ANSI AnsiC12xx Driver

Filename	AnsiC12xx.dll
Manufacturer	ANSI
Devices	N/A
Protocol	ANSI C12.18, ANSI C12.19, and ANSI C12.21
Version	1.0.22
Last Update	06/13/2025
Platform	Win32
Dependencies	IOKit version 2.0 or later
Superblock Readings	No
Level	0

Introduction

This Driver implements the **C12.18** and **C12.21** communication protocols, defined by ANSI (*American National Standards Institute*), requesting data from tables according to the description of the **C12.19** standard, which allows **Elipse Software** products to communicate with power metering devices.

ANSI C12 is a set of standards for meters. **ANSI C12.18** is an ANSI standard that describes a protocol used for two-way communications with a meter. **C12.18** standard is written specifically for communications with meters via an ANSI type 2 optical port and specifies lower-level protocol details. **ANSI C12.19** specifies the data tables to use. **ANSI C12.21**, on the other hand, is an extension of **C12.18** protocol written for modem instead of optical communications.

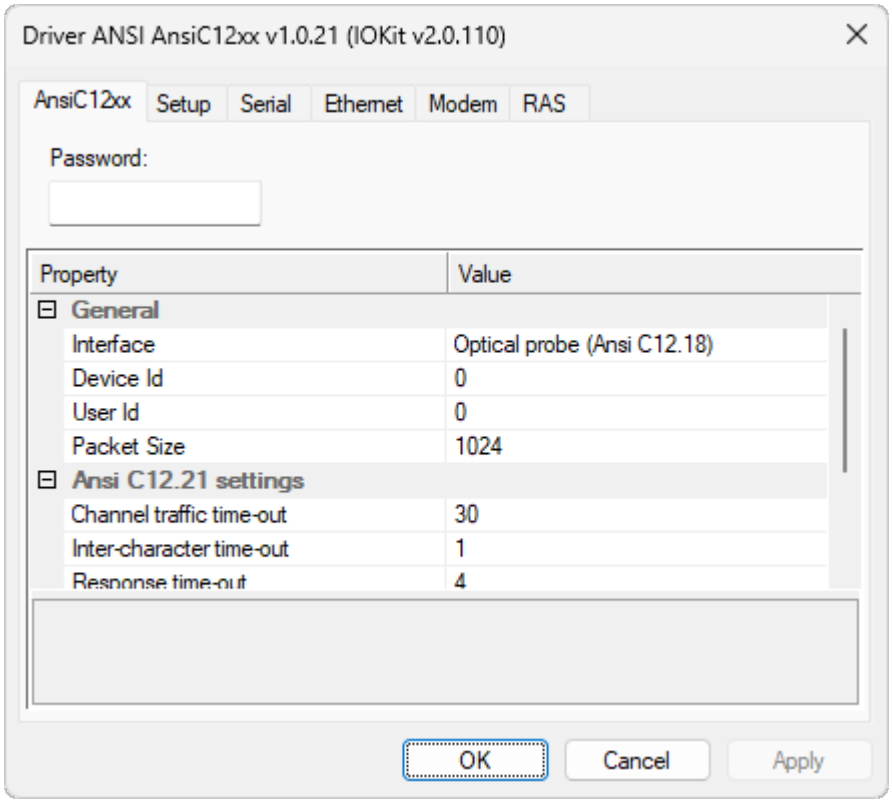
This Driver was developed using **Elipse Software**'s **IOKit** library. This library, after user configuration, is responsible for implementing the physical layer access, that is, **Serial**, **Ethernet**, **Modem**, or **RAS**. For more information, please check topic **Documentation of I/O Interfaces**.

Meters already tested with this Driver are **Itron Sentinel** and **Elster A1800**.

Driver Settings

This Driver's **[P]** parameters are not used. All settings are performed on this Driver's settings dialog box. For more information about **Setup**, **Serial**, **Ethernet**, **Modem**, and **RAS** tabs, please check topic **Documentation of I/O Interfaces**.

The **AnsiC12xx** tab contains specific settings for this Driver.



AnsiC12xx tab

The available options on this tab are described on the next table.

Available options on the AnsiC12xx tab

PARAMETER	DESCRIPTION
Password	Password used as a way to select access permissions. This password is sent to a meter and then compared with the stored password to grant access to its tables. Its maximum size is 20 characters
General - Interface	Indication of a type of interface used by a meter to communicate. The available options are **Optical probe (Ansi C12.18)** or **Modem (Ansi C12.21)**
General - Device Id	Identifier of a device. Defines a value that identifies an **ANSI C12.19** device on both request and response packets. The default value is 0 (zero)
General - User Id	User identifier. Defines a code, optionally stored by an **ANSI C12.18** device, indicating the identity of the operator who requested the creation of a session. The default value is 0 (zero)
General - Packet Size	Maximum size of packets as the maximum number of octets that can communicate during transactions with a meter. The minimum value is 64 octets and the maximum value is 8192 octets. If a meter negotiates a lower value, the configured value is used as the size for requests to read segments from the **ST-64** table. The default value is 1024
Ansi C12.21 settings - Channel traffic time-out	Time-out of a channel's traffic, in seconds. The ending device finishes all communications after waiting for some time for a valid packet or confirmation. The default value is 30 and the minimum allowed value is 1 (one)

PARAMETER	DESCRIPTION
Ansi C12.21 settings - Inter-character time-out	Time-out between octets, in seconds. The packet's recipient must handle the case when a packet segment is lost. A time-out between characters is defined as the minimum amount of time a recipient must wait between characters within a packet when receiving data over the communication channel. The recipient must wait at least this time-out before stopping to wait for the remaining package and send a denied acknowledgment signal. The default value and the minimum allowed value are equal to 1 (one)
Ansi C12.21 settings - Response time-out	Time-out of a response, in seconds. The packet's sender must handle the case when a denied acknowledgment or confirmation is lost. A response time-out is defined as the minimum amount of time a sender must wait after a packet is sent to receive an acknowledgment over the communication channel. If this time-out elapses, the sender sends the packet again. The default value is 4 (four) and the minimum allowed value is 1 (one)
Ansi C12.21 settings - Attempt retries	Maximum number of retries. The default value is 3 (three) and the maximum allowed value is 10

NOTE

The settings on the **Ansi C12.21 settings** group are only relevant when selecting the **Modem (Ansi C12.21)** item on the **General - Interface** option.

Performing a Data Collecting

The AnsiC12xx Driver was designed to turn an application created with an **Elipse Software**'s product into a telemetering application. Therefore, most likely the main usage is to perform data collecting tasks.

The main data collecting operations that can be performed on meters implementing **Ansi C12.19** data tables are **Load Profile**, **History Log, and Event Log**.

Reading a Load Profile

Reading a **Load Profile** of a meter is transferring and interpreting channel measurements with consumption data. Users generally refer to Load Profile collecting as **Mass Memory Reading**. To perform a Load Profile reading, execute the next procedures.

1. Write to a **Start Read** Tag (1, 64, 0, 0) with two Elements with the initial and final date and time to start the transfer process.
2. If needed, monitor the transfer by reading the **Status Read** Tag (2, 64, 0, 0).
3. When the transfer ends, read the **Read Table** Tag configured for the **ST-64** table (0, 64, 0, 0).
4. If needed, read the **Read Profile Channels** Tag (4, 0, 0, 0) to interpret each one of the Load Profile channels.

History and Event Logs

Although very similar in design, the role of **History Log** data is significantly different from **Event Log** data. Event Log data are designed to facilitate establishing and maintaining consumer's confidence on a meter's operation. For that reason, the

implementation of an event logger must comply with strict guidelines and uniform rules consistent with the implementation of an audit trail capable of controlling all meter changes to metrology entities. On the other hand, a History logger provides monitoring capabilities of any meter activity or events that can be of general interest for operators. Both History logger and Event logger may coexist in a meter. To perform a reading of History Logs, execute the next procedures.

1. Write to a **Start Read** Tag (1, 74, 0, 0) with two Elements with the initial and final date and time to start the transfer process.
2. If needed, monitor the transfer by reading the **Status Read** Tag (2, 74, 0, 0).
3. When the transfer ends, read the **Read Table** Tag configured for the **ST-74** table on section zero (0, 74, 0, 0) and another Tag on section one (0, 74, 1, 0).
4. To perform an Event Log reading, create the same Tags with the *N2* or *B2* parameter from the previous steps with the value 76.

Tag Reference

This section contains information about the configuration of **[N/B]** Tags of this Driver.

Read Table

Read-Only

N1/B1	0 (zero)
N2/B2	Number of an **ANSI C12.19** table
N3/B3	Number of a table section, if this is a table with sections
N4/B4	Not used, leave it in 0 (zero)

The **ANSI C12.19** standard defines a table structure to pass utility application data between a meter and a computer. The purpose of these tables is to define structures for data transport to and from meters. The **Read Table** Tag performs a data reading from a table, which can already be cached or can be transferred from a meter immediately when executing. Depending on the type of table, the execution of data transfer behaves in different ways. Ordinary tables perform data transfers synchronously, but vector tables require an execution of the **Start Read** Tag so that a data transfer previous to execution occurs and then data can be read.

Ordinary Tables

Ordinary Tables are the simplest tables, which perform a reading to perform communication with a device and display values right after finishing it.

ST-00: General Configuration

Read-Only

B1	0 (zero)
B2	0 (zero)
B3	Not used, leave it in 0 (zero)
B4	Not used, leave it in 0 (zero)
Size	22

- **Element 1**: Format control 1 (one, bit field)
- **Element 2**: Format control 2 (two, bit field)
- **Element 3**: Format control 3 (three, bit field)
- **Element 4**: Device class (**String**)
- **Element 5**: Nameplate type. Possible values are **0**: Gas, **1**: Water, or **2**: Electrical
- **Element 6**: Default configuration. Possible values are **0**: Not used, **1**: Simple meter register, **2**: Simple demand meter, **3**: Simple TOU meter, or **4**: Simple profile recorder
- **Element 7**: Maximum length of procedure parameters
- **Element 8**: Maximum length returned by procedures
- **Element 9**: Version of standard tables
- **Element 10**: Revision number of standard tables
- **Element 11**: Number of octets required to represent standard tables
- **Element 12**: Number of octets required to represent manufacturer's tables
- **Element 13**: Number of octets required to represent standard procedures
- **Element 14**: Number of octets required to represent manufacturer's procedures
- **Element 15**: Number of octets allocated for specific manufacturer's status indicators
- **Element 16**: Number of sets of pending statuses on the table of pending statuses (table 4)
- **Element 17**: **String** as hexadecimal characters indicating which standard tables are implemented by a device
- **Element 18**: **String** as hexadecimal characters indicating which manufacturer's tables are implemented by a device
- **Element 19**: **String** as hexadecimal characters indicating which standard procedures are implemented by a device
- **Element 20**: **String** as hexadecimal characters indicating which manufacturer's procedures are implemented by a device
- **Element 21**: **String** as hexadecimal characters indicating which standard tables are writable
- **Element 22**: **String** in hexadecimal characters indicating which manufacturer's tables are writable

ST-05: Device Identification

Read-Only

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

- **Value property**: Identification (**String** with 20 character used to uniquely identify a meter)

ST-06: Util Info

Read-Only

B1	0 (zero)
B2	6 (six)
B3	Not used, leave it in 0 (zero)
B4	Not used, leave it in 0 (zero)
Size	19

- **Element 1**: Owner's name
- **Element 2**: Utility's division number
- **Element 3**: Service point's identifier number
- **Element 4**: Logical electrical address of a device
- **Element 5**: Hardware's attached identifier
- **Element 6**: Utility's specific serial number
- **Element 7**: Identifier of a customer
- **Element 8**: X coordinate
- **Element 9**: Y coordinate
- **Element 10**: Z coordinate
- **Element 11**: Identifier of a billing tariff
- **Element 12**: Name 1 (one) of the manufacturer providing configuration or programming software
- **Element 13**: Number 1 (one) of software's version number

- **Element 14**: Number 1 (one) of software's revision number
- **Element 15**: Name 2 (two) of the manufacturer providing configuration or programming software
- **Element 16**: Number 2 (two) of software's version number
- **Element 17**: Number 2 (two) of software's revision number
- **Element 18**: Name of the last programmer or programming device
- **Element 19**: Miscellaneous identifier

ST-11: Actual Data Sources Limiting

Read-Only

B1	0 (zero)
B2	11
B3	Not used, leave it in 0 (zero)
B4	Not used, leave it in 0 (zero)
Size	8 (eight)

- **Element 1**: Bit fields indicating demand and register functions a device can execute
- **Element 2**: Number of entries on Table 12
- **Element 3**: Number of entries on Table 13
- **Element 4**: Number of octets for each element on Table 14
- **Element 5**: Number of entries on Table 14
- **Element 6**: Number of entries on Tabela 15
- **Element 7**: Selector of constants, type of register on Table 15
- **Element 8**: Number of entries on Table 16

ST-21: Act Regs

Read-Only

B1	0 (zero)
B2	21
B3	Not used, leave it in 0 (zero)
B4	Not used, leave it in 0 (zero)
Size	10

- **Element 1**: **RegFunc1** flags (bits field)
- **Element 2**: **RegFunc2** flags (bits field)
- **Element 3**: Number of auto-readings
- **Element 4**: Number of sum registers in each data block
- **Element 5**: Number of demand blocks in each data block
- **Element 6**: Number of coinciding values
- **Element 7**: Number of occurrences
- **Element 8**: Number of tiers
- **Element 9**: Number of demands available
- **Element 10**: Number of values available

ST-52: Clock

Read-Only

B1	0 (zero)
B2	52
B3	Not used, leave it in 0 (zero)
B4	Not used (leave it in zero)
Size	2 (two)

- **Element 1**: Date and time in **TIME** format
- **Element 2**: Value of quality

ST-53: Time Offset

Read-Only

B1	0 (zero)
B2	53
B3	Not used, leave it in 0 (zero)
B4	Not used, leave it in 0 (zero)
Size	3 (three)

- **Element 1**: The time of the day in which a meter shifts to daylight saving time from standard time
- **Element 2**: Number of minutes to adjust for daylight saving time
- **Element 3**: Timezone offset

ST-55: Clock State

Read-Only

B1	0 (zero)
B2	55
B3	Not used, leave it in 0 (zero)
B4	Not used, leave it in 0 (zero)
Size	3 (three)

- **Element 1**: Current date and time of a device in **TIME** format
- **Element 2**: Value of quality
- **Element 3**: Status

ST-60: Load Profile Dimension Limits

Read-Only

B1	0 (zero)
B2	60
B3	Not used, leave it in 0 (zero)
B4	Not used, leave it in 0 (zero)
Size	7 (seven)

This table specifies the maximum dimensional values, the control selectors, and data options of a Load Profile.

- **Element 1**: Memory size of a Load Profile
- **Element 2**: Flags of a Load Profile
- **Element 3**: Flags of acceptable formats of a Load Profile
- **Element 4**: Maximum number of blocks that can be contained in set 1 (one) of a Load Profile (table **64**)
- **Element 5**: Maximum number of intervals per block that can be contained in set 1 (one) of a Load Profile's data (table **64**)
- **Element 6**: Maximum number of channels in set 1 (one) of a Load Profile's data (table **64**)
- **Element 7**: Maximum time, in minutes, for the duration of an interval that can be contained in set 1 (one) of a Load Profile's data (table **64**)

ST-61: Actual Load Profile

Read-Only

B1	0 (zero)
B2	61
B3	Not used, leave it in 0 (zero)
B4	Not used, leave it in 0 (zero)
Size	7 (seven)

This table specifies the actual dimensional values of data from a Load Profile.

- **Element 1**: Memory size of a Load Profile
- **Element 2**: Flags of a Load Profile
- **Element 3**: Flags of acceptable formats of a Load Profile
- **Element 4**: Number of blocks that can be contained in set 1 (one) of a Load Profile (table **64**)

- **Element 5**: Number of intervals per block that can be contained in set 1 (one) of a Load Profile (table **64**)
- **Element 6**: Number of channels in set 1 (one) of a Load Profile's data (table **64**)
- **Element 7**: Time, in minutes, for the duration of an interval that can be contained in set 1 (one) of a Load Profile's data (table **64**)

ST-63: Load Profile Status

Read-Only

B1	0 (zero)
B2	63
B3	Not used, leave it in 0 (zero)
B4	Not used, leave it in 0 (zero)
Size	6 (six)

- **Element 1**: Status flags of a Load Profile set (bit fields)
- **Element 2**: Number of valid data blocks of a Load Profile in their respective tables
- **Element 3**: Index of the most recent element of the data block array on a Load Profile data array
- **Element 4**: Sequence number of the most recent valid data block on a Load Profile data array
- **Element 5**: Number of blocks not yet read
- **Element 6**: Number of valid intervals generated on the most recent block of a Load Profile

ST-71: Actual Log Limiting Table

Read-Only

B1	0 (zero)
B2	71
B3	Not used, leave it in 0 (zero)
B4	Not used, leave it in 0 (zero)
Size	7 (seven)

- **Element 1**: Status flags (bit fields)
- **Element 2**: Actual number of octets in the supported set of standard events (table 72)

- **Element 3**: Actual number of octets in the supported set of events (table 72)
- **Element 4**: Actual number of octets in historic arguments (table **74**)
- **Element 5**: Actual number of octets in event arguments (table 76)
- **Element 6**: Actual maximum number of entries in the history log
- **Element 7**: Actual maximum number of entries in the event log

Vector Tables

Vector Tables are tables presented bi-dimensionally, that is, they are data blocks with size *S* arranged on a list with dimension *N*, returned at once in a Block Tag. Therefore, if there is cached data during the execution of a reading of Tags from these array tables, a list with *N* blocks is returned, in which *N* is the number of data blocks read. Otherwise, returns an empty list.

ST-12: Units of Measure Entry

Read-Only

N1	0 (zero)
N2	12
N3	Not used, leave it in 0 (zero)
N4	Not used, leave it in 0 (zero)

- **Value property**: Measurement unit's entry (a 32-bit value separated in bit fields)

NOTE

This table requires a writing to the **Start Read** Tag to command its data transfer in background.

ST-15: Primary Metering Information

Read-Only

B1	0 (zero)
B2	15
B3	Not used, leave it in 0 (zero)
B4	Not used, leave it in 0 (zero)
Size	5 (five)

- **Element 1**: Multiplier

- **Element 2**: Offset
- **Element 3**: Flags to apply constants. Possible values are **0**: Must be applied or **1**: Already applied
- **Element 4**: Ratio of current transformer
- **Element 5**: Ratio of voltage transformer

NOTE

This table requires a writing to the **Start Read** Tag to command its data transfer in background.

ST-16: Sources Definition

Read-Only

N1	0 (zero)
N2	16
N3	Not used, leave it in 0 (zero)
N4	Not used, leave it in 0 (zero)

This table contains information regarding the sources selected by other tables.

- **Value property**: Source flags (an 8-bit value separated in bit fields)

NOTE

This table requires a writing to the **Start Read** Tag to command its data transfer in background.

ST-62: Load Profile Control

Read-Only

B1	0 (zero)
B2	62
B3	Not used, leave it in 0 (zero)
B4	Not used, leave it in 0 (zero)
Size	5 (five)

- **Element 1**: Control flags
- **Element 2**: Selected source

- **Element 3**: End Block Read Source Select
- **Element 4**: Scalar
- **Element 5**: Divider

NOTE

This table requires a writing to the **Start Read** Tag to command its data transfer in background.

ST-64: Load Profile Data Set

Read-Only

B1	0 (zero)
B2	64
B3	Not used, leave it in 0 (zero)
B4	Not used, leave it in 0 (zero)
Size	Determined by the number of channels of a Load Profile

NOTE

This table requires a writing to the **Start Read** Tag, with an initial and final date and time interval, to command its data transfer in background.

This Tag is presented as a list of *N* blocks, in which *N* is the number of data blocks read and determined between the time space between the initial date and time and the ending date and time written in the **Start Read** Tag. If there is no data, returns an empty list.

The structure of blocks of this list cannot be determined without knowing the number of channels of the Load Profile, so the size of the blocks is determined at run time during the reading of available data on other tables. The size of blocks (*S*) is determined by twice the number of channels (*C*) plus one, that is, $S = 1 + (2 \times C)$. For example, for 4 (four) channels the size of blocks is 9 (nine), that is, $S = 1 + (2 \times 4) \rightarrow S = 1 + 8 \rightarrow S = 9$.

The disposition of elements of each one of these blocks is the following:

- **Element 1**: Common flags. For more information, please check **Note 1**
- **Element 2**: Value of Channel 1 (one) with an applied scale
- **Element 3**: Status of Channel 1 (one). For more information, please check **Note 2**
- **Element 4**: Value of Channel 2 (two) with an applied scale
- **Element 5**: Status of Channel 2 (two). For more information, please check **Note 2**
- ...
- **Element S - 1**: Value of Channel ($S \div 2$) with an applied scale

- **Element S**: Status of Channel (*S* ÷ 2). For more information, please check **Note 2**

The timestamp of each Block Tag corresponds to the date and time of a demand interval.

NOTE 1

To interpret common flags, users must map the element into bits as follows:

- **Bit 00**: Effective daylight saving time during the interval
- **Bit 01**: Power failed or restored during the interval
- **Bit 02**: Clock reset forward during the interval
- **Bit 03**: Clock reset backwards during the interval

NOTE 2

Interpretation for Channel statuses is the following:

- **0**: No status
- **1**: Overflow
- **2**: Partial interval due to a common status
- **3**: Long interval due to a common status
- **4**: Ignored interval due to a common status
- **5**: Interval contains data from test mode

This is the longest to read in **ANSI C12.18/21** communication, and the duration of transfer time may vary depending on the required time interval.

Users can identify each channel and their measured quantities by performing a reading from the **Read Profile Channels** Tag.

MT-17: Alternate Source Definition

Read-Only

B1	0 (zero)
B2	2065
B3	Not used, leave it in 0 (zero)
B4	Not used, leave it in 0 (zero)
Size	7 (seven)

This table is specific to Elster manufacturer. This Tag is dimensioned as a list and contains 255 blocks with 7 (seven) Elements with the following meanings:

- **Element 1**: Measurement unit (please check table)

- **Element 2**: Net flow (bit field)
- **Element 3**: Billing and profile usage (bit field)
- **Element 4**: Harmonic (bit field)
- **Element 5**: Scale factor
- **Element 6**: Demand selection
- **Element 7**: Multiplier selection

NOTE

This table requires a writing to the **Start Read** Tag to command its data transfer in background.

Vector Tables with Sections

Vector Tables with Sections are vector tables divided into header and data list sections. A full reading of this type of table must be performed by reading all of its sections, otherwise table data cannot be updated again.

ST-74: History Log Data

Read-Only

B1	0 (zero)
B2	74
B3	Number of a table section. Possible values are **0**: Header data or **1**: Registry data
B4	Not used, leave it in 0 (zero)
Size	Depends on the section

A full reading of this table must be performed by reading all of its sections, otherwise table data cannot be updated again.

NOTE

This table requires a writing to the **Start Read** Tag, with initial and final date and time interval, to command its data transfer in background.

Section 0

Read-Only

B1	0 (zero)
B2	74
B3	0 (zero)
B4	Not used, leave it in 0 (zero)
Size	5 (five)

- **Element 1**: Event flags
- **Element 2**: Number of valid entries
- **Element 3**: Element of the last entry
- **Element 4**: Sequential number of the last entry
- **Element 5**: Number of unread entries

Section 1

Read-Only

B1	0 (zero)
B2	74
B3	1 (one)
B4	Not used, leave it in 0 (zero)
Size	6 (six)

The next Block Tags of this List are Tags with six Elements that are repeated in a number equal to the value of the **Element 2 (two) of Section 0 (zero)**. The Elements of this Block Tag are the following:

- **Element 1**: Event number
- **Element 2**: Sequential number of the event linked to History Log
- **Element 3**: User identifier
- **Element 4**: History code
- **Element 5**: **String** of the history code
- **Element 6**: Block with one to four elements with arguments

NOTE

The timestamp of this Block Tag corresponds to the event's date and time.

ST-76: Event Log Data

Read-Only

B1	0 (zero)
B2	76
B3	Number of a table section. Possible values are **0**: Header data or **1**: Registry data
B4	Not used, leave it in 0 (zero)
Size	Depends on the section

A full reading of this table must be performed by reading all of its sections, otherwise table data cannot be updated again.

NOTE

This table requires a writing to the **Start Read** Tag, with initial and final date and time interval, to command its data transfer in background.

Section 0

Read-Only

B1	0 (zero)
B2	76
B3	0 (zero)
B4	Not used, leave it in 0 (zero)
Size	5 (five)

- **Element 1**: Event flags
- **Element 2**: Number of valid entries
- **Element 3**: Element of the last entry
- **Element 4**: Sequential number of the last entry

- **Element 5**: Number of unread entries

Section 1

Read-Only

B1	0 (zero)
B2	76
B3	1 (one)
B4	Not used, leave it in 0 (zero)
Size	6 (six)

The next Block Tags of this List are Tags with six Elements that are repeated in a number equal to the value of the **Element 2 (two) of Section 0 (zero)**. The Elements of this Block Tag are the following:

- **Element 1**: Event number
- **Element 2**: Sequential number of the event linked to Event Log
- **Element 3**: User identifier
- **Element 4**: Event code
- **Element 5**: **String** of the event code
- **Element 6**: Block with one to four elements with arguments

NOTE

The timestamp of this Block Tag corresponds to the event's date and time.

Start Read

Write-Only

N1/B1	1 (one)
N2/B2	Number of an **ANSI C12.19** table
N3/B3	Not used, leave it in 0 (zero)
N4/B4	Not used, leave it in 0 (zero)

If the value of the N2/B2 parameter is equal to 64, this Tag must be a Block with two Elements, indicating a period of time for a data collecting.

- **Element 1**: Initial date and time of data collecting in **TIME** format

- **Element 2**: Final date and time of data collecting in **TIME** format

For all other tables, this Tag must be a **PLC** type and its content of the **Value** property is not relevant. Tables requiring a writing to command readings are **ST-12**, **ST-15**, **ST-16**, **ST-62**, **ST-64**, **ST-74**, **ST-76**, and **MT-17** (Elster).

Status Read

Read-Only

N1/B1	2 (two)
N2/B2	Number of an **ANSI C12.19** table
N3/B3	Not used, leave it in 0 (zero)
N4/B4	Not used, leave it in 0 (zero)

This Tag displays the reading status of the respective table indicated by the *N2* parameter. This Tag is especially useful when reading tables considered too long and may take tens of seconds to update, such as table **64**. The **Value** property of this Tag displays the reading status in background with number values corresponding to the current status. Possible values are **0**: Idle, **1**: Transfer in progress, **2**: Transfer successful, or **3**: Transfer failed.

Clock Sync

Write-Only

N1/B1	3 (three)
N2/B2	Not used, leave it in 0 (zero)
N3/B3	Not used, leave it in 0 (zero)
N4/B4	Not used, leave it in 0 (zero)

A PLC Tag that writes a new value, in the **Value** property, of the current date and time, in **TIME** format, to a meter's clock.

Read Profile Channels

Reading and Writing

B1	4 (four)
B2	Not used, leave it in 0 (zero)
B3	Not used, leave it in 0 (zero)
B4	Not used, leave it in 0 (zero)
Size	6 (six)

A Block Tag with six Elements that performs a profile reading of each one of the channels. If there is data, a list with *N* blocks is returned, where *N* is the number of data blocks read. Otherwise, returns an empty list.

Writing any value to this Tag forces the enumeration of the channels, which are added to the list of channels to read during the execution of this Tag's reading.

- **Element 1**: Profile index
- **Element 2**: Measurement unit. For more information, please check the **Table of quantities**
- **Element 3**: Quadrant
- **Element 4**: Net flow. Possible values are **0**: Absolute power per time unit or **1**: Net delivered or received
- **Element 5**: Segmentation
  - **0**: Not phase-related or n/a
  - **1**: Phase A to B
  - **2**: Phase B to C
  - **3**: Phase C to A
  - **4**: Neutral to ground or no information
  - **5**: Phase A to Neutral
  - **6**: Phase B to Neutral
  - **7**: Phase C to Neutral
- **Element 6**: Scale factor

Table of quantities

OPTION	QUANTITY
0	Active power (W)
1	Reactive power (VAR)
2	Apparent power (VA)
3	Phasor power (VA = sqrt(W^2 + VAR^2))
4	Quantity power Q(60)

OPTION	QUANTITY
5	Quantity power Q(45)
6	Phasor reactive power
7	Distortion volt-amperes
8	RMS voltage
9	Average voltage
10	RMS voltage squared
11	Instantaneous voltage
12	RMS current
13	Average current
14	RMS current squared
15	Instantaneous current
16	Total harmonic distortion, voltage (IEEE)
17	Total harmonic distortion, current (IEEE)
18	Total harmonic distortion, voltage (Industry Canada)
19	Total harmonic distortion, current (Industry Canada)
20	Voltage phase angle
21	Voltage phase angle XY, in which X and Y are phases defined in phase selector
22	Current phase angle
23	Current phase angle XY, in which X and Y are phases defined in phase selector
24	Power Factor, computed using apparent power
25	Power Factor, computed using phasor power
26	Average power factor
29	Time of day
30	Date
31	Time of day and date
32	Interval Timer
33	Frequency
34	Counter
35	Sense Input
36	Pulse output form A
37	Pulse output form B
38	Pulse output form C
39	Zero flow duration
40	Voltage sag
41	Voltage swell

OPTION	QUANTITY
42	Power outage
43	Voltage excursion low
44	Voltage excursion high
45	Normal voltage level
46	Phase voltage phase imbalance
47	Voltage total harmonic distortion excess
48	Current total harmonic distortion excess
49	Power Outage (End Device)
50	Number of power outages
51	Number of demand resets
52	Number of times programmed
53	Number of minutes on battery carryover
54	Number of inversion tampers
55	Number of inversion tampers
56	Number of removal tampers
57	Number of reprogramming tampers
58	Number of power loss tampers
59	Number of reverse rotation tampers
60	Number of physical tampers
61	Number of encoder tampers
62	Number of watchdog timeouts
63	Number of diagnostic alarm

Documentation of I/O Interfaces

This section contains the documentation of I/O Interfaces referring to the **AnsiC12xx** 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.

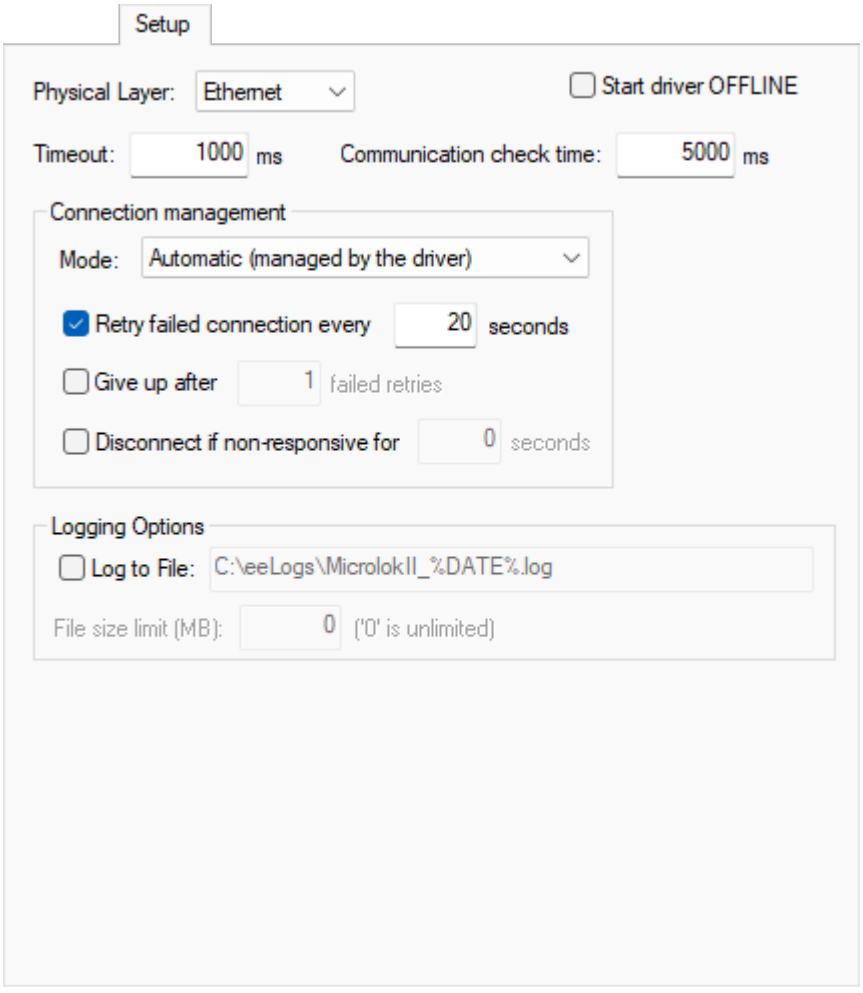
Configuration Dialog Box

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

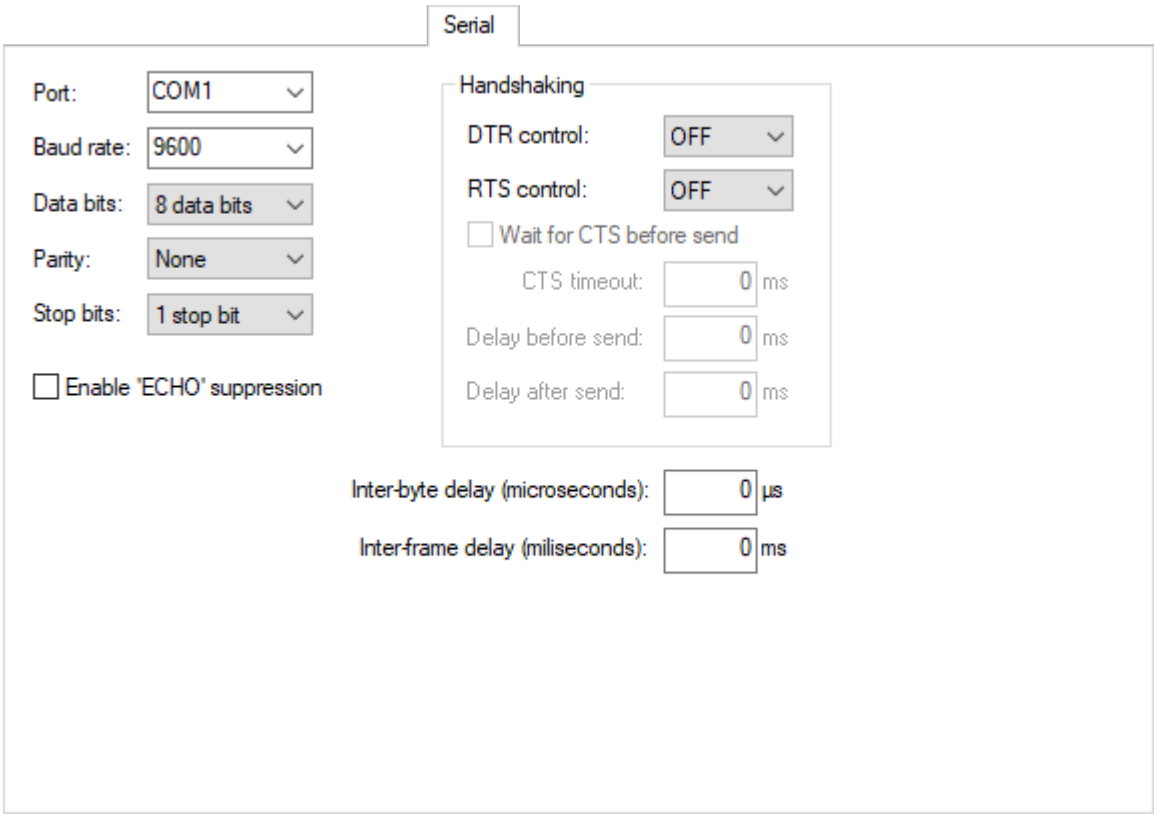
OPTION	DESCRIPTION
Mode	Selects a management mode of a connection. Selecting the **Automatic** option allows a Driver to manage the connection automatically, as specified in the next options. Selecting the **Manual** option allows an application to fully manage a connection
Retry failed connection every ... 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

Serial Tab

Use this tab to configure parameters for a **Serial** Interface.



Serial tab

General options on the Serial tab

OPTION	DESCRIPTION
Port	Select a serial port on the list, from **COM1** to **COM4**, or type the name of a serial port in the format **COM*n***, such as "COM15". When typing the name of a serial port manually, the dialog box only accepts names of serial ports starting with the expression "COM"
Baud rate	Select a baud rate on the list (**1200**, **2400**, **4800**, **9600**, **19200**, **38400**, **57600**, or **115200**) or type a baud rate, such as 600
Data bits	Select 7 (seven) or 8 (eight) data bits on the list
Parity	Select a parity on the list. The available options are **None**, **Even**, **Odd**, **Mark**, or **List**
Stop bits	Select the number of stop bits on the list. The available options are **1**, **1.5**, or **2** stop bits
Enable 'ECHO' supression	Enable this option to remove the echo received after the I/O Interface sends data via serial port. If this echo is not equal to the bytes just sent, then the I/O Interface aborts communication
Inter-byte delay (microseconds)	Defines a delay between each byte transmitted by the I/O Interface, in millionths of a second, that is, 1000000 is equal to a second. This option must be used with small delays of less than a millisecond
Inter-frame delay (milliseconds)	Defines a delay between packets sent or received by the I/O Interface, in thousandths of a second, that is, 1000 is equal to a second. This delay is applied if the I/O Interface

OPTION	DESCRIPTION
	sends two consecutive packets, or between a received packet and the next sending

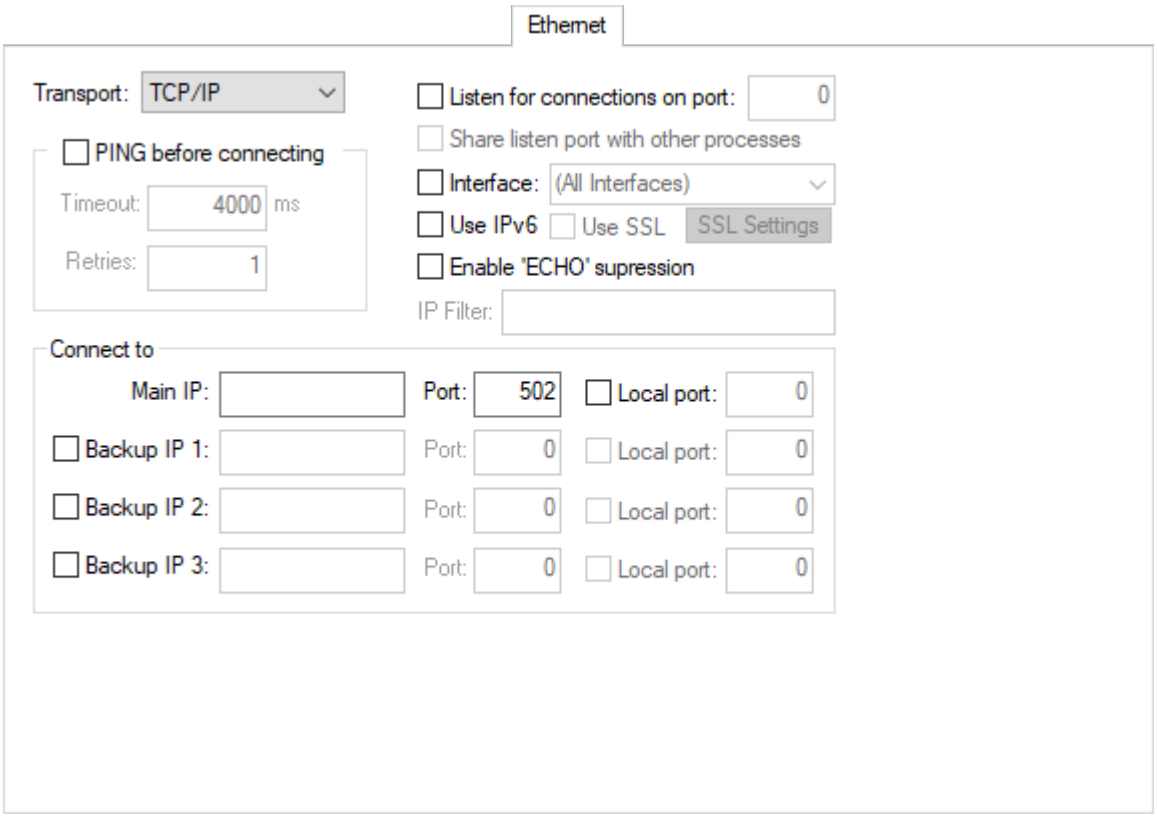
The **Handshaking** group configures the usage of **RTS**, **CTS**, and **DTR** signals in the handshaking process, that is, it controls when data can be sent or received via serial line. Most of the time, configuring the **DTR control** option to **ON** and the **RTS control** option to **Toggle** works with **RS232**-type serial lines as well as with **RS485**-type serial lines.

Available options on the Handshaking group

OPTION	DESCRIPTION
DTR control	Select the value **ON** to keep the **DTR** signal always on while the serial port is open. Select the value **OFF** to turn the **DTR** signal off while the serial port is open. Some devices require the **DTR** signal always on to allow communication
RTS control	Select the value **ON** to keep the **RTS** signal always on while the serial port is open. Select the value **OFF** to turn the **RTS** signal off while the serial port is open. Select the value **Toggle** to turn the **RTS** signal on while sending bytes via serial port and turn it off when not sending bytes, therefore enabling the reception
Wait for CTS before send	Available only when the **RTS control** option is configured with the value **Toggle**. Use this option to force a Driver to check the **CTS** signal before sending bytes via serial port, after turning the **RTS** signal on. In this mode, the **CTS** signal is handled as a permission flag for sending
CTS timeout	Determines a maximum time, in milliseconds, that a Driver waits for the **CTS** signal after turning the **RTS** signal on. If the **CTS** signal is not turned on within this time-out, that Driver then fails the current communication and returns an error
Delay before send	Some serial port devices have a delay when enabling a data sending circuit after the **RTS** signal is turned on. Configure this option to wait a certain number of milliseconds after turning the **RTS** signal on and before sending the first byte. **IMPORTANT**: This delay must be used carefully, because it uses 100% of CPU resources while waiting. System's general performance degrades as this value increases
Delay after send	This is the same effect of the **Delay before send** option, but in this case the delay is performed after sending the last byte, before turning the **RTS** signal off

Ethernet Tab

Use this tab to configure parameters of an **Ethernet** Interface. These parameters, except port configurations, must also be configured for use in the **RAS** Interface.



Ethernet tab

Available options on the Ethernet tab

OPTION	DESCRIPTION
Transport	Select the value **TCP/IP** for a TCP socket (*stream*) or select the value **UDP/IP** to use a UDP socket (*connectionless datagram*)
Listen for connections on port	Use this option to wait for new connections in a specific IP port, common in Slave Drivers. If this option remains unselected, a Driver connects to the address and port specified in the **Connect to** option
Share listen port with other processes	Select this option to share the listening port with other Drivers and processes
Interface	Select the local network interface, identified by its IP address, that a Driver uses to establish and receive connections, or select the value **(All Interfaces)** to allow connection in any network interface
Use IPv6	Select this option to force a Driver to use addresses in **IPv6** format on all Ethernet connections. Leave this option deselected to use the **IPv4** format
Enable 'ECHO' suppression	Enable this option to remove the echo from received data. An echo is a copy of sent data, which can be returned before a reply message
IP Filter	List of restricted or allowed IP addresses from where a Driver accepts connections (*Firewall*). Please check the **IO.Ethernet.IPFilter** property for more information
PING before connecting	Enable this option to execute a **ping** command, that is, to check whether a device can be reached on a network, for a device before trying a socket connection. This is a quick way

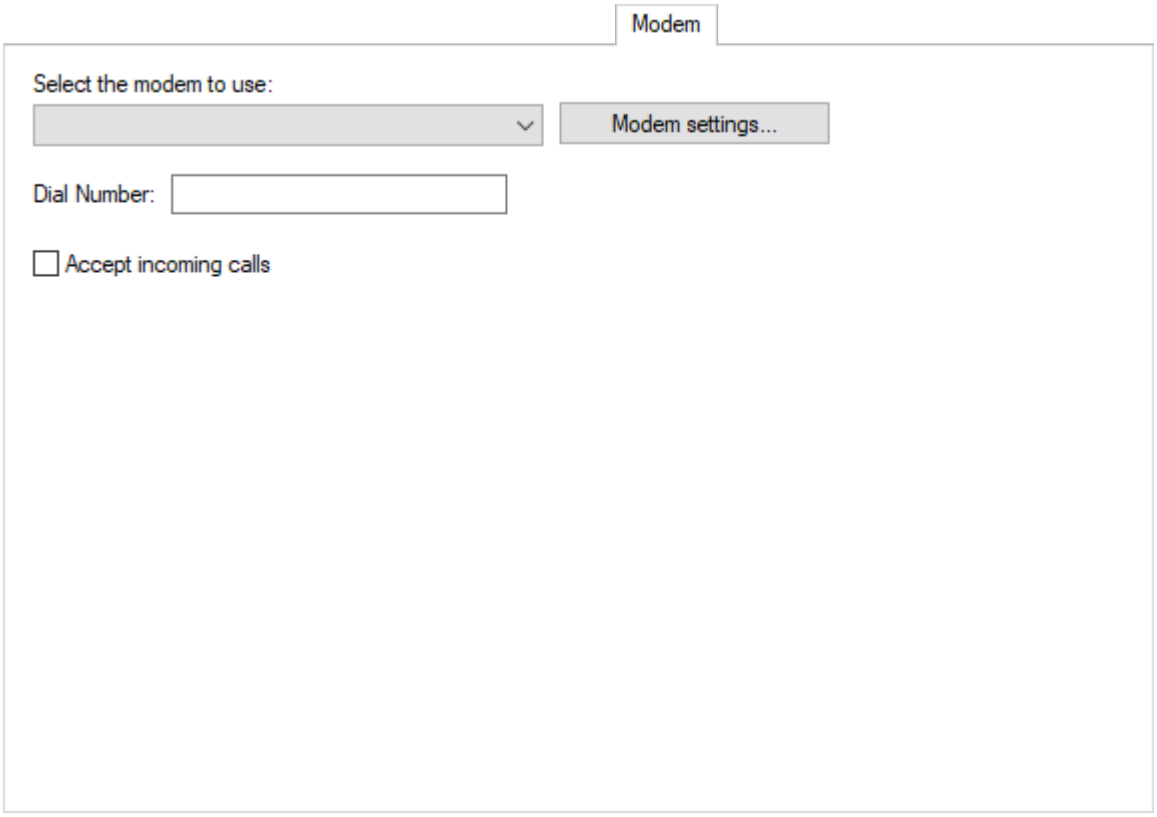
OPTION	DESCRIPTION
	of determining a successful connection before trying to open a socket with a device. The time-out of a connection with a socket can be very high. The available options are: • **Timeout**: Specify the number of milliseconds to wait for a reply from a **ping** command. Users must use a **ping** command to check the normal reply time, configuring this option for a value above that average. Usually this value can be configured between 1000 and 4000 milliseconds, that is, between 1 (one) and 4 (four) seconds • **Retries**: Number of retries of a **ping** command, not counting the first attempt. If all attempts fail, then the socket connection is aborted

Available options on the Connect to group

OPTION	DESCRIPTION
Main IP	Type the IP address of a remote device. Users can use an IP address separated by dots, as well as a URL. In case of a URL, a Driver uses the available DNS service to map that URL to an IP address, such as "192.168.0.13" or "Server1"
Port	Type the IP port of a remote device, between 0 (zero) and 65535
Local port	Select this option to use a fixed local IP port when connecting to a remote device
Backup IP 1, 2, and 3	Indicate the IP address, the IP port, and the fixed local IP port of up to 3 (three) backup addresses of a remote device

Modem Tab

Use this tab to configure parameters of a **Modem** Interface. Some options on the **Serial** tab affect the configuration of a modem, therefore users must also configure the **Serial** Interface.



Modem tab

The **Modem** Interface uses the TAPI modems installed on the computer.

Available options on the Modem tab

OPTION	DESCRIPTION
Select the modem to use	Select a modem on the list of available modems on the computer. If the value **Default modem** is selected, then the first available modem is used. Selecting this option is recommended specially when an application is used on another computer
Modem settings	Click to open the configuration window of the selected modem
Dial Number	Type a default number for dialing. This value can be changed at run time. Users can use the **w** character to represent a pause or a waiting time for a dial tone. For example, "0w33313456" dials the number 0 (zero), waits, and then dials the number "33313456"
Accept incoming calls	Enable this option so that a Driver answers the phone when receiving an external call. To use this option, users must configure the **Connection management** option on the **Setup** tab to the value **Manual**

RAS Tab

Use this tab configure parameters of a **RAS** Interface. Users must also configure the **Ethernet** tab.

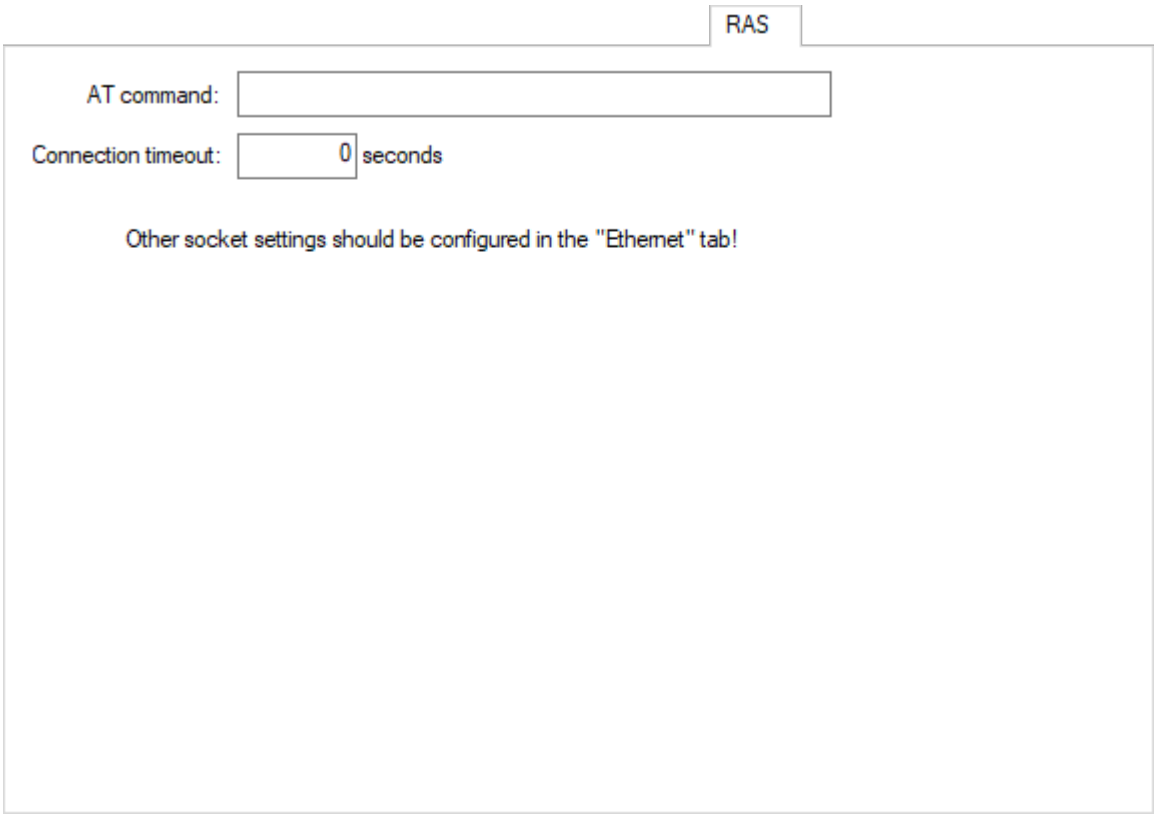
A **RAS** Interface opens a socket connection with a RAS device. A RAS device is a server of modems available through TCP/IP, waiting for socket connections on an IP port. For each connection accepted on this port, users have access to one modem.

When connecting to a RAS device, first the I/O Interface**IOKit** connects to the socket on the IP address and port configured on the **Ethernet** tab. After opening the socket, the following initialization or connection steps are performed:

1. Clears the socket, that is, removes any **TELNET** greeting message received from a RAS device.
2. Sends an **AT** dial message, in **ASCII** format, in the socket.
3. Waits for a **CONNECT** reply.
4. If the time-out expires, the connection is aborted.
5. If the **CONNECT** reply is received within the time-out, the socket is available for communication with a device, that is, the connection was established.

If step 5 (five) is successful, then the socket behaves as a normal socket, with the RAS device working as a router between a Driver and the device. Bytes sent by a Driver are received by the RAS device and sent to the destination device using a modem. Bytes received by the modem's RAS device are sent back to a Driver using the same socket.

After establishing a connection, the **RAS** interface monitors data received by a Driver. If a "NO CARRIER" **String** is found, the socket is closed. If the RAS device does not send a **NO CARRIER** signal, the **RAS** Interface cannot detect when the modem connection between the RAS device and the final I/O device fails. To recover from this failure, users are strongly advised to enable the **Disconnect if non-responsive** option on the **Setup** tab.



RAS tab

Available options on RAS tab

OPTION	DESCRIPTION
AT command	A **String** with the full **AT** command used to dial to a destination device. For example, "ATDT33313456" dials by tone to number "33313456"
Connection timeout	Number of seconds to wait for a modem's **CONNECT** reply, after sending an **AT** command

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

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

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

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

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

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

IO.Type

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.

Ethernet Interface Configuration

This section contains information about the configuration of **I/O Tags** and **Properties** of an **Ethernet** Interface.

I/O Tags

Tags of an Ethernet Interface (N2/B2 = 4)

The Tags described next allow controlling and identifying an **Ethernet** Interface at run time and they are also valid when the **RAS** Interface is selected.

IMPORTANT
These Tags are available **ONLY** while a Driver is in **Online** mode.

IO.Ethernet.IPSelect

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

Indicates the active IP address. Possible values are **0**: The main IP address is selected, **1**: The first alternative or backup IP address is selected, **2**: The second alternative or backup IP address is selected, or **3**: The third alternative or backup IP address is selected.

If the **Ethernet** or **RAS** Interface is connected, this Tag indicates which one of the four configured IP addresses is in use. If the Interface is disconnected, this Tag indicates which IP address is used first on the next attempt to connect.

During the connection process, if the active IP address is not available, the I/O Interface tries to connect using the other IP address. If the connection with the alternative IP address works, it is configured as the active IP address (automatic switchover).

To force a manual switchover, write values from 0 (zero) to 3 (three) to this Tag. This forces a reconnection with the specified IP address (**0**: Main address or **1, 2, 3**: Alternative address) if a Driver is currently connected. If a Driver is disconnected, this Tag configures the active IP address for the next attempt to connect.

IO.Ethernet.IPSwitch

Type of Tag	I/O Tag
Type of Access	Write-Only
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	4 (four)
N4 Parameter	1 (one)
String Configuration	IO.Ethernet.IPSwitch

Any value written to this Tag forces a manual switchover. If the main IP address is active, then the first alternative or backup IP address is activated, and so on for all alternative IP addresses and returning to the main address until a connection is established.

If a Driver is disconnected, this Tag configures the active IP address for the next attempt to connect.

IO.Ethernet.SocketState

Type of Tag	I/O Tag
Type of Access	Read-Only
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	4 (four)
N4 Parameter	2 (two)
String Configuration	IO.Ethernet.SocketState

The Value property of this Tag corresponds to socket states as a map of bits:

- **Bit 0**: 0 (zero, not listening) or 1 (one, listening)
- **Bit 1**: 0 (zero, disconnected) or 1 (one, connected)

Properties

These properties control the configuration of an **Ethernet** Interface.

NOTE

The **Ethernet** Interface is also used by the **RAS** Interface.

IO.Ethernet.AcceptConnection

Configure to False if a Driver must not accept external connections, that is, if a Driver behaves as a master, or configure to True to enable the reception of connections, that is, if a Driver behaves as a slave.

IO.Ethernet.BackupEnable[2,3]

Configure to True to enable an alternative or backup IP address. If the reconnection attempt with the main IP address fails, a Driver tries to use an alternative IP address. Configure to False to disable its usage.

IO.Ethernet.BackupIP[2,3]

Alternative or backup IP address of a remote device. Users can use a numerical address, as well as a device's host name, such as "192.168.0.7" or "SERVER2".

IO.Ethernet.BackupLocalPort[2,3]

Local port number to be used when connecting to an alternative IP address of a remote device. Used only if **IO.Ethernet.BackupLocalPortEnable** is equal to True.

IO.Ethernet.BackupLocalPortEnable[2,3]

Configure to True to force the use of a specific local port when connecting to an alternative or backup IP address or configure to False to use any available local port.

IO.Ethernet.BackupPort[2,3]

Port number of an alternative or backup IP address of a remote device, used with the **IO.Ethernet.BackupIP** property.

IO.Ethernet.IPFilter

List with a comma-separated IPv4 or IPv6 addresses, which defines from which addresses a Driver accepts or blocks connections. Users can use asterisks, such as "192.168.*.*", or intervals, such as "192.168.0.41-50", in any part of IP addresses. To block an IP address or a range of IP addresses, use the tilde ("~") character at the beginning of the address, according to the next examples:

- **192.168.0.24**: Accepts only connections from IPv4 address 192.168.0.24
- **192.168.0.41-50**: Accepts connections from IPv4 addresses in the interval between 192.168.0.41 and 192.168.0.50
- **192.168.0.***: Accepts connections from IPv4 addresses in the interval between 192.168.0.0 and 192.168.0.255
- **fe80:3bf:877::*:* (expands to fe80:03bf:0877:0000:0000:0000:*:*)**: Accepts connections from IPv6 addresses in the interval between fe80:03bf:0877:0000:0000:0000:0000:0000 and fe80:03bf:0877:0000:0000:0000:ffff:ffff
- **192.168.0.10, 192.168.0.15, 192.168.0.20**: Accepts connections from IPv4 addresses 192.168.0.10, 192.168.0.15, and 192.168.0.20
- **~192.168.0.95, 192.168.0.***: Accepts connections from IPv4 addresses in the interval between 192.168.0.0 and 192.168.0.255, except the IPv4 address 192.168.0.95

When a Driver receives a connection attempt, the list of filters is scanned sequentially from left to right, searching for a specific authorization or block for the IP address where the connection comes from. If no element on the list corresponds to the IP address, the authorization or block are dictated by the last element of that list:

- If the last element on the list is an authorization, such as "192.168.0.24", then all IP addresses not found on the list are blocked
- If the last element on the list is a block, such as "~192.168.0.24", then all IP addresses not found on the list are authorized

If an IP address appears on more than one filter on the list, the leftmost filter has precedence. For example, in case of "~192.168.0.95, 192.168.0.*", the IP address 192.168.0.95 fits both rules, but the rule that wins is the leftmost one, "~192.168.0.95", and therefore this IP address is blocked.

When **IOKit** blocks a connection, it logs a message "Blocked incoming socket connection from {IP}!".

In case of UDP connections in broadcast listening mode, in which a Driver can receive packets from different IP addresses, blocks or permissions are performed at each packet received. If a packet is received from a blocked IP address, it logs a message "Blocked incoming packet from {IP} (discarding {N} bytes)!".

IO.Ethernet.ListenIP

A IP address of the local network interface that a Driver uses to establish and accept connections. Leave this property empty to establish and accepts connections using any local network interface.

IO.Ethernet.ListenPort

9 Number of the IP port used by a Driver to listen to connections.

IO.Ethernet.MainIP

A IP address of a remote device. Users can use a numerical address, as well as a device's host name, such as "192.168.0.7" or "SERVER2".

IO.Ethernet.MainLocalPort

9 Local port number to use when connecting to the main IP address of a remote device. This value is only used if the **IO.Ethernet.MainLocalPortEnable** property is equal to True.

IO.Ethernet.MainLocalPortEnable

◩ Configure to True to force the use of a specific local port when connecting to the main IP address of a remote device or configure to False to use any available local port.

IO.Ethernet.MainPort

9 Number of the IP port of a remote device, used with the **IO.Ethernet.MainIP** property.

IO.Ethernet.PingEnable

◩ Configure to True to enable sending a **ping** command to the IP address of a remote device, before trying to connect to the socket. This socket's connection time-out cannot be controlled, therefore sending a **ping** command before connecting is a fast way to detect if the connection is going to fail. Configure to False to disable a **ping** command.

IO.Ethernet.PingTimeoutMs

Delay time to wait for a response from a **ping** command, in milliseconds.

IO.Ethernet.PingTries

Maximum number of attempts of a **ping** command. Minimum value is 1 (one), including the first **ping** command.

IO.Ethernet.ShareListenPort

Configure to True to share a listening port with other Drivers and processes or False to open a listening port in exclusive mode. To successfully share a listening port, all Drivers and processes that use that port must open it in shared mode. When a listening port is shared, each incoming connection is distributed to one of the processes listening. This way, if a Slave Driver only supports one connection at a time, users can use several instances of this Driver listening on the same port, therefore simulating a Driver with support for multiple connections.

IO.Ethernet.SupressEcho

Configure to True to eliminate echoes in communication. An echo is the unwanted reception of an exact copy of all data packets a Driver sent to a device.

IO.Ethernet.Transport

Defines a transport protocol. Possible values are **T or TCP**: Uses the TCP/IP protocol or **U or UDP**: Uses the UDP/IP protocol.

IO.Ethernet.UseIPv6

Configure to True to use IPv6 addresses on all Ethernet connections or configure to False to use IPv4 addresses (default).

Modem Interface Configuration

This section contains information about the configuration of **I/O Tags** and **Properties** of a **Modem** (TAPI) Interface.

I/O Tags

Tags of a Modem Interface (N2/B2 = 3)

The Tags described next allow controlling and diagnosing a **Modem** (TAPI) Interface at run time.

IMPORTANT
These Tags are available **ONLY** while a Driver is in **Online** mode.

IO.TAPI.ConnectionBaudRate

Type of Tag	I/O Tag
Type of Access	Read-Only
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	3 (three)
N4 Parameter	5 (five)
String Configuration	IO.TAPI.ConnectionBaudRate

Indicates a baud rate value for the current connection. If a modem is not connected, returns the value 0 (zero).

IO.TAPI.Dial

Type of Tag	I/O Tag
Type of Access	Write-Only
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	3 (three)
N4 Parameter	1 (one)
String Configuration	IO.TAPI.Dial

Write any value to this Tag to force a **Modem** Interface to start a call. This is an asynchronous command, only starting the call process. Users can monitor the **IO.TAPI.IsModemConnected** Tag to detect when a call is established.

IO.TAPI.HangUp

Type of Tag	I/O Tag
Type of Access	Write-Only
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	3 (three)
N4 Parameter	4 (four)
String Configuration	IO.TAPI.HangUp

Any value written to this Tag hangs the current call up.

NOTE

Use this command only when managing the physical layer manually or when explicitly trying to force a Driver to restart the communication. If the physical layer is configured for automatic reconnection, a Driver immediately tries to reestablish the connection.

IO.TAPI.IsModemConnected

Type of Tag	I/O Tag
Type of Access	Read-Only
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	3 (three)
N4 Parameter	3 (three)
String Configuration	IO.TAPI.IsModemConnected

This Tag indicates the status of a modem connection. Possible values are **0**: The modem is not connected, but it may be performing or receiving an external call or **1**: The modem is connected and a Driver completed or received an external call successfully. While it is in this status, the physical layer can send or receive data.

IO.TAPI.IsModemConnecting

Type of Tag	I/O Tag
Type of Access	Read-Only
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	3 (three)
N4 Parameter	6 (six)
String Configuration	IO.TAPI.IsModemConnecting

This Tag indicates the status of a modem connection, with more details than the **IO.TAPI.IsModemConnected** Tag. Possible values are **0**: Modem is not connected, **1**: Modem is connecting, that is, performing or receiving an external call, **2**: Modem is connected. While in this status, the physical layer can send or receive data, or **3**: Modem is disconnecting the current call.

IO.TAPI.ModemStatus

Type of Tag	I/O Tag
Type of Access	Read-Only
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	3 (three)
N4 Parameter	2 (two)
String Configuration	IO.TAPI.ModemStatus

Returns a **String** with the current status of a modem. Possible values are the following:

- **"No status!"**: The **Modem** Interface was not open yet or was already closed
- **"Modem initialized OK!"**: The **Modem** Interface was initialized successfully
- **"Modem error at initialization!"**: A Driver could not initialize modem's line. Check that Driver's log file for more details
- **"Modem error at dial!"**: A Driver could not start or accept a call
- **"Connecting..."**: A Driver started a call successfully, and is currently processing that call
- **"Ringing..."**: Indicates that the modem is receiving an external call, but it did not accepted it yet
- **"Connected!"**: A Driver connected successfully, that is, completed or accepted an external call
- **"Disconnecting..."**: A Driver is turning the current call off
- **"Disconnected OK!"**: A Driver turned the current call off
- **"Error: no dial tone!"**: A Driver aborted a call because the available line signal was not detected
- **"Error: busy!"**: A Driver aborted a call because the line was busy
- **"Error: no answer!"**: A Driver aborted a call because no answer was received from the other modem
- **"Error: unknown!"**: Current call was aborted because of an unknown error

IO.TAPI.PhoneNumber

Type of Tag	I/O Tag
Type of Access	Reading or Writing
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	3 (three)
N4 Parameter	0 (zero)
String Configuration	IO.TAPI.PhoneNumber

This Tag is a **String** that reads or changes the telephone number used by the **IO.TAPI.Dial** Tag. When changing this Tag, the new value is used only on the next **Dial** command.

Properties

These properties control the configuration of a **Modem** (TAPI) Interface.

IO.TAPI.AcceptIncoming

Configure to False if a modem cannot accept external calls, that is, if a Driver behaves as a master, and configure to True to enable receiving calls, that is, if a Driver behaves as a slave.

IO.TAPI.ModemID

This is the identification number of a modem. This ID is created by Windows and used internally to identify a modem on a list of devices installed on a computer. This ID may not remain valid if a modem is reinstalled or an application is executed on another computer.

NOTE

It is advisable to configure this property as 0 (zero), indicating that a Driver must use the first available modem.

IO.TAPI.PhoneNumber

A telephone number used by **Dial** commands, such as "0w01234566", in which the "w" character forces a modem to wait for a call sign.

RAS Interface Configuration

This section contains information about the configuration of **I/O Tags** and **Properties** of a **RAS** Interface.

I/O Tags

Tags of a RAS Interface (N2/B2 = 5)

Currently, there are no Tags defined specifically to manage a **RAS** Interface at run time.

Properties

These properties control the configuration of a **RAS** Interface.

NOTE

A **RAS** Interface uses the **Ethernet** Interface, which therefore must be also configured.

IO.RAS.ATCommand

An **AT** command to send through a socket to force a RAS device to perform a call using the current RAS channel, such as "ATDT6265545".

IO.RAS.CommandTimeoutSec

Time to wait for a **CONNECT** message in response to an **AT** command, in seconds.

Serial Interface Configuration

This section contains information about the configuration of **I/O Tags** and **Properties** of a **Serial** Interface.

I/O Tags

Tags of a Serial Interface (N2/B2 = 2)

Currently, there are no Tags defined specifically to manage a **Serial** Interface at run time.

Properties

These properties control the configuration of a **Serial** Interface.

IO.Serial.Baudrate

Specifies a baud rate of a serial port, such as 9600.

IO.Serial.CTSTimeoutMs

Time to wait for a **CTS** signal, in milliseconds. After turning the **RTS** signal on, a timer is started to wait for a **CTS** signal. If this timer expires, a Driver aborts sending bytes through the serial port. Available only when the **IO.Serial.RTS** property is configured with the value **Toggle** and the **IO.Serial.WaitCTS** property is configured to True.

IO.Serial.DataBits

Specifies the number of data bits to configure a serial port. Possible values are **5**: Five data bits, **6**: Six data bits, **7**: Seven data bits, or **8**: Eight data bits.

IO.Serial.DelayAfterMs

Number of milliseconds to delay after the last byte is sent through a serial port, but before turning the **RTS** signal off. Available only when the **IO.Serial.RTS** property is configured with the value **Toggle** and the **IO.Serial.WaitCTS** property is configured to False.

IO.Serial.DelayBeforeMs

Number of milliseconds to delay after turning the **RTS** signal on, but before data is sent. Available only when the **IO.Serial.RTS** property is configured with the value **Toggle** and the **IO.Serial.WaitCTS** property is configured to False.

IO.Serial.DTR

Indicates how a Driver deals with the **DTR** signal. Possible values are **OFF**: **DTR** signal is always turned off or **ON**: **DTR** signal is always turned on.

IO.Serial.InterbyteDelayUs

Delay time, in milliseconds (1/1000000 of a second), for each two bytes sent through a **Serial** Interface.

IO.Serial.InterframeDelayMs

Delay time, in milliseconds, before sending a packet after the last packet sent or received.

IO.Serial.Parity

Specifies a parity for the configuration of a serial port. Possible values are **E or Even**: Even parity, **N or None**: No parity, **O or Odd**: Odd parity, **M or Mark**: Mark parity, or **S or Space**: Space parity.

IO.Serial.Port

Number of the local serial port. Possible values are **1**: Uses the COM1 port, **2**: Uses the COM2 port, **3**: Uses the COM3 port, or **n**: Uses the COM*n* port.

IO.Serial.RTS

Indicates how a Driver deals with the **RTS** signal. Possible values are **OFF**: **RTS** signal always off, **ON**: **RTS** signal always on, or **Toggle**: Turns the **RTS** signal on when transmitting data and turns the **RTS** signal off when not transmitting data.

IO.Serial.StopBits

Specifies the number of stop bits for the configuration of a serial port. Possible values are **1**: One stop bit, **2**: One and a half stop bit, or **3**: Two stop bits.

IO.Serial.SupressEcho

Use a value different from 0 (zero) to enable suppressing the echo or 0 (zero) to disable it.

IO.Serial.WaitCTS

Configure to True to force a Driver to wait for the **CTS** signal before sending bytes when the **RTS** signal is turned on. Available only when the **IO.Serial.RTS** property is configured with the value **Toggle**.

Driver Revision History

VERSION	DATE	AUTHOR	COMMENTS
1.0.22	06/13/2025	M. Ludwig	• Driver updated to **IOKit** library version **3.0** and Visual Studio 2022 (*Case 37938*).
1.0.20	08/21/2019	M. Ludwig	• Driver ported to Visual Studio 2017 (*Case 27318*).
1.0.19	02/07/2018	M. Ludwig F. Englert	• First 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)

Copyright © 2013-2025 Elipse Software Ltda. All rights reserved.