GraphWorX32 - Local Aliasing and Tag Indirection

APPLICATIONS NOTE

August 2004

Description: Information on GraphWorX32 local aliasing and tag indirection.
OS Requirement: Windows XP/2000/NT/95/98/Me.
General Requirement:

Aliasing is the ability to define a string (usually a short name) to represent another string (usually part, or all of a tag name). This feature is very powerful and will significantly reduce development time and increase reusability.

For example, on the factory floor there are several actuators all of the same type with similar I/O. If a detail window were to be created for each actuator, the system integrator would have to create one display per actuator, resulting in many files that essentially contain the same information, but have different OPC Tags. GraphWorX32 will allow you to re-use one display and associate it with different OPC Tags using Aliasing. This feature will reduce the cost of maintenance on your application; there will be fewer files to maintain and when changes have to be made, they'll only have to be made in one display!

The "**Edit Local Aliases**" command can be found on the Dynamics Menu in GraphWorX32. When including a local alias name in a data connection, the alias name should be surrounded by the special tokens "<<" and ">>". These tokens are used to identify the beginning and end of the alias name.

For this applications note we will use the following OPC Tags as an example:

- ICONICS.Simulator\SimulatePLC.Ramp
- ICONICS.Simulator\SimulatePLC.Sine
- ICONICS.Simulator\SimulatePLC.Random

Set Aliases Directly:

OPC tags are often very similar in syntax, and only differ in one or two places or numbers. It can be very redundant work to create multiple process points that are so similar on a display. If we could instead create just one process point and have it switch what tag it displays with the push of a button, this would declutter our display and simplify configuration. This is where GraphWorX32 local aliasing and tag indirection is very useful.

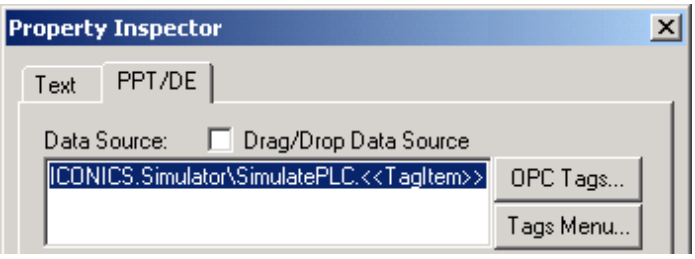

Fig. 1: Example GraphWorX32 Aliased Process Point

In the example in Figure 1, the alias name is TagItem. In order to resolve the alias:

1. Create a process point with name: **<<TagItem>>**
2. Create a push button and assign it to "**Set Aliases**".
3. Click on the "**Set Aliases**" button.
4. This will bring up the Local Aliases Configuration.
5. If you do not remember the name of the alias, click on the "**Aliases…**" button in the dialog, and a list of aliases in the display will be shown.
6. Select an alias name and hit OK.
7. Specify "Ramp" in the Alias Definition field.
8. Click on "**Add**" to create the alias.

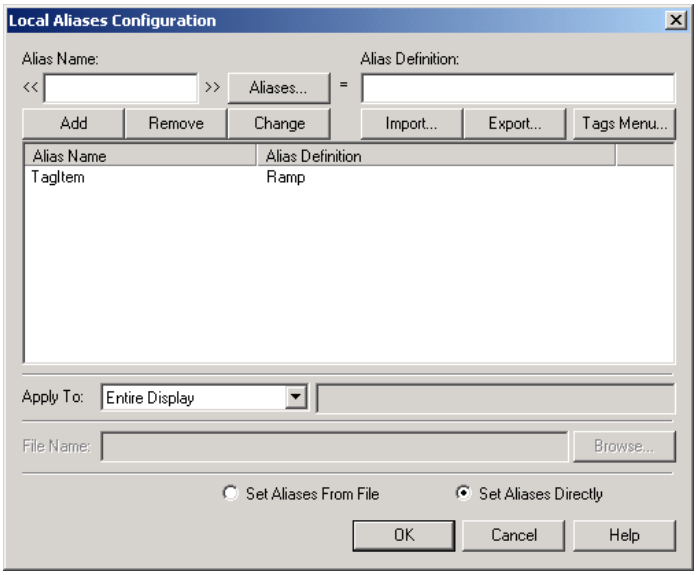

Fig. 2: Local Aliases Configuration Window

Place a second button on the display and repeat the same steps but now for the "Sine" tag. And repeat this once more for the "Random" tag. Go into runtime and press the buttons one by one and you will notice the values changing accordingly!

Aliases can be used in any combination for an OPC Tag. For example:
<<Vendor>>.<<OPCServer>>\<<Device>>.<<TagItem>>
In this example all four aliases need to be resolved before the process point will show any data. GraphWorX32 can assign many aliases to a single object. You could set the aliases for the example above to be:

<<Vendor>>	= ICONICS
<<OPCServer>>	= Simulator
<<Device>>	= SimulatePLC
<<TagItem>>	= Ramp

One alias can also contain the entire OPC Connection String. For example:
<<Tag>> = ICONICS.Simulator\SimulatePLC.Ramp

Set Aliases through Alias Files:

To even further reduce the amount of engineering time, GraphWorX32 supports aliasing with the use of alias files. These files can be used for different displays. Instead of configuring the aliases in each display, an alias file can be specified and will automatically resolve the aliases for that display.

- Using the GraphWorX32 Alias File Editor:
  1. Go to "Tools", "Local Alias File Editor" and specify the alias involved.
  2. Export the Alias information to a standard text file.
  3. Using a text editor like Notepad, the file needs to comply with the following syntax:

     aliasName<TAB>aliasDefinition<CR>

  4. Specify the aliasName then press the Tab key, followed by the aliasDefinition, and then the enter key. Repeat this for all aliases to be included in the alias file.

  Here is an example alias file for the 4-alias example above (Note that there is exactly one tab between the aliasName and the aliasDefinition, which is why the columns may or may not line up):

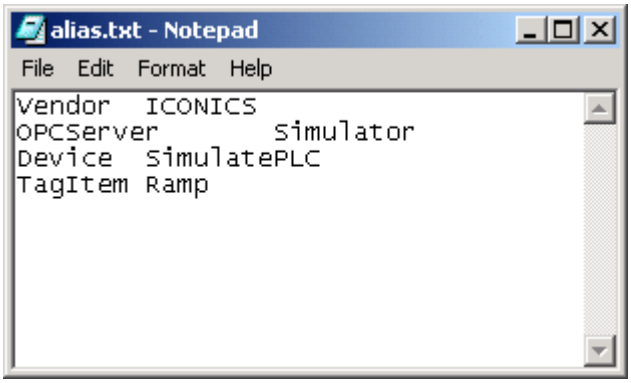

Fig. 3: Example Alias File

Local Aliasing through VBA:

GraphWorX32 also has a Visual Basic automation interface that supports aliasing. Use the following automation calls in your GraphWorX32 display to create alias files.

- Function SetAliasDefinition (aliasName As String, newDefinition As String) As Boolean

  Sets the alias definition of the specified alias name for all dynamic objects in the display.

- Function SetAliases (commandString As String) as Boolean

  Sets multiple alias definitions for the entire display. The commandString parameter has the following format: aliasName1 Chr(9) aliasDef1 Chr(13) Chr(10) aliasName2 Chr(9) aliasDef2 Chr(13) Chr(10) … etc.

- Function SetAliasesFromFile (filename As String) As Boolean

  Sets multiple alias definitions for the entire display. Using the same file as format as specified above.