

# Keller Series 30 Driver

<b>File Name</b>	KellerSeries30.dll
<b>Manufacturer</b>	Keller Pressure AG
<b>Devices</b>	Pressure transmitters from series 30
<b>Protocol</b>	Proprietary
<b>Version</b>	2.0.1
<b>Last Update</b>	02/03/2026
<b>Platform</b>	Win32
<b>Dependencies</b>	IOKit version 1.6 or later
<b>Superblock Readings</b>	No
<b>Level</b>	0

## Introduction

This Driver allows communication with pressure transmitters from series 30 by Keller Pressure AG. There are functions that allow defining and reading settings and also reading and writing values in transmitters.

Communication must be performed using a serial port, by using a converter from RS-485 to USB model K-104, from the same manufacturer, or any other converter from RS-485 to RS-232. This Driver was originally tested with that USB converter to RS-485. When using this converter, a virtual serial port for communication with a device is created. This virtual port is usually port COM3. A device must be connected to the converter via RS-485 output, and then the converter connects to a computer via USB port. For more information, please check manufacturer's documentation. For more information, about the configuration of **IOKit** library, please check topic **Documentation of I/O Interfaces**.

## Preparing a Device

The configuration parameters of a device can be read by using the Block Tag **Reading the Configuration of a Device**. The table **Configuration parameters of a device** of this Tag lists the parameters that can be read by this Driver. These parameters are described next.

- **CFG\_P**: Channels measured with maximum priority, typically for **P1** pressure
- **CFG\_T**: Channels measured at each **CNT\_T** seconds, typically for **TOB1** temperature

After the number of seconds calculated by  $CNT\_TCOMP \times CNT\_T$ , the correction of temperature for the pressure sensor is then recalculated. The **CFG\_P** and **CFG\_T** parameters are encoded according to the next table.

**Encoding of bytes of CFG\_P and CFG\_T parameters**

BIT	7	6	5	4	3	2	1	0
<b>Description</b>	-	-	TOB2	TOB1	T	P2	P1	-

- **DAC**: Configuration for the analog output. If this value is equal to 0 (zero), there is no analog output available. The bit 0 (zero) indicates whether it is a current (zero, mA) or voltage (one, V) output. If bit 3 (three) is equal to 1 (one), this indicates that channel **P1** is an output channel. If bit 3 (three) is equal to 0 (zero), the channel of differential pressure

**P1** - **P2** of devices with **CLASS.GROUP** equal to 5.1 is an output channel and the channel **CH0** of devices with **CLASS.GROUP** equal to 5.20 is an output channel

- **CFG\_CH0**: Defines the function for the calculated channel **CH0**. This channel is the only one available for devices with **CLASS.GROUP** equal to 5.20. The next table contains the available calculations

**Functions available for the calculated channel CH0**

CFG_CH0	CH0 FUNCTION	UNIT
0	Inactive	-
1	Differential pressure <b>P1</b> - <b>P2</b>	bar
2	Differential pressure <b>P2</b> - <b>P1</b>	bar
3	Square root calculation of <b>P1</b>	-
4	Square root calculation of <b>P2</b>	-
5	Square root calculation of <b>P1</b> - <b>P2</b>	-
6	Square root calculation of <b>P2</b> - <b>P1</b>	-
7	Density calculation <b>SF6</b>	kg/m <sup>3</sup>
8	Pressure at 20 °C <b>SF6</b>	bar
11	Absolute value of <b>P1</b>	bar
12	Absolute value of <b>P1</b> - <b>P2</b>	bar

**NOTES**

- For the calculation of density **SF6**: Pressure in **P1**, temperature in **T**, and density in **CH0**.
- For the calculation of pressure at 20 °C **SF6**: Pressure in **P1**, temperature in **T**, and pressure in scale in **CH0**.

## Driver Configuration

This section contains information about the configuration of **[P]** parameters of this Driver.

### [P] Parameters

<b>P1</b>	Not used, leave it in 0 (zero)
<b>P2</b>	Not used, leave it in 0 (zero)
<b>P3</b>	Not used, leave it in 0 (zero)
<b>P4</b>	Not used, leave it in 0 (zero)

This Driver does not use the **[P]** configuration parameters. All settings of **IOKit** library and the ones specific to this Driver must be performed on the properties window of **Elipse E3**, **Elipse Power**, or **Elipse Water**, or on the window for extra configurations of **Elipse SCADA**. On this window, the **KellerSeries30** tab contains specific settings of this Driver, while the other tabs contain communication settings of **Elipse Software's IOKit** library. For more information about the settings of this library, please check topic **Documentation of I/O Interfaces**.

The available option on the **KellerSeries30** tab is **Use Old Modbus Address**. If this option is selected, when using the function 3 (three) of Modbus protocol, that is, a **Reading Current Pressure and Temperature in Modbus Format** Tag,

the provided register address is subtracted from 1 (one) unit before added to protocol's message. This feature is useful for devices that comply with old standard of Modicon controllers, the first ones to implement the Modbus protocol, in which addresses of registers start at 1 (one) but are encoded in protocol's message starting at 0 (zero). Therefore, users can provide a logical address in the parameters of a Tag and let this Tag subtract the value 1 (one) before representing this address in a protocol's message. If this option is unselected, the value configured in the *N3* parameter of this Tag is added to a Modbus message unaltered.

Devices use a virtual serial port, usually port COM3, for communication configured with a baud rate of 9600 or 115200 bps (only for devices with **CLASS.GROUP** equal to 5.20), 1 (one) start bit, 8 (eight) data bits, 1 (one) stop bit, and parity can be configured for devices with **CLASS.GROUP** equal to 5.20.

## Tag Reference

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

## Reading Current Pressure and Temperature in Modbus Format

### Read-Only

<b>N1</b>	0 (zero)
<b>N2</b>	Address of a <b>Slave</b> device on the network
<b>N3</b>	Address of the first register to read
<b>N4</b>	Format of data

This Tag reads current values of pressure and temperature from a device, using function 3 (three) of Modbus protocol. This function is available for devices with **CLASS.GROUP** equal to 5.20 and **Year.Week** greater than or equal to 02.40.

The *N2* parameter is the address of a **Slave** device on the network, between 1 (one) and 249. The address 250 is transparent, that is, any device responds to this address, regardless of the configured address, therefore this address must be used only in operations with a single device.

The *N3* parameter is the address of the first register to read and the *N4* parameter is the format of data. Possible values for this parameter are **0**: A 16-bit integer (divided by 100) or **1**: A 32-bit floating-point value (32-bit **IEEE754** format).

The **Value** property of this Tag contains the value of pressure or temperature of the specified channel. The next tables contain the registers for reading in the available formats.

### Registers for reading channels in 32-bit floating-point format

CHANNEL	DESCRIPTION	INITIAL ADDRESS (HEX)	REGISTERS	UNIT
<b>CH0</b>	Calculated value	00	02	--
<b>P1</b>	Pressure of sensor 1 (one)	02	02	bar
<b>P2</b>	Pressure of sensor 2 (two)	04	02	bar
<b>T</b>	Temperature in temperature sensor	06	02	°C

CHANNEL	DESCRIPTION	INITIAL ADDRESS (HEX)	REGISTERS	UNIT
TOB1	Temperature of sensor 1 (one)	08	02	°C
TOB2	Temperature of sensor 2 (two)	0A	02	°C

**Registers for reading channels in 16-bit integer format**

CHANNEL	DESCRIPTION	INITIAL ADDRESS (HEX)	REGISTERS	UNIT
CH0	Calculated value	10	01	1/100 --
P1	Pressure of sensor 1 (one)	11	01	1/100 bar
P2	Pressure of sensor 2 (two)	12	01	1/100 bar
T	Temperature in temperature sensor	13	01	1/100 °C
TOB1	Temperature of sensor 1 (one)	14	01	1/100 °C
TOB2	Temperature of sensor 2 (two)	15	01	1/100 °C

## Reading or Writing Coefficients

### Reading or Writing

N1	1 (one)
N2	Address of a <b>Slave</b> device on the network
N3	Number of a coefficient
N4	Not used, leave it in 0 (zero)

This Tag reads or writes the coefficient specified in the *N3* parameter, in a 32-bit floating-point format. The **calibration** values can be read or written, while **information** values can only be read.

The *N2* parameter is the address of a **Slave** device on the network, between 1 (one) and 249. The address 250 is transparent, that is, any device responds to this address, regardless of the configured address, therefore this address must be used only in operations with a single device.

The **Value** property of this Tag contains the value read or written in the specified coefficient.

**Values of coefficients for calibration**

COEFFICIENT	DESCRIPTION	UNIT
53	Threshold for the roof function	

COEFFICIENT	DESCRIPTION	UNIT
64	Offset of pressure in P1 sensor	bar
65	Gain factor of pressure in P1 sensor	
66	Offset of pressure in P2 sensor	bar
67	Gain factor of pressure in P2 sensor	
68	Offset of analog output	bar
69	Gain factor of analog output	
70	Offset of channel <b>CH0</b>	
71	Gain factor of channel <b>CH0</b>	
72	Upper threshold for switched output 1 (one)	
73	Lower threshold for switched output 1 (one)	
78	Upper threshold for switched output 2 (two)	
79	Lower threshold for switched output 2 (two)	

#### Values of coefficients for information

COEFFICIENT	DESCRIPTION	UNIT
80	Minimum pressure of P1 sensor	bar
81	Maximum pressure of P1 sensor	bar
82	Minimum pressure of P2 sensor	bar
83	Maximum pressure of P2 sensor	bar
84	Minimum temperature of temperature sensor	°C
85	Maximum temperature of temperature sensor	°C
86	Minimum temperature of P1 sensor	°C
87	Maximum temperature of P1 sensor	°C
88	Minimum temperature of P2 sensor	°C
89	Maximum temperature of P2 sensor	°C
90	Minimum value for channel <b>CH0</b>	
91	Maximum value for channel <b>CH0</b>	
92	Pressure for minimum analog signal	bar
93	Pressure for maximum analog signal	bar
94	Minimum analog signal	mA, V
95	Maximum analog signal	mA, V

# Initializing

## Write-Only

<b>B1</b>	2 (two)
<b>B2</b>	Address of a <b>Slave</b> device on the network
<b>B3</b>	Not used, leave it in 0 (zero)
<b>B4</b>	Not used, leave it in 0 (zero)

Each time a device is turned on when applying a supply voltage, or after interrupting its power supply, this device must be initialized by a writing operation to this Tag. An attempt to execute any other function provokes an exception error 32. Sending a command is performed by writing any value to this Tag.

This initialization command, if successful, returns a block of status information, which can be read by using the **WriteEx** method of scripts in **Elipse E3**, **Elipse Power**, or **Elipse Water**. This block of information is returned in the *WriteStatus* parameter of this method, whose Elements are described on the next table. For more information about using this Tag in **Elipse E3**, **Elipse Power**, or **Elipse Water**, please check topic **Sending an Initialization Command**.

The *B2* parameter is the address of a **Slave** device on the network, between 1 (one) and 249. The address 250 is transparent, that is, any device responds to this address, regardless of the configured address, therefore this address must be used only in operations with a single device.

### Elements returned in the WriteStatus parameter of the WriteEx method

ELEMENT	DESCRIPTION
<b>1</b>	Identifier of a device
<b>2</b>	Subdivision in the class of a device. Possible values are <b>1</b> : Pressure transmitter from series 30 of 1999 or newer or <b>20</b> : Pressure transmitter from series 30 of 2002 or newer
<b>3</b>	Firmware version in <b>String</b> format
<b>4</b>	Size of the internal reception buffer
<b>5</b>	Status information. Possible values are <b>0</b> : A device was addressed for the first time after turned on or <b>1</b> : A device has already been initialized

# Configuring a New Address to a Slave

## Write-Only

<b>B1</b>	3 (three)
<b>B2</b>	Address of a <b>Slave</b> device on the network
<b>B3</b>	Not used, leave it in 0 (zero)
<b>B4</b>	Not used, leave it in 0 (zero)

This Tag allows defining a new address to a device.

The *B2* parameter is the address of a **Slave** device on the network, between 1 (one) and 249. The address 250 is transparent, that is, any device responds to this address, regardless of the configured address, therefore this address must be used only in operations with a single device.

The Elements of this Block Tag are described on the next table.

**Elements of the Configuring a New Address to a Slave Tag**

ELEMENT	DESCRIPTION
1	Current address of the destination device
2	Address to configure in the destination device

## Reading the Serial Number of a Device

### Read-Only

<b>N1</b>	4 (four)
<b>N2</b>	Address of a <b>Slave</b> device on the network
<b>N3</b>	Not used, leave it in 0 (zero)
<b>N4</b>	Not used, leave it in 0 (zero)

This Tag returns an unsigned 32-bit integer in the **Value** property representing the serial number of a device, defined by its manufacturer.

The *N2* parameter is the address of a **Slave** device on the network, between 1 (one) and 249. The address 250 is transparent, that is, any device responds to this address, regardless of the configured address, therefore this address must be used only in operations with a single device.

## Reading the Value of a Channel in Floating-Point Format

### Read-Only

<b>B1</b>	5 (five)
<b>B2</b>	Address of a <b>Slave</b> device on the network
<b>B3</b>	Channel to read. For more information, please check table <b>Registers for reading channels in 32-bit floating-point format</b>
<b>B4</b>	Not used, leave it in 0 (zero)

This Tag reads the value of a channel in 32-bit floating-point format. Each device can read up to 5 (five) channels. Users can read 2 (two) independent pressure sensors, **P1** and **P2**, or the temperature of pressure sensors, **TOB1** and **TOB2**, and

also an additional temperature sensor **T**. The temperature of pressure sensors **TOB1** and **TOB2** are needed for temperature compensation of pressure sensors.

The **B2** parameter is the address of a **Slave** device on the network, between 1 (one) and 249. The address 250 is transparent, that is, any device responds to this address, regardless of the configured address, therefore this address must be used only in operations with a single device.

The Elements of this Block Tag are described on the next table.

**Elements of the Reading the Value of a Channel in Floating-Point Format Tag**

ELEMENT	DESCRIPTION
1	Value read from a channel, in <b>IEEE754</b> floating-point format
2	Status byte, composed by a map of bits with values <b>0</b> : Computation or measurement error in channel <b>CH0</b> , <b>1</b> : Computation or measurement error in channel <b>P1</b> , <b>2</b> : Computation or measurement error in channel <b>P2</b> , <b>3</b> : Computation or measurement error in channel <b>T</b> , <b>4</b> : Computation or measurement error in channel <b>TOB1</b> , <b>5</b> : Computation or measurement error in channel <b>TOB2</b> , <b>6</b> : Computation error in the process of calculating for analog output, and <b>7</b> : Device is in <b>Power Up</b> mode or in <b>Standard</b> mode

## Reading the Value of a Channel in Integer Format

### Read-Only

<b>B1</b>	6 (six)
<b>B2</b>	Address of a <b>Slave</b> device on the network
<b>B3</b>	Channel to read. For more information, please check table <b>Registers for reading channels in 16-bit integer format</b>
<b>B4</b>	Not used, leave it in 0 (zero)

This Tag reads the value of a channel in 32-bit long integer format. Each device can read up to 5 (five) channels. Users can read 2 (two) independent pressure sensors, **P1** and **P2**, or the temperature of pressure sensors, **TOB1** and **TOB2**, and also an additional temperature sensor **T**. The temperatures of pressure sensors **TOB1** and **TOB2** are needed for temperature compensation of pressure sensors.

The values of channels **CH0**, **P1**, and **P2** are read in Pascal, in which 1 Pa is equal to 10<sup>-5</sup> bar. The values of channels **T**, **TOB1**, and **TOB2**, on the other hand, are read in units of 0.01 °C. This Tag works only in devices with **CLASS.GROUP** equal to 5.20.

The **B2** parameter is the address of a **Slave** device on the network, between 1 (one) and 249. The address 250 is transparent, that is, any device responds to this address, regardless of the configured address, therefore this address must be used only in operations with a single device.

The Elements of this Block Tag are described on the next table.

### Elements of the Reading the Value of a Channel in Integer Format Tag

ELEMENT	DESCRIPTION
1	Value read from a channel, in 32-bit integer format
2	Status byte, composed by a map of bits with values <b>0</b> : Computation or measurement error in channel <b>CH0</b> , <b>1</b> : Computation or measurement error in channel <b>P1</b> , <b>2</b> : Computation or measurement error in channel <b>P2</b> , <b>3</b> : Computation or measurement error in channel <b>T</b> , <b>4</b> : Computation or measurement error in channel <b>TOB1</b> , <b>5</b> : Computation or measurement error in channel <b>TOB2</b> , <b>6</b> : Computation error in the process of calculating for analog output, and <b>7</b> : Device is in <b>Power Up</b> mode or in <b>Standard</b> mode

## Daisy Chain Function for Pressure and Temperature

### Read-Only

<b>B1</b>	7 (seven)
<b>B2</b>	Address of a <b>Slave</b> device on the network
<b>B3</b>	Not used, leave it in 0 (zero)
<b>B4</b>	Not used, leave it in 0 (zero)

A Daisy Chain function is a special function for fast buses with many users. With this function, users can read simultaneously the values of pressure and temperature of all devices available on a network, as long as the addresses are in ascending order, without gaps. Users must always address the smaller address value among the **Slave** devices. When a device responds, this response provokes the next address to send its respective response, and so on.

The values read are returned on a list of events, that is, each reading operation of this Tag starts a sequence of **OnRead** events, in which each event corresponds to the values read from a device. This function is only available for devices with **CLASS.GROUP** equal to 5.20.

The **B2** parameter is the address of the first **Slave** device on the network, between 1 (one) and 249, that is, the device with the smaller address of the sequence to read.

The Elements of this Tag returned at each **OnRead** event and relative to a specific device are described on the next table.

### Elements of the Daisy Chain Function for Pressure and Temperature Tag

ELEMENT	DESCRIPTION
1	Pressure in mbar
2	Temperature (0.1 Kelvin)
3	Address of a device

## Configuring a Zero Point

### Write-Only

<b>N1</b>	8 (eight)
<b>N2</b>	Address of a <b>Slave</b> device on the network
<b>N3</b>	Command. Please check the next <b>table</b> for more information
<b>N4</b>	Using a Setpoint. Please check later on this topic for more information

This Tag allows defining a Zero Point of channels of a device. Zero Points of channels **P1**, **P2**, and **CH0** can also be read or written using the **Reading or Writing Coefficients** Tag.

The *N2* parameter is the address of a **Slave** device on the network, between 1 (one) and 249. The address 250 is transparent, that is, any device responds to this address, regardless of the configured address, therefore this address must be used only in operations with a single device.

The available commands for the *N3* parameter are described on the next table.

#### Commands available for the N3 parameter

COMMAND	DESCRIPTION
<b>0</b>	Defines the Zero Point for channel <b>P1</b>
<b>1</b>	Configures the Zero Point of channel <b>P1</b> to its default value
<b>2</b>	Defines the Zero Point for channel <b>P2</b>
<b>3</b>	Configures the Zero Point of channel <b>P2</b> to its default value
<b>4</b>	Updates <b>DAC</b> scale, only for devices with <b>CLASS.GROUP</b> equal to 5.1
<b>6</b>	Defines the Zero Point for channel <b>CH0</b> , only for devices with <b>CLASS.GROUP</b> equal to 5.20
<b>7</b>	Configures the Zero Point of channel <b>CH0</b> to its default value, only for devices with <b>CLASS.GROUP</b> equal to 5.20

The *N4* parameter defines whether the value of this Tag must be sent as a Setpoint or not. For commands 0 (zero), 2 (two), and 6 (six), if the value is not sent as a Setpoint, the Zero Point is calculated so that the current measured value is equal to 0 (zero). If the value is sent as a Setpoint for these commands, the Zero Point is calculated so that the value of this Setpoint is equal to the measured value. The value of this Tag can have the values **0**: Not sent as a Setpoint or **1**: Value is sent as a Setpoint.

The **Value** property of this Tag contains the Setpoint value to send if the *N4* parameter is equal to 1 (one). This value must be in a 32-bit unsigned integer format.

## Reading the Configuration of a Device

### Read-Only

<b>B1</b>	9 (nine)
<b>B2</b>	Address of a <b>Slave</b> device on the network
<b>B3</b>	Index of parameters to read. Possible values for this parameter are 2 (two) or 3 (three). Please check table <b>Configuration parameters of a device</b> for more information about the relationship between the meaning of this parameter and the selected index
<b>B4</b>	Not used, leave it in 0 (zero)

This Tag returns the configuration of a device, as defined by its manufacturer. These configurations cannot be changed.

The *B2* parameter is the address of a **Slave** device on the network, between 1 (one) and 249. The address 250 is transparent, that is, any device responds to this address, regardless of the configured address, therefore this address must be used only in operations with a single device.

**Configuration parameters of a device**

INDEX	PARAM0	PARAM1	PARAM2	PARAM3	PARAM4
2	CFG_P	CFG_T	CFG_CH0	CNT_T	CNT_TCOMP
3	--	--	--	--	DAC

The Elements of this Block Tag are described on the next table.

**Elements of the Reading the Configuration of a Device Tag**

ELEMENT	DESCRIPTION
1	PARAM0
2	PARAM1
3	PARAM2
4	PARAM3
5	PARAM4

For more information, please check topic **Preparing a Device**.

## Sending an Initialization Command

The next script can be used to send a initialization command to a device in **Elipse E3**, **Elipse Power**, or **Elipse Water** by using a writing operation in an **Initializing** Tag and at the same time read status information returned by this command.

```

Sub CommandButton1_Click()
    Dim vWriteStatus(4)
    if (Application.GetObject("[Driver Keller].InitAndRelease").WriteEx(vValue, vTimestamp, _
        vQuality, vWriteStatus)) Then
        Screen.Item("TextBox1").Text = vWriteStatus(0)
        Screen.Item("TextBox2").Text = vWriteStatus(1)
        Screen.Item("TextBox3").Text = vWriteStatus(2)
        Screen.Item("TextBox4").Text = vWriteStatus(3)
        Screen.Item("TextBox5").Text = vWriteStatus(4)
    End If
End Sub

```

#### NOTE

Users cannot read writing status information in **Elipse SCADA**.

## Scale of CH0, P1, and P2 Channels

Channels **CH0**, **P1**, and **P2** are linearly scalable with point 0 (zero) and gain factor equal to **Value = Gain Factor × Value + Offset**. The default value of offset is 0 (zero) and the default value of gain factor is 1.0. The offset value can be changed using a **Configuring Zero Point** Tag.

The gain factor must be used only for calibration purposes and not for changing pressure units. That last operation must be performed only by a **Master**. To represent other pressure units using the analog output, the conversion of units must be taken into consideration in the scale of that analog output.

## Scale of Analog Output

An analog output in pressure transmitters from series 30 can be programmed via interface. Once the paths **SIGNAL SENSORS -> TRANSFORMATION AND DIGITAL VALUE -> ANALOG SIGNAL** are calibrated regardless of their manufacturer, an output analog can be configured for different pressures and pressure units without recalibration. To configure the scale of an analog output, follow these steps:

- The **Reading the Configuration of a Device** Tag allows determining whether a device contains an analog output. The coefficients needed for that calculation can be read from the **Reading or Writing Coefficients** Tag. A new scale can be define by writing to that same Tag
- For devices with **CLASS.GROUP** equal to 5.1, the new scale must be defined in the **Configuring Zero Point** Tag
- For devices with **CLASS.GROUP** equal to 5.20, the analog output is updated automatically

### Reading a Range of Pressure Values for an Analog Output

**K[n]** coefficients must be read using the **Reading or Writing Coefficients** Tag to allow the calculation of upper and lower limits of an analog output using the next formulas:

- $A = (K[92] - K[68]) \div K[69]$
- $B = (K[93] - K[68]) \div K[69]$

### Configuring a New Range of Pressure Values for an Analog Output

**K[68]** and **K[69]** coefficients must be calculated and written to a device using the **Reading or Writing Coefficients** Tag using the next formulas:

- $A = K[92] - ((K[93] - K[92]) \div (B - A)) \times A$
- $B = (K[93] - K[92]) \div (B - A)$

In which:

- **K[x]** is a coefficient with the corresponding number  $x$ . For more information please check the **Reading or Writing Coefficients** Tag
- **A** is for the pressure, in bar, in which the signal **K[94]** must be written
- **B** is for the pressure, in bar, in which the signal **K[95]** must be written

Other pressure units must be converted to bar.

## Documentation of I/O Interfaces

This section contains the documentation of I/O Interfaces referring to **KellerSeries30** Driver.

### Configuration of a Driver

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

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

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

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

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

### Configuration Dialog Box

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

### Setup Tab

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

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

- **Logging options:** Controls the generation of log files

The screenshot shows the 'Setup' tab of a driver configuration window. It contains several sections:
 

- Physical Layer:** A dropdown menu set to 'Ethernet' and a checkbox for 'Start driver OFFLINE'.
- Timeout:** A text box with '1000' and 'ms'.
- Communication check time:** A text box with '5000' and 'ms'.
- Connection management:** A dropdown set to 'Automatic (managed by the driver)', a checked checkbox for 'Retry failed connection every 20 seconds', an unchecked checkbox for 'Give up after 1 failed retries', and an unchecked checkbox for 'Disconnect if non-responsive for 0 seconds'.
- Logging Options:** An unchecked checkbox for 'Log to File' with the path 'C:\eeLogs\MicrolokII\_%DATE%.log' and a text box for 'File size limit (MB)' set to '0' with the note '(0 is unlimited)'.

Setup tab

General options on the Setup tab

OPTION	DESCRIPTION
<b>Physical Layer</b>	Select the physical layer on a list. Available options are <b>Serial, Ethernet, Modem, and RAS</b> . The selected interface must be configured on its specific tab
<b>Timeout</b>	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
<b>Communication check time</b>	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 <b>IO.CommunicationStatus</b> Tag
<b>Start driver OFFLINE</b>	Select this option so that a Driver starts in <b>Offline</b> mode or stopped. This means that the I/O interface is not created until this Driver is configured to <b>Online</b> 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
<b>Mode</b>	Selects a management mode of a connection. Selecting the <b>Automatic</b> option allows a Driver to manage the connection automatically, as specified in the next options. Selecting the <b>Manual</b> option allows an application to fully manage a connection
<b>Retry failed connection every ... seconds</b>	Select this option to enable a Driver's connection retry in a certain interval, in seconds. If the <b>Give up after failed retries</b> option is not selected, this Driver keeps retrying until a connection is performed, or until the application is stopped
<b>Give up after ... failed retries</b>	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 <b>Offline</b> 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
<b>Disconnect if non-responsive for ... seconds</b>	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 <b>Timeout</b> option

**Options on the Logging Options group**

OPTION	DESCRIPTION
<p><b>Log to File</b></p>	<p>Enable this option and configure the name of a file to write a log. Log files can be large, so use this option for short periods of time, only for testing and debugging purposes. If the <b>%PROCESS%</b> 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 <b>Elipse E3</b>, 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 <b>OFDAh</b>. Users can also use the <b>%DATE%</b> macro in the file name. In this case a log file is generated every day, in the format <b>aaaa_mm_dd</b>. 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 <b>%DATE_HOUR%</b> macro generates one log file per hour, in the format <b>aaaa_mm_dd_hh</b></p>
<p><b>File size limit (MB)</b></p>	<p>Configure the log file size limit, in megabytes. A value equal to 0 (zero) means that there is no size limit for the log file</p>

**Serial Tab**

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

Serial

Port:

Baud rate:

Data bits:

Parity:

Stop bits:

Enable 'ECHO' suppression

Handshaking

DTR control:

RTS control:

Wait for CTS before send

CTS timeout:  ms

Delay before send:  ms

Delay after send:  ms

Inter-byte delay (microseconds):   $\mu$ s

Inter-frame delay (milliseconds):  ms

Serial tab

General options on the Serial tab

OPTION	DESCRIPTION
<b>Port</b>	Select a serial port on the list, from <b>COM1</b> to <b>COM4</b> , or type the name of a serial port in the format <b>COMn</b> , 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"
<b>Baud rate</b>	Select a baud rate on the list ( <b>1200, 2400, 4800, 9600, 19200, 38400, 57600, or 115200</b> ) or type a baud rate, such as 600
<b>Data bits</b>	Select 7 (seven) or 8 (eight) data bits on the list
<b>Parity</b>	Select a parity on the list. The available options are <b>None, Even, Odd, Mark, or List</b>
<b>Stop bits</b>	Select the number of stop bits on the list. The available options are <b>1, 1.5, or 2</b> stop bits
<b>Enable 'ECHO' suppression</b>	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
<b>Inter-byte delay (microseconds)</b>	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
<b>Inter-frame delay (milliseconds)</b>	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
<b>DTR control</b>	Select the value <b>ON</b> to keep the <b>DTR</b> signal always on while the serial port is open. Select the value <b>OFF</b> to turn the <b>DTR</b> signal off while the serial port is open. Some devices require the <b>DTR</b> signal always on to allow communication
<b>RTS control</b>	Select the value <b>ON</b> to keep the <b>RTS</b> signal always on while the serial port is open. Select the value <b>OFF</b> to turn the <b>RTS</b> signal off while the serial port is open. Select the value <b>Toggle</b> to turn the <b>RTS</b> signal on while sending bytes via serial port and turn it off when not sending bytes, therefore enabling the reception
<b>Wait for CTS before send</b>	Available only when the <b>RTS control</b> option is configured with the value <b>Toggle</b> . Use this option to force a Driver to check the <b>CTS</b> signal before sending bytes via serial port, after turning the <b>RTS</b> signal on. In this mode, the <b>CTS</b> signal is handled as a permission flag for sending
<b>CTS timeout</b>	Determines a maximum time, in milliseconds, that a Driver waits for the <b>CTS</b> signal after turning the <b>RTS</b> signal on. If the <b>CTS</b> signal is not turned on within this time-out, that Driver then fails the current communication and returns an error
<b>Delay before send</b>	Some serial port devices have a delay when enabling a data sending circuit after the <b>RTS</b> signal is turned on. Configure this option to wait a certain number of milliseconds after turning the <b>RTS</b> signal on and before sending the first byte. <b>IMPORTANT:</b> This delay must be used carefully, because it uses 100% of CPU resources while waiting. System's general performance degrades as this value increases
<b>Delay after send</b>	This is the same effect of the <b>Delay before send</b> option, but in this case the delay is performed after sending the last byte, before turning the <b>RTS</b> 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

Transport: TCP/IP ▼

PING before connecting

Timeout: 4000 ms

Retries: 1

Listen for connections on port: 0

Share listen port with other processes

Interface: (All Interfaces) ▼

Use IPv6  Use SSL SSL Settings

Enable 'ECHO' suppression

IP Filter:

Connect to

<input type="checkbox"/> Main IP:	<span style="border: 1px solid gray; padding: 2px;"> </span>	Port:	<span style="border: 1px solid gray; padding: 2px;">502</span>	<input type="checkbox"/> Local port:	<span style="border: 1px solid gray; padding: 2px;">0</span>
<input type="checkbox"/> Backup IP 1:	<span style="border: 1px solid gray; padding: 2px;"> </span>	Port:	<span style="border: 1px solid gray; padding: 2px;">0</span>	<input type="checkbox"/> Local port:	<span style="border: 1px solid gray; padding: 2px;">0</span>
<input type="checkbox"/> Backup IP 2:	<span style="border: 1px solid gray; padding: 2px;"> </span>	Port:	<span style="border: 1px solid gray; padding: 2px;">0</span>	<input type="checkbox"/> Local port:	<span style="border: 1px solid gray; padding: 2px;">0</span>
<input type="checkbox"/> Backup IP 3:	<span style="border: 1px solid gray; padding: 2px;"> </span>	Port:	<span style="border: 1px solid gray; padding: 2px;">0</span>	<input type="checkbox"/> Local port:	<span style="border: 1px solid gray; padding: 2px;">0</span>

**Ethernet tab**

**Available options on the Ethernet tab**

OPTION	DESCRIPTION
<b>Transport</b>	Select the value <b>TCP/IP</b> for a TCP socket ( <i>stream</i> ) or select the value <b>UDP/IP</b> to use a UDP socket ( <i>connectionless datagram</i> )
<b>Listen for connections on port</b>	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 <b>Connect to</b> option
<b>Share listen port with other processes</b>	Select this option to share the listening port with other Drivers and processes
<b>Interface</b>	Select the local network interface, identified by its IP address, that a Driver uses to establish and receive connections, or select the value <b>(All Interfaces)</b> to allow connection in any network interface
<b>Use IPv6</b>	Select this option to force a Driver to use addresses in <b>IPv6</b> format on all Ethernet connections. Leave this option deselected to use the <b>IPv4</b> format
<b>Enable 'ECHO' suppression</b>	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
<b>IP Filter</b>	List of restricted or allowed IP addresses from where a Driver accepts connections ( <i>Firewall</i> ). Please check the <b>IO.Ethernet.IPFilter</b> property for more information
<b>PING before connecting</b>	Enable this option to execute a <b>ping</b> 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
	<p>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:</p> <ul style="list-style-type: none"> <li>• <b>Timeout:</b> Specify the number of milliseconds to wait for a reply from a <b>ping</b> command. Users must use a <b>ping</b> 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</li> <li>• <b>Retries:</b> Number of retries of a <b>ping</b> command, not counting the first attempt. If all attempts fail, then the socket connection is aborted</li> </ul>

#### Available options on the Connect to group

OPTION	DESCRIPTION
<b>Main IP</b>	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"
<b>Port</b>	Type the IP port of a remote device, between 0 (zero) and 65535
<b>Local port</b>	Select this option to use a fixed local IP port when connecting to a remote device
<b>Backup IP 1, 2, and 3</b>	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

Select the modem to use:

▼ Modem settings...

Dial Number:

Accept incoming calls

**Modem tab**

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

#### Available options on the Modem tab

OPTION	DESCRIPTION
<b>Select the modem to use</b>	Select a modem on the list of available modems on the computer. If the value <b>Default modem</b> is selected, then the first available modem is used. Selecting this option is recommended specially when an application is used on another computer
<b>Modem settings</b>	Click to open the configuration window of the selected modem
<b>Dial Number</b>	Type a default number for dialing. This value can be changed at run time. Users can use the <b>w</b> 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"
<b>Accept incoming calls</b>	Enable this option so that a Driver answers the phone when receiving an external call. To use this option, users must configure the <b>Connection management</b> option on the <b>Setup</b> tab to the value <b>Manual</b>

## RAS Tab

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

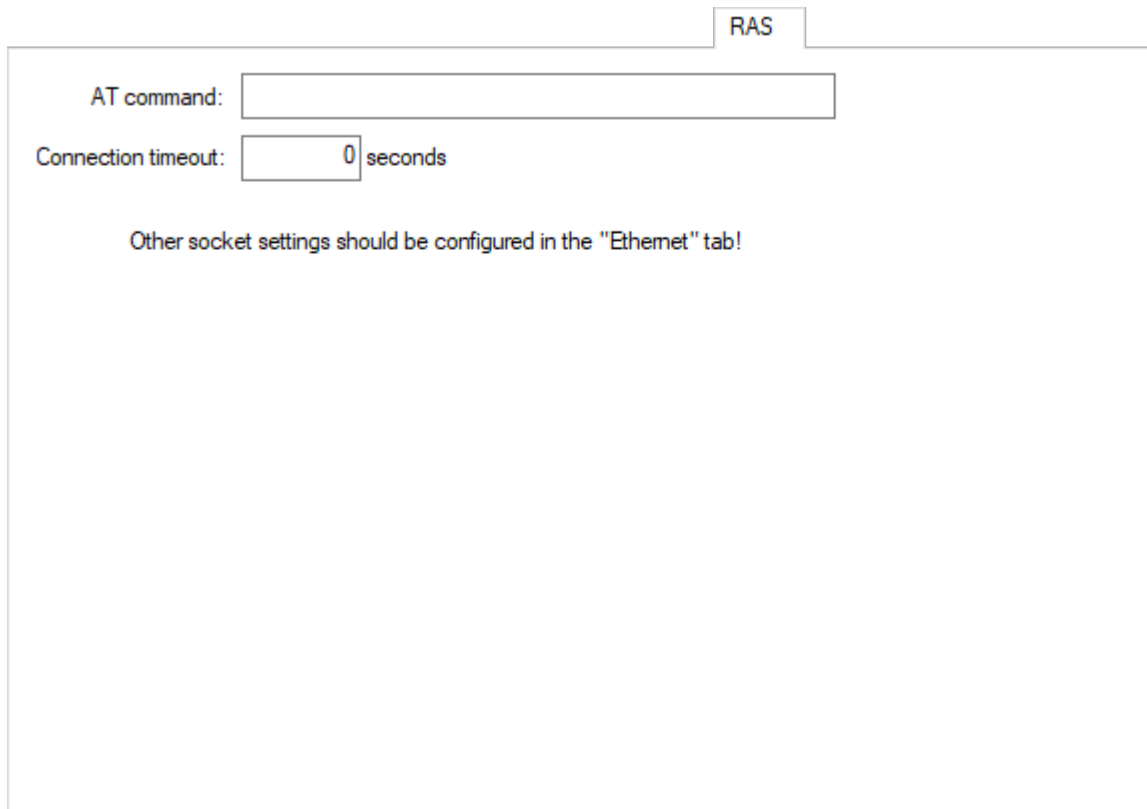
A **RAS** Interface opens a socket connection with a RAS device. A RAS device is a server of modems available through TCP/IP, waiting for socket connections on an IP port. For each connection accepted on this port, users have access to one modem.

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

<b>Type of Tag</b>	I/O Tag
<b>Type of Access</b>	Reading
<b>N1 Parameter</b>	-1 (minus one)
<b>N2 Parameter</b>	0 (zero)
<b>N3 Parameter</b>	0 (zero)
<b>N4 Parameter</b>	6 (six)
<b>String Configuration</b>	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

<b>Type of Tag</b>	Block Tag
<b>Type of Access</b>	Read-Only
<b>B1 Parameter</b>	-1 (minus one)
<b>B2 Parameter</b>	0 (zero)
<b>B3 Parameter</b>	0 (zero)
<b>B4 Parameter</b>	1 (one)
<b>Size Property</b>	4 (four)
<b>ParamItem Property</b>	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**

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

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

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

**IO.SetConfigurationParameters**

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

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

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

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

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

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

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

A variation of the previous example uses a bidimensional array.

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

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

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

## IO.WorkOnline

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

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

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

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

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

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

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

### IMPORTANT

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

## Properties

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

## IO.ConnectionMode

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

## IO.GiveUpEnable

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

## IO.GiveUpTries

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

## IO.InactivityEnable

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

## IO.InactivityPeriodSec

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

## IO.RecoverEnable

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

## IO.RecoverPeriodSec

9 Delay time between two connection attempts, in seconds.

### NOTE

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

## IO.StartOffline

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


### NOTE

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

## IO.TimeoutMs

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

## IO.Type

 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

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

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

#### IO.Stats.Partial.BytesSent

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

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

## IO.Stats.Partial.TimeConnectedSeconds

<b>Type of Tag</b>	I/O Tag
<b>Type of Access</b>	Read-Only
<b>N1 Parameter</b>	-1 (minus one)
<b>N2 Parameter</b>	0 (zero)
<b>N3 Parameter</b>	0 (zero)
<b>N4 Parameter</b>	1102
<b>Configuration by String</b>	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

<b>Type of Tag</b>	I/O Tag
<b>Type of Access</b>	Read-Only
<b>N1 Parameter</b>	-1 (minus one)
<b>N2 Parameter</b>	0 (zero)
<b>N3 Parameter</b>	0 (zero)
<b>N4 Parameter</b>	1103
<b>Configuration by String</b>	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

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

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

## IO.Stats.Total.BytesSent

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

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

## IO.Stats.Total.ConnectionCount

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

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

## IO.Stats.Total.TimeConnectedSeconds

<b>Type of Tag</b>	I/O Tag
<b>Type of Access</b>	Read-Only
<b>N1 Parameter</b>	-1 (minus one)
<b>N2 Parameter</b>	0 (zero)
<b>N3 Parameter</b>	0 (zero)
<b>N4 Parameter</b>	1002
<b>Configuration by String</b>	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

<b>Type of Tag</b>	I/O Tag
<b>Type of Access</b>	Write-Only
<b>N1 Parameter</b>	-1 (minus one)
<b>N2 Parameter</b>	0 (zero)
<b>N3 Parameter</b>	4 (four)
<b>N4 Parameter</b>	1 (one)
<b>String Configuration</b>	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

<b>Type of Tag</b>	I/O Tag
<b>Type of Access</b>	Read-Only
<b>N1 Parameter</b>	-1 (minus one)
<b>N2 Parameter</b>	0 (zero)
<b>N3 Parameter</b>	4 (four)
<b>N4 Parameter</b>	2 (two)
<b>String Configuration</b>	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:0877:0000:0000:0000:0000:0000:\*)**: 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

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

## IO.Ethernet.PingTries

9 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

A 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

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

<b>Type of Tag</b>	I/O Tag
<b>Type of Access</b>	Write-Only
<b>N1 Parameter</b>	-1 (minus one)
<b>N2 Parameter</b>	0 (zero)
<b>N3 Parameter</b>	3 (three)
<b>N4 Parameter</b>	1 (one)
<b>String Configuration</b>	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

<b>Type of Tag</b>	I/O Tag
<b>Type of Access</b>	Write-Only
<b>N1 Parameter</b>	-1 (minus one)
<b>N2 Parameter</b>	0 (zero)
<b>N3 Parameter</b>	3 (three)
<b>N4 Parameter</b>	4 (four)
<b>String Configuration</b>	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**

<b>Type of Tag</b>	I/O Tag
<b>Type of Access</b>	Read-Only
<b>N1 Parameter</b>	-1 (minus one)
<b>N2 Parameter</b>	0 (zero)
<b>N3 Parameter</b>	3 (three)
<b>N4 Parameter</b>	3 (three)
<b>String Configuration</b>	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**

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

**9** 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

**9** 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** 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

**A** 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

9 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

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

### IO.Serial.CTSTimeoutMs

9 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

9 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

9 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

9 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

A 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

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

## IO.Serial.InterframeDelayMs

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

## IO.Serial.Parity

**A** 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

**9** 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 COMn port.

## IO.Serial.RTS

**A** 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

**9** 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.SuppressEcho

**9** 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's Revision History

VERSION	DATE	AUTHOR	COMMENTS
2.0.1	02/03/2026	M. Ludwig	<ul style="list-style-type: none"> <li>Driver updated to <b>IOKit</b> library version <b>3.0</b> and Visual Studio 2022 (<i>Case 38837</i>).</li> </ul>
1.0.1	04/03/2007	A. Qutes	<ul style="list-style-type: none"> <li>Initial version of this Driver (<i>Case 7873</i>).</li> </ul>

**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](mailto: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](mailto:evan@elipse.com.br)**

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

**[www.elipse.com.br](http://www.elipse.com.br)**

**[kb.elipse.com.br](http://kb.elipse.com.br)**

**[forum.elipse.com.br](http://forum.elipse.com.br)**

**[www.youtube.com/elipsesoftware](http://www.youtube.com/elipsesoftware)**

**[elipse@elipse.com.br](mailto:elipse@elipse.com.br)**



Gartner, Cool Vendors in Brazil 2014, April 2014.

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

**Microsoft Partner**  
Gold Independent Software Vendor (ISV)