Disabled

MSB															LSB
16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1

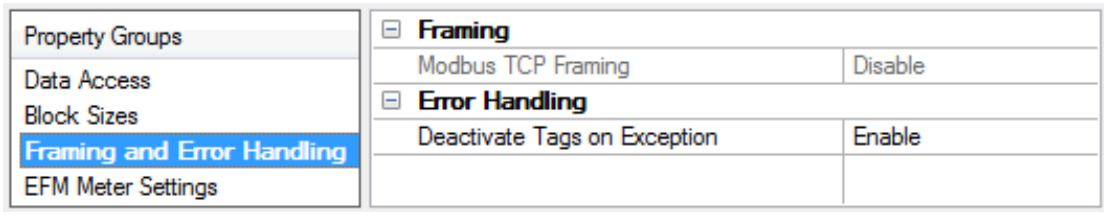
Device Properties - Block Sizes



Descriptions of the properties are as follows:

- **Boolean Variables Block Size:** Specify the Boolean variables. The valid range is 8 to 2000, in multiples of 8. The default is 32.
- **Numeric Variables Block Size:** Specify the Numeric variables. The valid range is 1 to 125. The default is 32.

Device Properties - Framing and Error Handling



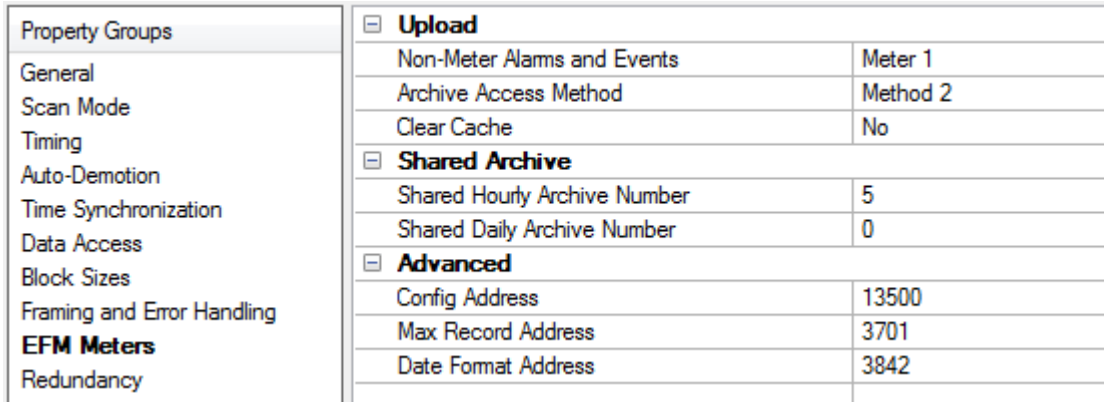
Descriptions of the properties are as follows:

- **Modbus TCP Framing:** When enabled, this option enables communications with native Modbus TCP devices using Ethernet Encapsulation. The default is disabled.
- **Deactivate Tags on Exception:** When enabled, the driver will stop polling for a block of data if the device returns Modbus exception code 2 (illegal address) or 3 (illegal data, such as number of points) in response to a read of that block. To read addresses that are accessible dynamically in the device, disable this option. The default is enabled.

Device Properties - EFM Meters

This dialog contains meter-specific EFM configuration and upload settings. The OMNI Flow Computer Driver supports up to four meters.

Upload



Upload

- **Non-Meter Alarms and Events**
  - **Ignore:** The alarms and events will not be sent to any meters.
  - **Meter 1, Meter 2, Meter 3, or Meter 4:** The alarms and events will be sent to one specified meter.
  - **All:** The alarms and events will be sent to all meters.
- **Archive Access Method:** Specify whether archives will be accessed using Method 1 or Method 2. Method 1 is the legacy method, and Method 2 is more efficient. The default is Method 2.
  Note: When Method 2 is selected but is not supported by the device, the read will fail and the driver will fall back to Method 1 and issue a warning.
- **Clear Cache:** Specify whether to clear the EFM cache, which is maintained by the server and stores history, alarms, and events data for each meter. When enabled, the cache will be cleared on the next poll. This feature will also remove pointer files, which are used to track EFM uploads in order to prevent uploading the same records twice. All EFM data in the device will be requested again on the next poll. Once the cache is cleared, this property will automatically be set back to No. The default is No.
  Note: This option should be used during testing, if the EFM mappings are not configured correctly, or in situations where it is beneficial to re-request all EFM data from the device. Some changes to meter properties clear the cache automatically.

Shared Archive

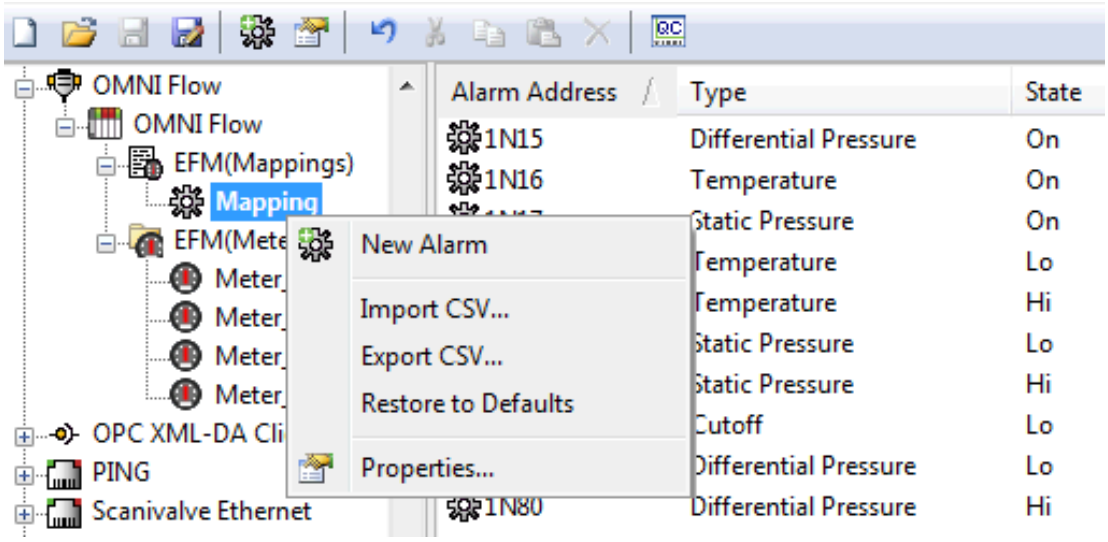
- **Shared Hourly Archive Number:** Specify the device archive number for shared hourly data. Each meter shares this archive. The valid range is 0 to 10. Setting this property to 0 will disable the archive. The default is 5.
- **Shared Daily Archive Number:** Specify the device archive number for shared daily data. Each meter shares this archive. The valid range is 0 to 10. Setting this property to 0 will disable the archive. The default is 0.
- **Shared Batch Archive Number:** Specify the device archive number for shared batch data. Each meter shares this archive. The valid range is 0 to 10. Setting this property to 0 will disable the archive. The default is 0.
  Note: This property is only supported by OMNI Liquid Firmware models.

Advanced

- **Config Address:** Specify the group configuration address for archive 1 in the device. The default is 13500. This setting should usually not be changed.
- **Max Record Address:** Specify the device address whose value is the maximum number of records for archive 1 in the device. The default is 3701. This setting should usually not be changed.
- **Date Format Address:** Specify the Date Format address that defines the date format for all archive data. The default is 3842. This setting should usually not be changed.

EFM Mapping

The Tree View configures the mapping of EFM data in the device to the server's EFM data model. The History and Alarms Mappings can be configured. Configuration and Event data is fixed.

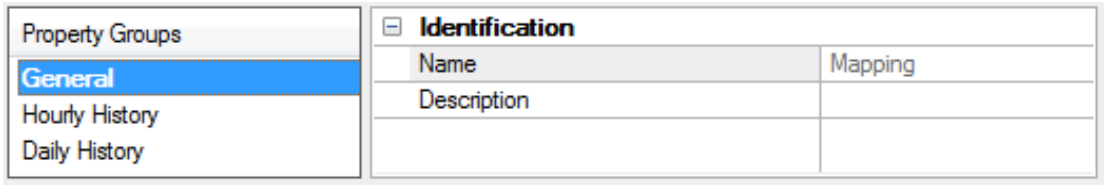


The following right-click options are available:

- **New Alarm**: used to map alarms to the server's alarms. The EFM Alarm Mapping is applied to all enabled meters. For more information, refer to **EFM Alarm Mapping**.
- **Import CSV**: used to import EFM History and EFM Alarm Mappings from a CSV file. The CSV file will replace all existing mappings.
- **Export CSV**: When clicked, this button launches the Export to CSV dialog, which is used to export the current EFM History and EFM Alarm Mappings to a CSV file for easy editing.

- **Restore to Defaults**: returns the EFM Mapping to the default settings.
- **Properties...** accesses the settings grid.

The General properties available for mappings include:



Name: Specify the name of the mapping. It is a user-defined name that can be up to 256 characters long.

Description: Enter a brief description of the mapping to help identify it in data and reports.

EFM Alarm Mapping

The Alarms dialog is used to assign alarms received from the device to specific meters, alarm types, and states. Alarms can apply to one or more meters depending on how the alarm address is specified.

Note: When an alarm is received from the device that does not match an address in the Alarm Mappings, the alarm will be logged as a user string alarm. It will be handled according to the "Non-Meter Alarms & Events" setting located in EFM Meter Settings. Furthermore, meter-specific registers that do not contain a meter-specific *n* designation will be routed to that same setting. For more information, refer to **EFM Meter Settings**.

Important: The default EFM Alarm Mapping contains some differential pressure mappings that are specific to Firmware versions 21.xx, 23.xx, and 27.xx. Devices running a different Firmware version than those listed can remove these mappings from the configuration.

Descriptions of the properties are as follows:

- **Address:** Specify the OMNI device address that generates the alarm. The default is blank.
- **Type:** Specify the type of alarm. The default is Differential Pressure. Options are as follows:
    - Differential Pressure
    - Static Pressure
    - Temperature
    - Cutoff
    - Backflow
    - Battery
- **State:** Specify the alarm state. The default is Off. Options are as follows:
    - Off
    - On
    - Lo
    - Hi

- **Add:** When clicked, this button launches a new Alarm dialog. For more information, refer to "Adding a New Alarm" below.
- **Delete:** When clicked, this button deletes the selected alarm from the mapping.
- **Edit:** When clicked, this button launches the Alarm dialog that contains the selected alarm's properties.

Alarm Syntax

An alarm address may use one of the following syntactic forms:

- **1N34:** This is a base address, and makes the alarm meter-specific. N will be replaced with the meter number.

  Note: For example, an address is "1N34". An alarm received from device address "1034" will be assigned to Meter 1. An alarm received from device address "1134" will be assigned to Meter 2.
- **1234:** This is a static address, and makes the alarm non-meter specific. An alarm that is received from address 1234 will be sent to all meters.

Adding a New Alarm

1. To start, right click **Mappings**.
2. Select **New Alarm**.
3. Next, in the Configure property group, specify the new alarm's Address, Type, and State.
4. Once finished, click **OK**.

EFM Event Mapping

EFM Event Mappings are not user-configurable: they depend on the Configuration Mapping.

When an event is received from an address that matches an address in any Configuration Mapping, it will be converted to an audit event. For example, a Configuration Mapping with address "7N48" is set to Pipe Diameter. An event will be generated from address 7148 when the user changes the Pipe Diameter in the device for Meter 1. The event will be converted to an audit event for the Pipe Diameter field on Meter 1. Both the old and new values will be displayed.

When an event is received from an address that does not match an address in any Configuration Mapping, it will be considered a non-meter event. The event will be converted to a string event, and then handled as defined in the "Non-Meter Alarms and Events" setting located in the **EFM Meter Settings** property group of **Device Properties**.

EFM History Mapping

The History group is used to assign each float to an EFM attribute using the float's unique index. History data pulled from the device is in record form, with each record containing an array of data points. Each data point has a unique index or position in the array. The valid range is 0 to 31. For model-specific information, select a link from the list below.

EFM History Mapping - Gas Models
EFM History Mapping - Liquid Models

EFM History Mapping - Gas Models

The EFM History Mapping for gas models contains two property groups: Hourly History and Daily History.

Descriptions of the property groups are as follows:

- **Hourly History:** The indices assigned to the attributes in this property group will apply to the Hourly History Archive in all configured meters. It will be used when parsing records that are read from a meter's hourly archive.
- **Daily History:** The indices assigned to the attributes in this property group will apply to the Daily History Archive in all configured meters. It will be used when parsing records that are read from a meter's daily archive.

History Syntax

A History Index uses the following syntax: *N* or *S<N>* where:

- **N:** This index in the record associates with an attribute. The valid range is 0 to 31.
- **S:** This optional index indicates that the element comes from the Shared Archive configured in the EFM Meter Settings group.

  Note: GC data is typically from a shared archive, and will not be requested from the device when S indices are not used.

Static values use the following syntax: *!<static>* where:

- **! :** This character indicates that the subsequent entry is static for the associated attribute.
- **Static:** Static values are always considered floats.

Scale factors use the following syntax: *N/<scale factor>* or *S<N>/<scale factor>* where:

- **/:** This character indicates that the subsequent entry is a scale factor for the associated attribute. The value read from the device will be divided by the scale factor before it is passed to the EFM Exporter.
- **Scale factor:** Scale factors are always considered floats.

  Note: Scale factors cannot be used with the static syntax or non-numeric attributes.

Examples
1. A meter attribute with scale factor could be "0/120".
2. A shared attribute with scale factor could be "S0/3.5".
3. A static attribute with no scale factor allowed could be "!128".

See Also: EFM Meter Settings

Meter History Attributes & Mappings

The tables below list all the attributes available in the History Mapping, and includes their name, CSV name, data type, and description. Attributes that are left blank will be ignored.

Flow

Attribute	CSV Name	Data Type	Description
Flow Time	flow_time	Float	Flow time for this record in minutes.*
Average Pressure	avg_pressure	Float	Average pressure.**

Attribute	CSV Name	Data Type	Description
Average Temperature	avg_temp	Float	Average temperature over the flow time. Fahrenheit for English and Celsius for Metric.
Cumulative Volume	cumulative_volume	Float	Volume added during this interval for orifice and turbine meters. Units are controlled by the Volume Units in the Configuration Mapping.
Differential Pressure	diff_pressure	Float	Average differential pressure for orifice meters.**
Average Extension	avg_extension	Float	Average extension for orifice meters.**
C Prime	c_prime	Float	Orifice flow constant.
Average FPV	avg_fpv	Float	Average Super Compressibility Factor.
Pulses	pulses	Float	Pulses for turbine meters.
Raw Volume	raw_volume	Float	Raw volume for turbine meters. Units are controlled by the Volume Units in the Configuration Mapping.
Flowing Condition Factor	flowing_condition_factor	Float	Flowing Condition Factor for turbine meters.
Coriolis Raw Mass	coriolis_raw_mass	Float	Raw mass for coriolis meters. Units are pounds for English and KG for Raw Mass.
Corrected Mass	corrected_mass	Float	Corrected mass for coriolis meters. Units are pounds for English and KG for Metric.
Coriolis Average Meter Factor	coriolis_avg_meter_factor	Float	Average meter factor for coriolis meters.
Liquid Mass	liquid_mass	Float	Mass for liquid meters. Units are pounds for English and KG for Metric.
Liquid Volume	liquid_volume	Float	Volume for liquid meters. Units are controlled by the Volume Units in the Configuration Mapping.
Liquid Energy	liquid_energy	Float	Energy for liquid meters. Units are BTU/cubic foot for English and MJ/cubic meter for Metric.
Total Volume	total_volume	Float	Total volume. Units are controlled by the Volume Units in the Configuration Mapping.
Total Energy	total_energy	Float	Total energy.

Attribute	CSV Name	Data Type	Description
			Units are BTU/cubic foot for English and MJ/cubic meter for Metric.

*The Flow Time resolution in the Omni device is measured in half seconds. The driver will round the Flow Time value down to the nearest second.
**Inches of Water for English and Kilopascals for Metric.

Gas Composition (Mole%)

Attribute	CSV Name	Data Type	Description
Average BTU	avg_btu	Float	Average heating value. Units are Dekatherms for English and Gigajoules for Metric.
Average Specific Gravity	avg_specific_gravity	Float	Average specific gravity.
Average CO2	avg_co2	Float	%
Average N2	avg_n2	Float	%
Average C1	avg_c1	Float	%
Average C2	avg_c2	Float	%
Average C3	avg_c3	Float	%
Average ISOC4	avg_isoc4	Float	%
Average NC4	avg_nc4	Float	%
Average ISOC5	avg_isoc5	Float	%
Average NC5	avg_nc5	Float	%
Average NEOC5	avg_neoc5	Float	%
Average C6	avg_c6	Float	%
Average C7	avg_c7	Float	%
Average C8	avg_c8	Float	%
Average C9	avg_c9	Float	%
Average C10	avg_c10	Float	%
Average O2	avg_o2	Float	%
Average H2O	avg_h2o	Float	%
Average H2S	avg_h2s	Float	%
Average HE	avg_he	Float	%
Average H2	avg_h2	Float	%
Average CO	avg_co	Float	%
Average AR	avg_ar	Float	%
Specific Heat Ratio	specific_heat_ratio	Float	Ratio of specific heat.
Viscosity	viscosity	Float	Viscosity.

Attribute	CSV Name	Data Type	Description
			Units of Pounds/Mass per Foot/Second for English and Centipoises for Metric.

EFM History Mapping - Liquid Models

The EFM History Mapping for liquid models contains three property groups: Hourly History, Daily History, and Batch.

Descriptions of the property groups are as follows:

- **Hourly History:** The indices assigned to the attributes in this property group will apply to the Hourly History Archive in all configured meters. It is used when parsing records that are read from a meter's hourly archive.
- **Daily History:** The indices assigned to the attributes in this property group will apply to the Daily History Archive in all configured meters. It is used when parsing records that are read from a meter's daily archive.
- **Batch:** The indices assigned to the attributes in this property group will apply to the Batch Archive in all configured meters. It is used when parsing records that are read from a meter's batch archive. This property group is only available to liquid models.

Note: For more information on the hourly, daily, and batch archives, refer to **EFM Meter Settings**.

History Syntax

A History Index uses the following syntax: *N* or *S<N>* where:

- **N:** This index in the record associates with an attribute. The valid range is 0 to 31.
- **S:** This optional index indicates that the element comes from the Shared Archive configured in the EFM Meter Settings group.

  Note: GC data is typically from a shared archive, and will not be requested from the device when S indices are not used.

Static values use the following syntax: *!<static>* where:

- **!:** This character indicates that the subsequent entry is static for the associated attribute.
- **static:** Static values are always considered floats.

Scale factors use the following syntax: *N/<scale factor>* or *S<N>/<scale factor>* where:

- **/:** This character indicates that the subsequent entry is a scale factor for the associated attribute. The value read from the device will be divided by the scale factor before it is passed to the EFM Exporter.
- **Scale factor:** Scale factors are always considered floats.

  Note: Scale factors cannot be used with the static syntax, Batch Start attribute X:Y syntax, or non-numeric attributes.

Examples

1. A meter attribute with scale factor could be "0/120".
2. A shared attribute with scale factor could be "S0/3.5".
3. A static attribute with no scale factor allowed could be "!128".

Important: The Batch Start attribute has different syntax than the other attributes. Its syntax is *X*:*Y*, where *X* is the first index in the record that associates with Batch Start attribute and *Y* is the number of record indices that the driver will use (starting with the value *X*) when setting the Batch Start attribute. For example, the Batch Start attribute contains the date and time that the batch was started. The OMNI device stores that data in two different addresses that are read, concatenated, and then stored in the attribute. The default value is 0:2, which means that Index 0 must contain the batch start date and Index 1 must contain the batch start time. The Batch Start attribute always requires two registers, so the only *Y* value allowed is 2. The Batch Start attribute's index notation does not allow the static and shared options described above.

See Also: EFM Meter Settings

Meter History Attributes & Mappings for Hourly History and Daily History

The table below lists all the attributes available in the History Mapping, and includes their name, CSV name, data type, and description. Attributes that are left blank will be ignored.

Flow

Attribute	CSV Name	Data Type	Description
Flow Time	flow_time	Integer	Flow time for this record in minutes.*
K Factor	k_factor	Float	Average K factor over the flow time.**
Meter Factor (MF)	meter_factor	Float	Average meter factor over the flow time.
Specific Heat Ratio	ratio_of_specific_heats	Float	Ratio of specific heat.
Viscosity	viscosity	Float	Viscosity.**
Observed Density	liquid_observed_density	Float	Hydrometer reading.**
Density Temperature	liquid_density_temperature	Float	Density temperature.**
Density Pressure	liquid_density_pressure	Float	Density pressure.**
Uncorrected Density	liquid_uncorrected_density	Float	Uncorrected densitometer result.
Meter Flowing Density	liquid_meter_flowing_density	Float	Meter flowing density.**
Meter Temperature	liquid_meter_temperature	Float	Meter temperature.**
Meter Pressure	liquid_meter_pressure	Float	Meter pressure.**
Indicated Volume (IV)	liquid_indicated_volume	Float	Indicated volume.**
IV Index Start	liquid_iv_index_start	Float	IV index start.
IV Index End	liquid_iv_index_end	Float	IV index end.
Gross Volume (GV)	liquid_gross_volume	Float	Gross volume.**
GV Index Start	liquid_gv_index_start	Float	GV index start.
GV Index End	liquid_gv_index_end	Float	GV index end.
Gross Standard Volume (GSV)	liquid_gross_standard_volume	Float	Gross standard volume.**
GSV Index Start	liquid_gsv_index_start	Float	GSV index start.
GSV Index End	liquid_gsv_index_end	Float	GSV index end.

Attribute	CSV Name	Data Type	Description
Mass	liquid_mass	Float	Mass.**
Mass Index Start	liquid_mass_index_start	Float	Mass index start.
Mass Index End	liquid_mass_index_end	Float	Mass index end.
Net Standard Volume (NSV)	liquid_net_standard_ volume	Float	Net standard volume.**
NSV Index Start	liquid_nsv_index_start	Float	NSV index start.
NSV Index End	liquid_nsv_index_end	Float	NSV index end.
S&W Volume	liquid_sw_volume	Float	Sediment and water volume.
S&W Index Start	liquid_sw_index_start	Float	Sediment and water index start.
S&W Index End	liquid_sw_index_end	Float	Sediment and water index end.
S&W Percent	liquid_sw_percent	Float	Sediment and water percent.
S&W Correction	liquid_sw_correction	Float	Sediment and water correction.
Pulses	liquid_pulses	Float	Pulses for turbine meters.
Pulse Index Start	liquid_pulse_index_start	Float	Pulse index start.
Pulse Index End	liquid_pulse_index_end	Float	Pulse index end.
Orifice Differential	liquid_orifice_differential	Float	Orifice differential.
Orifice Extension	liquid_orifice_extension	Float	Orifice extension.
Orifice C Prime	liquid_orifice_c_prime	Float	Orifice flow constant.
Gas Equivalent Volume	liquid_gas_equivalent_ volume	Float	Gas equivalent volume.**
Gas Equivalent Energy	liquid_gas_equivalent_ energy	Float	Gas equivalent energy.**
Densitometer Factor	liquid_densitometer_ factor	Float	Average densitometer factor over the flow time.
Equilibrium Vapor Pressure	liquid_equilibrium_vapor_ pressure	Float	Equilibrium vapor pressure.
CTL	ctl	Float	Correction for the effect of temperature on a liquid.
CPL	cpl	Float	Correction for the effect of pressure on a liquid.
CTPL	ctpl	Float	Correction for the temperature and pressure of a liquid.
CCF	ccf	Float	Combined correction factor equals MF * CTL * CPL.

*The Flow Time resolution in the Omni device is measured in half seconds. The driver will round the Flow Time value down to the nearest second.
**Units depend on the Firmware revision.

Liquid Composition (Mole%)

Attribute	CSV Name	Data Type	Description
Average CO2	co2	Float	Mole %
Average N2	n2	Float	Mole %
Average C1	c1	Float	Mole %
Average C2	c2	Float	Mole %
Average C3	c3	Float	Mole %
Average ISOC4	ic4	Float	Mole %
Average NC4	nc4	Float	Mole %
Average ISOC5	ic5	Float	Mole %
Average NC5	nc5	Float	Mole %
Average NEOC5	neoc5	Float	Mole %
Average C6	c6	Float	Mole %
Average C7	c7	Float	Mole %
Average C8	c8	Float	Mole %
Average C9	c9	Float	Mole %
Average C10	c10	Float	Mole %
Average O2	o2	Float	Mole %
Average H2O	h2o	Float	Mole %
Average H2S	h2s	Float	Mole %
Average HE	he	Float	Mole %
Average H2	h2	Float	Mole %
Average CO	co	Float	Mole %
Average AR	ar	Float	Mole %
Average Ethylene	ethylene	Float	Mole %
Average Propylene	propylene	Float	Mole %

Meter History Attributes & Mappings for Batch History

Flow

Attribute	CSV Name	Data Type	Description
Batch Start	when_start	Integer	Start time of the batch.
Batch Type	batch_type	Integer	Type of batch. Options include Unknown, Normal, Maintenance, and Unauthorized.
Batch ID	batch_id	String	Batch ID.
Report Number	report_number	Integer	Batch report number.
MF is in GSV	mf_in_gsv	Char	Meter factor is in the gross standard volume calculation.
Observed Density	observed_density	Float	Hydrometer reading or corrected densitometer result.
Density Temperature	density_temp	Float	Density temperature.*

Attribute	CSV Name	Data Type	Description
Density Pressure	density_pressure	Float	Density pressure.*
Density Correction Factor	dcf	Float	This is also known as the Pyc factor or Density Meter Factor (DMF).
Uncorrected Density	uncorrected_density	Float	Uncorrected densitometer result.
Meter Flowing Density	meter_flowing_density	Float	Meter flowing density.*
Meter Temperature	meter_temp	Float	Meter temperature.*
Meter Pressure	meter_pressure	Float	Meter pressure.*
Meter Factor	meter_factor	Float	Average meter factor for the batch.
K Factor	k_factor	Float	Average K factor for the batch.*
Equilibrium Vapor Pressure	evp	Float	Equilibrium vapor pressure.
CTL	ctl	Float	Correction for the effect of temperature on a liquid.
CPL	cpl	Float	Correction for the effect of pressure on a liquid.
CTPL	ctpl	Float	Correction for the temperature and pressure of a liquid.
CCF	ccf	Float	Combined correction factor equals MF * CTL * CPL.
Liquid Product Name	liquid_product_name	String	Product name for the batch.
Pulses	pulses	Float	Pulses for turbine meters.
Pulse Index Start	pulse_index_start	Float	Pulse index start.
Pulse Index End	pulse_index_end	Float	Pulse index end.
Orifice Differential	orifice_differential	Float	Orifice differential.
Orifice Extension	orifice_extension	Float	Orifice extension.
Orifice C Prime	orifice_c_prime	Float	Orifice flow constant.
Indicated Volume (IV)	iv	Float	Indicated volume.*
IV Index Start	iv_index_start	Float	IV index start.
IV Index End	iv_index_end	Float	IV index end.
Gross Volume (GV)	gv	Float	Gross volume.*
GV Index Start	gv_index_start	Float	GV index start.
GV Index End	gv_index_end	Float	GV index end.
Gross Standard Volume (GSV)	gsv	Float	Gross standard volume.*

Attribute	CSV Name	Data Type	Description
GSV Index Start	gsv_index_ start	Float	GSV index start.
GSV Index End	gsv_index_end	Float	GSV index end.
Net Standard Volume (NSV)	nsv	Float	Net standard volume.*
NSV Index Start	nsv_index_ start	Float	NSV index start.
NSV Index End	nsv_index_end	Float	NSV index end.
S&W Volume	sw	Float	Sediment and water volume.
S&W Index Start	sw_index_start	Float	Sediment and water index start.
S&W Index End	sw_index_end	Float	Sediment and water index end.
S&W Percent	sw_percent	Float	Sediment and water percent.
S&W Correction	sw_correction	Float	Sediment and water correction.
Mass	mass	Float	Mass.*
Mass Index Start	mass_index_ start	Float	Mass index start.
Mass Index End	mass_index_ end,	Float	Mass index end.
Gas Equivalent Volume	gas_eq_ volume	Float	Gas equivalent volume.*
Gas Equivalent Energy	gas_eq_ energy	Float	Gas equivalent energy.*
Viscosity	viscosity	Float	Viscosity.*
Specific Heat Ratio	specific_heat_ ratio	Float	Ratio of specific heat.

*Units depend on the Firmware revision.

Liquid Composition (Mole%)

Attribute	CSV Name	Data Type	Description
Average CO2	avg_co2	Float	Mole %
Average N2	avg_n2	Float	Mole %
Average C1	avg_c1	Float	Mole %
Average C2	avg_c2	Float	Mole %
Average C3	avg_c3	Float	Mole %
Average ISOC4	avg_isoc4	Float	Mole %
Average NC4	avg_nc4	Float	Mole %
Average ISOC5	avg_isoc5	Float	Mole %
Average NC5	avg_nc5	Float	Mole %
Average NEOC5	avg_neoc5	Float	Mole %

Attribute	CSV Name	Data Type	Description
Average C6	avg_c6	Float	Mole %
Average C7	avg_c7	Float	Mole %
Average C8	avg_c8	Float	Mole %
Average C9	avg_c9	Float	Mole %
Average C10	avg_c10	Float	Mole %
Average Ethylene	ethylene	Float	Mole %
Average Propylene	propylene	Float	Mole %
Average H2	avg_h2	Float	Mole %
Average CO	avg_co	Float	Mole %
Average AR	avg_ar	Float	Mole %
Average O2	avg_o2	Float	Mole %
Average H2O	avg_h2o	Float	Mole %
Average H2S	avg_h2s	Float	Mole %
Average HE	avg_he	Float	Mole %

EFM Cache

The OMNI Flow Computer Driver caches EFM data per device. During polls, the driver will only request new data from the device and then add it to its local cache. This minimizes communication between the physical device and the driver. The cache that is maintained by the driver will be cleared under the following scenarios:

1. The server is reinitialized, restarted, or a new project is loaded.
2. The channel or device is deleted.
3. The cache is cleared through the **Clear Cache** property located in EFM Meter Settings.
4. A meter's Archive Number changes.
5. The Max History Archive Size changes.

See Also: EFM Meter Settings

CSV Import/Export

The EFM Mappings support the import and export of data in a Comma Separated Variable (CSV) file. CSV import and export supports the efficient configuration of many devices. For more information on a specific aspect of CSV Import/Export, select a link from the list below.

Creating a Template
Exporting EFM Mappings
Importing EFM Mappings
Using Other Characters as the Delimiter

Creating a Template

The easiest way to create an import CSV file is to create a template. For more information, refer to the instructions below.

1. To start, create a new device using the default settings. Then, click **OK**.
2. Next, right-click on the device and select **Properties** | **EFM Mapping**. Then, click **Export**.
3. Save the file to an accessible location.
4. Use the exported template in a spreadsheet application that supports CSV files, and then modify the file as desired.

Note: Microsoft Excel is an excellent tool for editing large groups of tags outside the server. Once a template CSV file has been exported, it can be loaded directly into Excel for editing.

Exporting EFM Mappings

Exporting an EFM Mapping will generate a CSV text file that contains sections for History and Alarms. Each section has a heading record followed by a record for each item. Column names must match those listed; however, columns may be in any order.

Note: For Gas models, the CSV file will contain two sections for Hourly History and Daily History mapping data. For Liquid models, the CSV file will contain three sections for Hourly History, Daily History, and Batch History mapping data. CSV files cannot contain data for both.

History Mapping

The table below displays the History Mapping for the Hourly History, Daily History, and Batch History mapping data. The required columns are listed in **bold**.

Column Name	Description
Attribute	This is the name of the History Mapping attribute. Attributes can be in any order. Attributes that are not included on an import will be left blank in the mapping. **Note:** All possible attribute names are listed in the History group. *For more information, refer to* ***EFM History Mapping****.*
Value	This is the address syntax for the attribute. It can be blank, static, or take the form *S<N>*. For the Batch Start attribute available in liquid models, the syntax is *X:Y*. For more information, refer to **EFM History Mapping - Liquid Models**. **Note:** For information on each attribute's value limitations, refer to **EFM History Mapping**.

Alarm Mapping

The required columns are listed in **bold**.

Column Name	Description
Address	This is the address of the alarm. It can take the form *1N34.* For more information, refer to **EFM Alarm Mapping**.
Alarm Type	This is the type of the alarm. The default is Differential Pressure. The valid types are as follows: 1 = Differential Pressure 2 = Static Pressure

Column Name	Description
	3 = Temperature 4 = Cutoff 5 = Backflow 6 = Battery
Alarm State	This is the state of the alarm. The default is Off. 1 = Off 2 = On 3 = Hi 4 = Lo

Importing EFM Mappings

Once the CSV file has been created and exported, it may be re-imported into an EFM Mapping. To do so, open **EFM Mapping** and then click **Import**.

Note: For History and Alarms, importing will replace all existing settings with the settings specified in the CSV file. When the import is complete, the configured mapping should match one for one with the file.

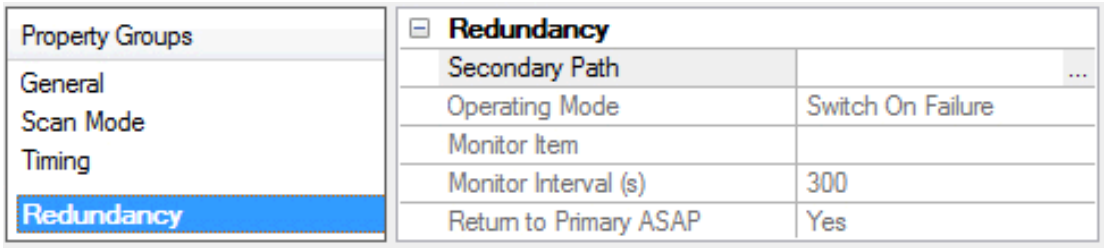
Using Other Characters as the Delimiter

When utilizing a CSV file that does not use a comma or semi-colon delimiter, users should do one of the following:

- Save the project in XML. Then, perform mass configuration on the XML file instead of using CSV.
- Perform a search-and-replace on the delimiter in the CSV file and then replace the delimiter with a comma or semicolon. The delimiter being used by the OPC server (either comma or semicolon) must be set to the replacement character.

Note: For information on specifying which character to use as the variable (comma or semicolon), refer to "Options - General" in the server help file.

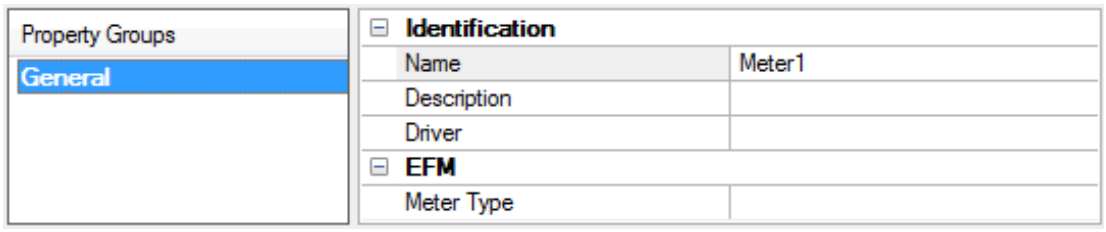
Device Properties — Redundancy



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

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

Meters



Identification

Name: Specify the name of the meter. It is a user-defined name that can be up to 256 characters long.

Description: Enter a brief description of the meter to help identify it in data and reports.

Driver: Verify the meter displays the correct driver.

EFM

Meter Type: Select the most appropriate type of meter for the hardware and data from the physical meter; typically liquid or gas.

Note: EFM properties are not available in all drivers.

Data Types Descriptions

Data Type	Description
Boolean	Single 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
BCD	Two byte packed BCD Value range is 0-9999. Behavior is undefined for values beyond this range.
LBCD	Four byte packed BCD Value range is 0-99999999. Behavior is undefined for values beyond this range.
String	Null terminated ASCII string Supports 8 byte and 16 byte string data.
Float*	32 bit floating point value The driver interprets two consecutive registers as a single precision value by making the last register the high word and the first register the low word.
Float Example	If register 40001 is specified as a float, bit 0 of register 40001 would be bit 0 of the 32 bit data type and bit 15 of register 40002 would be bit 31 of the 32 bit data type.

Address Descriptions

The default data types are shown in **bold**.

Note: The address ranges provided in the tables below are not necessarily supported on all Firmware revisions.

Address	Range	Data Type	Access
Digital I/O Point	1001-1024	**Boolean**	Read/Write
Programmable Boolean Point	1025-1088	**Boolean**	Read/Write
Programmable Accumulator Points	1089-1099	**Boolean**	Read/Write
Meter Run Status and Alarm Points	1n01-1n99 n = Number of Meter Run	**Boolean**	Read/Write
User Scratch Pad Boolean Points	1501-1599 1601-1649	**Boolean**	Read/Write
User Scratch Pad One Shot Points	1650-1699	**Boolean**	Read/Write
Command Boolean Points/Variables	1700-1799	**Boolean**	Read/Write
Meter Station Alarm and Status Points	1801-1899	**Boolean**	Read/Write
Prover Alarm and Status Points	1901-2099	**Boolean**	Read/Write
Meter Totalizer Roll-Over Flags	2n01-2n84 n = Number of Meter Run	**Boolean**	Read/Write
Misc. Meter Station Alarm and Status	2601-2660	**Boolean**	Read/Write
Miscellaneous Boolean Points	2701-2799	**Boolean**	Read/Write
Station Totalizer Roll-over Flags	2801-2851	**Boolean**	Read/Write
Station Totalizer Decimal Resolution Flags	2852-2862	**Boolean**	Read/Write
Status Booleans Relating to Redundant Flow Computer Systems	2863-2864	**Boolean**	Read/Write
Boolean Command Outputs and Status Points used for Meter Tube Switching	2877-2896	**Boolean**	Read/Write

16 Bit Integer Data Addresses

These addresses support bit level access. For more information, refer to "Zero vs One Based Bit Addressing Within Registers" in **Data Access**.

Address	Range	Data Type	Access
Custom Data Packet #1	3001-3040	**Short**, Word, BCD	Read/Write
Custom Data Packet #2	3041-3056	**Short**, Word, BCD	Read/Write
Custom Data Packet #3	3057-3096	**Short**, Word, BCD	Read/Write
Trapil function related data	3665-3699	**Short**, Word, BCD	Read/Write

Address	Range	Data Type	Access
Misc. 16 bit Integer Data	3097-3099 3737-3799 3880-3899	**Short**, Word, BCD	Read/Write
Meter Run 16 Bit Integer Data	3n01-3n99 n = Number of Meter Run	**Short**, Word, BCD	Read/Write
Scratch Pad 16 Bit Integer Data	3501-3599	**Short**, Word, BCD	Read/Write
User Display #1	3601-3608	**Short**, Word, BCD	Read/Write
User Display #2	3609-3616	**Short**, Word, BCD	Read/Write
User Display #3	3617-3624	**Short**, Word, BCD	Read/Write
User Display #4	3625-3632	**Short**, Word, BCD	Read/Write
User Display #5	3633-3640	**Short**, Word, BCD	Read/Write
User Display #6	3641-3648	**Short**, Word, BCD	Read/Write
User Display #7	3649-3656	**Short**, Word, BCD	Read/Write
User Display #8	3657-3664	**Short**, Word, BCD	Read/Write
Access Raw Data Archive Records	3701-3736	**Short**, Word, BCD	Read/Write
Honeywell Multivariable 16 Bit Integer Data	3753-3793	**Short**, Word, BCD	Read/Write
Meter Station 16 Bit Integer Data	3800-3842	**Short**, Word, BCD	Read/Write
Danalyzer Gas Chromatograph Data	3843-3864	**Short**, Word, BCD	Read/Write
Flow Computer Time and Date Variables	3867-3879	**Short**, Word, BCD	Read/Write
Prover 16 Bit Integer Data	3901-3944	**Short**, Word, BCD	Read/Write

8 Character ASCII String Data

Address	Range	Data Type	Access
Meter Run ASCII Data	4n01-4n43 n = Number of Meter Run	**String**	Read/Write
Scratch Pad ASCII Data	4501-4599	**String**	Read/Write

Address	Range	Data Type	Access
User Display Definition Variables	4601-4640	**String**	Read/Write
Station Auxiliary Input Variables	4707-4714	**String**	Read/Write
Meter Station ASCII Data	4801-4850	**String**	Read/Write
Prover ASCII String Data	4901-4999	**String**	Read/Write

32 Bit Integer Data

These addresses support bit level access. For more information, refer to "Zero vs One Based Bit Addressing Within Registers" in **Data Access**.

Address	Range	Data Type	Access
Meter Run 32 Bit Integer Data	5n01-5n99 n = Number of Meter Run	**Long**, DWord, LBCD, Float	Read/Write
Scratch Pad 32 Bit Integer Data	5501-5599	**Long**, DWord, LBCD, Float	Read/Write
Product 32 Bit integer data	5601-5799	**Long**, DWord, LBCD, Float	Read/Write
Station 32 Bit Integer Data	5801-5891	**Long**, DWord, LBCD, Float	Read/Write
Prover 32 Bit Integer Data	5901-5999	**Long**, DWord, LBCD, Float	Read/Write
Meter Run Premium Level 32 Bit Integer Data	6n01-6n99	**Long**, DWord, LBCD, Float	Read/Write
Station Premium Level 32 Bit Integer Data	6801-6855	**Long**, DWord, LBCD, Float	Read/Write

32 Bit IEEE Floating Point Data

These addresses support bit level access. For more information, refer to "Zero vs One Based Bit Addressing Within Registers" in **Data Access**.

Address	Range	Data Type	Access
Digital to Analog Outputs	7001-7024	**Float**, Long, DWord, LBCD	Read/Write
User Variables	7025-7088	**Float**, Long, DWord, LBCD	Read/Write
Programmable Accumulator	7089-7099	**Float**, Long, DWord, LBCD	Read/Write
Meter Run Data	7n01 - 7n99 n = Number of Meter Run	**Float**, Long, DWord, LBCD	Read/Write
Scratch Pad Data	7501-7599	**Float**, Long, DWord, LBCD	Read/Write

Address	Range	Data Type	Access
PID Control Data	7601-7623	**Float**, Long, DWord, LBCD	Read/Write
Miscellaneous Meter Run Data	7624-7699	**Float**, Long, DWord, LBCD	Read/Write
Miscellaneous Variables	7701-7778	**Float**, Long, DWord, LBCD	Read/Write
Meter Station Data	7801-7899	**Float**, Long, DWord, LBCD	Read/Write
Prover Data	7901-8499	**Float**, Long, DWord, LBCD	Read/Write
Miscellaneous Meter Run #1	8501-8599	**Float**, Long, DWord, LBCD	Read/Write
Miscellaneous Meter Run #2	8601-8699	**Float**, Long, DWord, LBCD	Read/Write
Miscellaneous Meter Run #3	8701-8799	**Float**, Long, DWord, LBCD	Read/Write
Miscellaneous Meter Run #4	8801-8899	**Float**, Long, DWord, LBCD	Read/Write
Station Previous Batch Average Data	8901-8999	**Float**, Long, DWord, LBCD	Read/Write

16 Bit Integer Configuration Data

These addresses support bit level access. For more information, refer to "Zero vs One Based Bit Addressing Within Registers" in **Data Access**.

Address	Range	Data Type	Access
Meter Run #1	13001-13013	**Short**, Word, BCD	Read/Write
Meter Run #2	13014-13026	**Short**, Word, BCD	Read/Write
Meter Run #3	13027-13039	**Short**, Word, BCD	Read/Write
Meter Run #4	13040-13052	**Short**, Word, BCD	Read/Write
Meter Run Configuration 16 Bit Integer Data	13053-13073 13300-13499	**Short**, Word, BCD	Read/Write
General Flow Configuration	13074-13084	**Short**, Word, BCD	Read/Write
Serial Port Configuration	13085-13128	**Short**, Word, BCD	Read/Write
PID Configuration	13129-13160	**Short**, Word, BCD	Read/Write
PLC Data	13161-13299	**Short**, Word, BCD	Read/Write
Peer to Peer Setup	13300-13477	**Short**, Word, BCD	Read/Write

16 Character ASCII String Data

Address	Range	Data Type	Access
Flow Computer Configuration	14001-14499	**String**	Read/Write

32 Bit Integer Data

These addresses support bit level access. For more information, refer to "Zero vs One Based Bit Addressing Within Registers" in **Data Access**.

Address	Range	Data Type	Access
Flow Computer Configuration	15001-16999	**Long**, DWord, LBCD, Float	Read/Write

32 Bit IEEE Floating Point Data

These addresses support bit level access. For more information, refer to "Zero vs One Based Bit Addressing Within Registers" in **Data Access**.

Address	Range	Data Type	Access
Flow Computer Configuration	17001-19999	**Float**, Long, DWord, LBCD	Read/Write

Array Support

Arrays are supported for register locations for all data types except for strings. Arrays are also supported for input and output coils (Boolean data types). There are two methods of addressing an array. The following examples use register locations:

3*xxx* [rows] [cols]
3*xxx* [cols] this method assumes rows is equal to one.

For arrays, rows multiplied by cols cannot exceed the maximum number of registers or Booleans (depending on the data type) that can be read by the protocol in a single access.

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 size is out of range for address '<address>'
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
Received block length of '<received length>' does not match expected length of '<expected length>' for address '<address>' on device '<device>'

Device Status Messages

Device '<device name>' is not responding
Unable to write to address '<address>' on device '<device>': Device responded with exception code '<code>'
Unable to write to '<address>' on device '<device name>'
Write failed for '<tag name>' on device '<device name>'. Maximum path length of '<number>' characters exceeded

OMNI Flow Computer Error Messages

'<device name>' - A starting address of '<address>' in the archive's record structure is invalid. May not receive data for address '<address>'
'<device name>' - Alarm record parse for device failed
'<device name>' - Archive '<archive number>' is not configured correctly for address '<address>'. Max number of records is zero
'<device name>' - Archive record contains an invalid address. BOOL, 8-byte strings, and 16-byte strings are not supported
'<device name>' - Archive record for address '<address>' contains an unexpected number of bytes. Expected '<number of bytes>' bytes, received '<number of bytes>' bytes
'<device name>' - Config data attribute for meter tap location read from device address '<address>' is '<value>', and does not map to any valid meter tap locations. Expecting 0 for flange, or 1 for pipe. Defaulting to flange
'<device name>' - Config data attribute for meter type read from device address '<address>' is '<value>', and does not map to any valid meter types. Expected values are 0, 2, or 3 for orifice, 1 for turbine, 4 or 8 for ultra sonic, and 5 for vcone. Defaulting to orifice
'<device name>' - Config data attribute for static pressure tap read from device address '<address>' is '<value>', and does not map to any valid static pressure tap locations. Expecting 0 for up, or 1 for down. Defaulting to up

'<device name>' - Config data attribute for static pressure unit read from device address '<address>' is '<value>', and does not map to a valid pressure unit. Expecting 0 for kPa, 1 for Bar, or 2 for kg/cm2. Defaulting to kPa
'<device name>' - Config data attribute for totalizer digits read from the device address '<address>' is '<value>', and does not map to a valid number of totalizer digits. Expecting 0 for 9 digits, or 1 for 8 digits. Defaulting to 9 digits
'<device name>' - Date format for address '<address>' is invalid. Device returned '<value>', valid values are 0 or 1
'<device name>' - Device Firmware version '<Firmware version>' is not supported by the '<model name>' model
'<device name>' - Device password write not successful. Value in response is different from the written value
'<device name>' - Event record parse for device failed
'<device name>' - Failed to read EFM pointer file. <Extended Error>
'<device name>' - Failed to write EFM pointer file. <Extended Error>
'<device name>' - Meter and shared archives are not in sync. Records will only contain flow data (no analysis)
'<device name>' - Meter archive record parse failed
'<device name>' - Read invalid Firmware version '<Firmware version>' from address '<address>', config upload complete
'<device name>' - Shared '<archive type>' archive address is not configured, records will only contain flow data (no analysis)
'<device name>' - Shared archive record parse failed
'<device name>' - The <archive type> mapping contains more configured attributes than the device. Some attributes will not contain valid data
'<device name>' - The max alarm archive size was changed from '<previous size>' to '<current size>'
'<device name>' - The max event archive size was changed from '<previous size>' to '<current size>'
'<device name>' - Time sync write not successful. Value in response is different from the written value
'<device name>' - Unable to read '<number of registers>' registers in config register block at address '<address>'
'<device name>' - Unable to read date format register for address '<address>'. Response is not the correct size
'<device name>' - Unable to read index registers. Response is not the correct size
'<device name>' - Unable to read record format registers for address '<address>'. Response is not the correct size
'<device name>' - Unable to write requested record register for address '<address>'. Response is not the correct size
'<device name>' - Unable to write requested record register for address '<address>'. Wrote '<value>', read back '<value>'
Alarm mapping for address '<address>' is invalid and will be ignored
Alarm state for address '<address>' is invalid. Setting the state to <state>
Alarm type for address '<address>' is invalid. Setting the type to <type>

Bad address in block [<start address> to <end address>] on device '<device name>'
Bad array spanning [<address> to <address>] on device '<device>'
Device password invalid for device '<device name>'
Device password write for device '<device name>' was successful
History attribute '<attribute index>' is unknown and will be ignored
History mapping for attribute '<attribute name>' is invalid and will be ignored
Received "needs password" exception from device '<device name>' with 'fail after successive timeouts' set to 1. Set the 'fail after successive timeouts' setting to a value greater than 1 and verify that the 'device password' setting is correct
Serialization of EFM data to temporary file '<file name>' failed. Reason: '<file I/O error>'
The '<archive type>' archive number for meter '<meter name>' is already being used. XML project load not successful
The shared '<archive type>' archive number is already in use by another meter. XML project load not successful
Unable to read '<address>' from device '<device name>'. The device is configured for broadcast writes only
Unable to read block address ['<start address>' to '<end address>'] on device '<device name>'. Unexpected characters in response
Warning loading '<mapping type>' mapping from CSV. '<warning type>'

Serial Communications

Communications error on '<channel name>' [<error mask>]
COMn does not exist
COMn is in use by another application
Error opening COMn
Unable to set comm parameters on COMn

See Also: Modbus Exception Codes

Modbus Exception Codes

The following data is from Modbus Application Protocol Specifications documentation.

Code Dec/Hex	Name	Meaning
01/0x01	ILLEGAL FUNCTION	The function code received in the query is not an allowable action for the server (or slave). This may be because the function code is only applicable to newer devices, and was not implemented in the unit selected. It could also indicate that the server (or slave) is in the wrong state to process a request of this type, for example, because it is unconfigured and is being asked to return register values.
02/0x02	ILLEGAL DATA ADDRESS	The data address received in the query is not an allowable address for the server (or slave). More specifically, the combination of reference number and transfer length is invalid. For a controller with 100 registers, a request with offset 96 and length 4 would succeed. A request with offset 96 and length 5 will generate exception 02.

Code Dec/Hex	Name	Meaning
03/0x03	ILLEGAL DATA VALUE	A value contained in the query data field is not an allowable value for server (or slave). This indicates a fault in the structure of the remainder of a complex request, such as that the implied length is incorrect. It specifically does not mean that a data item submitted for storage in a register has a value outside the expectation of the application program, since the MODBUS protocol is unaware of the significance of any particular value of any particular register.
04/0x04	SLAVE DEVICE FAILURE	An unrecoverable error occurred while the server (or slave) was attempting to perform the requested action.
05/0x05	ACKNOWLEDGE	The slave has accepted the request and is processing it, but a long duration of time will be required to do so. This response is returned to prevent a timeout error from occurring in the master. The master can next issue a Poll Program Complete message to determine if processing is completed.
06/0x06	SLAVE DEVICE BUSY	The slave is engaged in processing a long-duration program command. The master should retransmit the message later when the slave is free.
07/0x07	NEGATIVE ACKNOWLEDGE	The slave cannot perform the program function received in the query. This code is returned for an unsuccessful programming request using function code 13 or 14 decimal. The master should request diagnostic or error information from the slave.
08/0x08	MEMORY PARITY ERROR	The slave attempted to read extended memory, but detected a parity error in the memory. The master can retry the request, but service may be required on the slave device.
10/0x0A	GATEWAY PATH UNAVAILABLE	Specialized use in conjunction with gateways indicates that the gateway was unable to allocate an internal communication path from the input port to the output port for processing the request. This usually means that the gateway is misconfigured or overloaded.
11/0x0B	GATEWAY TARGET DEVICE FAILED TO RESPOND	Specialized use in conjunction with gateways indicates that no response was obtained from the target device. This usually means that the device is not present on the network.

Note: For this driver, the terms Slave and Unsolicited are used interchangeably.

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

Error Type:
Warning

Possible Cause:
A tag address that has been specified statically references a location that is beyond the range of supported locations for the device.

Solution:
Verify that the address is correct; if it is not, re-enter it in the client application.

Array size is out of range for address '<address>'

Error Type:
Warning

Possible Cause:
A tag address that has been specified statically is requesting an array size that is too large for the address type or block size of the driver.

Solution:
Re-enter the address in the client application to specify a smaller value for the array or a different starting point.

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.

Received block length of '<received length>' does not match expected length of '<expected length>' for address '<address>' on device '<device>'

Error Type:
Warning

Possible Cause:
The driver attempted to read a block of memory but the PLC did not provide the driver with the requested size of data. No error code was returned.

Solution:
N/A

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 communications 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 setting.

Solution:

1. Verify the cabling between the PC and the PLC device.
2. Verify the specified communications properties match those of the device.
3. Verify the Network ID given to the named device matches that of the actual device.
4. Increase the Request Timeout setting so that the entire response can be handled.

Unable to write to '<address>' on device '<device name>'

Error Type:
Serious

Possible Cause:

1. The serial connection between the device and the Host PC is broken.
2. The communications properties for the serial connection are incorrect.
3. The named device may have been assigned an incorrect Network ID.

Solution:

1. Verify the cabling between the PC and the PLC device.
2. Verify the specified communications properties match those of the device.
3. Verify that the Network ID given to the named device matches that of the actual device.

Unable to write to address '<address>' on device '<device>': Device responded with exception code '<code>'

Error Type:
Warning

Possible Cause:
See **Modbus Exception Codes** for a description of the exception code.

Solution:
See **Modbus Exception Codes**.

Write failed for '<tag name>' on device '<device name>'. Maximum path length of '<number>' exceeded

Error Type:
Warning

Possible Cause:
Path length is limited to the indicated number of characters.

Solution:
Devise a shorter path.

'<device name>' - A starting address of '<address>' in the archive's record structure is invalid. May not receive data for address '<address>'

Error Type:
Warning

Possible Cause:

1. The Config Address property (located in the **EFM Meter Settings** property group of **Device Properties**) is configured incorrectly.
2. The archive record structure's configuration is invalid.

Solution:

1. Verify that the Config Address property is configured using the base address of the raw data archive record structure section from the Omni memory map. If unsure of the value, use the Config Address's default value.
2. Verify that the record structure for the archive associated with the address in the message is valid.

'<device name>' - Alarm record parse for device failed

Error Type:
Warning

Possible Cause:

1. A failure occurred when parsing the alarm record for EFM attribute data.
2. The device's archive memory may have been cleared since the last upload.

Solution:

1. Ensure that the time and date have been set correctly.
2. Verify the cabling between the PC and the PLC device.
3. Verify that the specified communications properties match those of the device.

'<device name>' - Archive '<archive number>' is not configured correctly for address '<address>'. Max number of records is zero

Error Type:
Warning

Possible Cause:
The meter's archive configuration is invalid.

Solution:
Verify that the configuration of the archive's group and max records are correct.

'<device name>' - Archive record contains an invalid address. BOOL, 8-byte strings, and 16-byte strings are not supported

Error Type:
Warning

Possible Cause:
The meter's archive group configuration is invalid.

Solution:
Ensure that the archive's group configuration does not contain any addresses that correspond to Boolean or String data.

'<device name>' - Archive record for address '<address>' contains an unexpected number of bytes. Expected '<number of bytes>' bytes, received '<number of bytes>' bytes

Error Type:
Warning

Possible Cause:

1. The Config Address property (located in the **EFM Meter Settings** property group of **Device Properties**) is configured incorrectly.
2. The archive record structure's configuration is invalid.
3. The serial connection between the device and the Host PC is bad.
4. The communications properties for the serial connection are incorrect.

Solution:

1. Verify that the Config Address property is configured using the base address of the raw data archive record structure section from the Omni memory map. If unsure of the value, use the Config Address's default value.
2. Verify that the record structure for the archive associated with the address in the message is valid.

3. Verify the cabling between the PC and the PLC device.
4. Verify that the specified communications properties match those of the device.

'<device name>' - Config data attribute for meter tap location read from device address '<address>' is '<value>', and does not map to any valid meter tap locations. Expecting 0 for flange, or 1 for pipe. Defaulting to flange

Error Type:
Warning

Possible Cause:
The meter tap location register contains an invalid value.

Solution:
Set the value to one of the expected values from the error string.

'<device name>' - Config data attribute for meter type read from device address '<address>' is '<value>', and does not map to any valid meter types. Expected values are 0, 2, or 3 for orifice, 1 for turbine, 4 or 8 for ultra sonic, and 5 for vcone. Defaulting to orifice

Error Type:
Warning

Possible Cause:
The meter type register contains an invalid value.

Solution:
Set the value to one of the expected values from the error string.

'<device name>' - Config data attribute for static pressure tap read from device address '<address>' is '<value>', and does not map to any valid static pressure tap locations. Expecting 0 for up, or 1 for down. Defaulting to up

Error Type:
Warning

Possible Cause:
The static pressure tap location register contains an invalid value.

Solution:
Set the value to one of the expected values from the error string.

'<device name>' - Config data attribute for static pressure unit read from device address '<address>' is '<value>', and does not map to a valid pressure unit. Expecting 0 for kPa, 1 for Bar, or 2 for kg/cm2. Defaulting to kPa

Error Type:
Warning

Possible Cause:
The static pressure unit register contains an invalid value.

Solution:
Set the value to one of the expected values from the error string.

'<device name>' - Config data attribute for totalizer digits read from the device address '<address>' is '<value>', and does not map to a valid number of totalizer digits. Expecting 0 for 9 digits, or 1 for 8 digits. Defaulting to 9 digits

Error Type:
Warning

Possible Cause:
The totalizer digits register contains an invalid value.

Solution:
Set the value to one of the expected values from the error string.

'<device name>' - Date format for address '<address>' is invalid. Device returned '<value>', valid values are 0 or 1

Error Type:
Warning

Possible Cause:
The date format register contains an invalid value.

Solution:
Set the value to one of the expected values from the error string.

'<device name>' - Device Firmware version '<Firmware version>' is not supported by the '<model name>' model

Error Type:
Warning

Possible Cause:
The device is running a Firmware version that is not supported by the chosen model.

Solution:
Download one of the supported Firmware versions to the device or select the appropriate model.

'<device name>' - Device password write not successful. Value in response is different from the written value

Error Type:
Warning

Possible Cause:
1. The serial connection between the device and the Host PC is bad.
2. The communications properties for the serial connection are incorrect.

Solution:
1. Verify the cabling between the PC and the PLC device.
2. Verify that the specified communications properties match those of the device.

'<device name>' - Event record parse for device failed

Error Type:
Warning

Possible Cause:
1. A failure occurred when parsing the event record for EFM attribute data.
2. The device's archive memory may have been cleared since the last upload.

Solution:
1. Ensure that the time and date have been set correctly.
2. Verify the cabling between the PC and the PLC device.
3. Verify that the specified communications properties match those of the device.

'<device name>' - Failed to read EFM pointer file. <Extended Error>

Error Type:
Warning

Extended Error:
When supplied by the operating system, this describes the file error that occurred.

Possible Cause:
1. A permission error was encountered when the EFM pointer cache was read.
2. The EFM pointer cache file is corrupt.

Solution:
The driver will automatically generate a new EFM pointer file; however, the server will re-poll (uploading all EFM data) during the next EFM poll for meters in the device.

Note:
For more information, refer to the extended error.

'<device name>' - Failed to write EFM pointer file. <Extended Error>

Error Type:
Warning

Extended Error:
When supplied by the operating system, this describes the file error that occurred.

Possible Cause:
1. The disk is full.
2. A permission error was encountered when the EFM pointer cache was written.

Solution:
The server will attempt to update the EFM pointer file periodically, in addition to when the server is shut-down. If the pointer file cannot be written, the server will re-poll (uploading all EFM data) during the next EFM poll for meters in the device.

Note:
For more information, refer to the extended error.

'<device name>' - Meter and shared archives are not in sync. Records will only contain flow data (no analysis)

Error Type:
Warning

Possible Cause:
1. The number of records in the meter and shared archives are not equal.
2. The timestamps of the records located at the same index are not equal.
3. Records that had not yet been uploaded by the server were overwritten in the device by newer ones.

Solution:
1. Ensure that the shared archive and all meter archives are configured to use the same trigger.
2. Ensure that the values in the Current Record Number registers for all meter archives and the shared archive are the same.

3. Increase the frequency of the meter's EFM poll to ensure that unconsumed records in the device will not be overwritten.

'<device name>' - Meter archive record parse failed

Error Type:
Warning

Possible Cause:

1. A failure occurred when parsing the meter history record for EFM attribute data.
2. The device's archive memory may have been cleared since the last upload.

Solution:

1. Verify that the meter's archive configuration is correct, and that it matches the History Mapping.
2. Ensure that the time and date have been set correctly.

'<device name>' - Read invalid Firmware version '<Firmware version>' from address '<address>', config upload complete

Error Type:
Warning

Possible Cause:
An unsupported Firmware version is installed on the device.

Solution:
Download one of the supported Firmware versions.

Note:
For a list of supported Firmware versions, refer to **Device Setup**.

'<device name>' - Shared '<archive type>' archive address is not configured, records will only contain flow data (no analysis)

Error Type:
Warning

Possible Cause:
The shared hourly, daily, or batch archive number (located in the **EFM Meter Settings** property group of **Device Properties**) is set to zero.

Solution:
Configure the shared hourly, daily, or batch archive number with a value other than zero.

Note:
The batch archive number is only supported by OMNI Liquid Firmware models.

See Also:
EFM Meter Settings

'<device name>' - Shared archive record parse failed

Error Type:
Warning

Possible Cause:
1. A failure occurred when parsing the shared data history record for EFM attribute data.
2. The device's archive memory may have been cleared since the last upload.

Solution:
1. Verify that the shared data archive's configuration is correct, and that it matches the Gas Quality mapping.
2. Ensure that the time and date have been set correctly.

'<device name>' - The <archive type> mapping contains more configured attributes than the device. Some attributes will not contain valid data

Error Type:
Warning

Possible Cause:
The number of registers configured in an archive's group configuration does not equal the number of attributes with a configured index in the hourly, daily, or batch mapping.

Solution:
Verify that the archive's group configuration matches the hourly, daily, or batch mapping configuration.

'<device name>' - The max alarm archive size was changed from '<previous size>' to '<current size>'

Error Type:
Warning

Possible Cause:
The size of the alarm archive in the device has been changed since the last alarm archive upload.

Solution:
Change the alarm archive size back to its default value for the Firmware version that is being used.

Note:
Data may be lost if the archive size is not set to the default value.

'<device name>' - The max event archive size was changed from '<previous size>' to '<current size>'

Error Type:
Warning

Possible Cause:
The size of the event archive in the device has been changed since the last event archive upload.

Solution:
Change the event archive size back to its default value for the Firmware version that is being used.

Note:
Data may be lost if the archive size is not set to the default value.

'<device name>' - Time sync write not successful. Value in response is different from the written value

Error Type:
Warning

Possible Cause:
1. The serial connection between the device and the Host PC is bad.
2. The communications properties for the serial connection are incorrect.

Solution:
1. Verify the cabling between the PC and the PLC device.
2. Verify that the specified communications properties match those of the device.

'<device name>' - Unable to read '<number of registers>' registers in config register block at address '<address>'

Error Type:
Warning

Possible Cause:
1. The serial connection between the device and the Host PC is bad.
2. The communications properties for the serial connection are incorrect.

Solution:
1. Verify that the Firmware version running on the device is supported.
2. Verify the cabling between the PC and the PLC device.
3. Verify that the specified communications properties match those of the device.

'<device name>' - Unable to read date format register for address '<address>'. Response is not the correct size

Error Type:
Warning

Possible Cause:

1. The serial connection between the device and the Host PC is bad.
2. The communications properties for the serial connection are incorrect.

Solution:

1. Verify the cabling between the PC and the PLC device.
2. Verify that the specified communications properties match those of the device.

'<device name>' - Unable to read index registers. Response is not the correct size

Error Type:
Warning

Possible Cause:

1. The serial connection between the device and the Host PC is bad.
2. The communications properties for the serial connection are incorrect.

Solution:

1. Verify the cabling between the PC and the PLC device.
2. Verify that the specified communications properties match those of the device.

'<device name>' - Unable to read record format registers for address '<address>'. Response is not the correct size

Error Type:
Warning

Possible Cause:

1. The serial connection between the device and the Host PC is bad.
2. The communications properties for the serial connection are incorrect.

Solution:

1. Verify the cabling between the PC and the PLC device.
2. Verify that the specified communications properties match those of the device.

'<device name>' - Unable to write requested record register for address '<address>'. Wrote '<value>', read back '<value>'

Error Type:
Warning

Possible Cause:

1. The serial connection between the device and the Host PC is bad.
2. The communications properties for the serial connection are incorrect.

Solution:

1. Verify the cabling between the PC and the PLC device.
2. Verify that the specified communications properties match those of the device.

'<device name>' - Unable to write requested record register for address '<address>'. Response is not the correct size

Error Type:
Warning

Possible Cause:

1. The serial connection between the device and the Host PC is bad.
2. The communications properties for the serial connection are incorrect.

Solution:

1. Verify the cabling between the PC and the PLC device.
2. Verify that the specified communications properties match those of the device.

Alarm mapping for address '<address>' is invalid and will be ignored

Error Type:
Warning

Possible Cause:
An invalid Alarm Mapping was imported from a CSV file or loaded from an XML project file.

Solution:
Correct the Alarm Mapping in the CSV import file or the XML project file.

See Also:
EFM Alarm Mapping

Alarm state for address '<address>' is invalid. Setting the state to <state>

Error Type:
Warning

Possible Cause:
An invalid Alarm Mapping was imported from a CSV file.

Solution:
Correct the Alarm Mapping in the CSV import file.

See Also:
EFM Alarm Mapping

Alarm type for address '<address>' is invalid. Setting the type to <type>

Error Type:
Warning

Possible Cause:
An invalid Alarm Mapping was imported from a CSV file.

Solution:
Correct the Alarm Mapping in the CSV import file.

See Also:
EFM Alarm Mapping

Bad address in block [<start address> to <end address>] on device '<device name>'

Error Type:
Serious

Possible Cause:

1. An attempt has been made to reference a nonexistent location in the specified device.
2. An attempt has been made to read more registers than allowed by the protocol.

Solution:

1. Verify the tags assigned to addresses in the specified range on the device and eliminate ones that reference invalid locations.
2. Decrease the register block size value to 125.

See Also:
Framing & Error Handling
Block Sizes

Bad array spanning [<address> to <address>] on device '<device>'

Error Type:
Serious

Possible Cause:

1. An attempt has been made to reference a nonexistent location in the specified device.
2. An attempt has been made to read more registers than allowed by the protocol.

Solution:

1. Verify that all the register addresses requested in the array exist in the device and reduce the array size such that only valid addresses (that exist in the device) are requested by the array.
2. Reduce the array size value to the number of addresses that can be read by the protocol in a single access. For example, set the value to 125 for 16 bit registers.

See Also:
Framing & Error Handling
Block Sizes

Device password invalid for device '<device name>'

Error Type:
Warning

Possible Cause:
The device responded with the "needs password" exception code, even though the configured password was written to the device.

Solution:
Ensure that the device password that is configured in the Data Access property group of Device Properties matches the password that is configured on the device.

Device password write for device '<device name>' was successful

Error Type:
Information

Possible Cause:
After receiving the "Device password invalid for device '<device name>'" error message, a valid password was written to the device.

Solution:
N/A.

Note:
This message is logged to indicate that the device's password requirements have been met.

See Also:
Device password invalid for device '<device name>'

History attribute '<attribute index>' is unknown and will be ignored

Error Type:
Warning

Possible Cause:
An invalid History Mapping was imported from a CSV file or loaded from an XML project file.

Solution:
Correct the History Mapping in the CSV import file or the XML project file.

See Also:
EFM History Mapping

History mapping for attribute '<attribute name>' is invalid and will be ignored

Error Type:
Warning

Possible Cause:
An invalid History Mapping was imported from a CSV file or loaded from an XML project file.

Solution:
Correct the History Mapping in the CSV import file or the XML project file.

See Also:
EFM History Mapping

Received "needs password" exception from device '<device name>' with 'fail after successive timeouts' set to 1. Set the 'fail after successive timeouts' setting to a value greater than 1 and verify that the 'device password' setting is correct

Error Type:
Warning

Possible Cause:
A password is required to access a register in the device, and the Fail After *x* Successive Timeouts setting is configured with a value of 1.

Solution:

To start, enter the Device Password (located in the **Data Access** property group of **Device Properties**) and ensure that it is set correctly. Then, change the Fail After *x* Successive Timeouts setting (located in the **Timing** property group of **Device Properties**) to a value greater than 1.

See Also:
Data Access

Serialization of EFM data to temporary file '<file name>' failed. Reason: '<file I/O error>'

Error Type:
Warning

Possible Cause:

1. The driver was unable to create the specified file directory.
2. The driver was unable to access the specified file.

Solution:

1. Verify that the disk has sufficient disk space.
2. Verify user permissions for the specified file directory.

The '<archive type>' archive number for meter '<meter name>' is already being used. XML project load not successful

Error Type:
Warning

Possible Cause:
The archive number for the meter's hourly, daily, or batch archive is already being used by another meter.

Solution:
Change the archive number for the meter's hourly, daily, or batch archive to a value that is not already in use.

The shared '<archive type>' archive number is already in use by another meter. XML project load not successful

Error Type:
Warning

Possible Cause:
The archive number for the shared hourly, daily, or batch archive is already being used by another meter.

Solution:
Change the archive number for the shared hourly, daily, or batch archive to a value that is not already in use.

Unable to read '<address>' from device '<device name>'. The device is configured for broadcast writes only

Error Type:
Warning

Possible Cause:
The device is configured for broadcast writes only, and an EFM Poll was triggered. The Device ID is set to 0.

Solution:
1. Disable EFM polling for broadcast devices.
2. Do not use a Device ID of 0 for EFM-enabled devices.

Unable to read block address ['<start address>' to '<end address>'] on device '<device name>'. Unexpected characters in response

Error Type:
Warning

Possible Cause:
The calculated CRC did not match the CRC that was sent by the device.

Solution:
1. Verify the cabling between the PC and the PLC device.
2. Verify that the specified communications properties match those of the device.

Warning loading '<mapping type>' mapping from CSV. '<warning type>'

Error Type:
Information

Possible Cause:
A new EFM mapping was imported from the CSV file.

Solution:
N/A.

Communications error on '<channel name>' [<error mask>]

Error Type:
Serious

Error Mask Definitions:
B = Hardware break detected.
F = Framing error.
E = I/O error.
O = Character buffer overrun.

R = RX buffer overrun.
P = Received byte parity error.
T = TX buffer full.

Possible Cause:

1. The serial connection between the device and the Host PC is bad.
2. The communications properties for the serial connection are incorrect.

Solution:

1. Verify the cabling between the PC and the PLC device.
2. Verify that the specified communications properties match those of the device.

COMn does not exist

Error Type:
Fatal

Possible Cause:
The specified COM port is not present on the target computer.

Solution:
Verify that the proper COM port has been selected.

COMn is in use by another application

Error Type:
Fatal

Possible Cause:
The serial port assigned to a device is being used by another application.

Solution:

1. Verify that the correct port has been assigned to the channel.
2. Verify that only one copy of the current project is running.

Error opening COMn

Error Type:
Fatal

Possible Cause:
The specified COM port could not be opened due an internal hardware or software problem on the target computer.

Solution:
Verify that the COM port is functional and may be accessed by other Windows applications.

Unable to set comm parameters on COMn

Error Type:
Fatal

Possible Cause:
The serial parameters for the specified COM port are not valid.

Solution:
Verify the serial parameters and make any necessary changes.

Index

<

A

B

C

D

E

F

G

H

I

L

M

N

O

P

R

S

T

U

V

W