

OSIsoft PIConnector Driver

| | |
|----------------------------|-------------------------------|
| File Name | PIConnector.dll |
| Manufacturer | OSIsoft (AVEVA Group Limited) |
| Devices | PI System |
| Protocol | |
| Version | 1.0.2 |
| Last Update | 04/06/2026 |
| Platform | Win32 |
| Dependencies | PI API (PIAPI32.dll library) |
| Superblock Readings | No |
| Level | 32000 |

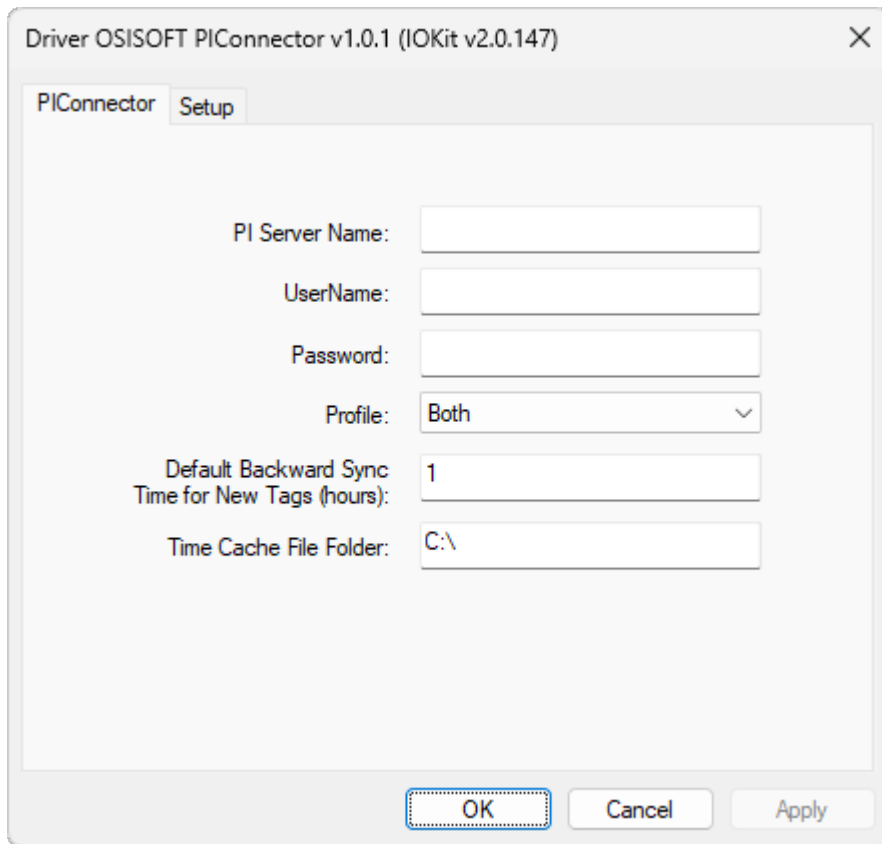
Introduction

This Driver, by using the PI API (*Application Programming Interface*), enables an application developed by **Elipse Software** to communicate with a PI System by OSIsoft (AVEVA Group Limited).

The purpose of this Driver is to facilitate integration between **Elipse Software's** applications and a PI database, allowing both real-time monitoring and synchronization of historical data, in a reliable and efficient way.

Driver's Configuration Parameters

The **[P]** configuration parameters of this Driver are not used. All settings are performed on the configuration window, shown on the next figure.



PIConnector tab

The available options on the **PIConnector** tab are described on the next table.

Available options on the PIConnector tab

| OPTION | DESCRIPTION |
|--|--|
| PI Server Name | Name of a PI server or a <i>PI Data Archive's</i> node to which users are trying to connect. This parameter identifies the connection destination |
| UserName | Name of the user trying to authenticate. This must be a valid user configured in a <i>PI Data Archive</i> |
| Password | The password corresponding to the provided user, used to verify identity |
| Profile | Profile for capturing data from a PI Tag. The available options are RealTime : Monitors real-time values through continuous subscriptions, thus tracking live data updates, Historical : Retrieves exclusively past data for analysis, focusing on archived data, or Both : Integrates both modes, first syncing the historical mode and then monitoring real-time values |
| Default Backward Sync Time for New Tags (hours) | Determines a retroactive time interval, in hours, for which historical data of a PI Tag must be retrieved when performing the initial syncing of a new Tag |

| OPTION | DESCRIPTION |
|-------------------------------|--|
| Time Cache File Folder | Defines a directory where the time cache file is stored, allowing to save the last timestamps of PI Tags when the program is closed and restore them during initialization. This ensures continuity in data syncing between sessions, optimizing both historical and real-time data collecting |

Tag Reference

This section contains information about the configuration of this Driver's [N] Tags.

PLC Tag

Read-Only By Callback

Use a PLC Tag to represent and retrieve data from a PI Tag in a *PI Data Archive*, the component of a PI System that stores values from PI Tags.

Parameters of a PLC Tag

| Item | Name of a PI Tag in a <i>PI Data Archive</i> |
|--------------|--|
| N1/B1 | 0 (zero) |
| N2/B2 | 0 (zero) |
| N3/B3 | 0 (zero) |
| N4/B4 | 0 (zero) |

While syncing historical data, this Tag returns lists of previous values from the corresponding PI Tag. Each list contains up to 100 values, organized from the oldest values to the most recent values. While monitoring in real time, this Tag provides only the current value of the corresponding PI Tag.

Documentation of I/O Interfaces

This section contains the documentation of I/O Interfaces referring to **PIConnector** Driver.

Configuration of a Driver

I/O Interface configuration is performed on a Driver's configuration dialog box. To access the configuration of this dialog box in **Eclipse E3** in version 1.0, follow these steps:

1. Right-click a Driver object (IODriver).
2. Select the **Properties** item on the contextual menu.
3. Select the **Driver** tab.
4. Click **Other parameters**.

In **Eclipse E3** version 2.0 or later, click **Configure driver**  on a Driver's toolbar. In **Eclipse SCADA**, follow these steps:

1. Open the Organizer.
2. Select a Driver on Organizer's tree.
3. Click **Extras** on the **Driver** tab.

Currently, an I/O Interface allows opening only one connection for each Driver. This means that, if users want to access two serial ports, they must add two Drivers to an application and then configure each one of these Drivers for each serial port.

Configuration Dialog Box

The dialog box of I/O Interfaces allows configuring the I/O connection used by a Driver. This dialog box contains the **Setup**, **Serial**, **Ethernet**, **Modem**, and **RAS** tabs, described on the next topics. If a Driver does not implement a specific I/O connection, its corresponding tab is not available for configuration. Some Drivers may contain additional tabs, specific for that Driver, on the configuration dialog box.

Setup Tab

The **Setup** tab contains general configurations of a Driver. This tab is divided into the following groups:

- **General configurations:** Configurations of a Driver's physical layer, time-out, and initialization mode
- **Connection management:** Configurations on how the I/O Interface keeps a connection and which recovery policy is used on failure
- **Logging options:** Controls the generation of log files

The screenshot shows the 'Setup' tab configuration window. It is divided into several sections:

- Physical Layer:** A dropdown menu set to 'Ethernet'. To the right is a checkbox labeled 'Start driver OFFLINE' which is unchecked.
- Timeout:** A text input field containing '1000' followed by 'ms'.
- Communication check time:** A text input field containing '5000' followed by 'ms'.
- Connection management:** A sub-section containing:
 - A dropdown menu for 'Mode' set to 'Automatic (managed by the driver)'.
 - A checked checkbox 'Retry failed connection every' followed by a text input '20' and 'seconds'.
 - An unchecked checkbox 'Give up after' followed by a text input '1' and 'failed retries'.
 - An unchecked checkbox 'Disconnect if non-responsive for' followed by a text input '0' and 'seconds'.
- Logging Options:** A sub-section containing:
 - An unchecked checkbox 'Log to File:' followed by a text input 'C:\eeLogs\MicrolokII_%DATE%.log'.
 - A text input 'File size limit (MB):' followed by '0' and '(0 is unlimited)'.

Setup tab

General options on the Setup tab

| OPTION | DESCRIPTION |
|---------------------------------|---|
| Physical Layer | Select the physical layer on a list. Available options are Serial , Ethernet , Modem , and RAS . The selected interface must be configured on its specific tab |
| Timeout | Configure a time-out, in milliseconds, for the physical layer. This is the amount of time an I/O interface waits to receive any byte from the reception's buffer |
| Communication check time | Set the time, in milliseconds, to define the interval at which communication is considered to be in an inactive state. As long as an I/O Driver receives valid data, its communication state is considered active. However, if during operation an I/O Driver does not receive valid data inside this period of time, the state is considered inactive. The communication state is shown in the IO.CommunicationStatus Tag |

| OPTION | DESCRIPTION |
|-----------------------------|--|
| Start driver OFFLINE | Select this option so that a Driver starts in Offline mode or stopped. This means that the I/O interface is not created until this Driver is configured to Online mode by using a Tag in an application. This mode enables a dynamic configuration of an I/O interface at run time |

Options on the Connection management group

| OPTION | DESCRIPTION |
|---|---|
| Mode | Selects a management mode of a connection. Selecting the Automatic option allows a Driver to manage the connection automatically, as specified in the next options. Selecting the Manual option allows an application to fully manage a connection |
| Retry failed connection every ... seconds | Select this option to enable a Driver's connection retry in a certain interval, in seconds. If the Give up after failed retries option is not selected, this Driver keeps retrying until a connection is performed, or until the application is stopped |
| Give up after ... failed retries | Enable this option to define a maximum number of connection retries. When the specified number of consecutive connection retries is reached, a Driver goes to the Offline mode, assuming that a hardware problem was detected. If a Driver establishes a successful connection, the number of unsuccessful retries is cleared. If this new connection is lost, then the retry counter starts at zero |
| Disconnect if non-responsive for ... seconds | Enable this option to force a Driver to disconnect if no byte was received by the I/O interface during the specified time-out, in seconds. This time-out must be greater than the time-out configured in the Timeout option |

Options on the Logging Options group

| OPTION | DESCRIPTION |
|------------------------------------|---|
| <p>Log to File</p> | <p>Enable this option and configure the name of a file to write a log. Log files can be large, so use this option for short periods of time, only for testing and debugging purposes. If the %PROCESS% macro is used in the log file name, it is replaced by the identifier of the current process. This option is particularly useful when using several instances of the same Driver in Elipse E3, thus allowing each instance to generate a separate log file. For example, when configuring this option with value "c:\e3logs\drivers\sim_%PROCESS%.log", it generates a file named c:\e3logs\drivers\sim_00000FDA.log for process OFDAh. Users can also use the %DATE% macro in the file name. In this case a log file is generated every day, in the format aaaa_mm_dd. For example, when configuring this option with value "c:\e3logs\drivers\sim_%DATE%.log", it generates a file named c:\e3logs\drivers\sim_2005_12_31.log in 12/31/2005 and a file named c:\e3logs\drivers\sim_2006_01_01.log in 01/01/2006. Similarly, the %DATE_HOUR% macro generates one log file per hour, in the format aaaa_mm_dd_hh</p> |
| <p>File size limit (MB)</p> | <p>Configure the log file size limit, in megabytes. A value equal to 0 (zero) means that there is no size limit for the log file</p> |

General Configurations

This section contains information about the configuration of general **I/O Tags** and **Properties** of I/O Interfaces.

I/O Tags

General I/O Interfaces Tags (N2/B2 = 0)

The Tags described next are provided for all supported I/O Interfaces.

IO.CommunicationStatus

| | |
|-----------------------------|------------------------|
| Type of Tag | I/O Tag |
| Type of Access | Reading |
| N1 Parameter | -1 (minus one) |
| N2 Parameter | 0 (zero) |
| N3 Parameter | 0 (zero) |
| N4 Parameter | 6 (six) |
| String Configuration | IO.CommunicationStatus |

This Tag informs the communication status of a Driver. It indicates how communication works relative to receiving valid data within a time period arbitrated in the configuration. For more information, please check topic **Setup Tab**. Possible values are **0 - Inactive communication**: The Driver did not receive valid data or stopped receiving data after *n* milliseconds, as configured in the properties window, or **1 - Active communication**: The Driver is receiving valid data.

IO.IOKitEvent

| | |
|---------------------------|----------------|
| Type of Tag | Block Tag |
| Type of Access | Read-Only |
| B1 Parameter | -1 (minus one) |
| B2 Parameter | 0 (zero) |
| B3 Parameter | 0 (zero) |
| B4 Parameter | 1 (one) |
| Size Property | 4 (four) |
| ParamItem Property | IO.IOKitEvent |

This Block returns Driver events generated by several sources in I/O Interfaces. The **TimeStamp** property of this Block represents the moment this event occurred. The Block Elements are the following:

- **Element 0:** Type of event. Possible values are **0:** Information, **1:** Warning, or **2:** Error
- **Element 1:** Source of an event. Possible values are **0:** Driver (specific of a Driver), **-1:** IOKit (generic events of I/O Interfaces), **-2:** **Serial** Interface, **-3:** **Modem** Interface, **-4:** **Ethernet** Interface, or **-5:** **RAS** Interface
- **Element 2:** Error number, specific for each source of event
- **Element 3:** Message of an event, a **String** specific for each event

NOTE

A Driver keeps a maximum number of 100 events internally. If additional events are reported, older events are discarded.

IO.PhysicalLayerStatus

| | |
|-----------------------------|------------------------|
| Type of Tag | I/O Tag |
| Type of Access | Read-Only |
| N1 Parameter | -1 (minus one) |
| N2 Parameter | 0 (zero) |
| N3 Parameter | 0 (zero) |
| N4 Parameter | 2 (two) |
| String Configuration | IO.PhysicalLayerStatus |

This Tag indicates the status of a physical layer. Possible values are the following:

- **0:** Physical layer stopped, that is, a Driver is in **Offline** mode, the physical layer failed when initializing, or exceeded the maximum number of reconnection attempts

- **1:** Physical layer started but not connected, that is, a Driver is in **Online** mode but the physical layer is not connected. If the **Connection management** option is configured with the value **Automatic**, the physical layer can be connecting, disconnecting, or waiting for a reconnection attempt. If the **Connection management** option is configured with the value **Manual**, then the physical layer remains in this status until forced to connect
- **2:** Physical layer connected, that is, the physical layer is ready for use. This **DOES NOT** mean a device is connected, only that the access layer is working

IO.SetConfigurationParameters

| | |
|---------------------------|-------------------------------|
| Type of Tag | Block Tag |
| Type of Access | Read-Only |
| B1 Parameter | -1 (minus one) |
| B2 Parameter | 0 (zero) |
| B3 Parameter | 0 (zero) |
| B4 Parameter | 3 (three) |
| Size Property | 2 (two) |
| ParamItem Property | IO.SetConfigurationParameters |

Use this Tag to change any property of a Driver's configuration dialog box at run time.

This Tag works only while a Driver is in **Offline** mode. To start a Driver in **Offline** mode, select the **Start driver OFFLINE** option on that Driver's configuration dialog box. Users can write to a PLC Tag or to a Block Tag containing the parameters to change. Writing individual Block Elements is not supported, the whole Block must be written at once.

In **Eclipse SCADA**, users must use a Block Tag. Every parameter to configure uses two Block Elements. For example, if users want to configure 3 (three) parameters, then the size of the Block must be 6 (six, 3×2). The first Element is the property's name, as a **String**, and the second Element is the property's value, according to the next example.

```
// 'Block' must be a Block Tag with automatic reading,
// scan reading, and automatic writings disabled.
// Configure all parameters
Block.element001 = "IO.Type" // Parameter 1
Block.element002 = "Serial"
Block.element003 = "IO.Serial.Port" // Parameter 2
Block.element004 = 1
Block.element005 = "IO.Serial.BaudRate" // Parameter 3
Block.element006 = 19200
// Writes the whole Block
Block.Write()
```

When using **Eclipse E3**, the ability to create arrays at run time allows using an I/O Tag as well as a Block Tag. Users can use the **Write** method of a Driver to send the parameters directly to that Driver, without creating a Tag, according to the next example.

```
Dim arr(6)
' Configure all array elements
arr(1) = "IO.Type"
arr(2) = "Serial"
arr(3) = "IO.Serial.Port"
arr(4) = 1
arr(5) = "IO.Serial.BaudRate"
arr(6) = 19200
' There are two methods to send parameters
' Method 1: Using an I/O Tag
tag.WriteEx arr
' Method 2: Without using a Tag
Driver.Write -1, 0, 0, 3, arr
```

A variation of the previous example uses a bidimensional array.

```
Dim arr(10)
' Configure all array elements. Notice the array was resized
' to 10 elements. Empty array elements are ignored by a Driver
arr(1) = Array("IO.Type", "Serial")
arr(2) = Array("IO.Serial.Port", 1)
arr(3) = Array("IO.Serial.BaudRate", 19200)
Driver.Write -1, 0, 0, 3, arr
```

A Driver does not validate parameter names or passed values, therefore be careful when writing parameters and values. The **Write** method fails if the configuration array is incorrectly created. Users can check the log of a Driver or use the *writeStatus* parameter of the **WriteEx** method to find out the exact cause of an error.

```
Dim arr(10), strError
arr(1) = Array("IO.Type", "Serial")
arr(2) = Array("IO.Serial.Port", 1)
arr(3) = Array("IO.Serial.BaudRate", 19200)
If Not Driver.WriteEx -1, 0, 0, 3, arr, , , strError Then
    MsgBox "Failed configuring Driver parameters: " + strError
End If
```

IO.WorkOnline

| | |
|-----------------------------|--------------------|
| Type of Tag | I/O Tag |
| Type of Access | Reading or Writing |
| N1 Parameter | -1 (minus one) |
| N2 Parameter | 0 (zero) |
| N3 Parameter | 0 (zero) |
| N4 Parameter | 4 (four) |
| String Configuration | IO.WorkOnline |

This Tag informs the current status of a Driver and allows starting or stopping the physical layer. Possible values are the following:

- **0 - Driver Offline:** Physical layer is closed or stopped. This mode allows a dynamic configuration of a Driver's parameters using the **IO.SetConfigurationParameters** Tag
- **1 - Driver Online:** Physical layer is open or executing. While in **Online** mode, the physical layer can be connected or disconnected and its current status can be checked using the **IO.PhysicalLayerStatus** Tag

In the next example, using **Eclipse E3**, a Driver is configured to **Offline** mode, its COM port is changed, and then configured to **Online** mode again.

```
'Configure to Offline mode
Driver.Write -1, 0, 0, 4, 0
'Change port to COM2
Driver.Write -1, 0, 0, 3, Array("IO.Serial.Port", 2)
'Configure to Online mode
Driver.Write -1, 0, 0, 4, 1
```

The **Write** method may fail when configuring a Driver to **Online** mode, that is, writing the value 1 (one). In this case, this Driver remains in **Offline** mode. The cause of failure can be:

- Type of physical layer incorrectly configured, probably an invalid value was configured in the **IO.Type** property
- This Driver may have run out of memory
- Physical layer probably did not create its working thread. Search the log file for a message "Failed to create physical layer thread!"
- Physical layer could not start. The cause of this failure depends on the type of physical layer. It can be an invalid serial port number, a failure when starting Windows Sockets, or a failure when starting TAPI (modem), among others. This cause is recorded on the log file

IMPORTANT

Even if the configuration of a Driver to **Online** mode is successful, this does not necessarily mean the physical layer is ready to use, that is, ready to execute input and output operations with an external device. The **IO.PhysicalLayerStatus** Tag must be checked to ensure the physical layer is connected and ready for communication.

Properties

These are general properties of all supported I/O Interfaces.

IO.ConnectionMode

9 Controls the management mode of a Connection. Possible values are **0**: Automatic mode, in which a Driver manages the connection or **1**: Manual mode, in which an application manages the connection.

IO.GiveUpEnable

■ When configured to True, defines a maximum number of reconnection attempts. If all reconnection attempts fail, a Driver enters the **Offline** mode. When configured to False, a Driver tries until a reconnection is successful.

IO.GiveUpTries

9 Number of reconnection attempts before this one is aborted. For example, if the value of this property is equal to 1 (one), a Driver tries only one reconnection when the connection is lost. If this one fails, this Driver enters the **Offline** mode.

IO.InactivityEnable

■ Configure to True to enable and to False to disable inactivity detection. The physical layer is disconnected if inactive for a certain period of time. The physical layer is considered inactive only if it is capable of sending data but not capable of receiving it back.

IO.InactivityPeriodSec

9 Number of seconds to check for inactivity. If the physical layer is inactive for this period of time, it is then disconnected.

IO.RecoverEnable

☑ Configure to True to enable a Driver to recover lost connections and to False to leave a Driver in **Offline** mode when a connection is lost.

IO.RecoverPeriodSec

9 Delay time between two connection attempts, in seconds.

NOTE

The first reconnection is executed immediately after a connection is lost.

IO.StartOffline

☑ Configure to True to start a Driver in **Offline** mode and to False to start a Driver in **Online** mode.

NOTE

It is pointless to change this property at run time, as it can only be changed when a Driver is already in **Offline** mode. To configure a Driver in **Online** mode at run time, write the value 1 (one) to the **IO.WorkOnline** Tag.

IO.TimeoutMs

9 Defines a time-out for the physical layer, in milliseconds. One second is equal to 1000 milliseconds.

IO.Type

A Defines the type of physical interface used by a Driver. Possible values are the following:

- **N or None:** Does not use a physical interface, that is, a Driver must provide a customized interface
- **S or Serial:** Uses a local serial port (COM n)
- **M or Modem:** Uses a local modem, internal or external, accessed via TAPI (*Telephony Application Programming Interface*)
- **E or Ethernet:** Uses a TCP/IP or UDP/IP socket
- **R or RAS:** Uses a **RAS** (*Remote Access Server*) Interface. A Driver connects to a RAS device using the **Ethernet** Interface and then sends an **AT** (*dial*) command

Statistical Configuration

This section contains information about the configuration of **I/O Tags** and **Properties** of I/O Interfaces statistics.

I/O Tags

Tags of I/O Interface Statistics (N2/B2 = 0)

The Tags described next display statistics for all I/O Interfaces.

IO.Stats.Partial.BytesRecv

| | |
|--------------------------------|----------------------------|
| Type of Tag | I/O Tag |
| Type of Access | Read-Only |
| N1 Parameter | -1 (minus one) |
| N2 Parameter | 0 (zero) |
| N3 Parameter | 0 (zero) |
| N4 Parameter | 1101 |
| Configuration by String | IO.Stats.Partial.BytesRecv |

This Tag returns the number of bytes received in the current connection.

IO.Stats.Partial.BytesSent

| | |
|--------------------------------|----------------------------|
| Type of Tag | I/O Tag |
| Type of Access | Read-Only |
| N1 Parameter | -1 (minus one) |
| N2 Parameter | 0 (zero) |
| N3 Parameter | 0 (zero) |
| N4 Parameter | 1100 |
| Configuration by String | IO.Stats.Partial.BytesSent |

This Tag returns the number of bytes sent through the current connection.

IO.Stats.Partial.TimeConnectedSeconds

| | |
|--------------------------------|---------------------------------------|
| Type of Tag | I/O Tag |
| Type of Access | Read-Only |
| N1 Parameter | -1 (minus one) |
| N2 Parameter | 0 (zero) |
| N3 Parameter | 0 (zero) |
| N4 Parameter | 1102 |
| Configuration by String | IO.Stats.Partial.TimeConnectedSeconds |

This Tag returns the number of seconds a Driver is connected in the current connection or 0 (zero) if a Driver is disconnected.

IO.Stats.Partial.TimeDisconnectedSeconds

| | |
|--------------------------------|--|
| Type of Tag | I/O Tag |
| Type of Access | Read-Only |
| N1 Parameter | -1 (minus one) |
| N2 Parameter | 0 (zero) |
| N3 Parameter | 0 (zero) |
| N4 Parameter | 1103 |
| Configuration by String | IO.Stats.Partial.TimeDisconnectedSeconds |

This Tag returns the number of seconds a Driver is disconnected since the last connection ended or 0 (zero) if a Driver is connected.

IO.Stats.Total.BytesRecv

| | |
|--------------------------------|--------------------------|
| Type of Tag | I/O Tag |
| Type of Access | Read-Only |
| N1 Parameter | -1 (minus one) |
| N2 Parameter | 0 (zero) |
| N3 Parameter | 0 (zero) |
| N4 Parameter | 1001 |
| Configuration by String | IO.Stats.Total.BytesRecv |

This Tag returns the number of bytes received since a Driver was loaded.

IO.Stats.Total.BytesSent

| | |
|--------------------------------|--------------------------|
| Type of Tag | I/O Tag |
| Type of Access | Read-Only |
| N1 Parameter | -1 (minus one) |
| N2 Parameter | 0 (zero) |
| N3 Parameter | 0 (zero) |
| N4 Parameter | 1000 |
| Configuration by String | IO.Stats.Total.BytesSent |

This Tag returns the number of bytes sent since a Driver was loaded.

IO.Stats.Total.ConnectionCount

| | |
|--------------------------------|--------------------------------|
| Type of Tag | I/O Tag |
| Type of Access | Read-Only |
| N1 Parameter | -1 (minus one) |
| N2 Parameter | 0 (zero) |
| N3 Parameter | 0 (zero) |
| N4 Parameter | 1004 |
| Configuration by String | IO.Stats.Total.ConnectionCount |

This Tag returns the number of connections a Driver already established, successfully, since it was loaded.

IO.Stats.Total.TimeConnectedSeconds

| | |
|--------------------------------|-------------------------------------|
| Type of Tag | I/O Tag |
| Type of Access | Read-Only |
| N1 Parameter | -1 (minus one) |
| N2 Parameter | 0 (zero) |
| N3 Parameter | 0 (zero) |
| N4 Parameter | 1002 |
| Configuration by String | IO.Stats.Total.TimeConnectedSeconds |

This Tag returns the number of seconds a Driver remained connected since it was loaded.

IO.Stats.Total.TimeDisconnectedSeconds

| | |
|--------------------------------|--|
| Type of Tag | I/O Tag |
| Type of Access | Read-Only |
| N1 Parameter | -1 (minus one) |
| N2 Parameter | 0 (zero) |
| N3 Parameter | 0 (zero) |
| N4 Parameter | 1003 |
| Configuration by String | IO.Stats.Total.TimeDisconnectedSeconds |

This Tag returns the number of seconds a Driver remained disconnected since it was loaded.

Properties

Currently, there are no properties defined specifically to display I/O Interface statistics at run time.

Driver's Revision History

| VERSION | DATE | AUTHOR | COMMENTS |
|---------|------------|-------------|--|
| 1.0.2 | 04/06/2026 | M. Ludwig | <ul style="list-style-type: none">Driver updated to IOKit library version 3.0 and Visual Studio 2022 (Case 38003). |
| 1.0.1 | 04/28/2025 | M. Salvador | <ul style="list-style-type: none">Initial version of this Driver (Case 22175). |

Headquarters

**Rua Mostardeiro, 322/Cj. 902, 1001 e
1002**

90510-002 — Porto Alegre — RS

Phone: (+55 51) 3346-4699

Fax: (+55 51) 3222-6226

E-mail: elipse-rs@elipse.com.br

Branch in Taiwan

9F., No.12, Beiping 2nd St., Sanmin Dist.

807 — Kaohsiung City — Taiwan

Phone: (+886 7) 323-8468

Fax: (+886 7) 323-9656

E-mail: evan@elipse.com.br

Check our website for information about a representative in your country.

www.elipse.com.br

kb.elipse.com.br

forum.elipse.com.br

www.youtube.com/elipsesoftware

elipse@elipse.com.br



Gartner, Cool Vendors in Brazil 2014, April 2014.

Gartner does not endorse any vendor, product or service depicted in its research publications, and does not advise technology users to select only those vendors with the highest ratings. Gartner research publications consist of the opinions of Gartner's research organization and should not be construed as statements of fact. Gartner disclaims all warranties, expressed or implied, with respect to this research, including any warranties of merchantability of fitness for a particular purpose.

Microsoft Partner
Gold Independent Software Vendor (ISV)