DataWorX32 Configuration Guide

October 2015
An ICONICS Whitepaper
www.iconics.com

CONTENTS

1 About This Document

1.1 Scope of the Document

This document is intended to quickly guide users through the capabilities of DataWorX32. It describes DataWorX32 configurations which can be easily reproduced by first time users who have a basic understanding of OPC. The set of example configurations provide a better understanding of how projects can benefit from using OPC. For a complete description of the product's menus and dialogs please refer to the DataWorX32 Online Help.

1.2 Revision History

Version 6 – Raymond Van der Tas, August 2001
Version 8 – Raymond Van der Tas, August 2004
Version 9 – Raymond Van der Tas, September, 2007
Version 9.1 – Martin Jonas, December 2007

2 Introduction

DataWorX32 is an OPC client as well as OPC Server which allows managing SCADA/HMI applications in a simple way. The product is intended to be used as an add-on to GENESIS32 or any third party OPC based product. DataWorX32 is licensed by default through ICONICS Licensing software, optionally one may choose to enable the software through a hardware key (Dongle).

DataWorX32 is available in three versions: **Professional, Standard** and **Lite**. Customers who are upgrading from previous DataWorX32 versions (V6, V7 or V8) will find the **Standard edition** to be functionally equal.

Functionality	LITE	STD	PRO
OPC Registers	x	x	x
OPC Expressions	x	x	x
OPC Data Aggregation	x	x	x
Connectivity to Databases records	x	x	x
Connectivity to OPC DA tags	x	x	x
Connectivity to SNMP tags	x	x	x
OPC HDA. tunneling over TCP/IP or SOAP/XML	x	x	x
OPC A&E tunneling over TCP/IP or SOAP/XML	x	x	x
OPC DA tunneling over TCP/IP or SOAP/XML	x	x	x
Aliasing Support in OPC DA tags		x	x
Switch Alias Support in OPC DA tags		x	x
Redundancy Switching support for OPC DA Servers		x	x
Store &Forward technology for Trend Historian and Alarm Logger			x
Redundancy and switch-over support for Trend Historian			x
Redundancy and switch-over support for Alarm Server and Logger			x

Example project for a DataWorX32 LITE

a. You have a 3rd party SCADA package and purchase two (2) DWX-Lite licenses to create a tunnel between the SCADA and a remote OPC DA, AE, H.D.A Servers.
b. You have an OPC Server and would like to have OPC registers to do intelligent calculations on the OPC values, but you have no need to output these values back to an OPC Server.

Example project for a DataWorX32 Standard

a. You not only want to use intelligent calculations on OPC values, but also want to bridge some values from one OPC server to another OPC Server.
b. You want to add Redundancy Switch-over technology to a SCADA/HMI package.

Example projects for a DataWorX32 Pro

a. One (1) PRO license needs to be purchased to add Store & Forward capabilities for alarm/trend logging to a single GENESIS32 SCADA computer.
b. Two (2) PRO licenses are required to make two GENESIS32 V9 computers operate as a redundant pair.

3 Configure DataWorX32

The DataWorX32 configuration resides in a database. To configure DataWorX32 the Configurator needs to be started. Alternatively configurations can be generated in a CSV or XML format and then imported into the configuration database.

3.1 User Interface

The DataWorX32 Configurator user interface allows you to configure the required DataWorX32 functionality.

In order to start a new configuration database

1. Click Start, Programs, ICONICS, GENESIS32, DataWorX32, DataWorX32
2. Select from the menu **File, New**
3. Specify a name for the configuration: **myConfiguration.mdb** and click the **Save** button
4. Select **File, Make Active** in order to make this configuration the actively used configuration.

Folders and sub-folders can be created to logically organize the DataWorX32 items. When the configuration is completed, click the Traffic Light icon  on the toolbar to start **runtime**.

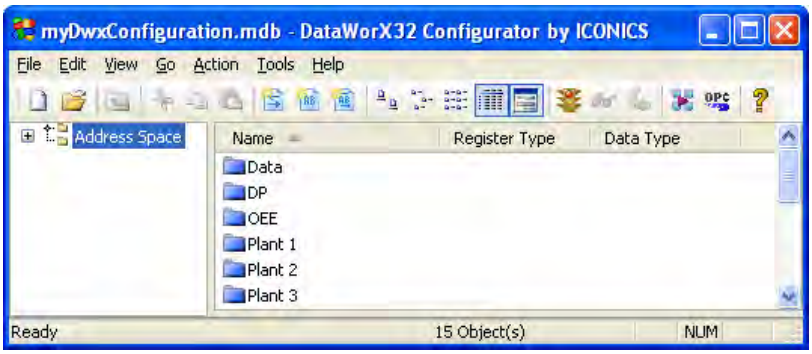


Figure 1: DataWorX32 Configurator

3.2 Online Configuration

The traffic light icon may be used to start/stop DataWorX32. However, changes to the configuration database can also be made while DataWorX32 is in runtime. When changes are made, the “Update Runtime” icon should be clicked before the changes are accepted by the DataWorX32 server.

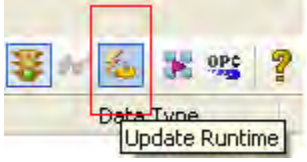


Figure 2: Update Runtime Button

3.3 Export / Import CSV, XML or Binary

The menu allows us to **export** the configuration to a CSV file. This way quickly configuration changes can be made using your favorite text editor. The changes can be **imported** again into the DataWorX32 configuration. Importing or Exporting a DWX Binary configuration file is only required when a version 7 or older configuration is to be upgraded to the new open database format in DataWorX32 V9.

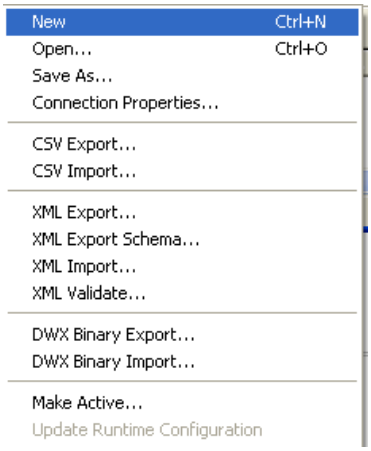


Figure 3: File Menu

3.4 OLE Automation

Automation can be used to programmatically connect to the objects in DataWorX. This may for instance be useful when a Visual Basic program periodically may want to read/write to Tags in DataWorX.

3.4.1 VBA Procedure Connects to DataWorX

Open your VBA hosting application (e.g. GraphWorX32) and select from the VBA menu **Tools, References** and reference

ICONICS DWXRuntime Automation Type Library

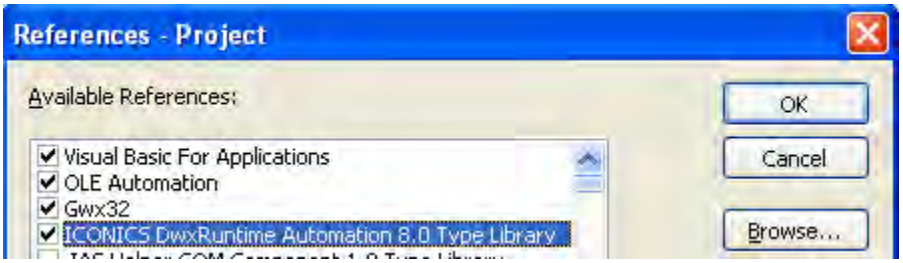


Figure 4: VBA References

The next VBA code example would read and increment a value of the register **Data.P1**

```
Dim dwx As DwxRuntime          '          DataWorX32
Dispatch pointer
  Dim reg As IRegister        ' DataWorX32 Registers
```

```
Set dwx = New DwxRuntime
Set reg = dwx.GetRegister("Data.P1")
reg.Value = reg.Value + 1
```

VBScript Code equivalent

```
Dim dwx
Dim reg

Set dwx = CreateObject("ICONICS.DwxRuntime")
Set reg = dwx.GetRegister("Data.P1")
reg.Value = reg.Value + 1
```

4 Example DataWorX32 configurations

Before exploring DataWorX32 we first need to have an OPC Server that exposes some process values. We could use the ICONICS Simulator OPC Server for this purpose.

4.1 ICONICS OPC Simulator

The ICONICS Simulator OPC Server allows us to create simulation OPC Tags.
The next example describes how to configure a simulated Boiler application.

To configure the ICONICS Simulator OPC Server:

1. Click, Start, Programs, ICONICS Tools, **Simulator OPC Simulator**
2. Select **Add, New Device…** from the menu.
3. Specify a Name for your device: **PLC**
4. Right mouse click on **PLC** in the left window, and select **New, Group…**
5. Specify a name for the folder, for instance **Boiler1** and select the **OK** button
6. Right mouse click on the **Boiler1** group in the left window, and select **New, Tag…**
7. Specify a name for the tag, for instance **Temperature** and configure the tag as seen below.

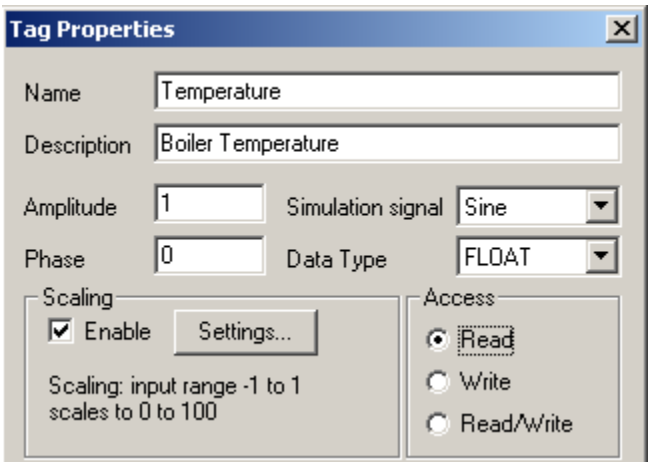
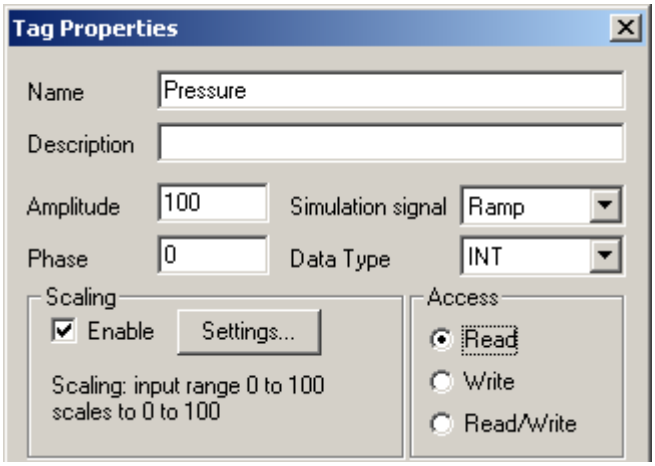


Figure 5: Tag Properties

8. To add another tag, right mouse click on the folder **Boiler1** in the left window, and select **New, Tag…**

9. Specify a name for the item, for instance **Pressure** and configure the tag as seen below.



10. Right mouse click once more on the folder **Boiler1**, and select **New, Tag...**
11. Specify a name for the item, e.g. **Setpoint** and configure the tag as seen below.

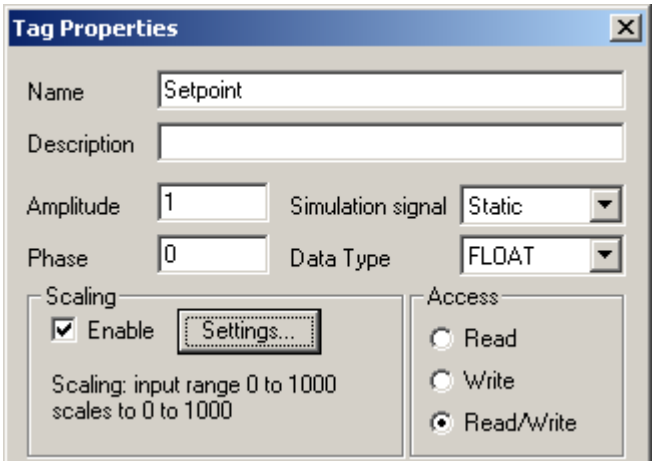


Figure 6: Tag Properties

12. Repeat steps 5 to 11 for another Group of tags. The Group can be called **Boiler2**

After completing these steps, the OPC Server configuration will look like this:

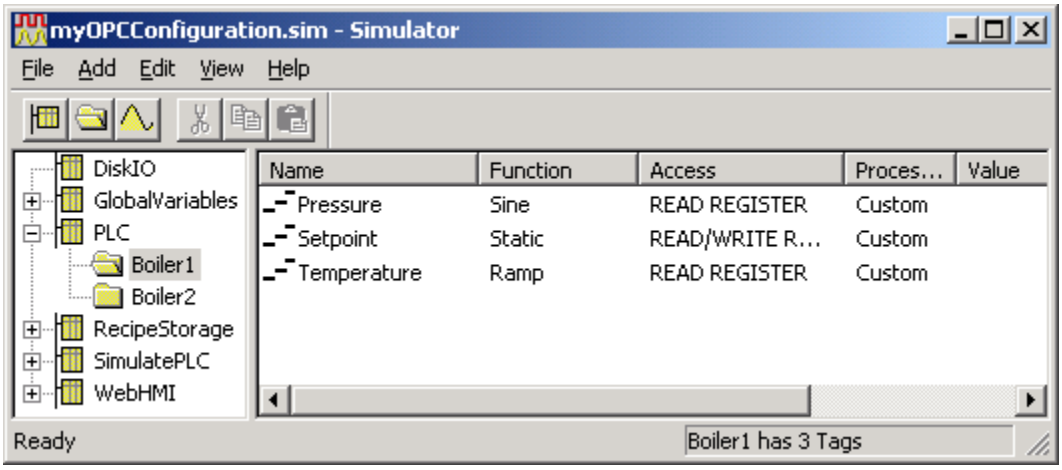


Figure 7: ICONICS Simulator OPC Server

13. Select **File, Save** from the menu.
14. You can select **File, Exit** in order to exit the Simulator Configurator.

4.2 OPC Server / Client capability

Information in the DataWorX32 address space (registers, expressions, aliases, etc.) can be accessed by OPC Clients such as GraphWorX32. DataWorX32 is not only an OPC Server, but it also performs OPC Client functionalities when it for instance connects to local and remote OPC Servers.

DataWorX32 is available as a standalone product and can be used in conjunction with 3rd party SCADA/HMI products to improve OPC connectivity and to provide added functionality such Tunneling, Redundancy, Data bridging, Expressions or network communications.

4.3 OPC Aliasing

Alias names can be configured as place holders for the actual OPC Tag names.
For example you configure an alias called TEMPERATURE and connect it to SomeOPCServer\SomeOPCItem.
The OPC client (GraphWorX32) can connect to the new name as ICONICS.DataWorX32\[[TEMPERATURE]]

As an example, machine builders appreciate the use of aliases since they deliver machines connected to different PLC brands depending customer demand. When a PLC is chosen, only the Alias reference list needs to be updated and the machine can be shipped.

An example project with aliases:

1. Select from the menu **Edit, New, Alias...** and specify the alias name: **Temperature**

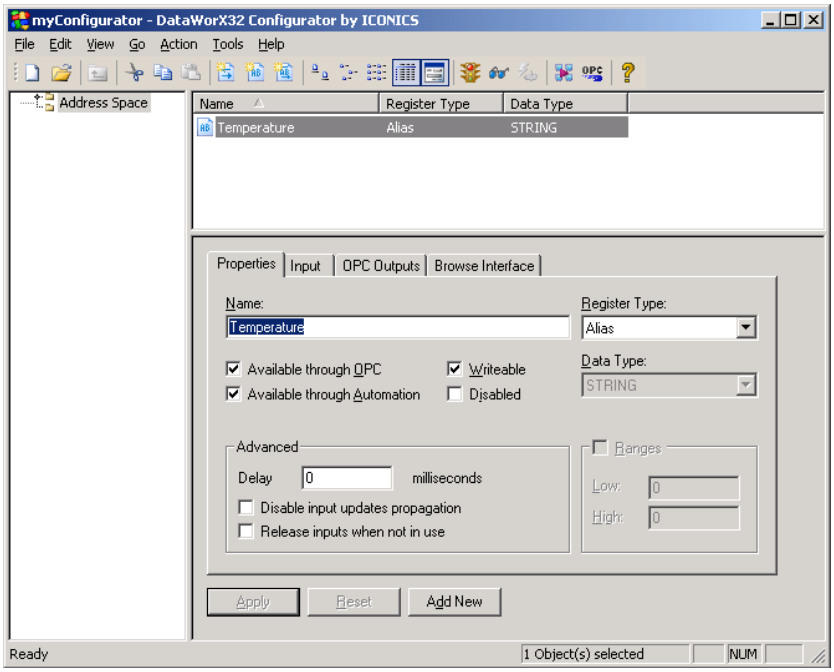


Figure 8: New Alias

2. Click the Input tab and specify the alias value: **ICONICS.Simulator.1\PLC.Boiler1.Temperature**

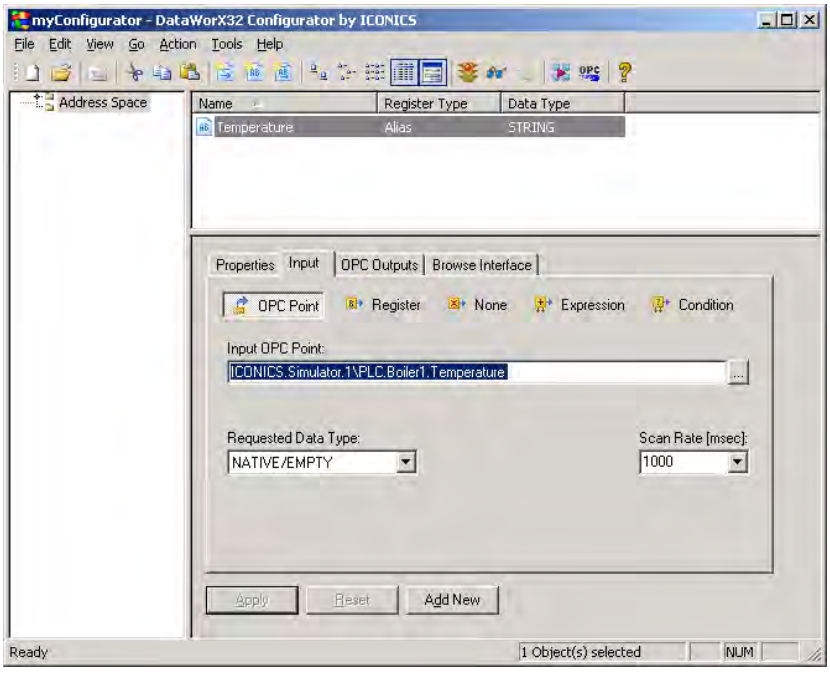


Figure 9: Alias Input

3. Set the Scan Rate to for instance 500 ms.
4. Perform the step 1 and 2 for two more aliases

Alias Name	Input OPC Point
Pressure	ICONICS.Simulator.1\PLC.Boiler1.Pressure
Setpoint	ICONICS.Simulator.1\PLC.Boiler1.Setpoint

5. Click **Apply** and click the traffic light icon on the DataWorX32 toolbar to start **runtime**.
6. Start **GraphWorX** and place a process points on the display
7. Click on **Data Tags...** and use the UDB to browse to the ICONICS.DataWorX32 Server.

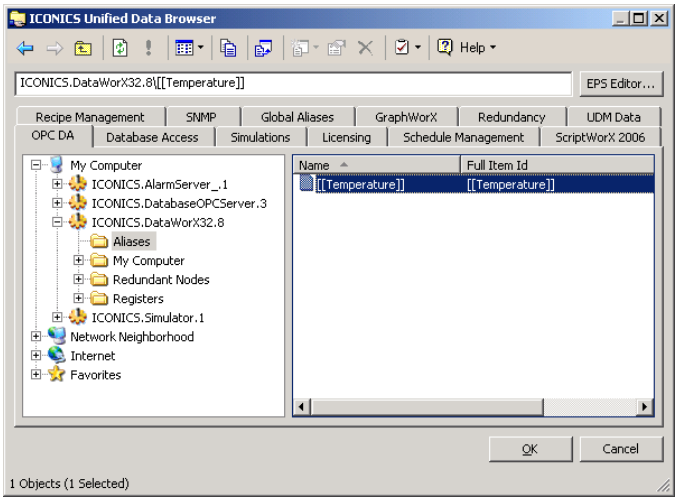
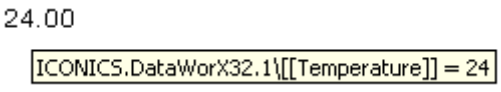


Figure 10: Unified Data Browser

8. The browser shows the Aliases folder where you can select **[[Temperature]]** and click **OK** twice.
9. Select **Runtime** from the menu and your process point will show something like:



Aliases can also be used in a more complex manner. You could for instance have a combined set of aliases to build up one tag. For example you create the following aliases

Alias Name	Default value
OPC Server	ICONICS.Simulator.1\
Device	PLC.
Boiler	Boiler1.

The tag in the OPC Client could then be

ICONICS.DataWorX32.1\[[OPCServer]][[Device]][[Boiler]]Temperature

You could use this technique if you want to dynamically change the content of for instance the [[Boiler]] alias. Your display could then multiplex between an unlimited amount of boilers: Boiler1, Boiler2, ... BoilerN. In order to control (read/write) the [[Boiler]] alias you browse to the Registers folder and select the string type tag

ICONICS.DataWorX32.1\Boiler

During Runtime this register will show something like...

4.4 OPC Bridging

You can bridge OPC Tags (registers) which reads OPC values from one tag and writes these to another OPC Tag when the input value changes. This functionality is used when you want to bridge process information between two OPC servers.

An example where the temperature of one boiler is used as a setpoint for another boiler:

1. Select from the menu **Register, Add...** and specify the register name **Gateway**
2. Select the **Input Tab** and Browse for the OPC Item **ICONICS.Simulator.1\PLC.Boiler1.Temperature** and specify a scan rate of **1000 ms**
3. Select the **Output Tab** and Browse for the OPC Item **ICONICS.Simulator.1\PLC.Boiler2.Setpoint**

The **Refresh Outputs** property allows optionally set a periodic refresh for the output. By default the value will only be written when the input changes in order to give the best communication performance.

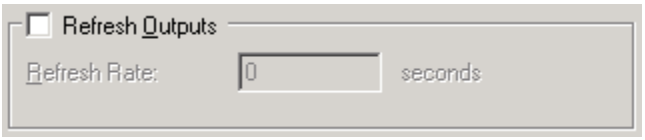


Figure 11: Refresh Outputs

The **Release inputs when not in use** checkbox must not be checked off. This feature must be turned off otherwise Bridging would not work because inputs would be disconnected unless a client is connected to the register.

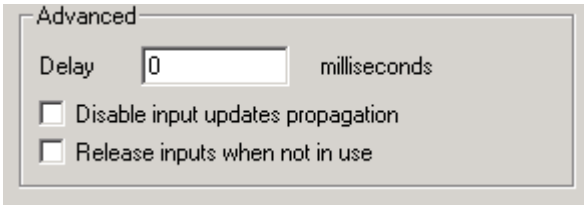


Figure 12: Advanced Properties

4.5 OPC Input Updates Propagation

The **Disable input updates propagation** parameter needs to be checked when the input and the output of the register are connected to one and the same OPC Item. This ensures that changes on the INPUT of the register are not written to the OUTPUT. Only changes written by OPC Clients connecting to the register will be written to the output.

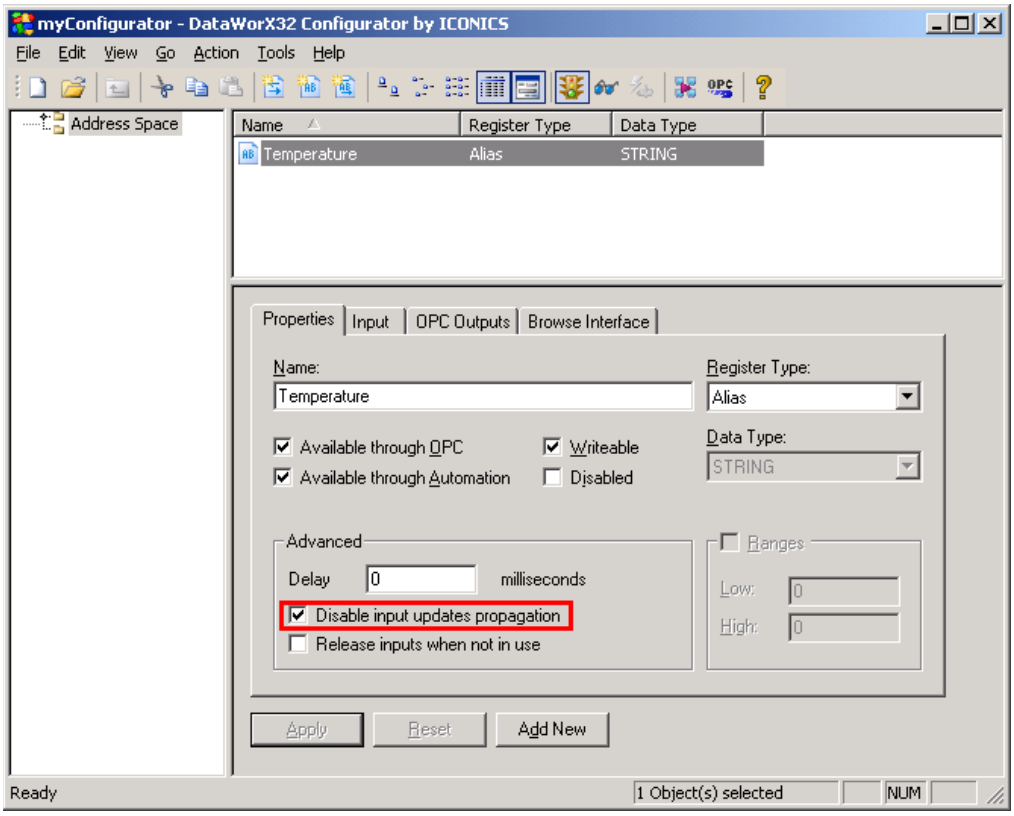


Figure 13: Disable input updates propagation

The **Delay** parameter is an internal delay that is used before a write to the Output occurs when the input value changed.

4.6 OPC Redundancy

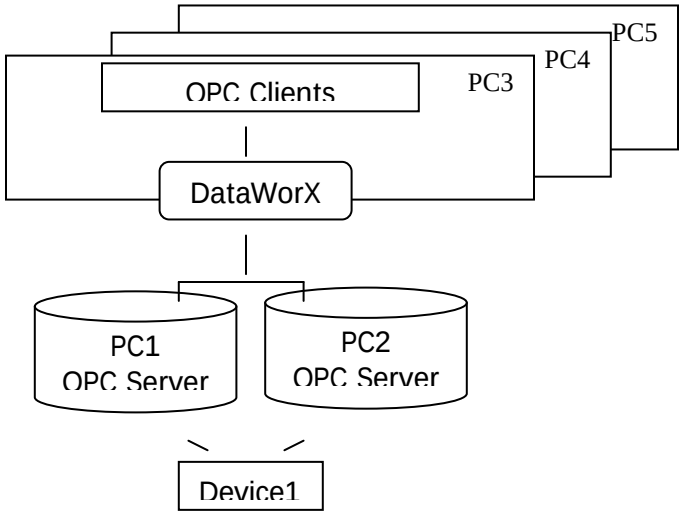
Having continuous access to the OPC Server is critical in many projects. DataWorX32 can be configured for setting up redundant OPC communication channels. If one communication channel fails, DataWorX32 automatically switches to the backup communication channel. Redundancy can be obtains in many ways. When implementing redundancy, you will need to consider which part in our application is most likely to fail. We will discuss a few options:

1. Computer Redundancy
2. Device Redundancy

Note: SCADA redundancy (synchronized alarm/trend servers and databases and automatic switching of the visualization layer to the active SCADA servers) is another topic which will not be discussed in this section. SCADA redundancy is available when installing a set of **DataWorX32 Professional** licenses.

4.6.1 Redundancy Alias for Computer redundancy

Computer redundancy solves the problem when an OPC Server computer fails. In case there's a redundant computer, this computer can still supply the OPC data. In the next example you will have three or more computers; one for DataWorX32 and one for each OPC Server. The configurations of the two OPC Server computers are identical.



4.6.2 Install GenBroker on the OPC Server computers

When using standard Microsoft DCOM the switch over between primary and backup OPC server may have a worst case 6 minute delay, for this reason we would like use Genbroker technology, which allows us to select a communication method **OPC Direct** which enables configurable timeouts, **OPC over TCP/IP** or **OPC over SOAP/XML**.

To install GenBroker technology on computers PC1 and PC2.

1. Insert the GENESIS32 Options CD ROM and click **Genbroker** to install it.

Figure 14: GENESIS32 Initial Installation Screen

2. Also select that you would like to include GenTray in the installation.

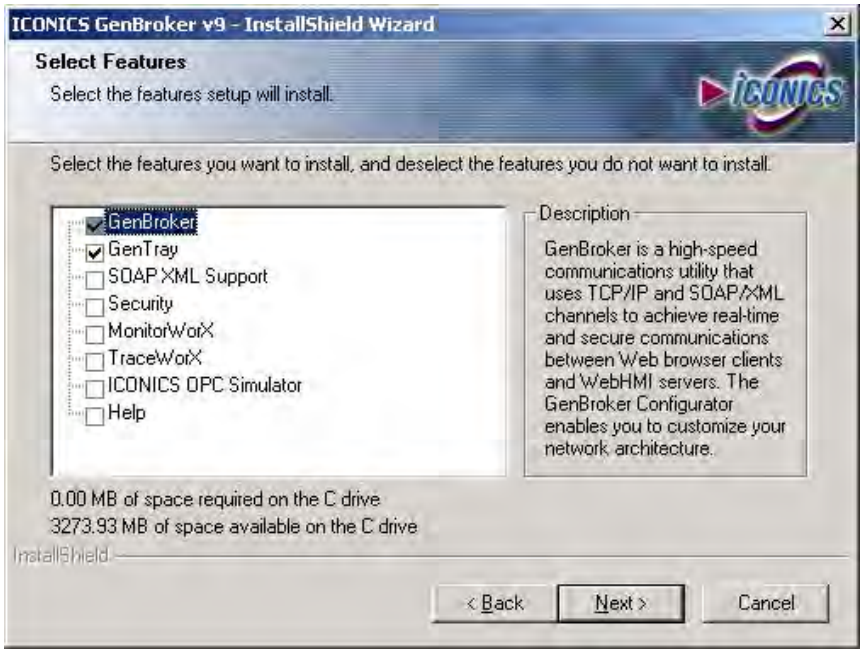


Figure 15: GENESIS32 Module Installation

3. Select Start, Programs, ICONICS Tools, GENESIS32 Tray

4. Use the GENESIS32 Tray (GenTray) system tray icon to start GenBroker Server as shown in the next figure.

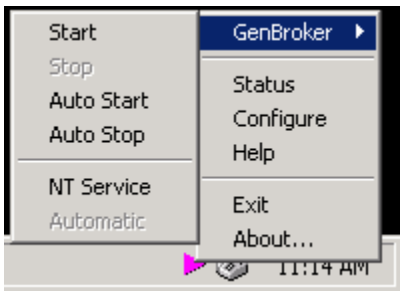


Figure 16: GenTray

4.6.3 Configure GenBroker on the DataWorX32 computer

When DataWorX32 is installed, GenBroker is already available and ready to be configured to communicate OPC over TCP/IP.

1. Select **Start, Programs, ICONICS Tools, GenBroker Configurator**
2. Click the **New...** button.

Figure 17: GenBroker Configurator

3. Select **Channels** in the left window pane
4. Right-Mouse click the **OPC over TCP/IP Direct Channels** and set it to **Default**.
5. Also add the **Node** names of the server computers e.g. PC1 and PC2 to this channel

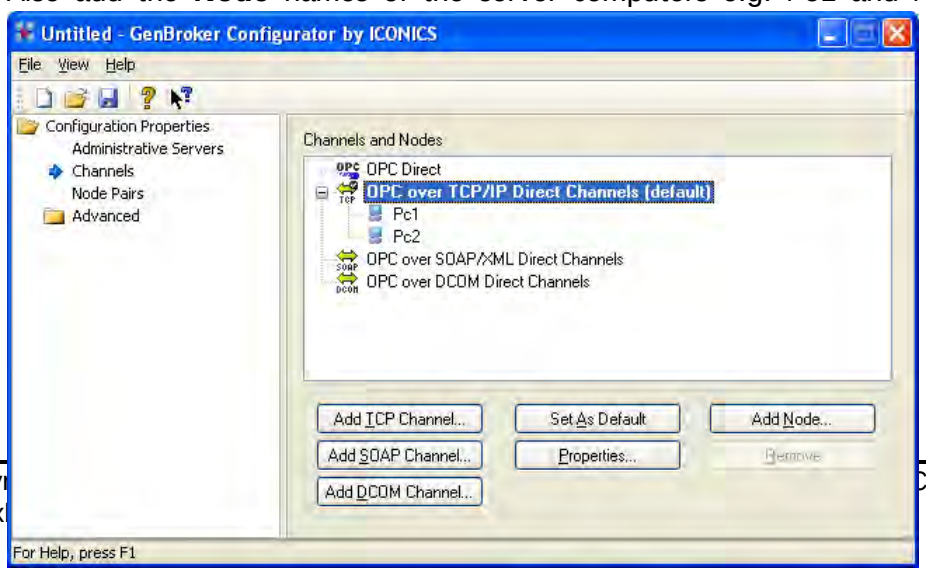


Figure 18: GenBroker Channels Configuration

6. Click **File, Save** specify a name for the configuration, for instance **mygenbroker.gbx**,
7. Select the **Save** button. Select **File, Exit** and click **OK..**

DataWorX32 can now connect to remote OPC Servers via the OPC over TCP/IP channel.

4.6.4 Configure Redundancy Alias in DataWorX32

The DataWorX32 redundancy configuration requires the following steps:

1. Select from the menu **Edit, New, Redundancy Alias**
2. Under Redundancy tab click **Add…** to add the primary node **PC1**
   (Trick: for test purposes you can specify the name of your DataWorX32 computer as the primary node)
3. Click **Add…** to add the backup node **PC2**
4. Specify the OPC Server Name, for instance **ICONICS.Simulator.1**
5. Click **Apply**.

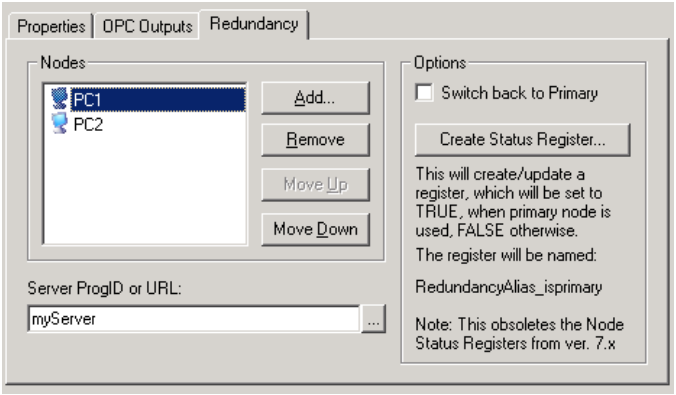


Figure 19: Redundancy Alias

6. Start **GraphWorX** and place a process points on the display
7. Click on **Data Tags…** and use UDB to browse the **ICONICS.DataWorX32** Server
8. Select the folder **Redundant Nodes** and browse **[[MyRedundancyAlias]]**

9. Select the tag **ICONICS.DataWorX32.1\[[MyRedundancyAlias]]\SimulatePLC.Ramp**
10. Click **OK** and start **Runtime**

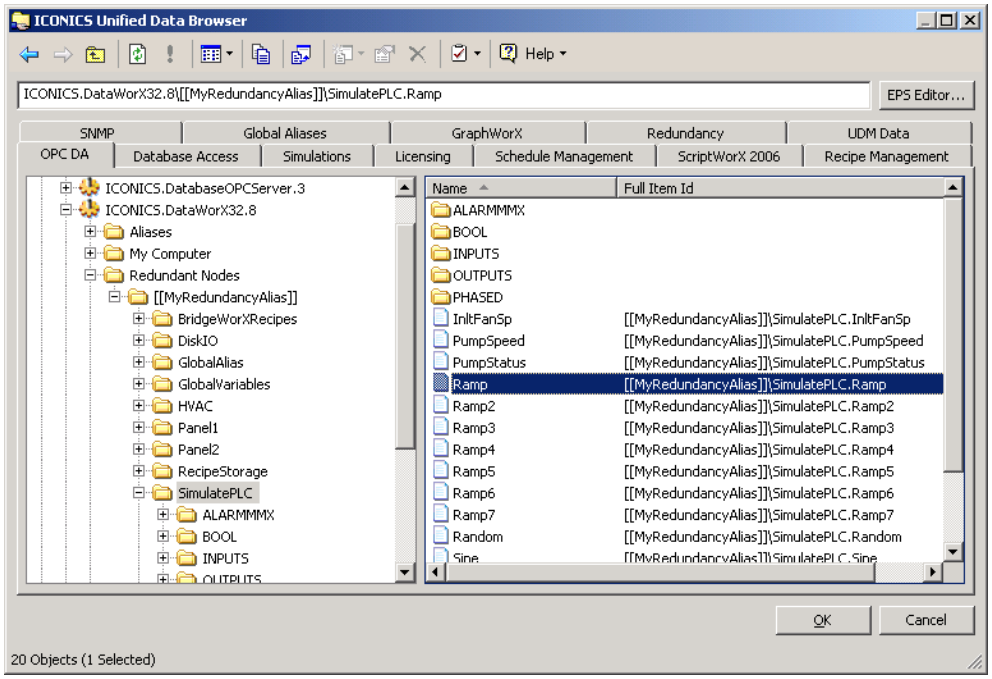
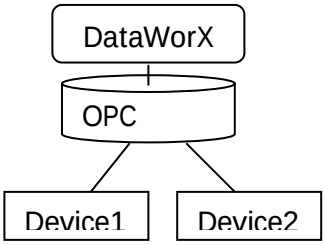


Figure 20: Unified Data Browser

4.6.5 Switch Alias for Device Redundancy

Device redundancy means that you will have at least two identical devices that are connected to your OPC Server.



Some OPC Servers have built-in device switch over. Your OPC Server will then automatically detect a failure in the communication and it will switch over to the backup device when needed. If this is the case with your server you do not need to configure DataWorX32 for the redundancy.

Let's assume your OPC Server is a simple server and you need the help of DataWorX32.

First you configure your OPC Server to allow communication to the two devices. In the example below, we configured two identical devices **DEVICE1** and **DEVICE2**.

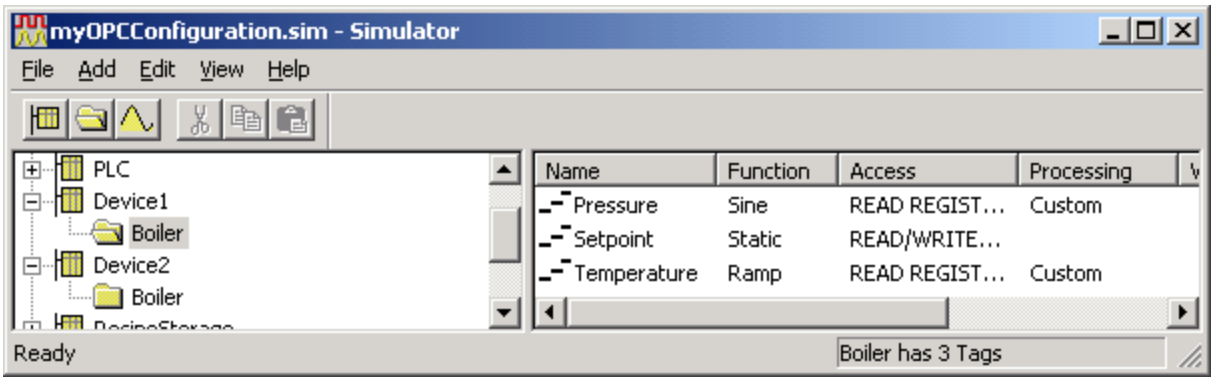


Figure 21: ICONICS Simulator OPC Server

In DataWorX32 we need to add a Switch Alias.

1. Select from the menu **Edit, New, Switch**
2. Specify the Switch name: **DEVICE**
3. Add two strings **DEVICE1** and **DEVICE2**

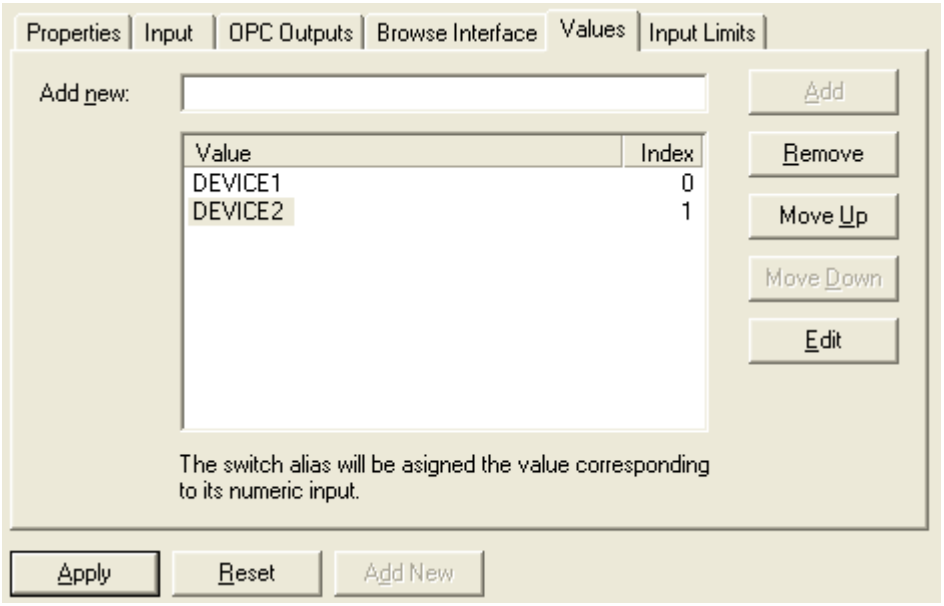


Figure 22: Switch Alias

4. Select the **Browse Interface** tab

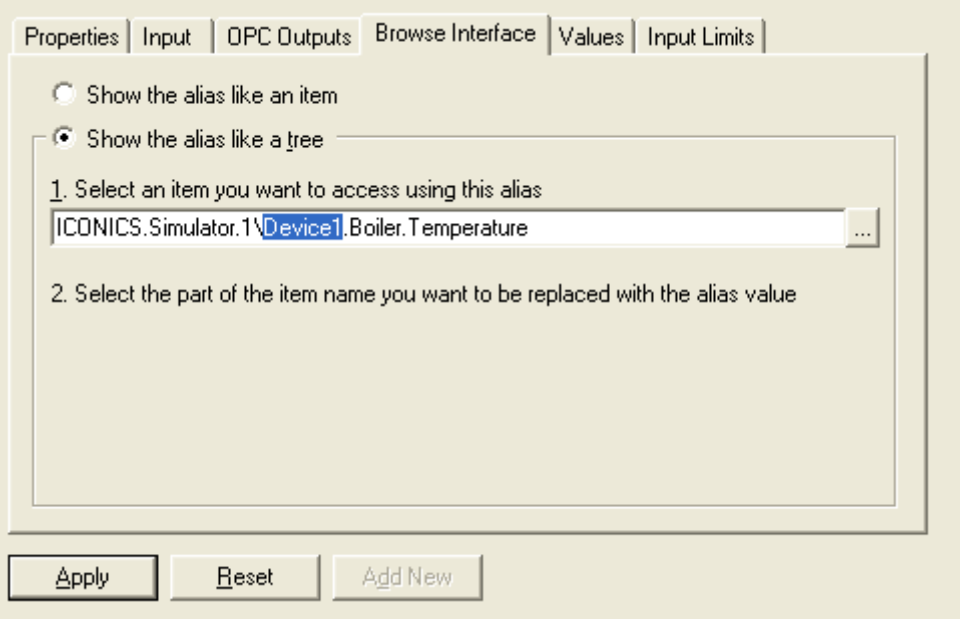


Figure 23: Switch Alias Browser Interface Tab

You can browse to your OPC Server and select one of the tags in Device1. You will see that the part **Device1** of the OPC Item is highlighted. You could also highlight this part yourself. This part will switch between Device1 and Device2 depending on your Device switch alias. OPC Clients will also be able to quickly select the proper tag names later on when browsing through the Device Switch Alias.

5. Click **Apply**
6. Select the **Input** tab to specify a register or OPC tag that controls the switch to position **0** which is Device1 or position **1** which is Device2.
7. Start **GraphWorX** and place a process points on the display
8. Click on **OPC Tag…** and use the Tag browser to browse to the ICONICS.DataWorX32 Server.
9. Under the **Aliases** folder you see the Switch alias [[Device]] and you can select the tag you want.
10. Click **OK** and start GraphWorX **Runtime**.

During Runtime tooltip of a GraphWorX32 Process Point will show something like

64.00

ICONICS.DataWorX32.1\ICONICS.Simulator.1\[[Device]].Boiler.Temperature = 64

4.7 OPC Registers

Registers can be configured in DataWorX32 which then can be used by OPC Clients such as GraphWorX32, AlarmWorX32 Server, TrendWorX32 Logger, etc. The registers can also be accessed by Automation via VBA and Visual Basic.

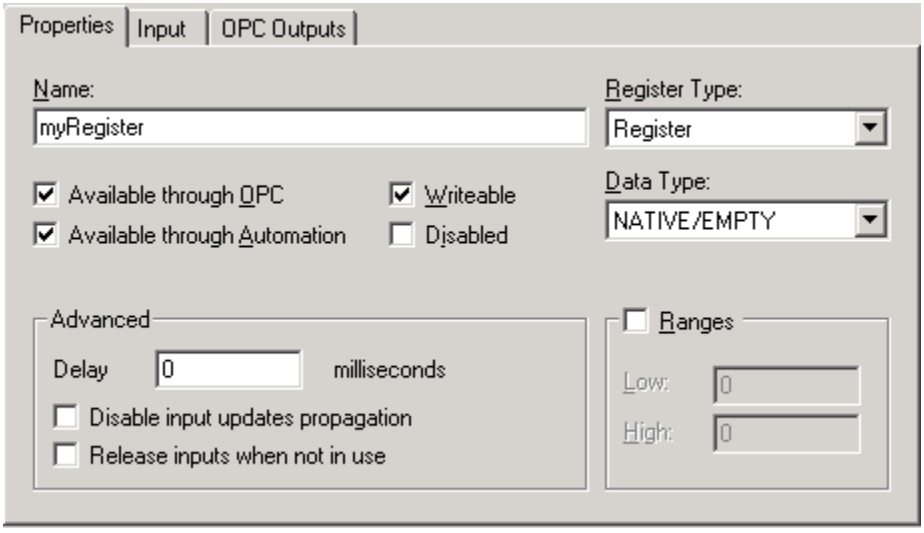


Figure 24: Register

A register can be configured to represent one of the following inputs:

1. The value of an OPC Item
2. The value of another DataWorX32 Register
3. The value of a calculation (expression)
4. The value of an OPC Condition
5. A preset value

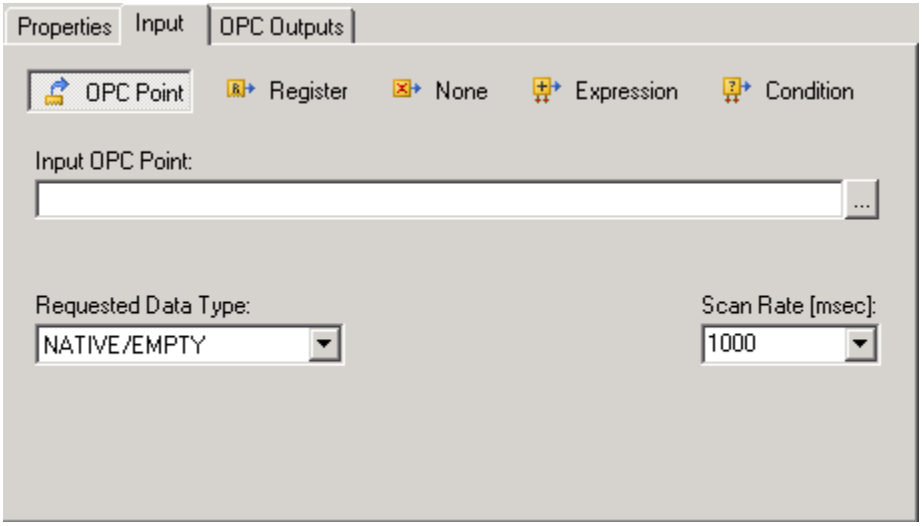


Figure 25: Input

The resulting register value can optionally be written to one or more Output OPC tags.

Some reasons why you would choose one of the above input types:

Register represents the value of an OPC Item

- Registers allow you to specify meaningful Tag Names and manage your OPC Tags in a central place.
- The register allows you set a specific scan rate. All OPC Clients that connect to this register will all get the same update rate. Data aggregation will reduce the load on the OPC Server.
- The Register is read-only and therefore protects the OPC Tag from being written. If you want the register to allow read/write, you will need to specify the Output tab to output to the same OPC Tag and select the "disable input propagation" property.

Register represents the value of another Register

- The register may get a more meaningful name than the register it gets the values from.
- The register may be temporarily connected to the register and connected to another input tag when the PLC becomes available to the engineer.

Register represents an expression

- The register holds a value that is the result of a calculation.
- The register can calculate the scaling from raw values to engineering units.

Register represents an OPC Condition

- The registers represent the condition of a set of OPC Tags, the largest value, smallest, etc.
- The change in condition can be used to trigger a switch alias.

Register represents an internal value

- The register contains a value that may be used as a global project parameter. The register value can be changed during runtime but the changes are not persistent.

4.8 OPC Expressions

OPC Registers can be configured to calculate user defined expressions. The calculated value is exposed as a regular OPC Tag. The advantage of this functionality is that you then no longer need to perform these calculations (expressions) in the OPC Client applications but have them available on a global level.

As an example we want to see the average temperature of two temperature sensors:

Follow the next steps

1. Select from the menu **Register, Add...** and specify a register name. For instance **AvgTemperature**
2. Select the **Input** tab and specify the expression **X = (** *Tag1* **+** *Tag2* **) / 2**

   Where Tag 1 is **{{ICONICS.Simulator.1\PLC.Boiler1.Temperature}}** and Tag 2 is **{{ICONICS.Simulator.1\PLC.Boiler2.Temperature}}**

3. Click the Traffic Light icon  on the DataWorX32 toolbar to start **runtime**.

The register is now available to OPC Clients as the OPC item *ICONICS.DataWorX32.1\AvgTemperature*

4.9 User Defined Simulation Signals

The next example demonstrates how to configure DataWorX32 as a storage place for simulation tags such as for instance ramping values, sine waves or pulse trains.

Right-Mouse Click the Address Space and select **New, Folder**

Specify **Data** as the name for the folder.

4.9.1 Saw Tooth

1. Right-Mouse Click the newly created **Data** Folder and select **New, Register** and set the name of the register to **Ramp** and set the High Range to the value **360**.

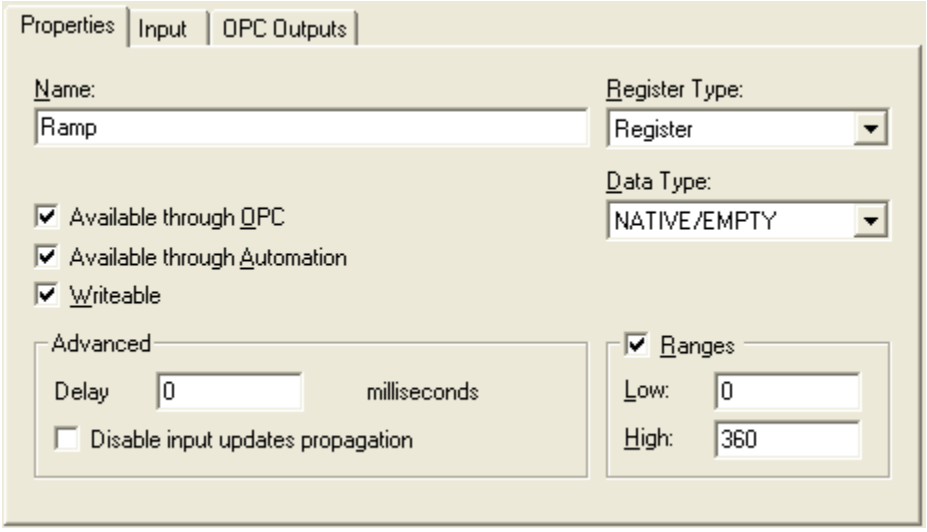


Figure 26: Properties

2. Click the **Input** tab and select **None** as the Input with an initial value of **0** and Click the **Apply** Button.

3. Add another register to the **Data** Folder and call it **RampGenerator.** It needs to generate a value from 0 to 360 which then will be stored back in the Data.Ramp register. See the next figure for the settings.

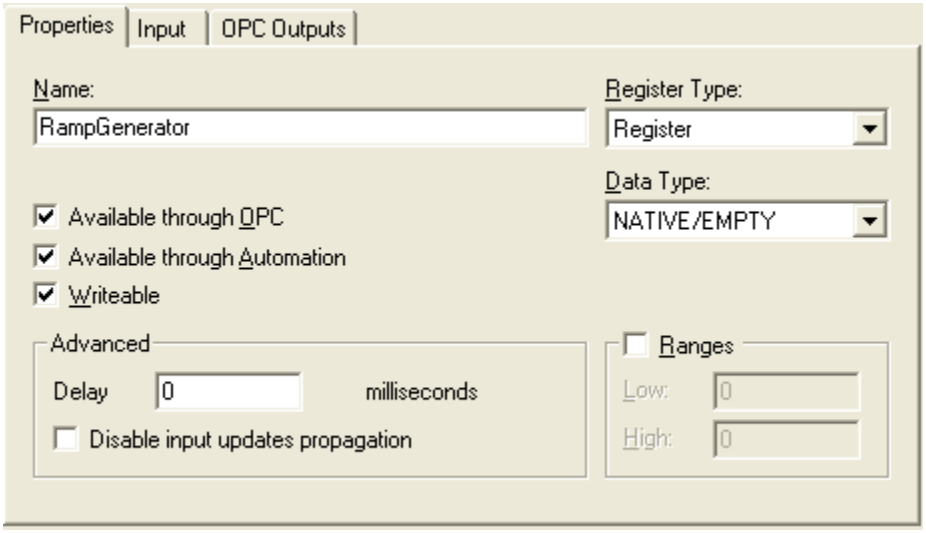


Figure 27: Properties

4. Click the **Input** tab and configure the following

Expression: **x= if({{Data.Ramp}} < 360, {{Data.Ramp}} + 1, 0)**

5. Click the **Output** tab and add the OPC Tag **ICONICS.DataWorX32.8\Data.Ramp**

4.9.2 Sine Wave

1. Add another register to the **Data** Folder and call it **Sine.** To let is behave like a sine we can use the Expression which uses the function

sin (2***pi***degrees_tag/360).

2. Click the **Input** tab and configure the following
   Expression: **x= sin(pi * {{Data.Ramp}}/180) * 100 + 100**

3. The sine function returns a value between -1 and 1 based on the degrees that the ramp provides. By multiplying it with 100 and adding 100 on top of it, the final expression returns a value between 0 and 200.

4.9.3 Pulse Train

1. Add another register to the **Data** Folder and call it **Pulse.**
2. Click the **Input** tab and select **None** as the Input with an initial value of **0** and Click the **Apply** Button.
3. Add another register to the **Data** Folder and call it **PulseGenerator** and set the **Delay** property to 1000 ms.

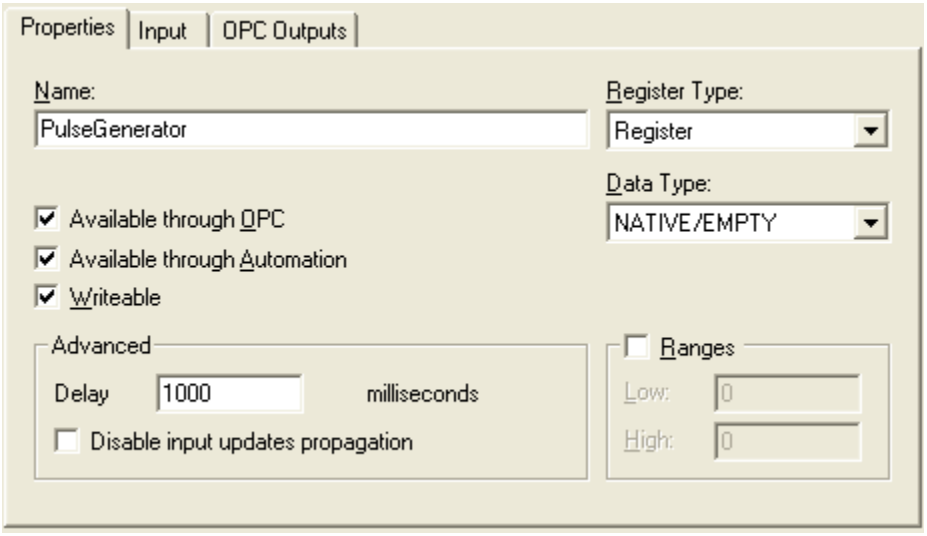


Figure 28: Properties

4. Click the **Input** tab and configure the following
   Expression: **x= if({{Data.Pulse}},0,1)**

5. Click the **Output** tab and add the OPC Tag **ICONICS.DataWorX32.8\Data.Pulse**

The next figure demonstrates that the DataWorX32 simulation signals behave as expected in the ICONICS TWXViewer32 ActiveX.

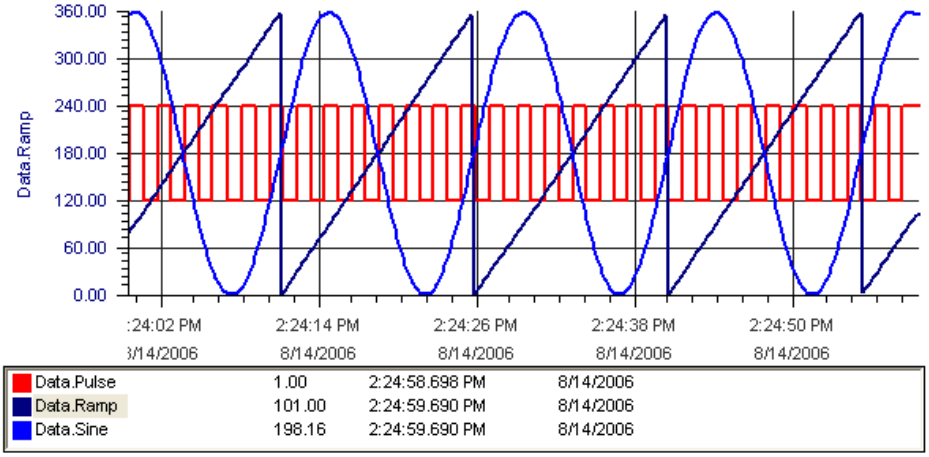


Figure 29: TWXViewer32 ActiveX

4.9.4 Snapshot Register

A snapshot register takes a snapshot of a process value and stores the value until the snapshot register is triggered again. Add a register to the **Data** Folder and call it **PV** which will be used to store the process value**.**

1. Click the **Input** tab and select **None** with an initial value **0**.
2. Add a register to the **Data** Folder and call it **SnapShot** which will be used to copy the process value when a bit goes high.
3. Click the **Input** tab and configure the following

Expression: **x= if({{***your_OPC_bit***}},{{***your_OPC-analog_value***}},{{Data.PV}})**
Scan Rate: **500 ms**

your_OPC_bit is the tag that goes high when we need to take the snapshot.
your_OPC-analog_value is the tag that represents you real process value.

4. Click the **Output** tab and add the OPC Tag **ICONICS.DataWorX32.8\Data.PV**

Whenever **your_OPC_bit** becomes active (non-zero), **your_OPC-analog_value** will be copied to the **Data.PV** register. The time stamp of the **Data.PV** register will be the moment **your_OPC_bit** became active.

4.9.5 Time Date Register

Registers in DataWorX32 can use the Extended Point Syntax (**EPS**) **tag:***some_OPC_tag*#**timestamp** to present for instance the time stamp of the OPC tag rather than the OPC value itself.

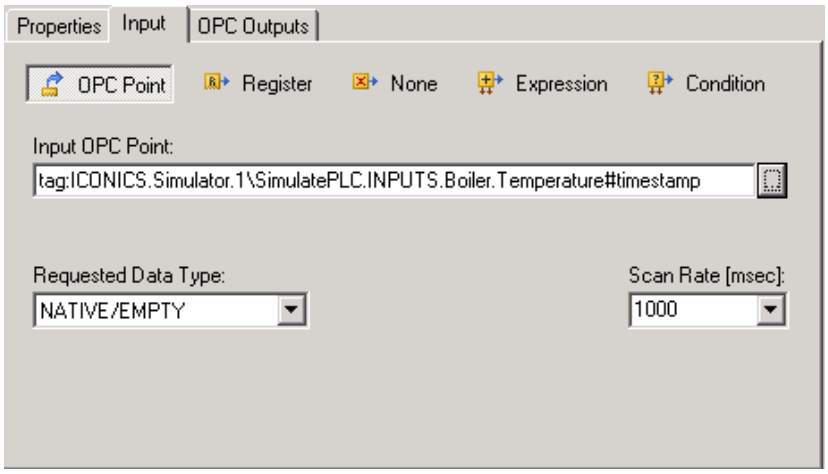


Figure 30: EPS Input OPC Point

4.10 OPC Condition Registers

You can configure OPC Registers that determine the condition of a set of OPC Tags. The condition register is exposed as a regular OPC Tag. As an example you want to determine if the communication cable to the PLC is broken. If it's broken you want to notify someone about the alarming situation. You may even want to use the condition to perform DataWorX32 Switch Aliasing.

To configure an OPC Condition Register, perform the next steps

1. Start **DataWorX32**
2. Select from the menu **Register, Add...** You can call the register **CableCondition**
3. Select the **Input** tab and click the **Condition** tab.

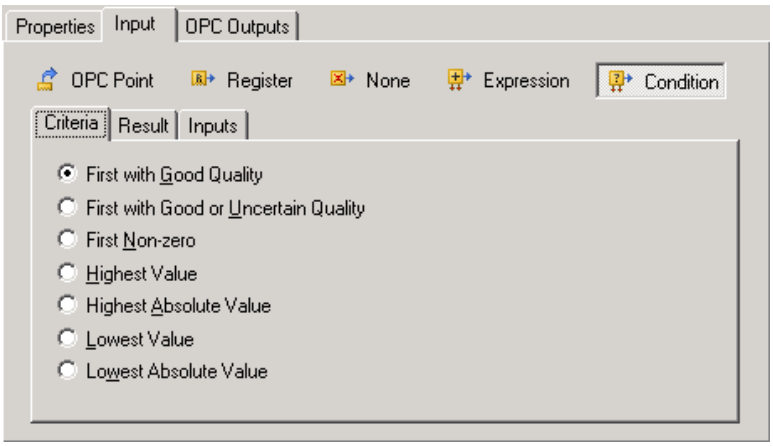


Figure 31: Condition Criteria

4. Select **First with Good Quality** and select the **Result** tab to specify **Zero-based index**

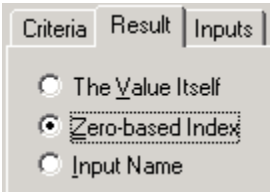


Figure 32: Condition Result

A cable failure should cause your OPC tag to provide a **bad quality**. We could use this state to determine the cable condition as soon as we have one OPC Tag which relies on the cable. We need another tag which always provides a **good quality** this could be another DataWorX32 Register or a tag from the Simulator OPC Server **ICONICS.Simulator\SimulatePLC.Ramp**.

5. Click the Inputs Tab of this condition and add for example the following two OPC Tags:

   YourServer\YourTag for example *I\PC1\Kepware.KepserverEX.V4\PLC.Boiler.Temperature*
   ICONICS.Simulator\SimulatePLC.Ramp

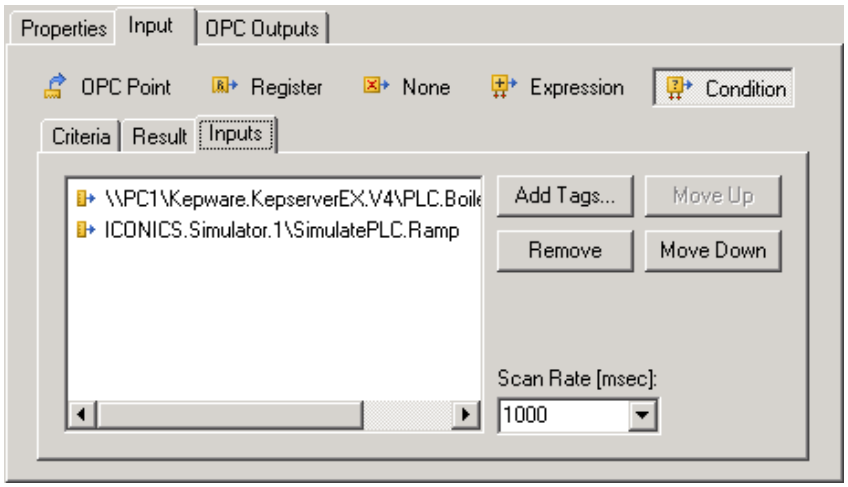


Figure 33: Condition Inputs

Make sure **YourServer\YourTag** is the top one in the list. It will correspond to Index 0. During runtime the **CableCondition** will provide value **0** in normal cases, but as soon as the cable breaks, the value will be **1**.

6. Click **Apply** and click the DataWorX32 traffic light to start **runtime**

CableCondition is now available to OPC Clients as the OPC Tag *ICONICS.DataWorX32.1\CableCondition*

Note: An identical functionality would be obtained if a regular register was configured with the expression **x= if(quality({{***Kepware.KepserverEX.V4\PLC.Boiler.Temperature* **}}) == 192, 0, 1)**. It would test the quality value of the OPC tag to be 192 (good quality) and return the value 0 and otherwise 1 to indicate a problem with the connection to the PLC.

5 DataWorX32 Settings

The **Tools, Options...** menu allows you to configure DataWorX32 Settings. In general you do not need to change these settings, but we will briefly explain them.

General Settings

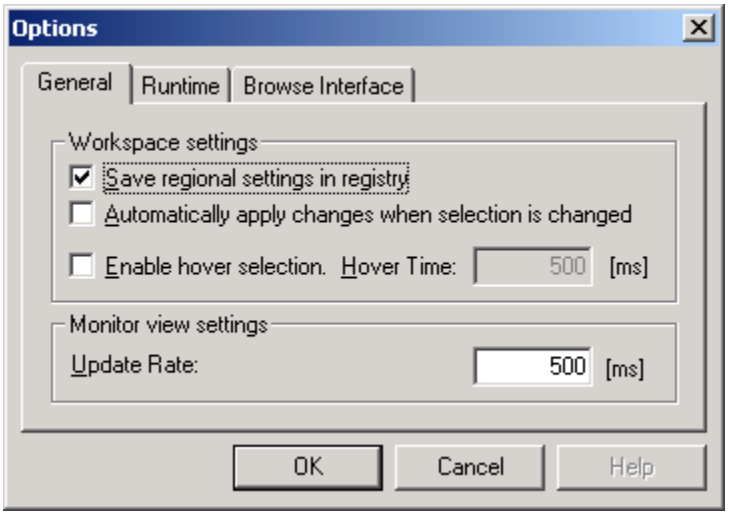


Figure 34: General Settings

- **Save regional settings in registry.** DataWorX32 is available in other languages as well such as Dutch, German etc. The GUI will typically appear in the language of your Windows regional settings, but one may also force to load the GUI in the language as specified in the registry.

- **Automatically apply changes** is most convenient for a user to edit configurations, otherwise the user would be prompted to confirm any change made.

- **Enable hover selection** is the effect that when the mouse points for a certain time at a leave in the configuration tree, the item is selected without having to click the item.

- **Monitor update rate** is the refresh rate of the monitor window when **View, Monitor View** is selected.

Browse Interface

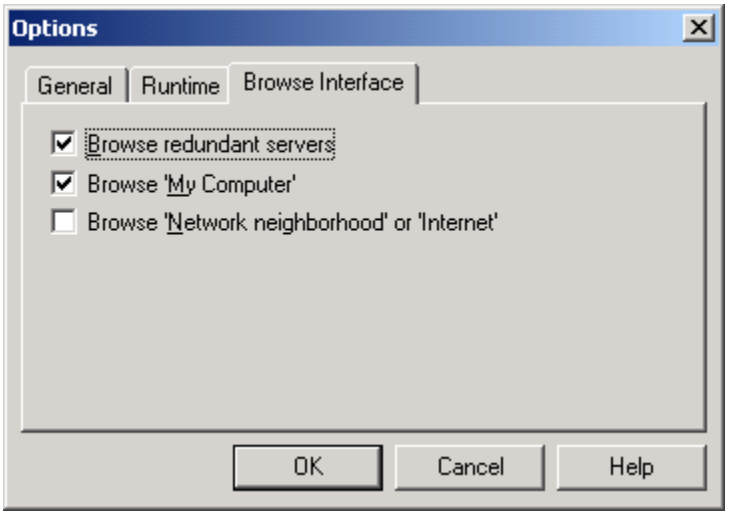


Figure 35: Browse Interface

The Browse interface settings have an effect on OPC Clients which may want to browse **through** DataWorX32.

- **Browse 'My Computer'** allows OPC Client to browse OPC Servers that are installed on the DataWorX32 computer.

The browsed tag could look like
ICONICS.DataWorX32.8\ICONICS.Simulator.1\SimulatePLC.Sine

- **Browse 'Network neighborhood or Internet'** allows the OPC Client to browse through DataWorX32 to OPC Servers which run on network nodes accessible through a GenBroker configuration "OPC over TCP/IP", "XML" or DCOM. Configure GenBroker on the DataWorX32 node before attempting this.

An example of a browsed tag could be
ICONICS.DataWorX32.8\\\PC123\ICONICS.Simulator.1\SimulatePLC.Sine

- **Browse redundant servers** allows the OPC Client to browse through a Redundancy Alias that is configured in DataWorX. The OPC Client will then see all tags that are exposed by the redundant OPC Servers.

An example of a browsed tag could be
ICONICS.DataWorX32.8\[[REDUNDANCY]]\Tank.Level

Runtime tab

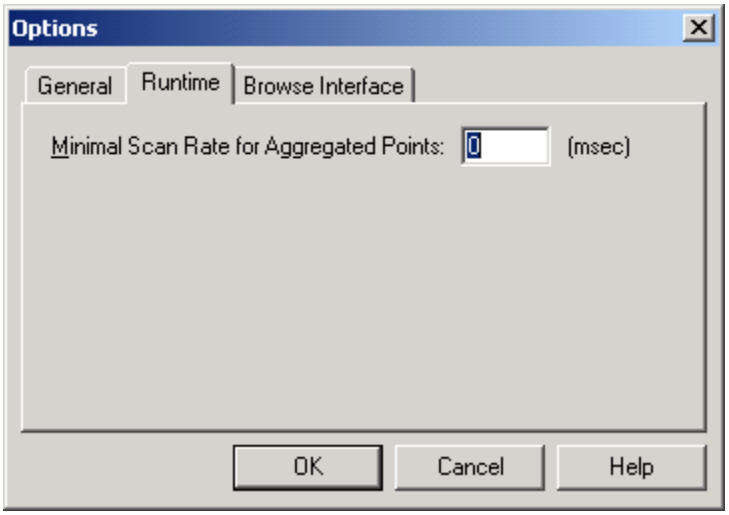


Figure 36: Runtime

The **minimal scan rate for aggregated points** is useful when you would like to limit the scan rate for subscribed OPC clients in order to reduce the load on the OPC Server.

6 Runtime Configuration

After configuring the DataWorX32 project, runtime can be started by clicking the traffic light. A green light indicates the Runtime state.

Figure 37: DataWorX32 Toolbar

In a real project, there's no need to click traffic lights. The DataWorX32 server acts like any OPC Server and starts runtime automatically as soon as OPC clients connect. However, it is often desired that DataWorX32 is already in runtime before clients connect. **GENESIS32 Tray** (GenTray) could be configured to start DataWorX32 automatically for instance 10 seconds after the computer boots. GenTray is the purple system tray icon which shows in the bottom right corner of the Windows task bar.

Figure 38: GenTray

To configure GenTray, complete the next steps:

1. Click **Start, Programs, ICONICS Tools, GENESIS32 Tray** to start the application.
2. Click the GenTray icon which shows in the Windows system tray.
3. Select **Configure** and the configuration dialog is shown. See figure 2.
4. Select for instance **DataWorX32** from the drop-down list.
5. Select **Include in AutoStart List**
6. Select **Include in AutoStop List** and click **OK**.

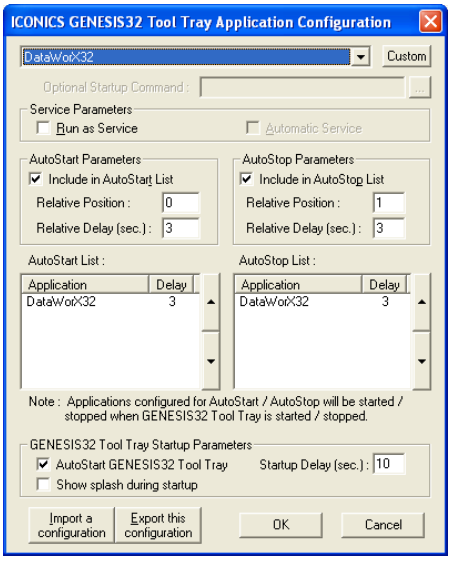


Figure 39: GenTray Configuration

While DataWorX32 is in Runtime, registers may be added or other online changes may be made. Changes are accepted after clicking the “Update Runtime” icon:

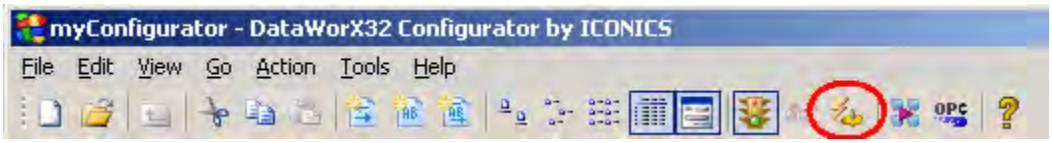


Figure 40: DataWorX32 Toolbar

DataWorX32 can also be registered as a service. The purpose of this is mainly that DataWorX32 runs even if there is no user logged in on the computer. Configuring DataWorX32 to run as a service can be done either through the **GenTray** user interface as shown above, or manually from the command prompt **“c:\Program Files\ICONICS\GENESIS-32\Bin\ DwxRuntime.exe” /service**

If DataWorX32 is configured to run as a service, you must configure DCOMCnfg.exe to verify the DCOM settings of ICONICS DataWorX32 and the OPC Servers accessed by DataWorX32. The Identity settings of the servers should be set to a user name and password of a user which has access rights to these servers.

For further information about DCOM setting refer to an appropriate ICONICS application note.

Founded in 1986, ICONICS is an award-winning independent software developer offering real-time visualization, HMI/SCADA, energy, fault detection, manufacturing intelligence, MES and a suite of analytics solutions for operational excellence. ICONICS solutions are installed in 70% of the Fortune 500 companies around the world, helping customers to be more profitable, agile and efficient, to improve quality and be more sustainable.

ICONICS is leading the way in cloud-based solutions with its HMI/SCADA, analytics, mobile and data historian to help its customers embrace the Internet of Things (IoT). ICONICS products are used in manufacturing, building automation, oil & gas, renewable energy, utilities, water/wastewater, pharmaceuticals, automotive and many other industries. ICONICS' advanced visualization, productivity, and sustainability solutions are built on its flagship products: GENESIS64™ HMI/SCADA, Hyper Historian™ plant historian, AnalytiX® solution suite and MobileHMI™ mobile apps. Delivering information anytime, anywhere, ICONICS' solutions scale from the smallest standalone embedded projects to the largest enterprise applications.

ICONICS promotes an international culture of innovation, creativity and excellence in product design, development, technical support, training, sales and consulting services for end users, systems integrators, OEMs and Channel Partners. ICONICS has over 300,000 applications installed in multiple industries worldwide.

World Headquarters
100 Foxborough Blvd.
Foxborough, MA, USA, 02035
Tel: 508 543 8600
Email: us@iconics.com
Web: www.iconics.com

European Headquarters
Netherlands
Tel: 31 252 228 588
Email: holland@iconics.com

Czech Republic
Tel: 420 377 183 420
Email: czech@iconics.com

France
Tel: 33 4 50 19 11 80
Email: france@iconics.com

China
Tel: 86 10 8494 2570
Email: china@iconics.com

Italy
Tel: 39 010 46 0626
Email: italy@iconics.com

UK
Tel: 44 1384 246 700
Email: uk@iconics.com

India
Tel: 91 22 67291029
Email: india@iconics.com

Germany
Tel: 49 2241 16 508 0
Email: germany@iconics.com

Australia
Tel: 61 2 9605 1333
Email: australia@iconics.com

Middle East
Tel: 966 540 881 264
Email: middleeast@iconics.com

Microsoft Partner
Gold Application Development

Microsoft Partner
2014 Partner of the Year Winner
Public Sector: CityNext

64™ | Windows 8 Compatible | Windows10 Compatible | Windows Server 2012 Certified | BTL | CERTIFIED OPC | vmware READY

VISUALIZE | ANALYZE | MOBILIZE | CLOUD

www.iconics.com