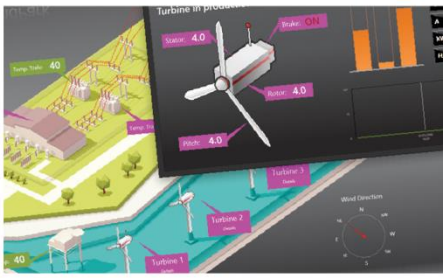
INDUSTRIAL AUTOMATION SOFTWARE
progea

Movicon.next™
Automation Platform.next generation



Movicon NExT

4.1 Drivers

Ver.3.4.268

Table of Contents

1. COMMUNICATION DRIVERS 1

1.1. General Concepts 3
1.2. Importing Tags from device 4
1.3. Driver Demo 6

2. DRIVERS CONFIGURATION 7

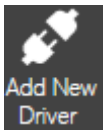
2.1. Communication Driver Configurations 7
2.2. General Settings 7
2.3. Channel Settings 8
2.4. TCP-IP Channel Settings 11
2.5. Channel Settings Serial 12
2.6. Station Settings 13
2.7. I/O Physical Address 15

1. Communication Drivers

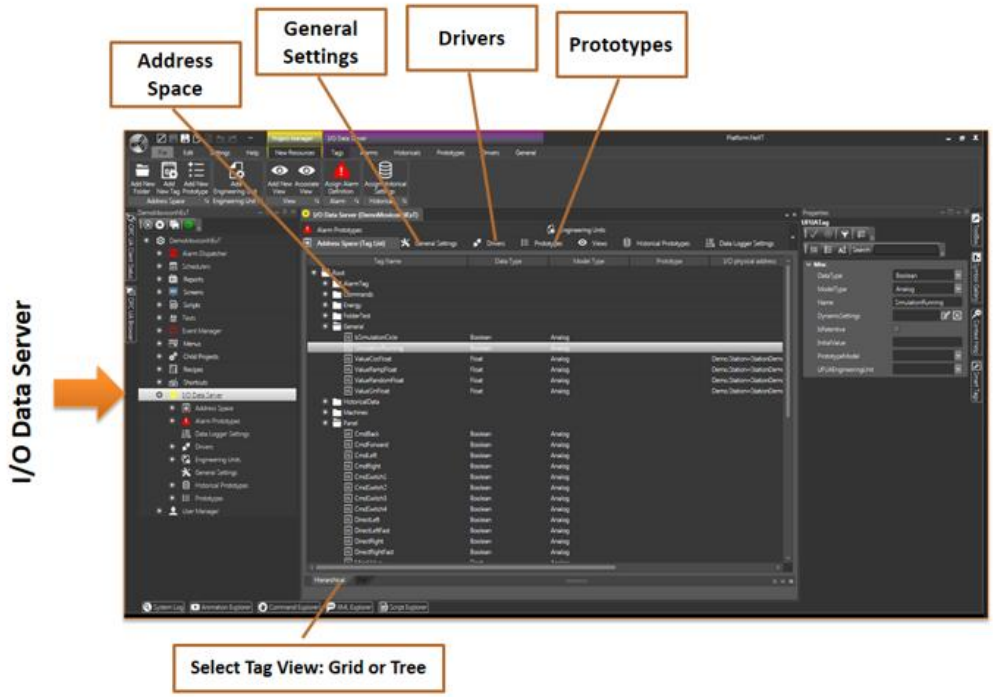
One or more communication drivers can be inserted and configured in the Platform.NExT project's I/O Data Server as required. Each driver inserted in the project must be configured according to it specific characteristics that depend on the device's propriety communication protocol. In order to do this, each specific detail of the addressable memory areas and communication parameters need to be referred to in the manual of each specific driver.
In this topic will examine the general concepts of use for a I/O Data Server driver which are common to all the drivers available in the platform.

Installing a new Communication Driver

To install a new Communication Driver in the project you will need to open the I/O Data Server and select the "Drivers" tab.

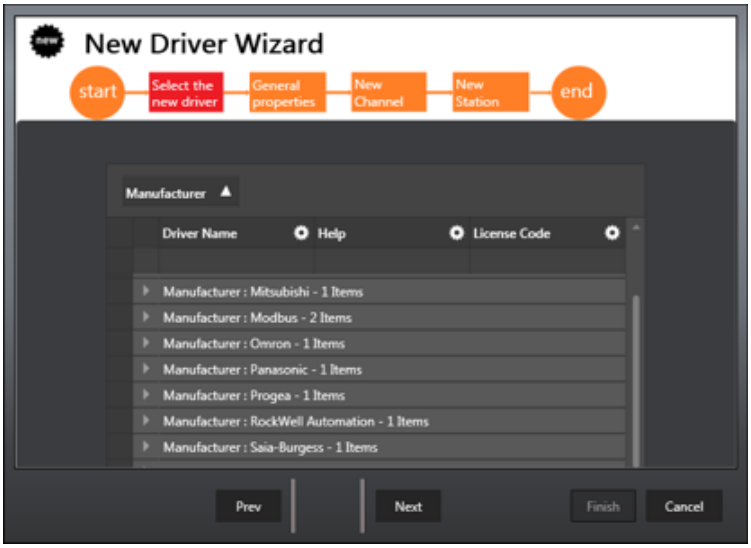


A new driver is added by using the "Add New Driver' command in the taskbar Ribbon or by right clicking within the area of the Driver window to open up a menu containing the same command.
Added Drivers are listed in the Drivers window. This window is used to edit the Driver's settings or remove the Driver from the project.



The "Drivers" window from the "I/O Data Server" resource contains a list of Drivers installed in the project.

When clicking the "Add New Driver" command a configuration wizard will open to guide you through the simple procedures. The first thing you must do is select the driver you wish to install from the list of drivers available in the system.
The available drivers are grouped by the manufacture name and then by device or protocol type.



The Wizard Window used for selecting and installing the Driver in the project.

Select the driver desired and then proceed to the next step to the specific driver configuration as indicated in the relating topic for each driver.

By following the simple steps it will be possible to:

1. Select the device manufacturer name and type (or protocol)
2. Set the Driver's general parameters (or leave the default values)
3. Set the main parameters required by the protocol or device setup
4. Define the addresses, channel and station

When the wizard terminates the driver will be installed in the project. The driver can be removed or modified at any given moment when required to by selecting it within the workspace and using the 'Open Settings' command from the ribbon or by with a right mouse click.

The I/O Data Server's Communication Drivers are installed with the server. Any new drivers will be available for downloading from the Progea website as well as others created by third parties.

Automatic Tag Importing

The communication drivers support automatic tag importation. This functionality allows you to select the data source (usually from the PLC or the export file from the PLC) and

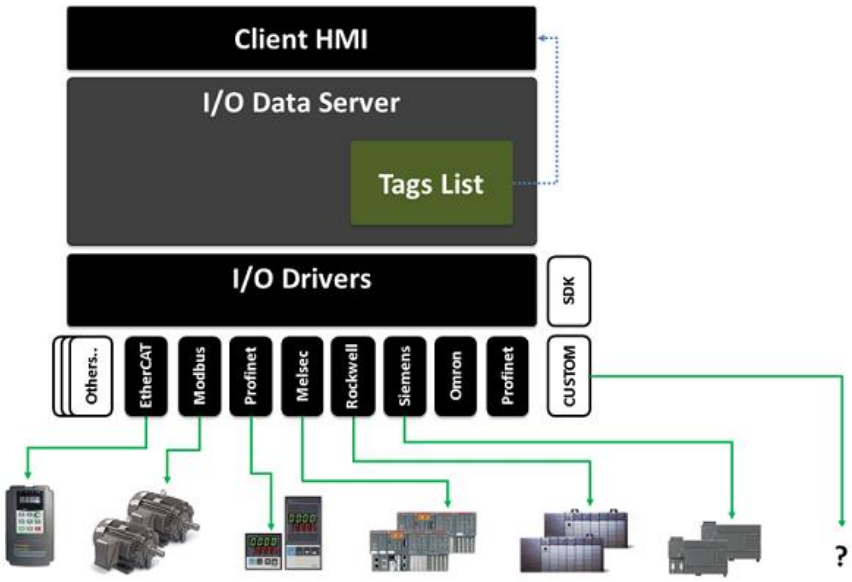
the tags to be imported (all or only part of them). The importer will automatically create the Tags in the progect's Tag List ((Address Space).

1.1. General Concepts

The Communication Drivers are I/O Data Server components which enable specific communication protocols to be managed for establishing physical connections to field devices.
The drivers are built with dynamic ".NET" libraries (.DLL files) which, according to "exception-based" logic, notify information they receive from connected device memory areas to the I/O Data Server memory areas and vice versa, according to the settings predefined for the specific communication protocols. By using the 'I/O Data Server's Address Space, which is the gathering point of dynamic information, dynamic information is placed in the Tags and hence made available to all the platform modules.

The list of Communication Drivers is available in the I/O Data Server and is continuously added to. The most popular drivers are those used for PLC devices (Siemens, Schneider, Rockwell, Omron, Mitsubishi, Saia, Panasonic, etc.), fieldbuses (Profibus, Profinet, Ethernet/IP, EtherCAT, Powerlink, etc.), RTUs and instrumentation (Modbus, etc.) or networks for infrastructures (IEC61870, IEC60850) and civil systems (Konnex, Bacnet, Lon, etc.).

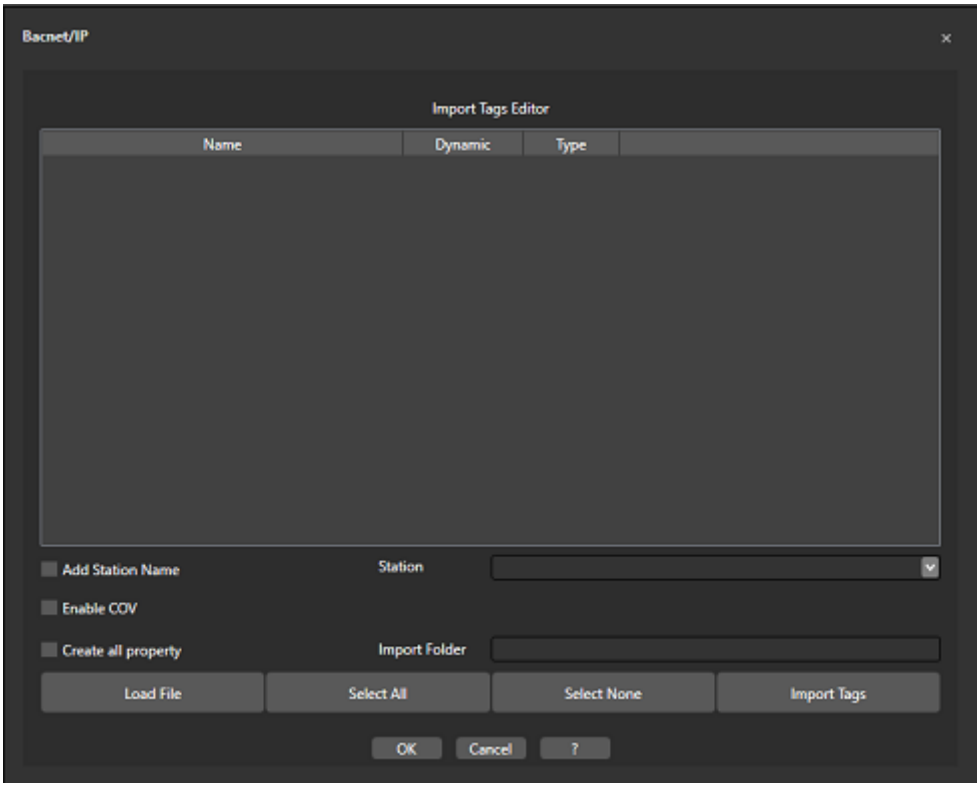
The system's open architecture is modular and allowsthe I/O Data Server to be integrated with third party I/O Drivers and thus the use of new communication protocols as well.



PC and field devices connections.

1.2. Importing Tags from device

Generally most communication drivers support the automatic data import from device functions that are available from the "**Import Tags**" command ribbon or by right mouse clicking on the driver selected.
Thanks to this important feature, you can directly access the database of a PLC or an equivalent data source of a device to import the desired Tags into the Supervisor project.
When the operation terminates, the project's Tag List (Address Space) will automatically populate with all the tags that have been imported. These tags will be defined as Dynamic Tags and will be inserted with the corresponding tag type and with the physical address already assigned for the device.



Load File

When activating this command, a window will display to select the "data source", which may be the PLC database, the symbolic file or .CSV file obtained by the exportation of data from the PLC or device.
Therefore you will need to provide a file and then select it to display a window showing the tags contained within as shown below.

⚠ Attention: Importing data from the PLC is supported in all the drivers for the most popular devices. Please check the relevant access modalities or 'data source' requirements, which may vary from one device to another. The parameters to access the device are defined in the selected station.

If tags are detected as already existing in the project's RealTimeDB while importing data, they will be overwritten without requiring confirmation. The existing tag properties that will be overwritten are the "Data Type" and Physcial I/O Address". The "Description" property will be left unchanged. Therefore the "Description" property will only be set if the tag being imported does not exist in the project's RealTimeDB.

The import operation may take a while to complete if a great number of tags have been selected for importing. The operation can be aborted by using the "ESC" key.

There are two different ways to import Structure and Array tags using the device's database's Import Tool:

- When the Structure's root is selected from the tool's dialog window and then imported using the "Import" key, a prototype will be created in the project. This prototype will be defined with the structure members and a tag of the same type making it then possible to import a structure multi-selection. The same operation can also be performed by using the "Select All" button and then the "Import" button.
- When one or more members of one or more Structures are selected and then imported, the tag types corresponding to the members selected will be created and not the structure prototypes. This same result can also be obtained by clicking the "Expand All" button to explore the Structure members, then the "Select All" to select all the members and then the "Import" button to export each individual member as distinct tags. Naturally, if individual tags and Structure tags both exist in the device's tag list, the "Select All" button will select all of them together.

As regards to importing Array tag types, if the Array root is selected and imported, a Array tag will be created in the project, that is, with "Type Data" and a corresponding "Array Size" equal to that import. This same operation can be performed by using the "Select All" button and then the "Import" button.

If one or more elements of one or more Array tags are selected and then imported, the tags corresponding to the element types selected will be created but not the Array type tags. The same result can be obtained by clicking on the "Expand All" button to explore the Array elements, then clicking the "Select All" button to select them all and finally the "Import" button to import each individual element as a distinct tag.

Naturally, if individual tags and Array tags both exist in the device's tag list, the "Select All" button will select them all.

Station:
This box is used for selecting the driver station to assign to the imported variables in cases where the driver has been defined with several stations.

Import folder
Allows you to specify an import folder for the tags.

Select All
Allows you to select all the variables from the importation file.
Use the CTRL+Click or SHIFT+Click keys for partially selecting variables.

Select None
Allows you to deselect all the variables from the importation file.
Use the CTRL+Click or SHIFT+Click combo keys for partially deselecting variables

Import Tags
Activates the importation of variables from the origin file (device's data source) to the Supervisor project. When the importation has terminated, the project's Real Time Database resource will result as being populated with all the imported variables.

As the 'data sources' depend on the device and might change, it is advised to always check the imported variables' properties to see whether the automatic parsing, the type assigned and the device's address have been executed correctly when the import procedure has terminated.

1.3. Driver Demo

The "Demo" Driver can be found in the list of Drivers available in the I/O Data Server. This is a Driver simulation that does not communicate with external devices and provides some simulation variables to be used for testing in demo projects. The types of simulation variables provided by the "Demo" Driver are:

- **Sin**: simulation variable which executes the Sin trigonometric function curve
- **Cos**: simulation variable which executes the Cosin trigonometric function curve
- **Ramp**: simulation variable which executes the curve with adapt to ramp
- **Random**: simulation variable which executes the curve with random adaptation

2. Drivers Configuration

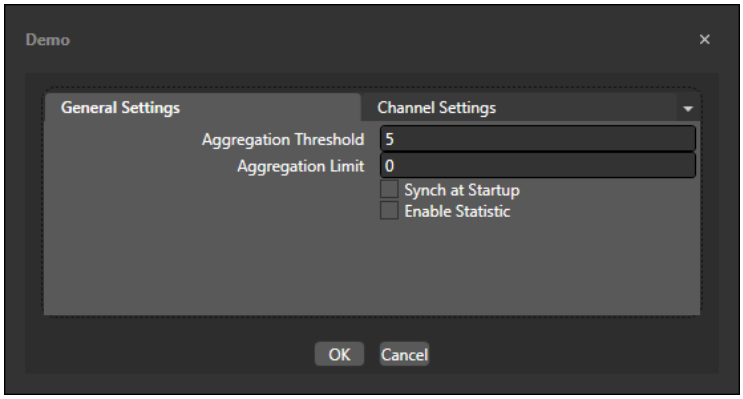
2.1. Communication Driver Configurations

During the procedures to insert a Communication Driver or any subsequent modifications, you should configure its general settings appropriately for the devices it is intended to communicate with.
A Wizard will provide the necessary configurations to insert a Communication Driver in a step by step guide using a few simple settings windows as described below. To open these windows at a later date, simply double click on the Driver desired among those inserted and presented in the project's Drivers window, or use the "Open Settings" window which opens with a right mouse click.

The Driver settings window is presented with three forms: "General Settings", "Channels Settings" and "Stations Settings". Some of the parameters presented in the three setting forms are the same for all Drivers, while others are specific to each individual Driver and depend directly on the type of protocol used by the Driver in question.

2.2. General Settings

The "General Settings" form is used to define the Driver's general settings. The parameters on this form are:



General Driver Settings Window.

Enable
Allows you to enable/disable the selected driver.

Aggregation Threshold
This parameter determines the minimum threshold for fragmenting data packets exchanged with the device. When managing communications, the Driver automatically calculates (at project startup) the size and quantity of communication tasks to be

created based on the variables and addresses exchanged with the device. The Supervisor will try to optimise communications by aggregating a major amount of data possible into one unique task. When data are connected in addresses at a distance from each other, this value determines the 'distance' between the addresses that consents the Driver to decide whether a new task should be created for the subsequent data block.
When this value is set to “zero”, no aggregation will be executed and a task will be created for each Tag. When this value is set to 'one', only those tags with consecutive addresses will be aggregated. When set with a value higher than 'one', those tags that point to addresses which have been set a distance lower than the set value, will be aggregated.

Aggregation Limit

This parameter is used to specify the maximum number of bytes which can be aggregated per communication task. When left set at 'zero', the Driver will use the maximum value set by the selected protocol instead. This value will need to be changed when using devices whose maximum number of bytes to be exchanged with one individual task is lower than the protocol's limit.

Synch at Startup

This option determines the synchronization between the project logic and Driver communication at project startup. When this option is enabled, Movicon will wait until the input or input/output communication tasks have been executed once before processing the project's logic and basic scripts.
Even though this option will cause the project to take longer to startup, it will nevertheless ensure that logic is executed with the most "updated" input values.

Enable Statistic

When this option is enabled, the Driver will keep statistics on the amount of data exchanged with the device, such as the number of tasks executed, number of variable in use, etc.

State/Command Variable

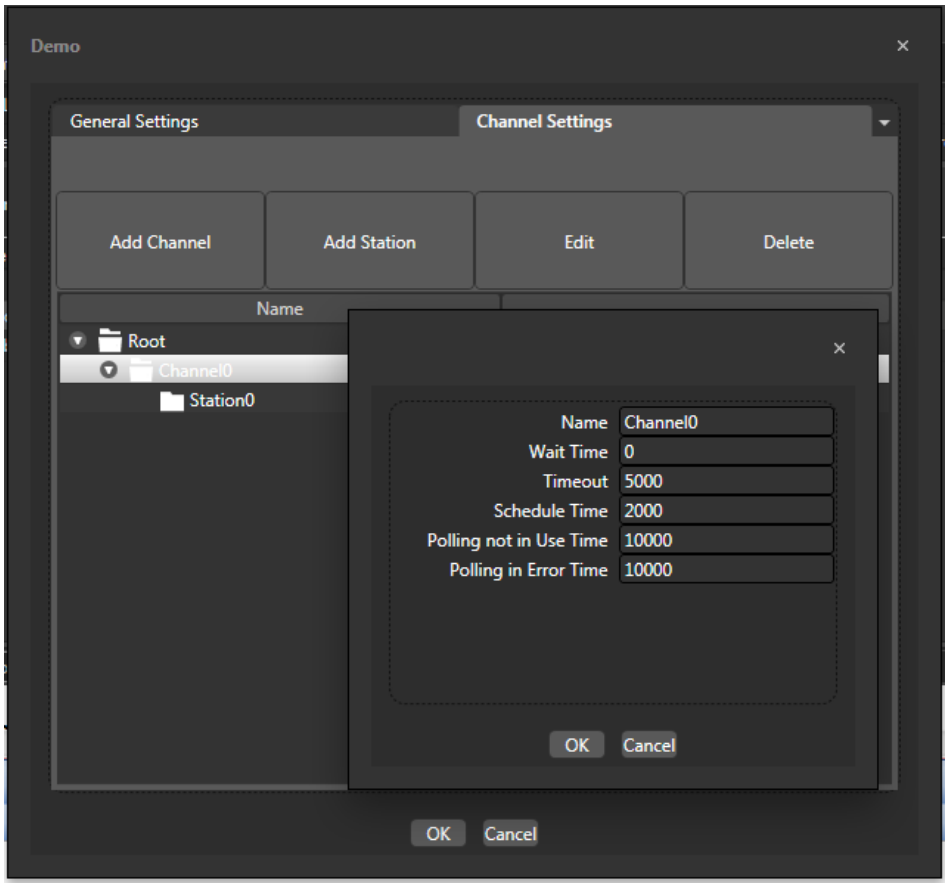
By assigning the name of a numeric variable of the Supervisor (Byte type recommended) to this property, it will be possible to refresh the TAG's list.
See the following table on variable bits. Please note that some bits can only be used to check the state of the station, while other bits can only be used to set the state of the station (commands).

Bit 0 (ChangeTagsSettings)	Command Load: 0 = Default, Tag's list load completed 1 = Force the driver to reload the tag's list from file
Bit 1 (ErrorLoadTagsSettings)	Load Result State: 0 = Load Operation Complete 1 = Load Operation Failed

2.3. Channel Settings

The "Channel Settings" form is used to define which Channel the Drivers will use for communications. The Channel parameters which are common to all Drivers and the "Demo" driver are described below. The other parameters differ according to the type of

protocol used by the Driver. For instance, if using a serial Driver, the COM port configuration parameters, such as "Baudrate" and "Stop Bit" for example, will be presented. On the other hand, if using an Ethernet Driver, the TCP/IP communication configuration parameters will be presented. More than one communication Channel can also be entered in the one same Driver when different devices are to be communicated with. Therefore, in cases such as this, each Channel must have different connection settings, such as different IP addresses or COM ports for example.
When selecting the "Channel Settings" form using the appropriate buttons, Channels can be added (Add Channel), deleted (Delete selecting the Channel) or edited (Edit Channel by selecting Channel or double clicking it).



Window used for setting Driver channels

One or more Stations can be inserted in each Channel as described in the paragraph on "Station Settings".

The general channel properties are:

Name

Name that identifies the Channel corresponding to the device to be communicated with. Each channel defined for the same Driver must have a different and unique name.

Wait Time

Wait time in milliseconds between the execution of one communication task and the next (data blocks). The default value is set at '0' (no pause). This '0' default value may have to be changed for devices that require a waiting time between each subsequent interrogation (i.e. low performing devices).

Timeout

Defines the time-out value in milliseconds. When expired, the Driver will notify the time-out of incoming communications. This time-out refers to incoming requested device data.

Schedule Time

This parameter, expressed in milliseconds, determines the minimum execution time of each individual data update task when variables are in use.

The zero value means that data will be updated with the highest velocity possible based on the number of listed tasks to be executed.

Higher values can be used, such as 10000 for instance, for data which do not need updating quickly.

Polling not in Use Time

This parameter, expressed in milliseconds, forces data updating of each individual task even when variables are not in use, by establishing a refresh time. For example, setting this value at 10000 (being 10 seconds), means that the task will be executed with a minimum time of 10 seconds even when variables are not in use.

When this parameter is set at '0', the tasks will not be executed if the variables are not in use. When this parameter is set with a different value, all variables that not in use with input or input/output tasks will be updated in read according to frequency set by value. This mechanism may slow down communications when a high number of variables are being dealt with.

Polling in Error Time

This parameter, expressed in milliseconds, represents the time during which tasks will not be executed for the station in error. Therefore when a stations goes into error, the Driver will wait the duration of time set here before retrying.

State command variable

Assigning a name to a supervisor numeric variable (Byte type recommended) to this property, it will be possible to control the communication status of the selected channel.

Bit 0 (State)	Connection channel: 0 = connected 1 = not connected
Bit 1 (State)	Primary host error state: 0 = Active 1 = Inactive
Bit 2 (State)	Backup host error state: 0 = Active 1 = Inactive
Bit 3 (State)	Connected host: 0 = active 1 = inactive

2.4. TCP-IP Channel Settings

State command variable

When assigning this property with the name of a numeric variable from the Supervisor (Byte type recommended), you will be able to verify the selected station's communication status. The following table describes what the different variable bit types represent:

Bit 0 (State)	Connection channel: 0 = connected 1 = not connected
Bit 1 (State)	Primary host error state: 0 = Active 1 = Inactive
Bit 2 (State)	Backup host error state: 0 = Active 1 = Inactive
Bit 3 (State)	Connected host: 0 = active 1 = inactive
Bit 4 (Command)	Commands communication switch over from primary station to backup station or vice versa.

Host Name

This is used to enter the IP address of the Server or network device to be connected to.

Example: 192.168.0.1; localhost; Server1

In cases where a connection requiring user authentication (User name and password) is used, the name of the computer, to be connected to, must be entered and not its IP address.

Host Port

This is the number of the server's or device's TCP port to be connected to. This value completes the device's IP address. For example, the 502 port is always used for the TCP-IP Modbus (as established by the protocol). Please refer to the relevant documentation for port details for other device types.

Backup host name

This is the address of the Backup Server. The driver will try to connect to the Backup Server address entered here when it is unable to communicate with the primary server. Basically, this is redundancy mode handled at driver level. Likewise, if communications with the Backup Server are interrupted, the driver will try to reconnect to the primary server and so forth.

This field accepts the name of one Server or a list of Servers separated by the ";" character. The list of servers permits the driver to handle several Backup Servers in cyclic mode so that if one server should become unavailable, the next one on the list of backup servers will automatically takeover. Once the last server on the list has been reached, the driver will start at the top of the list by reconnecting to the Primary Server again.

Change host time-out
This is the time in milliseconds that is used by the driver to connect to another Server when the one it is communicating with becomes unavailable.

Read Time-out
Default Value is 2000.
Sets the "Timeout" (in read) in milliseconds. When exceeded, the driver notifies a time-out upon communication reception. The time-out refers to data received.

Write Time-out
Default Value is 2000.
Sets the "Timeout" (in write) in milliseconds. When exceeded, the driver notifies a time-out upon communication reception. The time-out refers to data received.

For further information please refer to the general settings chapter on "Channel Settings"

2.5. Channel Settings Serial

Keep opened
When enabled, the Communication port will always remain open during communications.

Port
Defines the COM serial port number to be used for communications.

Make sure that no conflicts occur in Windows when using ports. For example, when installing the Com3 and Com4 ports you will need to check whether the assigned address and IRQ are compatible with the PC configuration. It is advised that you use addressable serial cards for this.

Baud Rate
Defines the communication serial baud rate. The baud rate value will need to correspond to that of the device to be communicated with.

Data Bits
This defines the number of bytes required by the communication protocol of interest.

Parity
This defines the parity type required by the communication protocol of interest.

Stop Bits
This defines number of Stop Bits required by the communication protocol of interest.

Handshake
This defines the handshake (also known as flow control) for the type of communication used. In particular, it negotiates with the connected device's serial port the required communication flow level needed. The driver sets "None" for default which means no handshake is needed, however it might be necessary to select a handshake type (e.g. in cases for signalling errors with code "1") according to the device used.

The options are:

- **None:** No Handshake needed. No flow control type is required by the protocol.
- **Xon/Xoff:** handshake type is Xon/Xoff.
- **RTS :** sets the serial for toggling the RTS control signal, meaning that the serial maintains a high signal as long as there are characters to be transmitted.
- **RTS XOnXOff:** sets the handshake (flow control) to NONE and set the serial for disabling the RTS signal managment.

RTS Enable

This selects the RS232 signal used for controlling the transmission direction. The possible values are: None (default value), RTS and DTR.

DTR Enable

This selects the RS232 signal used for controlling the transmission direction. The possible values are: None (default value), RTS and DTR.

RX Timeout

This defines the time-out value in milliseconds. When this value is exceeded, the driver will notify the communication time-out in reception.
This time-out refers to the reception of the requested data.

TX Timeout

This defines the time-out value in milliseconds. When this value is exceeded, the driver will notify the communication time-out in transmission.
This timeout refers to the data transmission.

State command variable

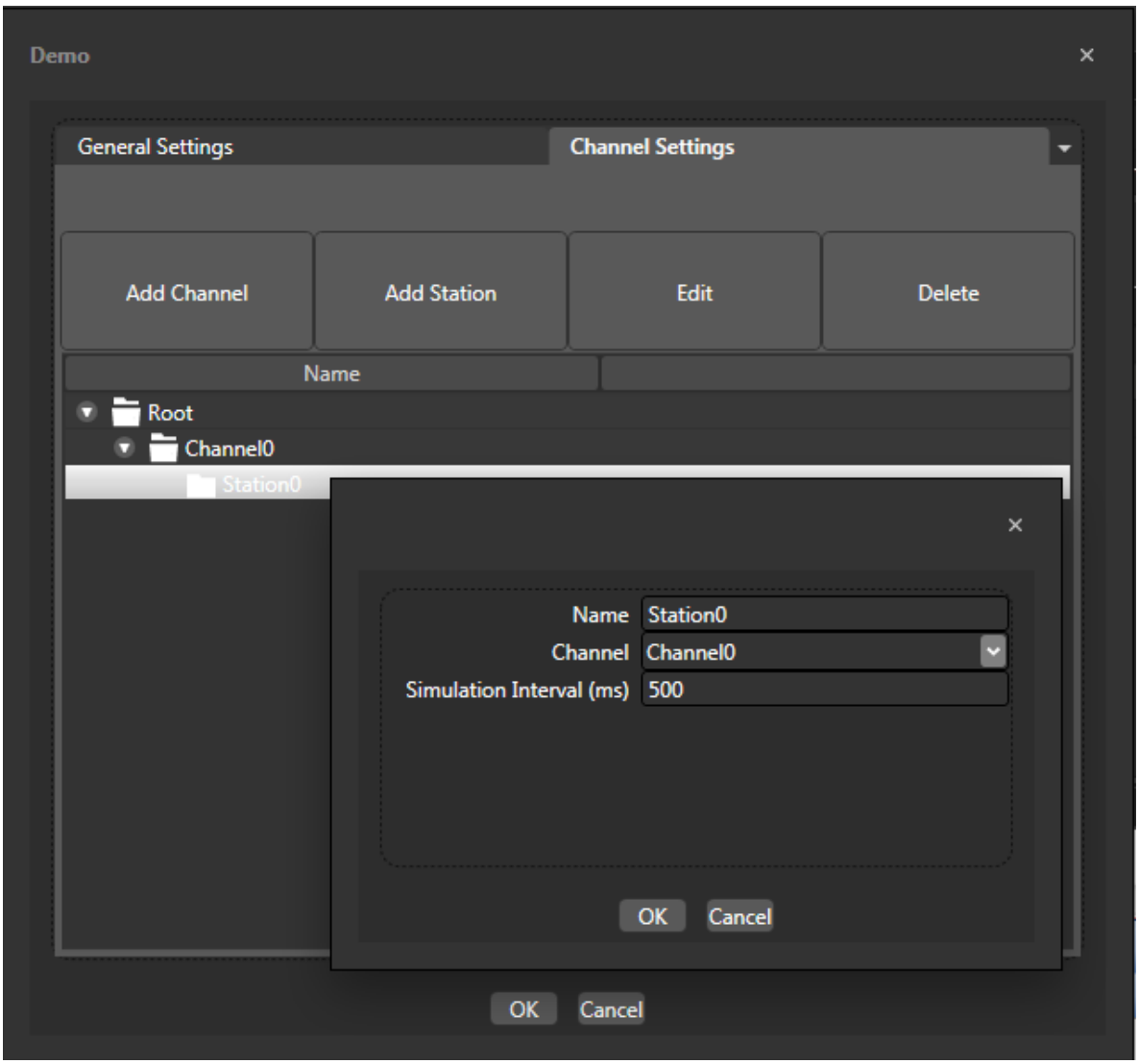
By assigning the name of a numeric variable of the Supervisor (Byte type recommended) to this property, it is possible to check the communication state with the selected station. See the following table for the meaning of the variable bits.

Bit 0 (State)	Connection channel: 0 = connected 1 = in Error

For further information please refer to the general settings chapter on "Channel Settings"

2.6. Station Settings

Once a Channel has been defined you can insert the Station, that is to be used by the Driver for communicating, in the "Channel Setting" form. The Station must be associated to a Communication Channel. To add a Station, select the Channel and activate the "Add Station" command. To delete a Station simply select it and use the Delete command. To edit the station's parameters, select the Station and double click or use the "Edit" command to open the edit window.



Window used for setting Driver Stations

The general station properties are:

Name

Name which identifies the Station corresponding to the device to be communicated with. If more than one Station is defined for the one same Driver, the name of each Station must be unique to that Station.

Channel

Name that identifies the Channel to be associated to the Station. The Channel can be selected from those already defined in the Driver. In most cases a different Channel is associated to each Station, but in specific cases it may be necessary to associate the same Channel to different Stations.

Simulation Interval (ms)

This parameter is only available for the Driver Demo and is used for defining the Driver value simulation time.

Max Retry Before Error

This parameter is used for defining the number of errors to be reached before Driver notifies communication error. When communication disturbances occur, the internal

meter consents attempts to re-establish communications before sending an error warning. Once the set number of retries has been reached, an error warning will be issued by the Driver.

State command variable

By assigning the name of a numeric variable of the Supervisor (Byte type recommended) to this property, it will be possible to control the selected station's communication state and enable/disable (start/stop) communications with this station.
See the following table on variable bits. Please note that some bits can only be used to check the state of the station, while other bits can only be used to set the state of the station (commands).

The variable's bit 1 can be used to check and modify the station's Active/Inactive state

Bit 0 (State)	Communication State: 0 = Ok 1 = Error
Bit 1 (State/Command)	Station State: 0 = Active/Enable 1 = Inactive/Disable

Allow rewriting of the same value

Inputoutput and Exceptionoutput type tasks force the value in write even when not changed.

2.7. I/O Physical Address

In this session on Variables we are going to configure the various properties that connect our driver to variables by means of the "I/O Physical Address"

Link Type

This selection of connection options is used to set the execution type to assign to the task.
The options are:

Input

This option defines the connection task type as 'Input Only'.
Therefore, when variables are in use, the driver will poll the device's associated addresses in input mode and transfer them to the desired project variables.

InputOutput

This option defines the connection task type as 'Input-Output'. Therefore, when the variables are in use, the driver will poll the device's associated addresses in input mode and transfer them to the desired project's variables. In cases where variable values are

changed by the Supervisor, the driver will execute a data output towards the device then return to Input mode.

ExceptionOutput
This option defines the connection task type as 'Output Only' to manage output on exception. This meaning only when data change.

UnconditionalOutput
This option defines the connection task type as 'Read Only' to manage continuous input independently from any data changes taking place.

Station
Name that identifies the station corresponding to the device with which to communicate. At least one station must be defined in the relevant settings card. When several stations have been defined, you must select the one, with which the task is to be executed, from with this box.

Swap Byte
This option inverts 'word' data type bytes. Therefore data connected between Supervisor and the device with 'high' bytes will be swapped with 'low' bytes and vice-versa for each of the task's word type data.

Swap Word
This option inverts double word data types with other double word data types. Therefore data connected between the Supervisor and device will be swapped over from 'high' word to 'low' word data types and vice-versa for each of the task's double word data type.

Run Script at Startup
This property is only valid for 'In "InputOutput" or "ExceptionOutput" tasks. When set to "True", the task will be executed in output at project startup.

Element Number
This property allows you to manage the conversion of data read by the device in respect to the one defined in the supervisor.

Element number is not managed by OPC Client and EIB.

- In cases where the data type to be read by the device (defined in the physical address) **is the same data type defined in the supervisor,** the user will have to set the **Element Number = 0** (default value).

- In cases where the data type defined in the supervisor **is greater** (being the number of occupied bytes), the user will have to set the **Element Number = 1;** by doing so, the number of elements will be calculated to use for reading/writing the data (of the same type indicated in the physical address of the tag from the device) based on the bytes of the data type defined in the supervisor.

E.g. Supervisor Variable : Int32 (4 bytes) Device Variable : Int (2 byte) Register No. of read elements : 2

E.g. Supervisor Variable : Float (4 bytes) Device Variable : Int (2 byte) Register No. of read elements : 2
E.g. Supervisor Variable : Double (8 bytes) Device Variable : Int (2 byte) Register No. of read elements : 4

N.B.: for Float/Double data types defined in the supervisor, data exchanged with the device will always be integer values (to scale data use the 'scaling' functions defined in the Engineering Unit within the supervisor's tag).

- In cases in which the data type defined in the supervisor **is smaller than the device's data type** (being the numb or occupied bytes) the Element Number value indicates which portion of the device's data to assign to the variables defined in the supervisor.

Eg. Device Variable : Int (2 byte) Register eg. Supervisor Supervisor : Bool Elements Number = 5 : identifies the bit 5

Conditional Variable

This is used to associate a project variable whose state will determine the conditioned execution of the communication task.
You can use the selection button, on the top right border, to select any one of the variables (Tags) that were previously inserted in the project's RealTime Database resource.
The variable (of any type) will condition the task's execution: when set at a value 'different from zero' (>< 0) the communication task will be executed by the driver.

Once the task execution has terminated successfully, the driver will automatically set the associated variable value to zero.

INDUSTRIAL AUTOMATION SOFTWARE

progea

Progea Srl
Via D'Annunzio, 295
I-41123 Modena
info@progea.com
Tel +39 059 451060

Progea International SA
via Sottobisio, 28
6828 Balerna (CH)
international@progea.com
Tel +41 91 96 76 610

Progea Deutschland GmbH
Marie-Curie Str., 12
D-78048 VS Villingen
info@progea.de
Tel +49 (0)7721 99838 0

Progea North America Corp.
2380 State Road 44, Suite C
Oshkosh, WI 54904
info@progea.us
Tel. +1 (888) 305-2999