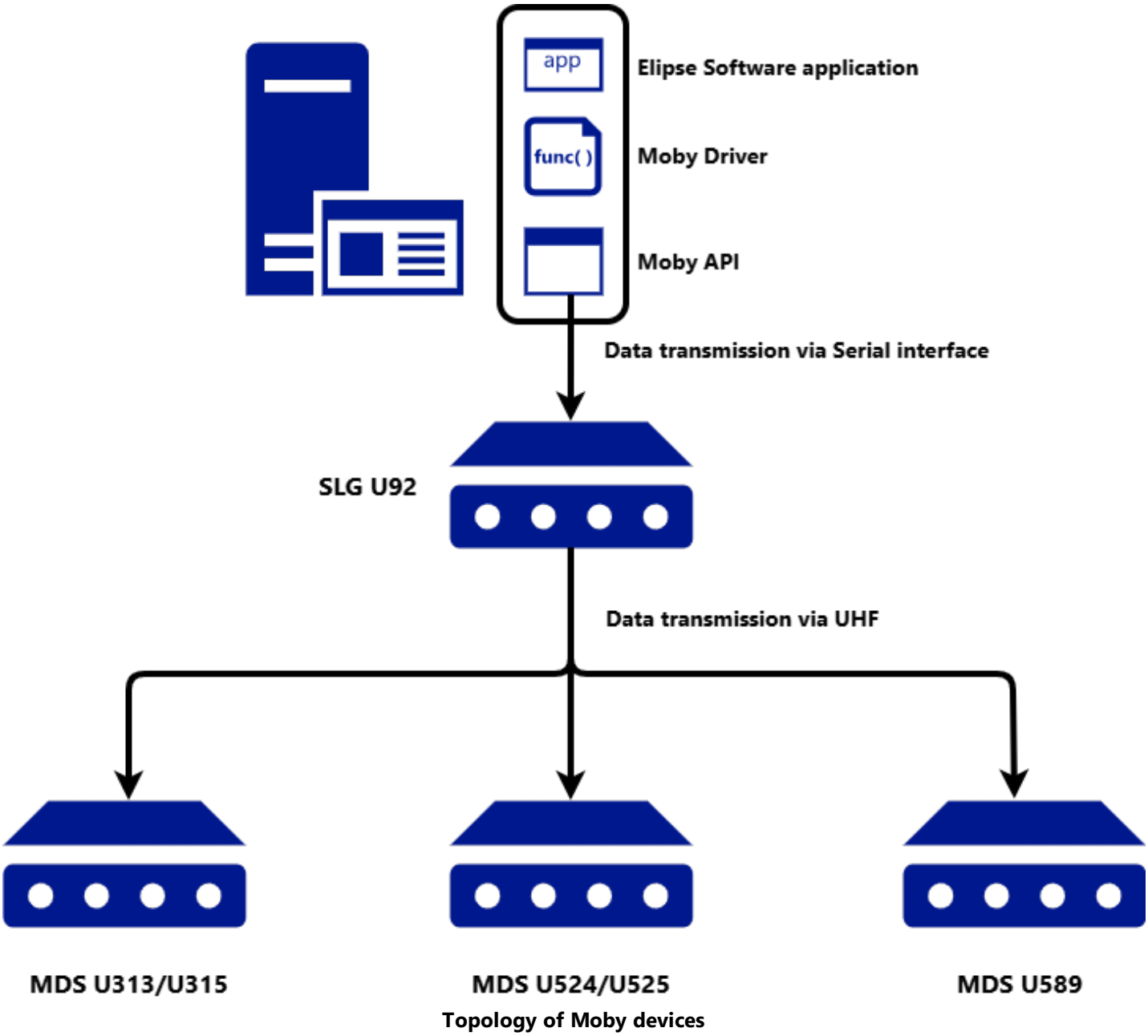
Siemens Moby Driver

File Name	MobyDrv.dll
Manufacturer	Siemens AG
Devices	Moby devices
Protocol	Moby commands packed in 3964R protocol over Serial interface
Version	2.0.1
Last Update	05/15/2026
Platform	Win32
Dependencies	Moby_API.dll and 3964R.dll libraries and IOKit version 1.15 or later
Superblock Readings	No
Level	0

Introduction

This is the Siemens Moby Driver for communication between **Elipse Software** applications and Moby devices by Siemens AG, according to the topology shown on the next figure.



Topology of Moby devices

Preparing a Device

Before using this Driver, please make sure that files Moby_API.dll and 3964R.dll are on the same directory of Moby file or on a directory of the **PATH** environment variable of the operating system. These files are provided by Siemens AG and this Driver connects to those libraries during the execution to request data. Although these files are needed for execution, this Driver can be configured without them.

Driver Configuration

This section contains information about the configuration of this Driver.

[P] Parameters

P1	Number of a serial port. Possible values are **1**: COM1, **2**: COM2, **3**: COM3, or **4**: COM4
P2	Number of a Moby channel (*MobyChannel*). This value refers to the number of the channel wherethe SLG, SIM, or SLA device is addressed
P3	Not used, leave it in 0 (zero)
P4	Time-out, in milliseconds

This Driver also has extra configurations, which determine values to start communication with an antenna device. These configurations are described on the next table.

Extra configurations of Siemens Moby Driver

CONFIGURATION	DESCRIPTION	DEFAULT VALUE
Mode	Operation mode. Possible values are **MOBY I** or **MOBY U**	MOBY U
Multitag/bunch	Number of MDS devices that can be processed in the are of an antenna, or zone 1 (one). Possible values range between 1 (one) and 12	1 (one)
Standby time	Time in which an MDS device must enter a stand-by mode after an MDS command is executed. This means that an MDS device remains active to process the next command, which must appear during that stand-by time, without delay. This value specifies a factor of 7 (seven) milliseconds and not an absolute time. For example, the value 10 corresponds to 10 × 7 ms, or 70 ms. The value 0 (zero) indicates the absence of a stand-by, that is, an MDS device enters inactivity mode again after each communication with an MDS command. Values between 1 (one) and 200 correspond to values between 7 ms and 1400 ms	30
Proximity switch mode (fcon)	Values of modes. Possible values are **0**: Mode 1 or without proximity switches or SLG syncing, **1**: Mode 2 or one or two proximity switches are turned on in an **OR** logical operation. While the first or second switch is turned on, the field remains turned on. Otherwise, the field remains turned off, **2**: Mode 3 or one or two proximity switches. The first switch turns on the field and the second switch turns off the field. When there are two proximity switches and the time of the proximity	0 (zero)

CONFIGURATION	DESCRIPTION	DEFAULT VALUE
	field is configured, the field is automatically turned off when the second proximity switch does not change its status inside that proximity time. When there are no proximity switch, the time of the proximity switch must be configured. After that time, the field is automatically turned off, or **3**: Mode 4 or SLG syncing. For more information, please check Moby U Manual for configuration, installation and services	
Distance limit (dili)	Distance limit. Values directly correspond to distances in meters, divided by 10. For example, the value 5 (five) corresponds to 0,5 m, the value 10 corresponds to 1,0 m, the value 15 corresponds to 1,5 m, the value 20 corresponds to 2,0 m, the value 25 corresponds to 2,5 m, the value 30 corresponds to 3,0 m, and the value 35 corresponds to 3,5 m. Users must add the value 128 when the sending capacity is reduced	30
Proximity switch time (ftim)	Time value. Possible values are **0**: **ftim** equal to 0 (zero, when **fcon** is equal to zero) or **1 to 255**: **ftim** between 1 (one) and 255 seconds	0 (zero)
ANW	Select this option for operations with presence check and deselect this option for operations without presence check	Selected

Tag Reference

The Tags of this Driver are referenced to commands by using the configuration of *N1* or *B1* parameters. The *N* or *B* parameters not mentioned are irrelevant in the configuration of Tags. It is recommended leaving them in 0 (zero).

Moby Status

Read-Only

B1	0 (zero)
B2	Not used, leave it in 0 (zero)
B3	Not used, leave it in 0 (zero)
B4	Not used, leave it in 0 (zero)

This Tag is used to check the status of an SLG, SLA, ASM, or SIM device. It can have up to 8 (eight) Elements, described on the next table.

Elements of a Block Tag Moby Status

ELEMENT	DESCRIPTION
1	ErrorCode
2	EccDone
3	DiagBatMDS507
4	BatMDS
5	AnwMDS
6	BusyASM
7	Dummy
8	Error

Moby Status U

Read-Only

B1	1 (one)
B2	Not used, leave it in 0 (zero)
B3	Not used, leave it in 0 (zero)
B4	Not used, leave it in 0 (zero)

This Tag is used to check the status from an SLG U92 device. It can have up to 26 Elements, described on the next table.

Elements of a Block Tag Moby Status U

ELEMENT	DESCRIPTION
1	ErrorCode
2	EccDone
3	DiagBatMDS507
4	BatMDS
5	AnwMDS
6	BusyASM
7	Dummy
8	Error
9	S_info
10	Hw_type
11	Hw_ver

ELEMENT	DESCRIPTION
12	Boot_ver
13	Fw_type
14	Fw_ver
15	Drv_type
16	Drv_ver
17	Interf
18	Baud
19	Díli
20	Mtag
21	Fcon
22	Ftim
23	Sema
24	Ant
25	Standby
26	Anw

Moby Diagnose Function Calls

Read-Only

B1	2 (two)
B2	2 (two)
B3	Not used, leave it in 0 (zero)
B4	Not used, leave it in 0 (zero)

This Tag is used to check diagnosis data from an SLG device, in which the last *n* function calls are requested. It can have up to 15 Elements, described on the next table.

NOTE

This Tag returns a list of data equal to the number of registers in each diagnosis mode. If there are *n* registers stored, this Tag contains a list of *n* blocks.

Elements of a Block Tag Moby Diagnose Function Calls

ELEMENT	DESCRIPTION
1	ErrorCode
2	EccDone
3	DiagBatMDS507

ELEMENT	DESCRIPTION
4	BatMDS
5	AnwMDS
6	BusyASM
7	Dummy
8	Error
9	Data [0]
10	Data [1]
11	Data [2]
12	Data [3]
13	Data [4]
14	Data [5]
15	Data [6]

Moby Diagnose Error Messages

Read-Only

B1	2 (two)
B2	3 (three)
B3	Not used, leave it in 0 (zero)
B4	Not used, leave it in 0 (zero)

This Tag is used to check diagnosis data from an SLG device, in which the last n error messages are requested. It can have up to 9 (nine) Elements, described on the next table.

NOTE
This Tag returns a list of data equal to the number of registers in each diagnosis mode. If there are *n* registers stored, this Tag contains a list of *n* blocks.

Elements of a Block Tag Moby Diagnose Error Messages

ELEMENT	DESCRIPTION
1	ErrorCode
2	EccDone
3	DiagBatMDS507
4	BatMDS
5	AnwMDS
6	BusyASM

ELEMENT	DESCRIPTION
7	Dummy
8	Error
9	Error message

Moby Diagnose Identified MDSs

Read-Only

B1	2 (two)
B2	4 (four)
B3	Not used, leave it in 0 (zero)
B4	Not used, leave it in 0 (zero)

This Tag is used to check diagnosis data from an SLG device, in which the last *n* identified MDS devices are requested. It can have up to 12 Elements, described on the next table.

NOTE

This Tag returns a list of data equal to the number of registers in each diagnosis mode. If there are *n* registers stored, this Tag contains a list of *n* blocks.

Elements of a Block Tag Moby Diagnose Identified MDSs

ELEMENT	DESCRIPTION
1	ErrorCode
2	EccDone
3	DiagBatMDS507
4	BatMDS
5	AnwMDS
6	BusyASM
7	Dummy
8	Error
9	Data [0]
10	Data [1]
11	Data [2]
12	Data [3]

Moby Status MDS

Leitura ou Escrita

B1	3 (three)
B2	Configure with the value 1 (one) to determine the lifetime of a battery or the value 0 (zero) to leave the lifetime of a battery undetermined
B3	Not used, leave it in 0 (zero)
B4	Not used, leave it in 0 (zero)

This Tag is used to check status and diagnosis data from an MDS device. It can have up to 15 Elements, described on the next table.

Elements of a Block Tag Moby Status MDS

ELEMENT	DESCRIPTION
1	ErrorCode
2	EccDone
3	DiagBatMDS507
4	BatMDS
5	AnwMDS
6	BusyASM
7	Dummy
8	Error
9	Mds_no
10	Mds_type
11	Strz
12	Ssmz
13	Mcod
14	Rbld
15	Sleep_time

Moby Memory

Leitura ou Escrita

N1 or B1	4 (four)
N2 or B2	Initial address of data located in an MDS device
N3 or B3	Not used, leave it in 0 (zero)
N4 or B4	Not used, leave it in 0 (zero)

This Tag is used to read or write data from or to an MDS device located in the antenna field of an SLG, SLA, or SIM device. This is a generic reading or writing, because an MDS device cannot be identified.

This Tag can be both a PLC Tag or a Block Tag, In case of a PLC Tag or a Block Tag with 1 (one) Element, it represents a simple **Byte**-type variable read from the referenced memory position. For Block Tags with more than 1 (one) Element, it represents a reading of blocks of memory bytes in sequence, in which each Element reflects a different memory position. To read data in an optimized way for several contiguous memory elements in a device, always prefer reading with Block Tags.

Moby Get ID

Read-Only

N1	5 (five)
N2	Not used, leave it in 0 (zero)
N3	Not used, leave it in 0 (zero)
N4	Not used, leave it in 0 (zero)

This Tag is used to read an identification number, or serial number, of an MDS device. This reading is only possible for MDS MOBY E, MOBY F, and MOBY U devices. The **Value** property contains that identification number.

Moby ANW Status

Read-Only

N1	0 (zero)
N2	Not used, leave it in 0 (zero)
N3	Not used, leave it in 0 (zero)
N4	Not used, leave it in 0 (zero)

This Tag is used to report the presence of MDS devices in the antenna field of an SLG device. The **Value** property contains the number of MDS devices in the antenna field of an SLG device.

Documentation of I/O Interfaces

This section contains the documentation of I/O Interfaces referring to **Moby** 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 **Elipse 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 **Elipse E3** version 2.0 or later, click **Configure driver** on a Driver's toolbar. In **Elipse 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.

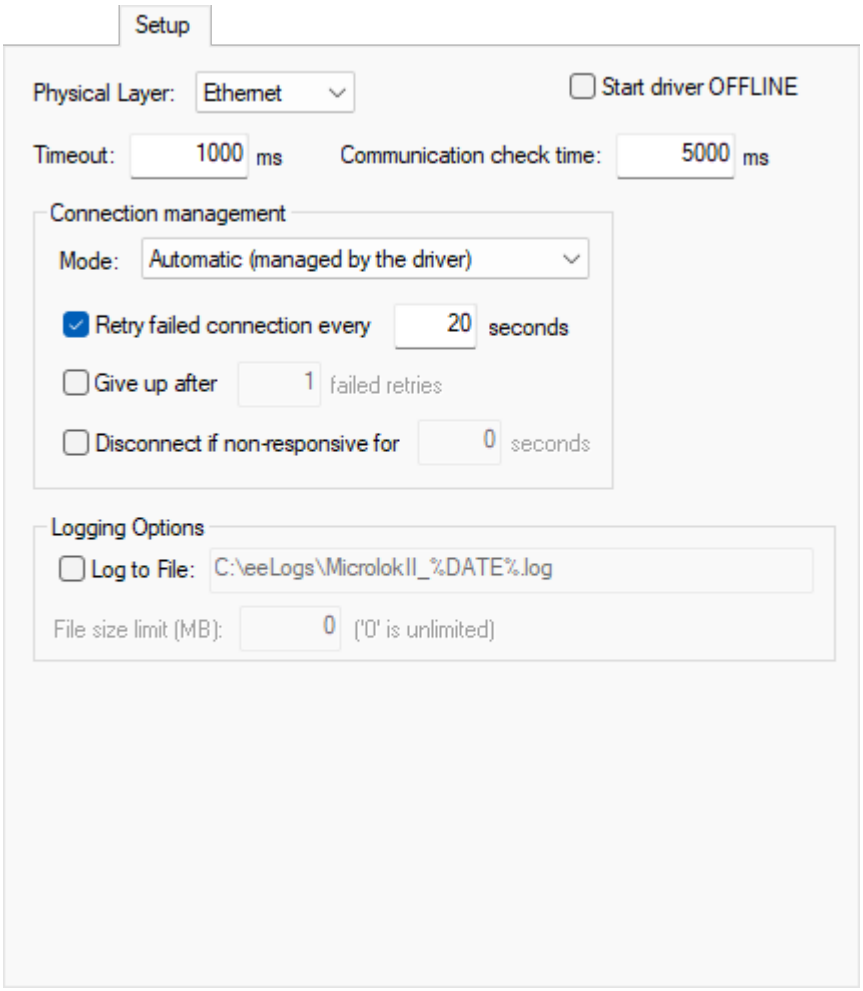
Configuration Dialog Box

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



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
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
Log to File	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 **0FDAh**. 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**
File size limit (MB)	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

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 vlaue **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 **Elipse 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 **Elipse 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 **Elipse 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
2.0.1	05/15/2026	M. Ludwig	• Driver updated to **IOKit** library version **3.0** and Visual Studio 2022 (*Case 39024*).
1.0.1	07/09/2010	M. Ludwig	• Initial version of this Driver.

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 2014 Cool Vendor

OPC FOUNDATION MEMBER

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)