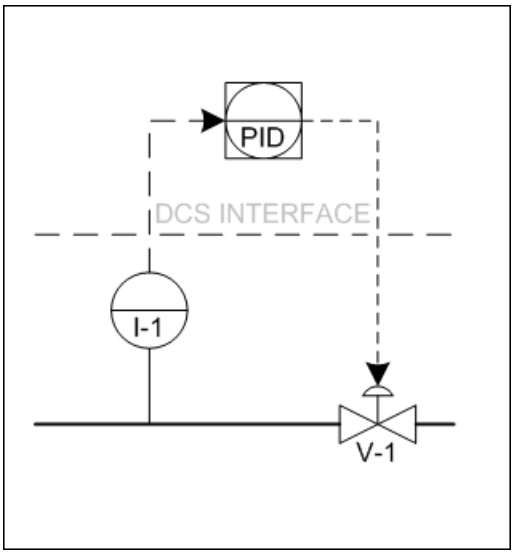
KONGSBERG

K-Spice DCS Link: OpcDaCom

User Guide

Release 1



1/A

Document history

Document 34148/H		
Rev. A	June 2014	Created document. Arbortext version of various Wiki pages.

The reader

This user guide is intended as a reference manual for the K-Spice® user. The manual is based on the assumption that the user is familiar with process modelling, as well as oil and gas production systems.

Note

Comments

To assist us in making improvements to the product and to this manual, we welcome comments and constructive criticism.

e-mail: kogt.documentation@kongsberg.com

For more information on K-Spice®, visit our website: www.kongsberg.com/K-Spice

Kongsberg Oil & Gas Technologies AS
www.kongsberg.com/kogt

Table of contents

1 Introduction

This is a user guide for the Distributed Control System (DCS) communications link between K-Spice® models and any supported DCS. The link is a separate program that connects to the K-Spice® *Simulation Manager* and handles all I/O exchanges between active timelines and the remote DCS.

Some of the links relates to an API specified by the DCS vendors. Other links are based on the OPC standard. The OPC links communicate with the DCS vendor machine following the OPC protocol. The DCS vendor machine is then configured as an OPC server.

Most of the images in this guide are from a test model, with a test link configuration. Many of the images are screen shots from real interfacing projects. Effort is made to disguise the data project's data structure and the real project's tag names. Some of these images are bit manipulated, characters are moved around. Some the tag names in the images are shortened down versions of the real names. Typically a project tag PI-430–754.PV is presented in the image as PI-754.PV, or an internet address 11.45.23.97 is presented in the image as 11.11.11.97. If there are errors in the images, or inconsistency between images, the error could be the result of unfortunate image manipulation.

1.1 Control systems and K-Spice DCS links

The companies that deliver control systems have proprietary DCS servers that K-Spice can connect to.

Table 1 Existing links

K-Spice link	DCS vendor	Control system	Link type
AbbItsCom	ABB	800xA	Vendor API for commands in combination with OPC for data.
AimInterfaceCom AimWinPsStarter	Kongsberg Maritime	AIM 2000	
AutoswitchCom			link-to-link
CccOpcCom	Compressor Controls Corporation		OPC
DeltaVCom	Emerson Process Management	DeltaV	OPC
ExatifCom	Yokogawa	Centum CS	Vendor API
HimaCom	HIMA	HIMA X-OTS Simulator	OPC
HimaElopIICom	HIMA	HIMA X-OTS Simulator	Vendor API for commands in combination with OPC for data.
OpcDaCom	Data source: Standard OPC		OPC DA
OpcHdaCom	Data source: Standard OPC		OPC HDA
ProSimCom	Oy Endat Ltd	Prosim	Vendor API
SimitCom	Siemens	Simatic PCS 7	
No installer for Sim4MeCom	Invensys	Foxboro Evo	Vendor API for commands and data.
SoftLogixCom	Allen-Bradley	ControlLogix	
TelepermMCom	Siemens	TelepermM OPC	Vendor API for commands in combination with OPC for data.

Table 1 Existing links (cont'd.)

VirtuosoCom	Virtuoso Automation Solutions		Vendor API
WmiCom	Data source: Windows Management Instrumentation	none	

1.2 Acronyms and concepts

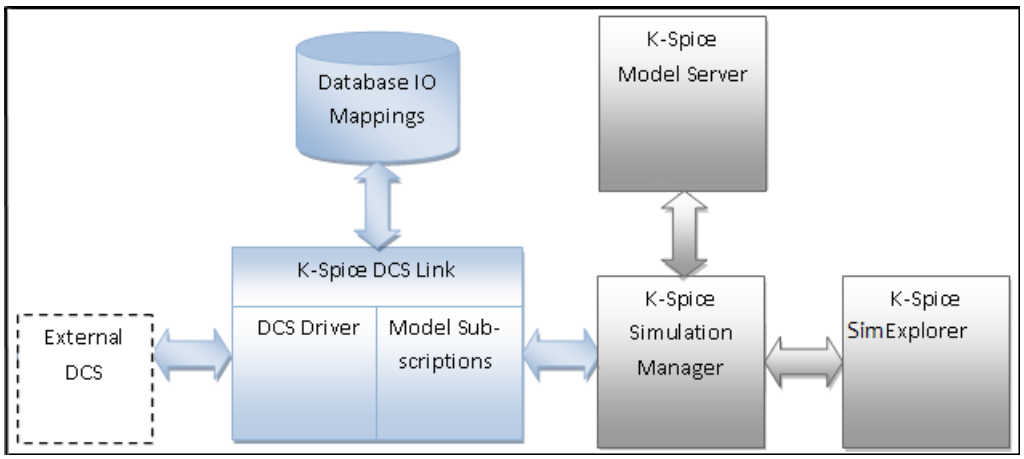
The basic concepts OPC, DCS and SCADA have a short explanation in the chapter Tips and Trouble Shooting. See *Basic Terminology* on page 157

Acronym	Full text	Description
Aspect Server	none	Name of the ABB ITS data servers
Bucket Brigade.Int4	none	Matrikon. Default tagname in the Matrikon Test tool.
COM	Component Object Model	Standard interactive object creation and inter process communication
DCOM	Distributed Component Object Model	Microsoft. Communication among applications distributed across networked computers.
DCS	Distributed Control System	Standard
EWS	Engineering Working Station	Yokogawa
FCS	Field Control System	Yokogawa
ITS	Industrial Training Simulator	ABB ITS is the name of the communication package used in interfaces to the ABB control system.
OLE	Object Linking and Embedding	Microsoft
OPC	OLE for Process Control	Standard
PI Server	none	OSIsoft. Storage of any form of real-time data.
PLC	Programmable Logic Controllers	Standard
Proplus	none	Name of the DeltaV system data servers
SCADA	Supervisory Control And Data Acquisition	Standard
WMI	Windows Management System	Microsoft. Management of data and operations on Windows-based operating systems.

1.3 Structure of a DCS communication system

The DCS Link connects to K-Spice® via the *Simulation Manager*, as shown in the figure below. There are different versions of a DCS Link depending on the DCS to be connected to, such as ABB, Emerson or Yokogawa. However, the interface to K-Spice is still the same. The K-Spice® DCS Link are connected to a Microsoft Access database that contains I/O mappings, also called the Cross Reference List, and DCS driver connection information.

Figure 1 K-Spice DCS communication system



The DCS communication link consists of two layers. The first layer is model variable subscriptions, which is independent of the DCS system and connected to *Simulation Manager*. The second layer is a communications driver which is to specific to the DCS system.

1.4 DCS driver functionality

The DCS driver for a particular DCS is a functional block with a defined interface towards the model variable subscriptions and internal logic implementing the system specific communication tasks.

The tasks solved by a driver are:

1 Transfer values between the external system and the model. Tag names used to identify a value in the external system are found as parameters in the entry for a variable in the IO list.

2 If applicable, convert commands received from the model variable subscriptions and forward them to the external system. Once a command has finished, return the proper acknowledgement to the model variable subscription.

The appropriate driver for a defined external DCS system is added through the Access database by selecting the appropriate DCS link for the control system.

Packing and unpacking variables

The DCS driver is also responsible for packing and unpacking data in cases where multiple, individual model variables are sharing a common DCS variable (for example, a mask of value limit switches.)

This HimaElopIICom is a link for HIMA H41/H51-q Systems programed with ELOP II. The interfacing is done through a Vizualization Gateway application from Kirchner Soft and logi.cals GmbH. The HIMA OPC server is used for data communication and Visualization Gateway with standard Windows COM technology, is used for commands.

The link must run on the same PC as Visualization Gateway.

2 Setting up a SimLink application

How to install the link on the PC

If you have an older version of K-Spice installed, uninstall K-Spice, and uninstall the old link.

Go to the build machine, and find the actual version. Double click on `KSpice.exe`. Then double click on `KSpiceExatifCom.exe`.

Often a new version of K-Spice and the link must be installed on a PC that cannot access the build computer directly. The installation program must copied to a memory stick. For i.e. the ExatifCom link, bring with you the following files and directories from the build machine:

- KSpice.exe
- KSpiceExatifCom.exe
- directory: **packages**
- directory: **redist**

If you do not have a K-Spice license installed, you must also bring with you:

- KOGTLicenseUtility.exe

2.1 The creation of a SimLink application

This is a general description for link installation and the initial link configuration. To make the description easier to read, it is necessary to select a name as the example for all the links. The **ExatifCom** link and Yokogawa's **Engineering Working Station** are used as examples in this chapter. The steps are more or less the same for all the links.

The Model PC and the EWS machine.

The normal configuration will be that all the K-Spice executables and the K-Spice model are installed on one machine, called the Model PC. The DCS vendor will normally set up a data server on a different machine. For a Yokogawa system, the machine with the data server is called a **Engineering Working Station (EWS)**. For the ABB ITS system,

the machine with the data server is called the **Aspect Server.** For DeltaV systems the data servers, normally one central server with multiple satellite servers, are running on a **Proplus** machine.

The link running on Model PC or on the EWS machine

There are two possibilities here. If *the link* is installed *on the Model PC*, the communication between K-Spice SimManager and the K-Spice SimLink is simple and straight forward. With this setup the SimLink must have read and write access to a DCS data server on a remote machine. There are fire wall issues involved. For the OPC links the main issue is to open up for DCOM access on the remote machine.

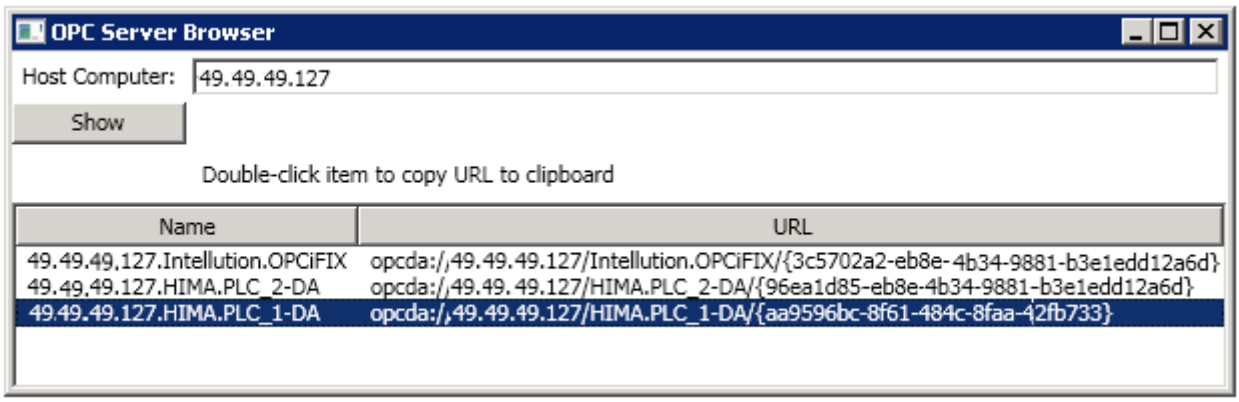
See the section *DCOM configuration for AbbItsCom* on page 112

The other possibility is that *the link* is installed on the same machine as *the DCS data server*. This simplifies the communication between the link and the data server, but complicates the communication between SimManager and the SimLink. The link has to be started by K-Spice SimManager on the remote machine. The application SimRemoteLauncher.exe, installed on the remote machine, is used to start the link application. There are fire wall issues and DCOM access configuration for this solution also.

One link application or two link applications

If there is more than 1 node in the OPC server, the normal configuration is that you create one SimLink application for each node. To read data from the HIMA servers in the figure below, one link application must be created to read data from PLC_1–DA, one link application must be created to read data from PLC_2–DA.

Figure 2 The OPC Server Browser for a HimaCom link



To read data from a DeltaV system you will only create one link application. The link will interface to what is called the Proplus Server. The Proplus OPC Server will communicate with the other nodes in the DeltaV system.

Installer for the K-Spice link

To set up the **ExatifCom** link its necessary to install **ExatifCom.exe** in the directory `C:\Program Files (x86)\Kongsberg\K-Spice\bin`.

The different links are installed with different installation files, i.e `KSpiceExatifCom.exe`

Either the installer needs to be run on the K-Spice Model PC, or the installer needs to be run on the Yokogawa **Engineering Working Station** (EWS) machine where the Yokogawa **Field Control Station** (FCS) test functions are launched.

The option where the link runs on the K-Spice Model PC:

In the directory: `\\xray\03208\KSPICE\Installations\KSpiceCombined\Official Releases\2.8.2.0\Installers\`

Double click on `KSpiceExatifCom.exe`

The option where the link runs on the remote machine.

Copy the following directories and files to the EWS machine:

From: `\\xray\03208\KSPICE\Installations\KSpiceCombined\Official Releases\2.8.2.0\Installers\`

The directory: `packages`

The directory: `redist`

The file: `KSpiceExatifCom.exe`

The file: `SimRemoteLauncher.exe`

On the remote machine, double click on the two executables.

If a full K-Spice runs on a remote machine:

In addition to the two directories and the two files in the previous section, copy the following two files:

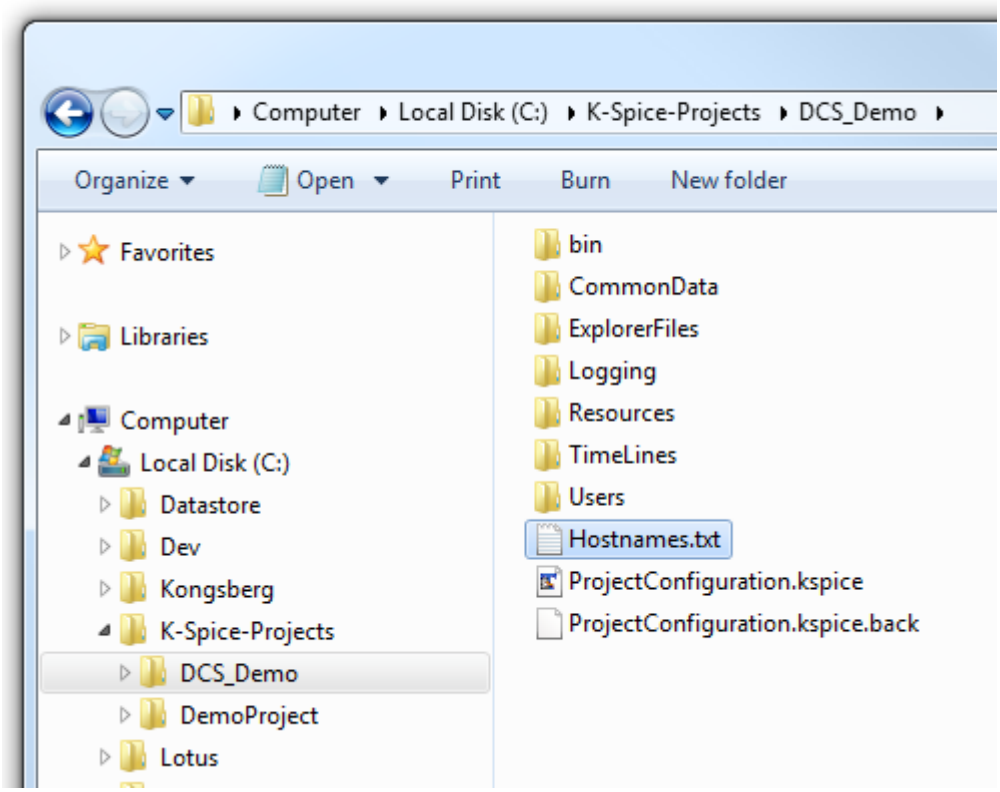
The file: `KOGTLicenseUtility.exe`

The file: `KSpice.exe`

2.2 The file Hostnames.txt

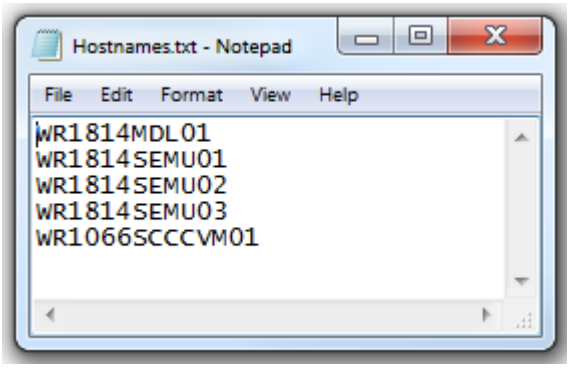
The setup of the ExatifCom link also needs a file **Hostnames.txt** to be placed inside the project folder. This file contains the names of the different EWS machines.

Figure 3 Hostnames.txt in the project folder



This text file consists of names of the remote machine where the different links are running. In this case it must contain the Yokogawa EWS machine name for e.g. **WR1814MDL01**.

Figure 4 Hostnames.txt

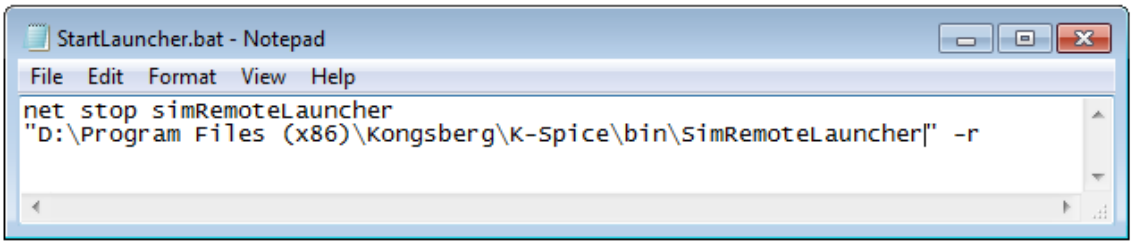


2.3 SimRemoteLauncher

If the link ExatifCom is installed on the EWS machine, the **SimRemoteLauncher** *must be launched* on the remote EWS machine running the FCS test functions *prior to* starting the ExatifCom link.

Create a start up batch file for the SimRemote launcher on the EWS machine. The batch file contains the following script.

Figure 5 Script to launch SimRemoteLauncher



Double click the batch file. This will launch the SimRemoteLauncher on the EWS machine.

2.4 Create a link application in the K-Spice project

To set up a link, a new link application must be created in the project folder, in parallel to the ProcessModel and other applications. When the link application exists, K-Spice SimExplorer will launch the link application, when the Timeline of the project is activated.

The Access database in the new .NET based links

A new Access database is created by the link application when you follow the steps below. When the configuration of the link application is finished, the Access database has all the values in it, that makes up the configuration of the link in the actual project. The database created in the steps below has all the necessary tables in it. Some of the keywords in the tables have default values when the database is first created.

Preparations before creating a link of the old type.

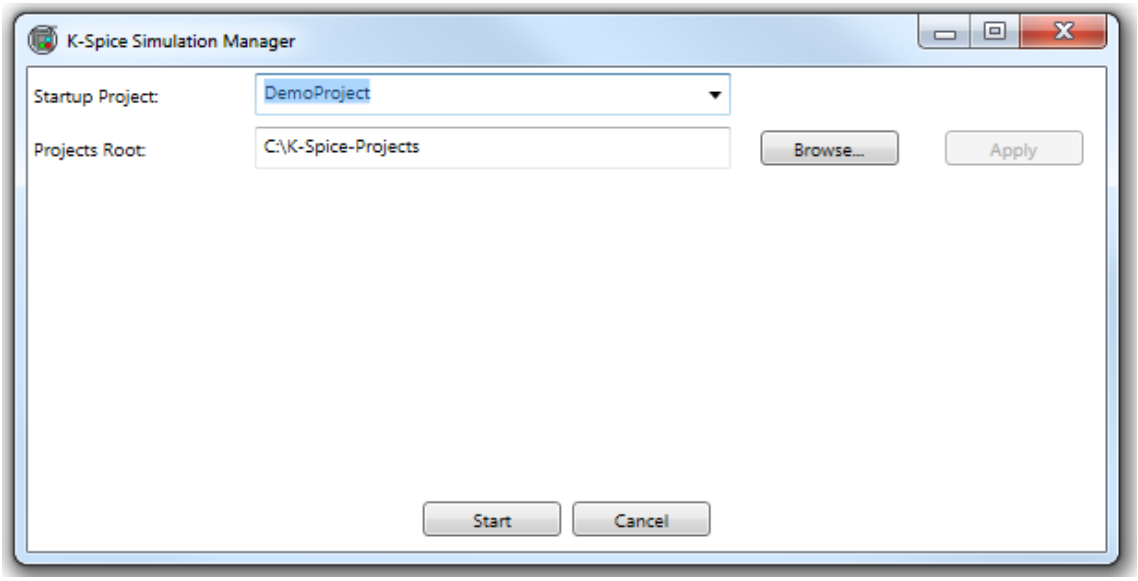
With the old C++ links, an Access database must exist in the project folder, before the link application is created in the K-Spice project. This is the case with the old version of the AbbItsCom link, 2013 and older. A database in the .mdb format, with a lot of configuration parameters in it, must be generated up front, based on databases from other projects. Create the folder below, and copy over the database file:

```
C:\K-Spice-Projects\DemoProject\TimeLines\Engineering\
Applications\AbbItsCom\AbbItsCom.mdb
```

Create a link application

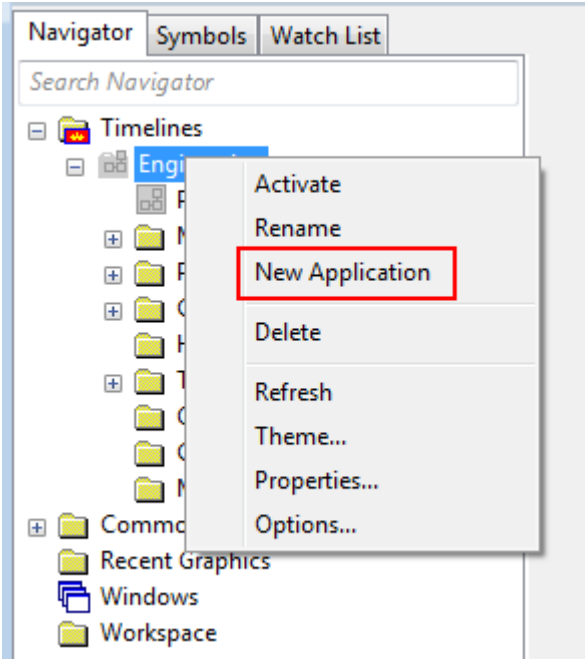
The link for the **Exatif** DCS system is used as an example. When you have finished, you have created a new application with the name **Exatif** in the project folder.

Figure 6 Start project



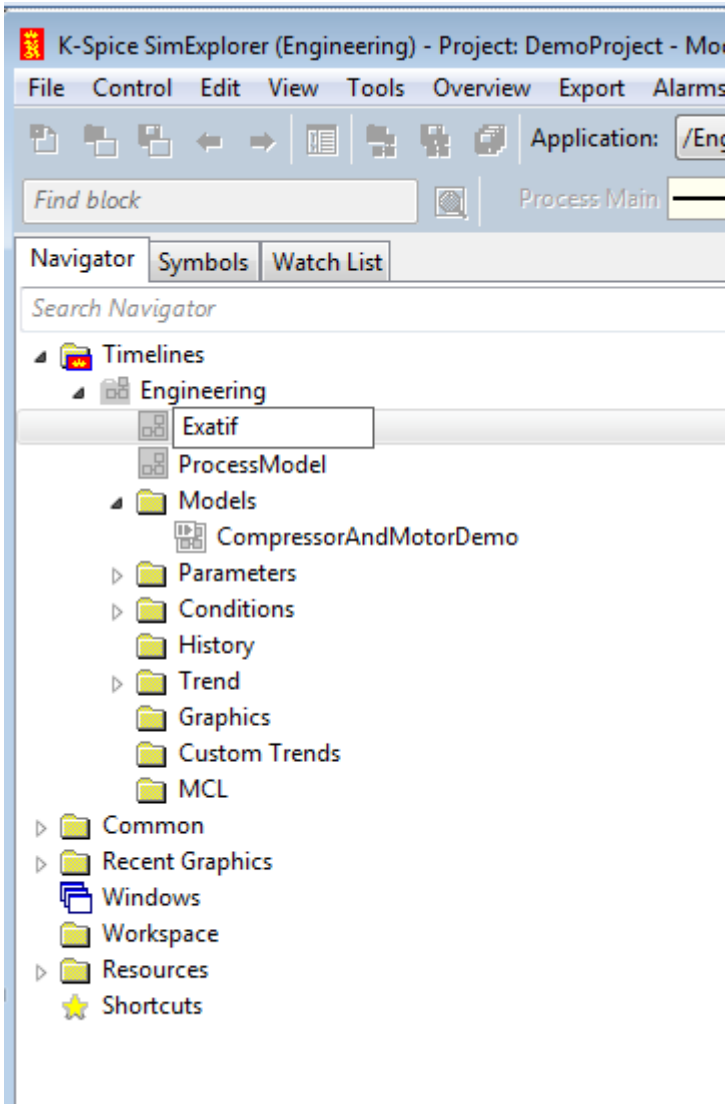
Click Start.

Figure 7 Set up the link as a new application



Open the SimExplorer. Right click on the Timeline **Engineering** and select →**New Application**

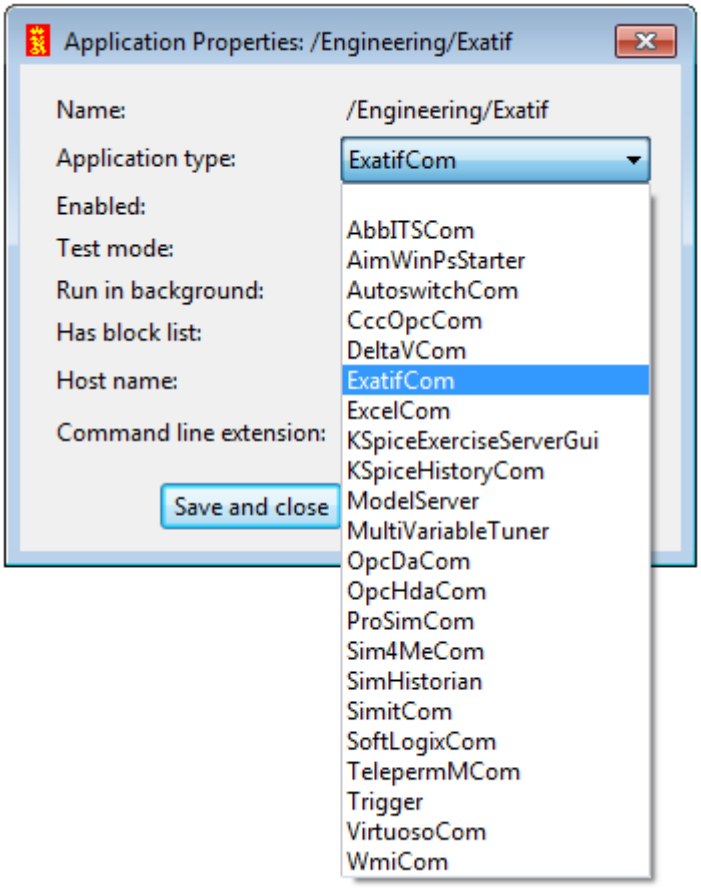
Figure 8 Exatif in Navigator



Type the name of the new application, in this case **Exatif**

The dialog Application Properties will come up automatically when you create the application. Enter the following:

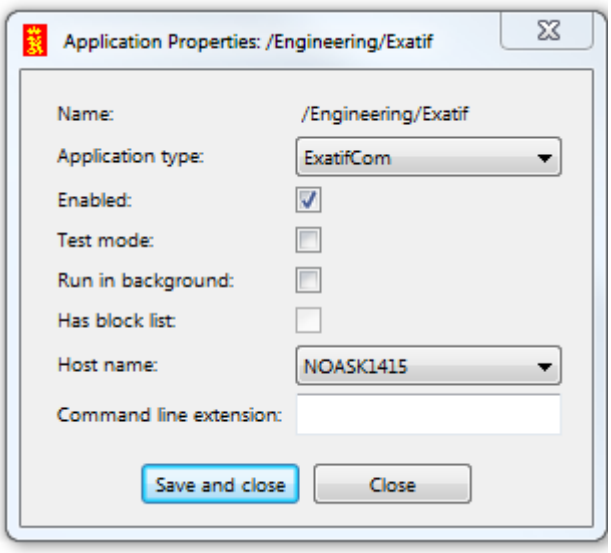
Figure 9 Application type



1 Select **ExatifCom** as the **Application Type** from the drop down list.

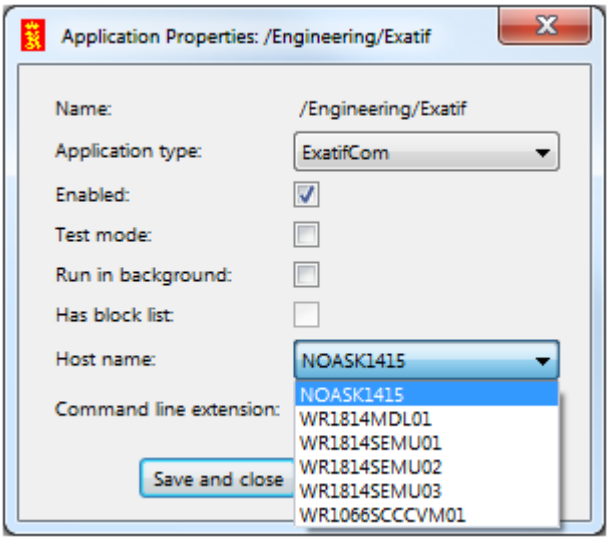
2 The default value is that **Enabled** is checked. Accept the default value. When **Enabled** is checked, the application will start automatically when the **Timeline** is activated

Figure 10 Enabled



3 Select **Host name** from the drop down list. This list is the machine names in the file **Hostnames.txt.**

Figure 11 Host name



4 Save the Application type and Host name configuration in the K-Spice project. In K-Spice SimExplorer:

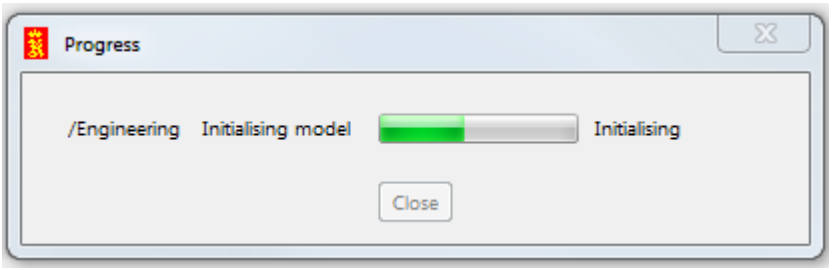
File →Save Project Properties

Now when the **Timeline** is activated, the K-Spice SimManager will automatically create the new folder **Exatif** in the existing folder `Timelines/Engineering/Applications`.

Activation of the timeline will also open the main window for the link, the window **K-Spice ExatifCom**, normally called the **SimLink GUI**

The first time you activate the timeline, the K-Spice dialog **Progress** will open with the text **Initializing model** and a running a progress bar.

Figure 12 The progress dialog opens when the timeline is activated



The dialog **Progress,** *with the running progress bar, will not disappear until you have created the link database, and reloaded the database, as described in the next section,* Creation of a new database.

Note

The initialization of the K-Spice model will not finish until you have a link database up and running, as described in the next section.

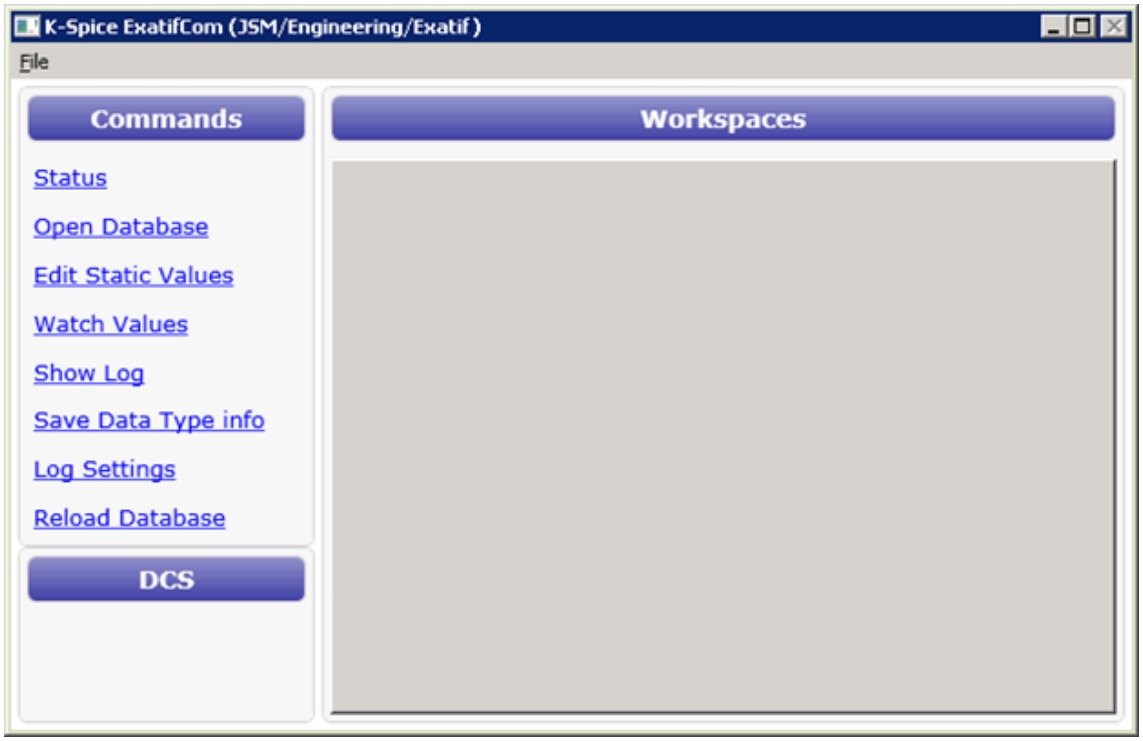
2.5 Creation of a new database

These are the steps for all .NET based links. The link **ExatifCom** is used as example.

Note

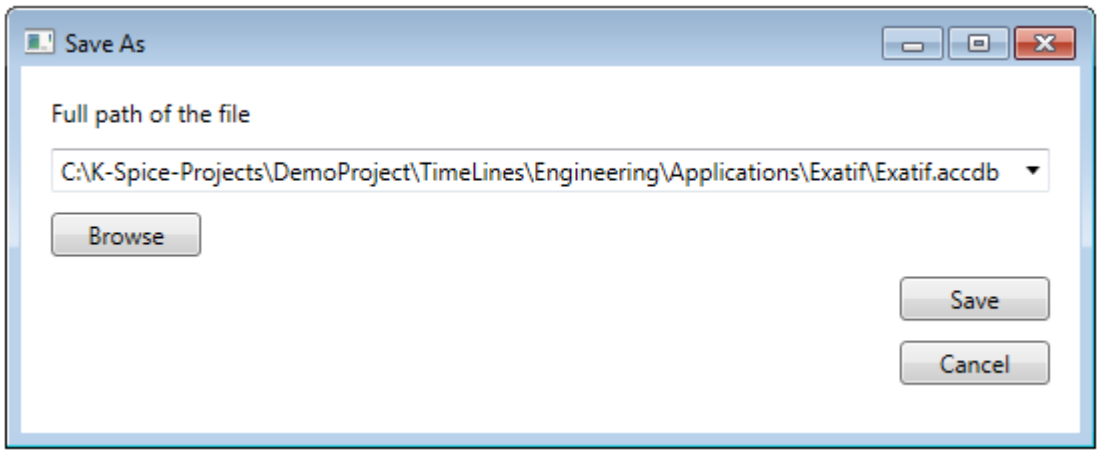
All SimLink applications require the 32-bit version of the Access Database engine and do not work with the 64-bit version. That also makes them incompatible with the 64 bit version of MS Office. Consequently, MS Office for 64-bit cannot be installed on the same machine as a SimLink.

Figure 13 K-Spice ExatifCom dialog, also called the SimLink GUI



To create a new database click on **File →Create empty database** in the SimLink GUI. The SimLink GUI is the dialog with the title **K-Spice ExatifCom**.

Figure 14 Save database as

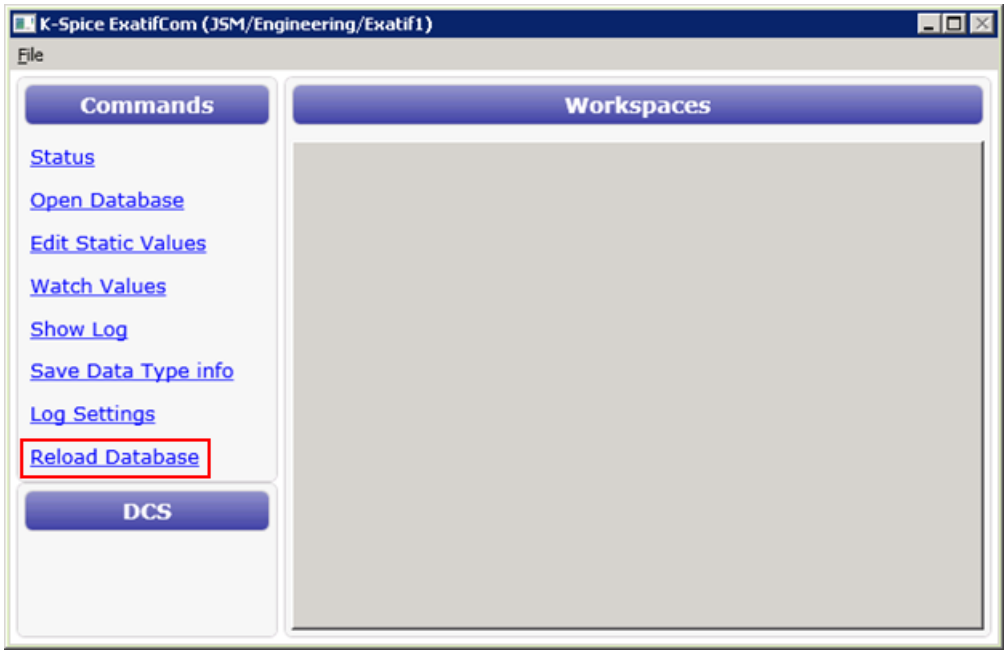


Reload database

After the database is created, it is necessary to load the values into the link. This is done with the button **Reload database**

When the database is reloaded, the activation of the K-Spice timeline is finished. The SimExplorer dialog **Progress** with the progress bar indicating that an initialization is going on, will be closed. The dialog closes with the text **Finished.**

Figure 15 Reload database



Directory structure with remote link installation for the Standard links

If a link is running on a remote (DCS vendor's) server, the cross reference database will reside in the link's working directory on that server. Any server running a K-Spice link must have a copy of the project's folder structure installed. The folder structure must have a path identical to the one found on the Model PC. I.e. if C:\K-Spice-Projects\"My project" is used on the Model PC, the directory C:\K-Spice-Projects\"My project" must be found on the remote server as well. The project folder on the remote server may be a file share, exposing the project folder on the Model PC.

2.6 The command line option **—database**

-database <name>

The ***default*** *database folder* is the application folder in the project timeline. The default database name is the name of the application created for the SimLink. For example, a project might configure an ExatifCom link with the name Exatif. The default database name will be Exatif.accdb

We recommend that you use the default folder and default name for the database. However, a command line option is provided to override the defaults.

The command line option `-database <name>` instructs the SimLink to open that database.

Example project specific database:

```
-database "C:\Configuration\Virtuoso.accdb"
```

The K-Spice project can be configured from SimExplorer to add the command line option for the SimLink.

3 The SimLink GUI

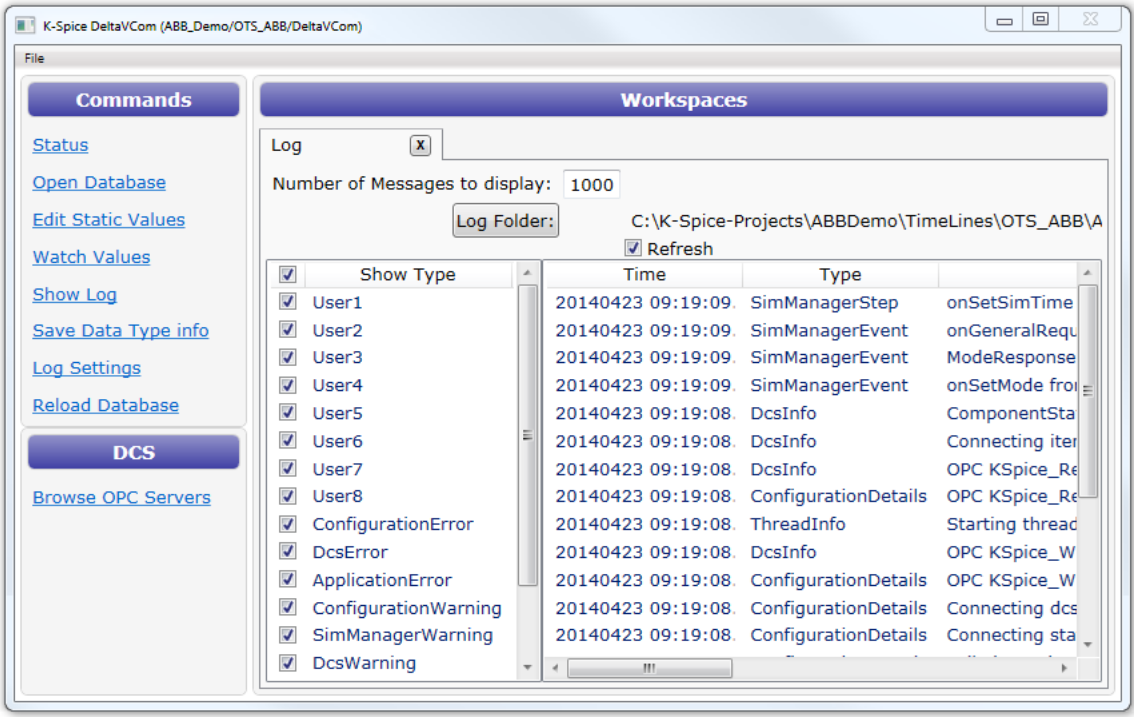
The .NET link dialog for configuration and monitoring and troubleshooting, is called the SimLink GUI. The dialog opens up when the timeline with the K-Spice link is activated. The title of the dialog is K-Spice ExatifCom for the ExatifCom link.

The SimLink GUI is used for four important tasks:

- A new Access database is created via the GUI the first time the timeline with a link application is activated.

  See *Creation of a new database* on page 25
- The Access database that is used for link configuration, is normally opened from the SimLink GUI.
- The feature The OPC Server Browser in the SimLink GUI will find the correct url adresses of the existing OPC servers.
- The log window in the SimLink GUI is essential when it comes to monitoring and troubleshooting the interface.

Figure 16 The SimLink GUI with workspace Log



3.1 Browse OPC Servers.

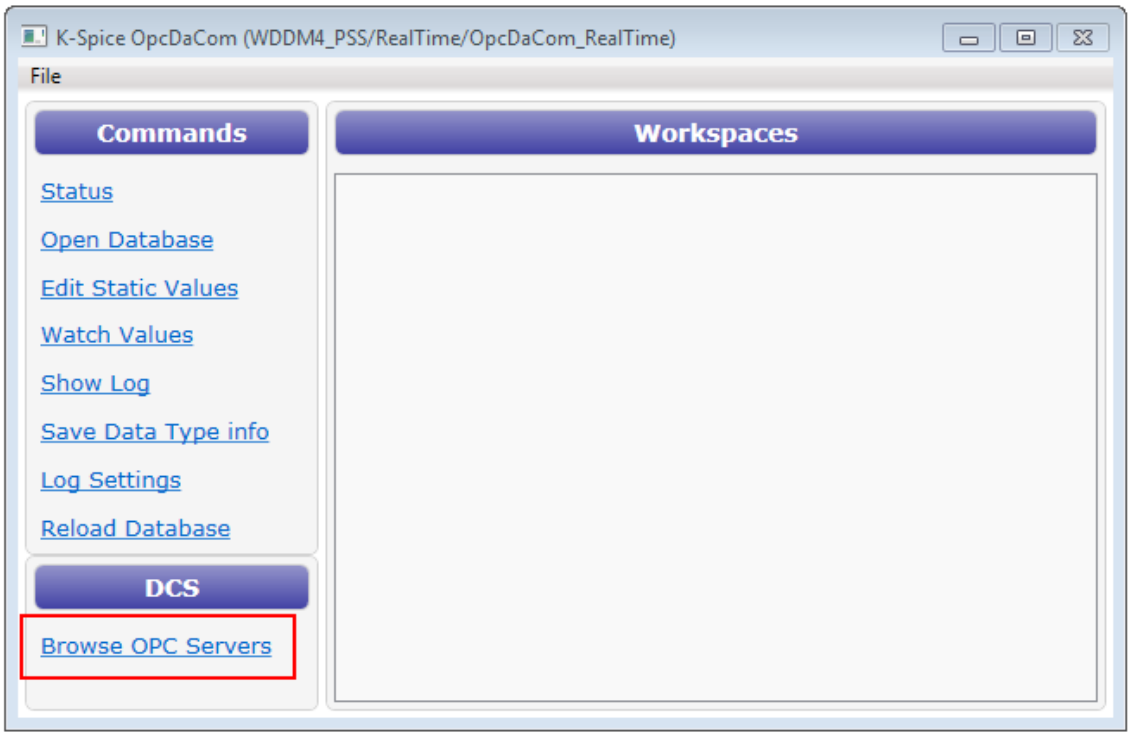
An important part of the OPC link configuration is to get the name and address of the OPC server correct in the **Config** table in the Access database. The actual keyword in the **Config** table is called **OpcServer**.

Different applications will show you the OPC servers and the corresponding url's on the local net. You will find them in **MatrikonOPC Explorer** and in **Softing's SOClient.** The best tool for getting the value of the keyword OpcServer correct, is the **SimLink GUI,** and the button **Browse OPC Servers.** See the detailed steps below.

The OpcServer connection URL

To find the URL, click→ **Browse OPC Servers** in the SimLink main window.

Figure 17 The SimLink main window



When the panel opens, click **Show**

If the OPC server is on another machine, enter the IP address of the other machine in the field **Host Computer**.

Figure 18 The OPC Server Browser dialog

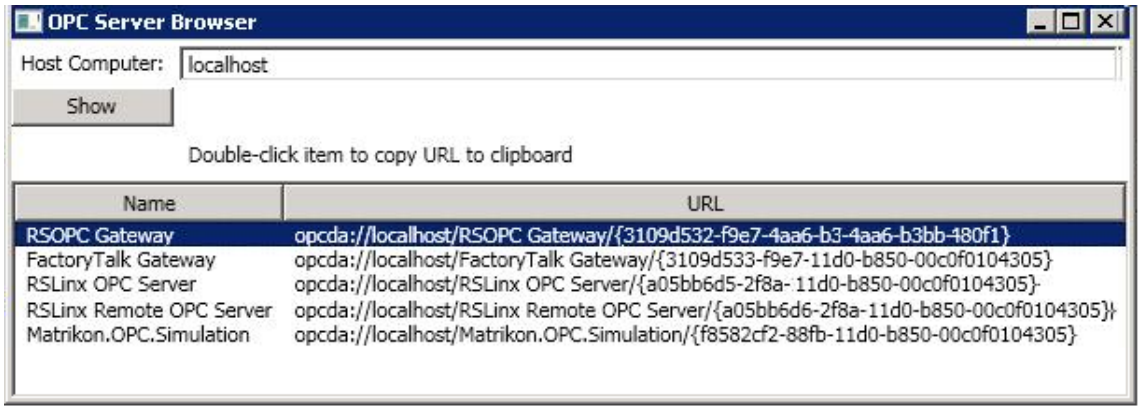
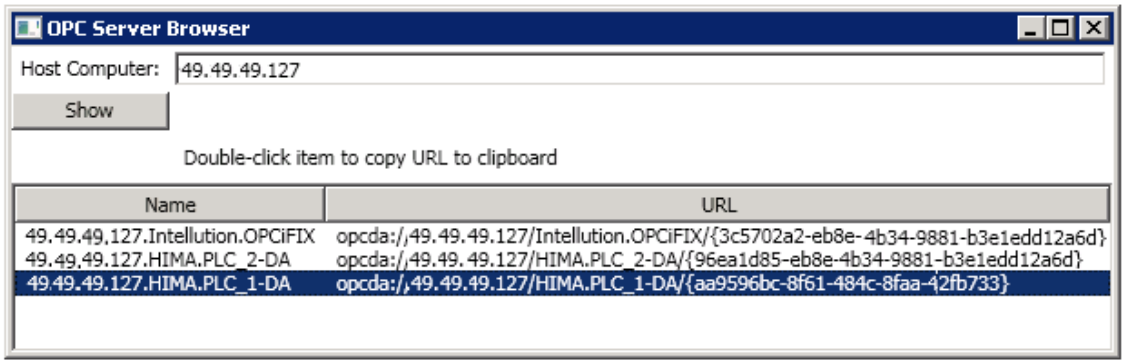


Figure 19 The OPC Server Browser for a HimaCom link



You can copy the OPC server connection URL from here. The URL shall be pasted in the **Value** cell of the of the **OpcServer** keyword in **Config** table. To copy the URL to clipboard, you must double-click the line!

If you double click on the line and paste the url directly from the clipboard, as described above, you are safe. If you for some reason paste the url to an editor and do a ctrl-C in the editor, and a ctrl-V in the Access database, you must look out for extra ctrl-LF (linefeed) in the Access database string. See *DcsError* on page 46

Note

*Use the URL as the SimLink GUI presents it. If you, in an OpcDaCom link GUI, ask for the URL for an OPC Server, the first 8 characters of the string is **opcda://** If you, in an OpcHdaCom link GUI, ask for the URL on the same machine, the URL will start with the 9 characters **opchda://***

Example of OpcServer value

```
opcda://localhost/Matrikon.OPC.Simulation
/{f8582cf2-88fb-11d0-b850-00c0f0104305}
```

The url of Matrikon OPC Server presented above is the real and correct url. Please notice that many screen shots in this guide are manipulated. The policy has been to disguise the real url's and the real names and structures of the DCS databases. Data items from a K-Spice demo project are normally untouched. Names of data items from real projects should all be manipulated. The DCS tagnames are manipulated, and the corresponding K-Spice data item names are manipulated.

3.2 Syntax in SimLink log files

An empty ><

The bigger than symbol and the smaller than symbol is used in log messages to demonstrate clearly where a text starts and where the text ends. If a link operation fails, the operation will often produce an error text. The error text will be presented as a part of a log message, encapsulated in > and <. If the operation succeeds, no error text is produced. If the operation succeeds, the log message will have an empty text in it, the two symbols ><.

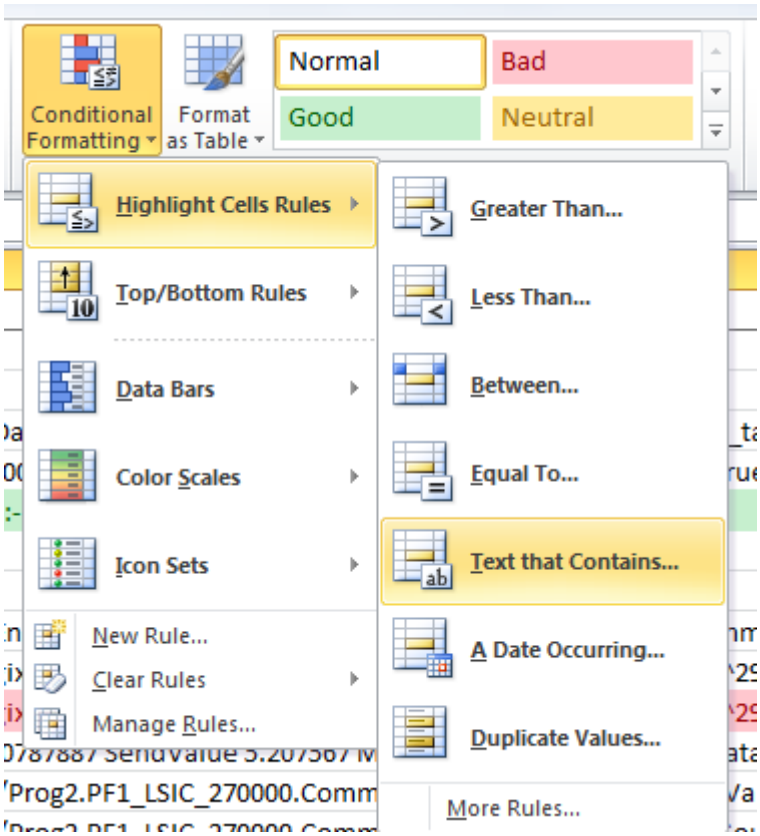
3.3 Using Excel conditional formatting on SimLink log files

Conditional formatting in Excel can be a powerful tool for analyzing the SimLink log file.

The link used as an example in this section, is an OPC based link

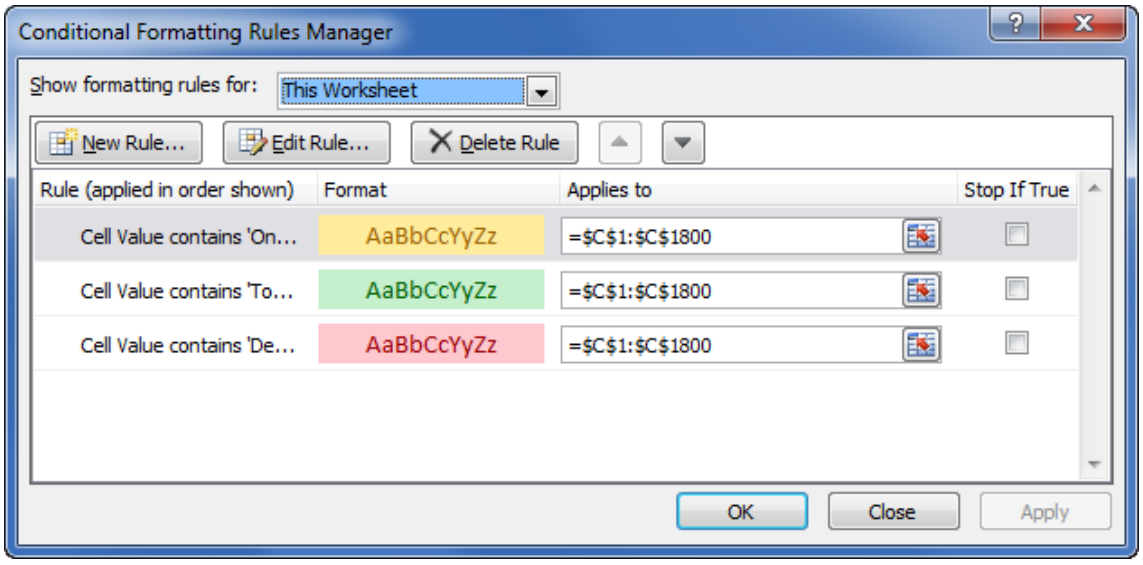
In Excel there is a formatting option called **Conditional Formatting: Text that Contains...**

Figure 20 Conditional Formatting



The setup is configured in the **Rules Manager**:

Figure 21 The Rules Manager



The three rules:

- Text containing "OnUpdateFromServer" gets the yellow highlighting
- Text containing "ToServerHasChanged" gets the green highlighting
- Text containing "Destination Update" gets the pink highlighting

In other words:

- Yellow highlighting is when the SimLink received data from the OPC Server
- Green highlighting is when the SimLink send data to the OPC Server
- Pink highlighting is when the SimLink updated values

Figure 22 The SimLink log file in Excel

	A	B	C
1	Messag	Timestamp	Message
1438	Debug	20141015 09:39:19.896	LSIC_2700PF1_LSIC_270000.Command.man_target OnUpdateFromServer 5.002 OK
1441	Debug	20141015 09:39:31.666	/Engineering/SoftLogixCom2/LSIC_2700PF1_LSIC_270000.Command.man_target^290787887 SendValue 5.002 Model Time 000 00:00:00.00, data Model Time: 000 00:00:02.00 Forced: True Trigger:False
1442	Debug	20141015 09:39:31.914	ksim(SoftLogixCom2/EMU0OTS1EMU02/LP_EMU_2,EMU_IO:6:O.Data[5]#-#)->dcs(LSIC_270.PF1_LSIC_270000.Command.man_target) OnUpdateSource SourceUpdates 5
1443	Debug	20141015 09:39:31.916	ksim(SoftLogixCom2/.EMU0OTS1EMU02/LP_EMU_2/EMU)IO:6:O.Data[5]#-#)->dcs(LSIC_270.PF1_LSIC_270000.Command.man_target) Destination Update 205 Data Model Time 000 00:00:00.00
1444	Debug	20141015 09:39:31.919	dcs(LSIC_2700PF1_LSIC_270000.Command.man_target)->ksim(/Engineering/SoftLogixCom2/lLSIC_270.PF1_LSIC_270000.Command.man_target^290787887#-#) OnUpdateSource SourceUpdates 9
1445	Debug	20141015 09:39:31.921	dcs(lLSIC_270.PF1_LSIC_270000.Command.man_target)->ksim(/Engineering/SoftLogixCom2/LSIC_270.PF1_LSIC_270000.Command.man_target^290787887#-#) Destination Update 5.002 Data Model Time 000 00:00:00.00
1446	Debug	20141015 09:39:32.845	LSIC_270.PF1_LSIC_270000.Command.man_target SendValue 5.002 Model Time 000 00:00:00.00, data Model Time: 000 00:00:00.00 Forced: True Trigger:False
1447	Debug	20141015 09:39:32.849	lLSIC_270.PF1_LSIC_270000.Command.man_target ToServerHasChanged 5.002 TS:---
1452	Debug	20141015 09:39:39.902	dcs(LSIC_2700PF1_LSIC_270000.Command.man_target)->ksim(/Engineering/SoftLogixCom2/lLSIC_270.PF1_LSIC_270000.Command.man_target^290787887#-#) OnUpdateSource SourceUpdates 10
1453	Debug	20141015 09:39:39.903	dcs(lLSIC_270.PF1_LSIC_270000.Command.man_target)->ksim(/Engineering/SoftLogixCom2/ LSIC_270.PF1_LSIC_270000.Command.man_target^290787887#-#) Destination Update 5.207567 Data Model Time 000 00:00:00.00
1454	Debug	20141015 09:39:39.904	LSIC_2700PF1_LSIC_270000.Command.man_target OnUpdateFromServer 5.207567 OK
1459	Debug	20141015 09:39:40.851	LSIC_270.PF1_LSIC_270000. Command.man_target SendValue 5.207567 Model Time 000 00:00:00.20, data Model Time: 000 00:00:00.00 Forced: False Trigger:False
1460	Debug	20141015 09:39:40.852	lLSIC_270.PF1_LSIC_270000.Command.man_target ToServerHasChanged 5.207567 TS:---
1462	Debug	20141015 09:39:41.705	/Engineering/SoftLogixCom2/ LSIC_270.PF1_LSIC_270000.Command.man_target^290787887 SendValue 5.207567 Model Time 000 00:00:01.00, data Model Time: 000 00:00:00.00 Forced: False Trigger:False
1476	Debug	20141015 09:39:47.801	ksim(SoftLogixCom2/EMU0OTS1EMU02/LP_EMU_2,EMU_IO:6:O.Data[5]#-#)->dcs(LSIC_270.PF1_LSIC_270000.Command.man_target) OnUpdateSource SourceUpdates 6
1477	Debug	20141015 09:39:47.803	ksim(SoftLogixCom2/.EMU0OTS1EMU02/LP_EMU_2/EMU)IO:6:O.Data[5]#-#)->dcs(LSIC_270.PF1_LSIC_270000.Command.man_target) Destination Update 213 Data Model Time 000 00:00:06.83
1478	Debug	20141015 09:39:47.805	dcs(LSIC_2700PF1_LSIC_270000.Command.man_target)->ksim(/Engineering/SoftLogixCom2/ LSIC_270.PF1_LSIC_270000.Command.man_target^290787887#-#) OnUpdateSource SourceUpdates 11
1479	Debug	20141015 09:39:47.806	dcs(LSIC_270.PF1_LSIC_270000.Command.man_target)->ksim(/Engineering/SoftLogixCom2/ LSIC_270.PF1_LSIC_270000.Command.man_target^290787887#-#) Destination Update 5.1972 Data Model Time 000 00:00:06.83
1699	Debug	20141015 09:41:47.801	ksim(SoftLogixCom2/EMU0OTS1EMU02/LP_EMU_2,EMU_IO:6:O.Data[5]#-#)->dcs(LSIC_270.PF1_LSIC_270000.Command.man_target) OnUpdateSource SourceUpdates 7
1700	Debug	20141015 09:41:47.802	ksim(SoftLogixCom2/.EMU0OTS1EMU02/LP_EMU_2/EMU)IO:6:O.Data[5]#-#)->dcs(LSIC_270.PF1_LSIC_270000.Command.man_target) Destination Update 2457 Data Model Time 000 00:01:57.00
1701	Debug	20141015 09:41:47.802	dcs(LSIC_2700PF1_LSIC_270000.Command.man_target)->ksim(/Engineering/SoftLogixCom2/ LSIC_270.PF1_LSIC_270000.Command.man_target^290787887#-#) OnUpdateSource SourceUpdates 12
1702	Debug	20141015 09:41:47.803	dcs(LSIC_270.PF1_LSIC_270000.Command.man_target)->ksim(/Engineering/SoftLogixCom2/ LSIC_270.PF1_LSIC_270000.Command.man_target^290787887#-#) Destination Update 59.9508 Data Model Time 000 00:01:57.00
1704	Debug	20141015 09:41:48.855	LSIC_270.PF1_LSIC_270000.Command.man_target SendValue 59.9508 Model Time 000 00:01:57.50, data Model Time: 000 00:01:57.00 Forced: False Trigger:False
1705	Debug	20141015 09:41:48.859	lLSIC_270.PF1_LSIC_270000.Command.man_target ToServerHasChanged 59.9508 TS:---
1708	Debug	20141015 09:41:49.712	/Engineering/SoftLogixCom2/ LSIC_270.PF1_LSIC_270000.Command.man_target^290787887 SendValue 59.9508 Model Time 000 00:01:58.30, data Model Time: 000 00:01:57.00 Forced: False Trigger:False
1709	Debug	20141015 09:41:49.933	dcs(LSIC_2700PF1_LSIC_270000.Command.man_target)->ksim(/Engineering/SoftLogixCom2/ LSIC_270.PF1_LSIC_270000.Command.man_target^290787887#-#) OnUpdateSource SourceUpdates 13
1710	Debug	20141015 09:41:49.934	dcs(lLSIC_270.PF1_LSIC_270000.Command.man_target)->ksim(/Engineering/SoftLogixCom2/ LSIC_270.PF1_LSIC_270000.Command.man_target^290787887#-#) Destination Update 59.9508 Data Model Time 000 00:01:58.50
1711	Debug	20141015 09:41:49.936	LSIC_2700PF1_LSIC_270000.Command.man_target OnUpdateFromServer 59.9508 OK
1714	Debug	20141015 09:41:50.932	dcs(LSIC_2700PF1_LSIC_270000.Command.man_target)->ksim(/Engineering/SoftLogixCom2/lLSIC_270.PF1_LSIC_270000.Command.man_target^290787887#-#) OnUpdateSource SourceUpdates 14
1715	Debug	20141015 09:41:50.939	dcs(lLSIC_270.PF1_LSIC_270000.Command.man_target)->ksim(/Engineering/SoftLogixCom2/LSIC_270.PF1_LSIC_270000.Command.man_target^290787887#-#) Destination Update 5.207567 Data Model Time 000 00:01:59.40
1716	Debug	20141015 09:41:50.943	LSIC_2700PF1_LSIC_270000.Command.man_target OnUpdateFromServer 5.207567 OK
1718	Debug	20141015 09:41:51.713	/Engineering/SoftLogixCom2/ LSIC_270.PF1_LSIC_270000.Command.man_target^290787887 SendValue 5.207567 Model Time 000 00:02:00.00, data Model Time: 000 00:01:59.40 Forced: False Trigger:False
1722	Debug	20141015 09:41:52.861	LSIC_270.PF1_LSIC_270000.Command.man_target SendValue 5.207567 Model Time 000 00:02:01.01, data Model Time: 000 00:01:59.40 Forced: False Trigger:False
1723	Debug	20141015 09:41:52.864	lLSIC_270.PF1_LSIC_270000.Command.man_target ToServerHasChanged 5.207567 TS:---
1728	Debug	20141015 09:41:55.803	ksim(SoftLogixCom2/GC640OTS1EMU02/LP_UCP_2/FPT_IO:6:O.Data[5]#-#)->dcs(LSIC_270.PF1_LSIC_270000.Command.man_target) OnUpdateSource SourceUpdates 8
1729	Debug	20141015 09:41:55.804	ksim(SoftLogixCom2/.EMU0OTS1EMU02/LP_EMU_2/EMU)IO:6:O.Data[5]#-#)->dcs(LSIC_270.PF1_LSIC_270000.Command.man_target) Destination Update 213 Data Model Time 000 00:02:03.50
1730	Debug	20141015 09:41:55.806	dcs(LSIC_2700PF1_LSIC_270000.Command.man_target)->ksim(/Engineering/SoftLogixCom2/lLSIC_270.PF1_LSIC_270000.Command.man_target^290787887#-#) OnUpdateSource SourceUpdates 15
1731	Debug	20141015 09:41:55.809	dcs(LSIC_270.PF1_LSIC_270000.Command.man_target)->ksim(/Engineering/SoftLogixCom2/ LSIC_270.PF1_LSIC_270000.Command.man_target^290787887#-#) Destination Update 5.1972 Data Model Time 000 00:02:03.50

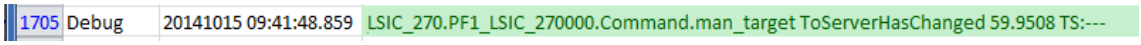
The tags in the image above are heavily manipulated to disguise real data structure. The manipulation has introduced inconsistencies. No doubt.

The following will demonstrate how a value of around 60 went through the system and then was reset back to around 5.

Step 1

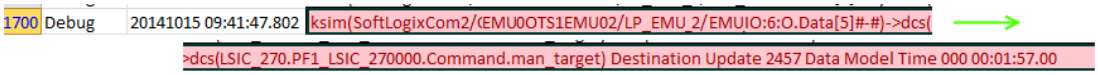
Track down when the value was sent to the OPC server. That is line 1705.

Figure 23 Value sent



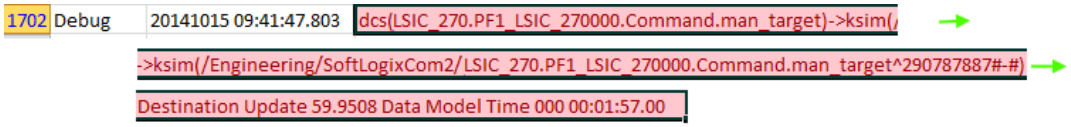
Step 2 Backing up from there, where did it come into this SimLink? In row 1700. Since the value is 2457, the project is using Gain/Bias on this link-to-link tag

Figure 24 The value comes into the link



Step 3 This is a bidirectional connection, 59.95098 is echoed back to the source link at row 1702

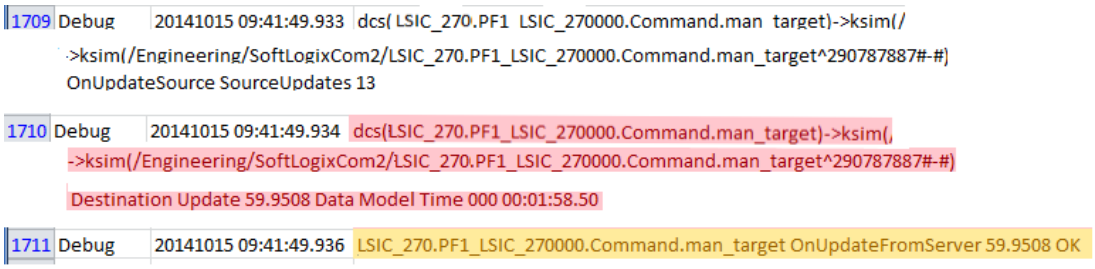
Figure 25 The value is echoed



Step 4 Note that difference in time between 1700 and 1705 is how long the SimLink had the value before sending it on to the OPC Server. Approximately 1 second.

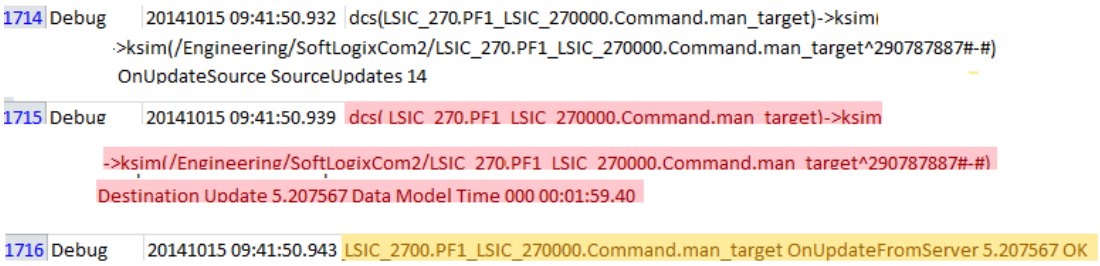
Step 5 Since there is a subscription to the OPC item for read & write, the data is echoed back to us at Row 1711. In the message to the log file, we actually see the log messages for updating the connection for this tag first at rows 1709 and 1710. But these were actually triggered by receiving the data from OPC at row 1711.

Figure 26 Updated Value



Step 6The value of 5.207... comes from the OPC Server at line 1716 which triggers sending it to the connected item on rows 1714 and 1714.

Figure 27 A new value is coming from the OPC Server



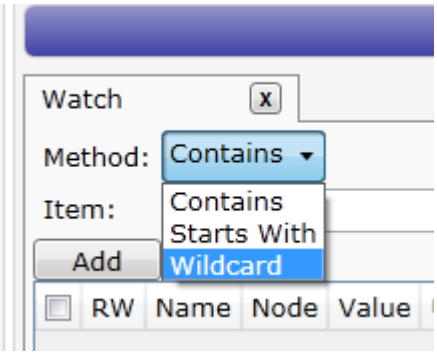
That final event is what we are looking for. What changed it back to around 5 after we set it to around 60? The answer is the DCS, using the OPC Server. At least according to our data trace messages.

3.4 Watch an item in the SimLink GUI

Adding a wildcard selection of tags

The SimLink **Watch** window supports a special syntax for adding multiple items simultaneously. This only works when the search method is Wildcard.

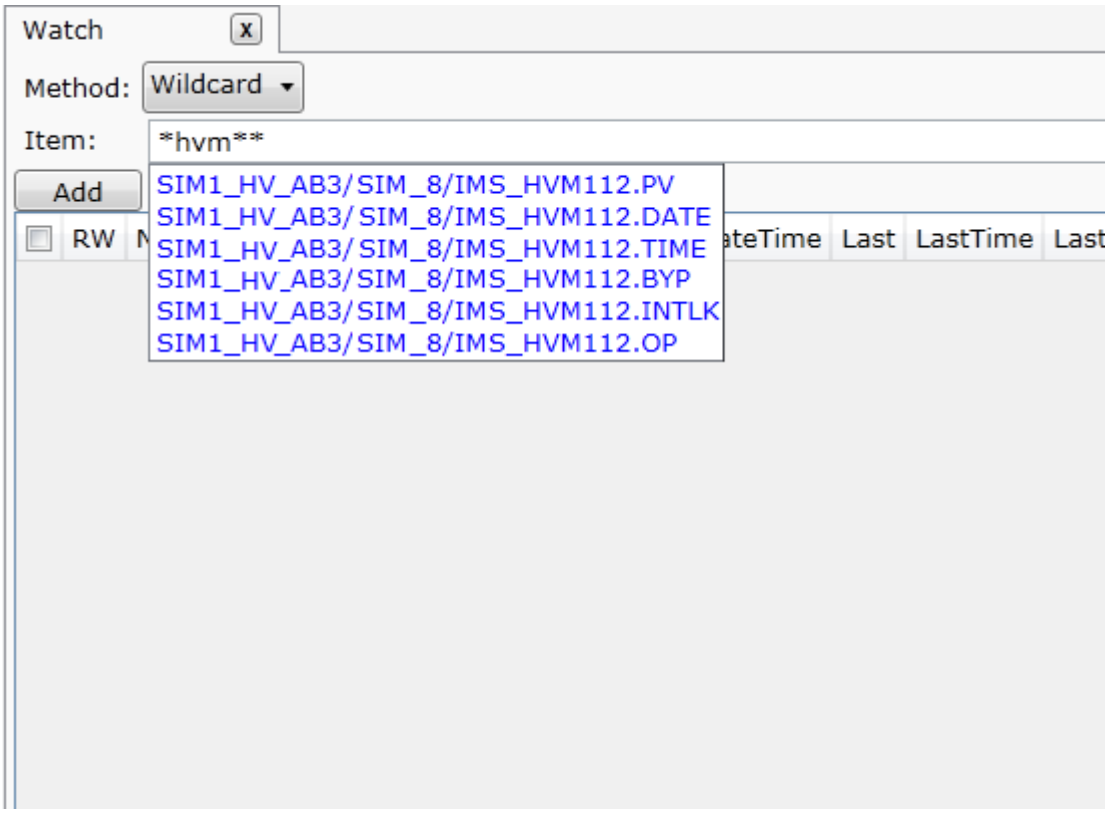
Figure 28 Set search method Wildcard



If you end the search pattern with two asterisks before clicking Add, this instructs Watch to add all matching items.

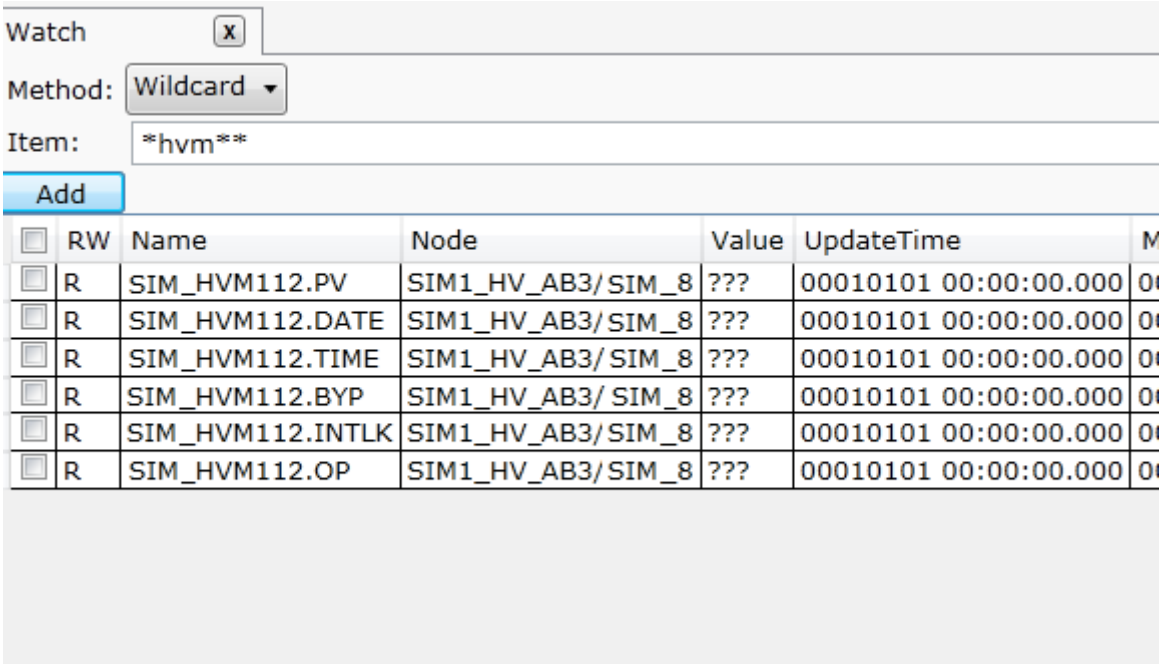
The normal search pattern for this example would be `*hvm*` – which indicates that Watch should show all items with `hvm` in the middle of the name

Figure 29 Set search Item



By adding the second asterisk at the end, Watch will add them all when Add is clicked.

Figure 30 Search item ending with two asterixes



3.5 Tag connection errors

The link will go into a diagnostic mode, when connection errors are encountered

This is an example from **ExatifCom**. When examining the log, you will see information similar to:

Figure 31 Log file in the SimLink GUI with connection errors

Time	Type	
20121109 11201211260.	DcsInfo	0000000:00:00 INFO: TagIOHandler::connect_read_tag HY6201211_ PV Failed
20121109 11:01:14.259.	DcsInfo	0000000:00:00 INFO: TagIOHandler::connect_read_tag mtag not found HY6201211_ PV
20121109 11:01:14.258.	DcsInfo	0000000:00:00 INFO: TagIOHandler::connect_read_tag HY620127B_ PV Failed
20121109 11:01:14.258.	DcsInfo	0000000:00:00 INFO: TagIOHandler::connect_read_tag mtag not found HY620127B_ PV
20121109 11:01:14.257.	DcsInfo	0000000:00:00 INFO: TagIOHandler::connect_read_tag HY620127A_ PV Failed
20121109 11:01:14.256.	DcsInfo	0000000:00:00 INFO: TagIOHandler::connect_read_tag mtag not found HY620127A_ PV
20121109 11:01:14.255.	DcsInfo	0000000:00:00 INFO: TagIOHandler::connect_read_tag HY620127A_ PV Failed
		0000000:00:00 INFO: TagIOHandler::connect_read_tag mtag not found HY620127A_ PV

When connecting to tags in a Station, Exatif either connects all tags successfully or it connects no tags when any error is encountered. So the error messages you are seeing may be for valid tags.

When an error is encountered, ExatifCom will enter a diagnostic mode and connect tags on an individual basis. This is so it can report errors. It does ***not*** automatically disable the incorrect tags and connect only the correct ones. You will see messages in the log file similar to:

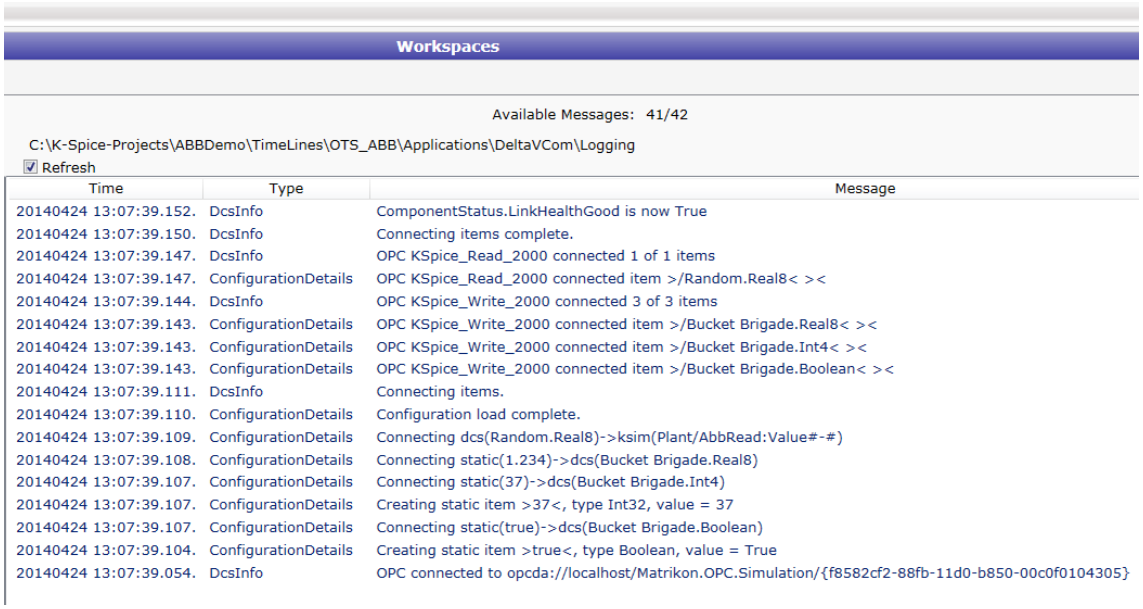
Figure 32 Log file in the SimLink GUI with one error

Time	Type	
20121109 11:01:14.252.	DcsInfo	0000000:00:00 INFO: LocalYokogawaExaTif::do_connect completed: connected
20121109 11:01:13.246.	DcsInfo	0000000:00:00 INFO: Station 3 - 14 80 of 80
20121109 11:01:03.245.	DcsInfo	0000000:00:00 INFO: Station 3 - 14 70 of 80
20121109 11:00:53.295.	DcsInfo	0000000:00:00 INFO: Station 3 - 14 60 of 80
20121109 11:00:43.245.	DcsInfo	0000000:00:00 INFO: Station 3 - 14 50 of 80
20121109 11:00:33.245.	DcsInfo	0000000:00:00 INFO: Station 3 - 14 40 of 80
20121109 11:00:23.245.	DcsInfo	0000000:00:00 INFO: Station 3 - 14 30 of 80
20121109 11:00:13.245.	DcsInfo	0000000:00:00 INFO: Station 3 - 14 20 of 80
20121109 11:00:03.245.	DcsInfo	0000000:00:00 INFO: Station 3 - 14 10 of 80
20121109 10:59:54.295.	DcsInfo	0000000:00:00 INFO: Station 3 - 14 trying 80 get tags
20121109 10:59:48.244.	DcsInfo	0000000:00:00 INFO: Station 3 - 14 150 of 155
20121109 10:59:38.246.	DcsInfo	0000000:00:00 INFO: Station 3 - 14 140 of 155
20121109 10:59:28.244.	DcsInfo	0000000:00:00 INFO: Station 3 - 14 130 of 155
20121109 10:59:18.243.	DcsInfo	0000000:00:00 INFO: Station 3 - 14 120 of 155
20121109 10:59:08.244.	DcsInfo	0000000:00:00 INFO: Station 3 - 14 110 of 155
20121109 10:59:04.363.	DcsInfo	0000000:00:00 INFO: Station 3 - 14 tag not connecting ZS20121109pe_ PV
20121109 10:58:58.243.	DcsInfo	0000000:00:00 INFO: Station 3 - 14 100 of 155
20121109 10:58:48.244.	DcsInfo	0000000:00:00 INFO: Station 3 - 14 90 of 155
20121109 10:58:38.296.	DcsInfo	0000000:00:00 INFO: Station 3 - 14 80 of 155
20121109 10:58:28.294.	DcsInfo	0000000:00:00 INFO: Station 3 - 14 70 of 155
20121109 10:58:18.243.	DcsInfo	0000000:00:00 INFO: Station 3 - 14 60 of 155
20121109 10:58:08.243.	DcsInfo	0000000:00:00 INFO: Station 3 - 14 50 of 155
20121109 10:57:58.256.	DcsInfo	0000000:00:00 INFO: Station 3 - 14 40 of 155
20121109 10:57:48.242.	DcsInfo	0000000:00:00 INFO: Station 3 - 14 30 of 155
20121109 10:57:38.245.	DcsInfo	0000000:00:00 INFO: Station 3 - 14 20 of 155
20121109 10:57:28.243.	DcsInfo	0000000:00:00 INFO: Station 3 - 14 10 of 155
20121109 10:57:19.312.	DcsInfo	0000000:00:00 INFO: Station 3 - 14 trying 155 put tags
20121109 10:57:19.311.	DcsInfo	0000000:00:00 INFO: Station 3 - 14 didn't connect all tags. Trying individually as a diagnostic.

3.6 Tag connection success

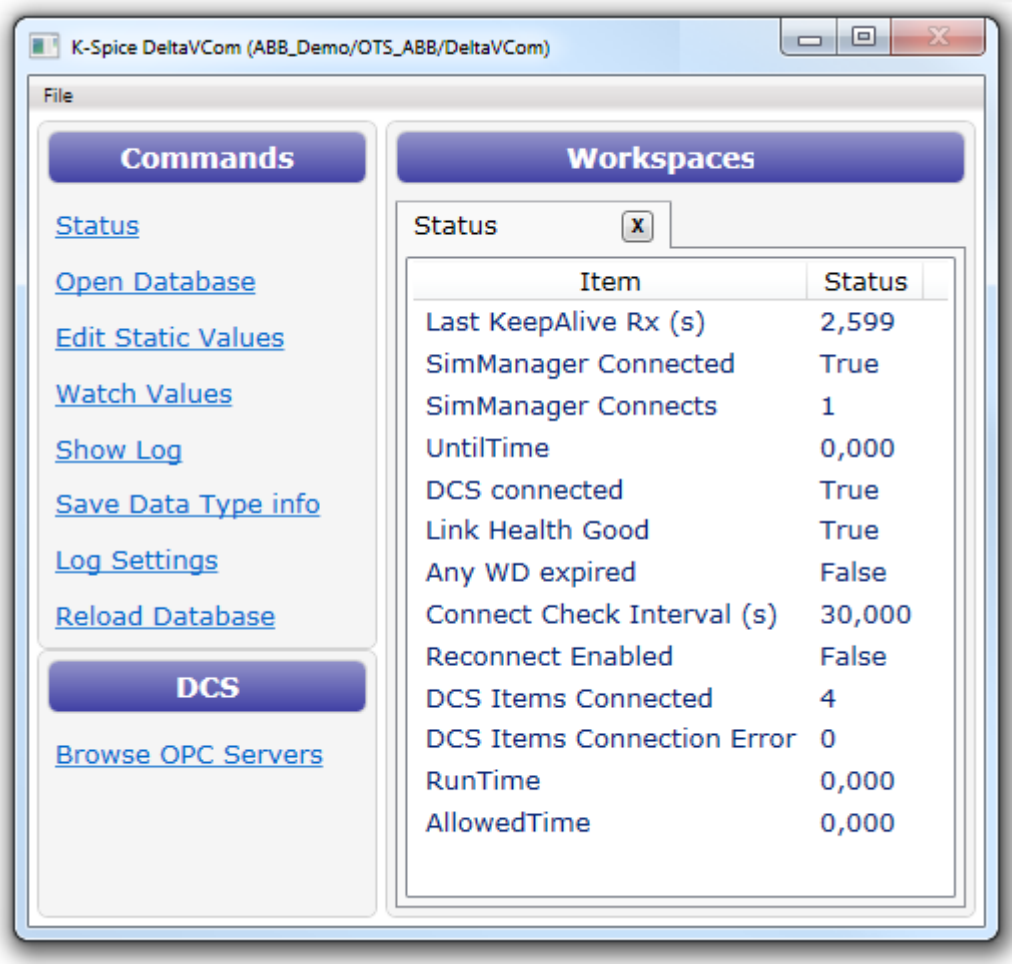
Log file from a test setup with a DeltaVCom link and default values in a Matrikon test server

Figure 33 Log file in the SimLink Gui from successful connection



Time	Type	Message
20140424 13:07:39.152.	DcsInfo	ComponentStatus.LinkHealthGood is now True
20140424 13:07:39.150.	DcsInfo	Connecting items complete.
20140424 13:07:39.147.	DcsInfo	OPC KSpice_Read_2000 connected 1 of 1 items
20140424 13:07:39.147.	ConfigurationDetails	OPC KSpice_Read_2000 connected item >/Random.Real8< ><
20140424 13:07:39.144.	DcsInfo	OPC KSpice_Write_2000 connected 3 of 3 items
20140424 13:07:39.143.	ConfigurationDetails	OPC KSpice_Write_2000 connected item >/Bucket Brigade.Real8< ><
20140424 13:07:39.143.	ConfigurationDetails	OPC KSpice_Write_2000 connected item >/Bucket Brigade.Int4< ><
20140424 13:07:39.143.	ConfigurationDetails	OPC KSpice_Write_2000 connected item >/Bucket Brigade.Boolean< ><
20140424 13:07:39.111.	DcsInfo	Connecting items.
20140424 13:07:39.110.	ConfigurationDetails	Configuration load complete.
20140424 13:07:39.109.	ConfigurationDetails	Connecting dcs(Random.Real8)->ksim(Plant/AbbRead:Value#-#)
20140424 13:07:39.108.	ConfigurationDetails	Connecting static(1.234)->dcs(Bucket Brigade.Real8)
20140424 13:07:39.107.	ConfigurationDetails	Connecting static(37)->dcs(Bucket Brigade.Int4)
20140424 13:07:39.107.	ConfigurationDetails	Creating static item >37<, type Int32, value = 37
20140424 13:07:39.107.	ConfigurationDetails	Connecting static(true)->dcs(Bucket Brigade.Boolean)
20140424 13:07:39.104.	ConfigurationDetails	Creating static item >true<, type Boolean, value = True
20140424 13:07:39.054.	DcsInfo	OPC connected to opcda://localhost/Matrikon.OPC.Simulation/{f8582cf2-88fb-11d0-b850-00c0f0104305}

Figure 34 Status log with 4 successful dcs connections



A full tag connection success implies that there is a tag connected *in both ends*. When the connection session starts, you will see this message as a verification of tag connection in both ends, :

```
ConfigurationDetails Connecting
dcs(OTS_R9.Global.R9_TAG-TAG-0002_DO)
->ksim(ProcessModel/HimaRead:DisplayValue#-#)
```

Note

*The value in the Status log for **DCS Items Connected** refers only to a success on the DCS side of the connection*

The DCS side of the connection is normally configured first, and setting the DCS tags correctly is normally the difficult part of the link configuration. To avoid that the logs are flooded with warnings about missing K-Spice data items, the status log gives feedback of the DCS part of the configuration, only.

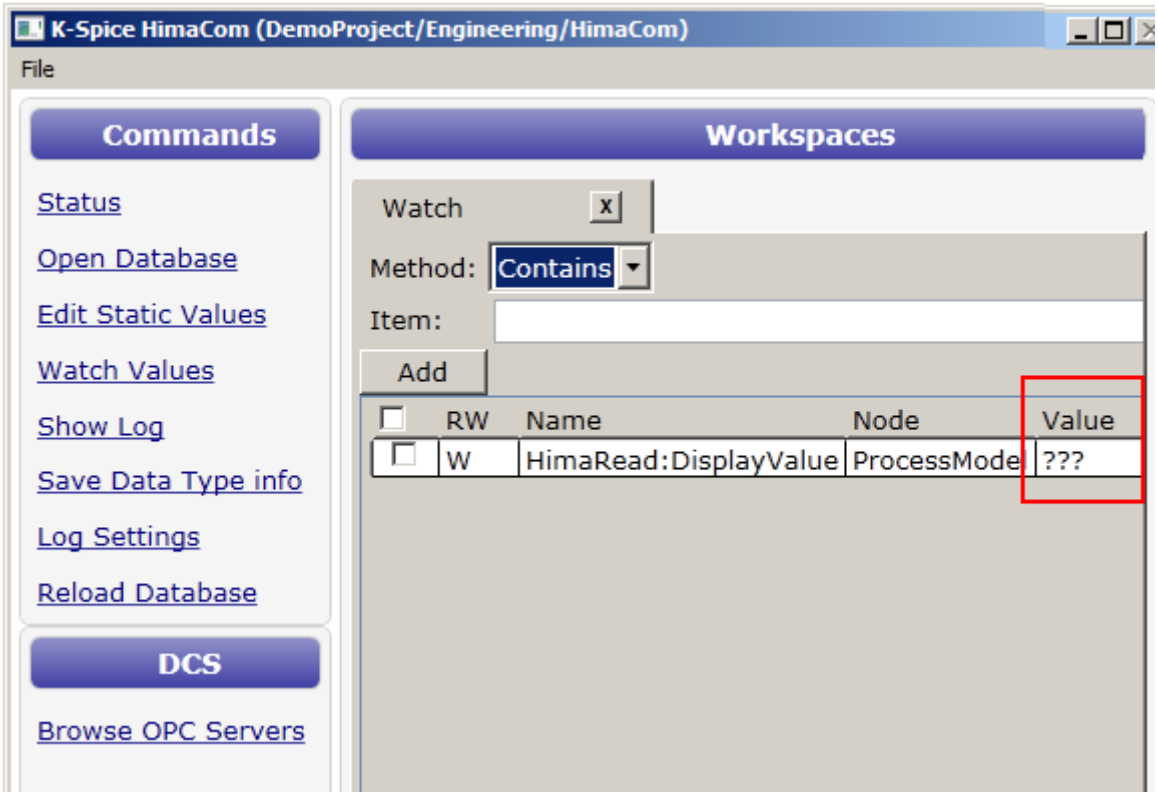
See next session **The ksim end of the connection has failed**

3.7 The ksim end of the connection has failed

If you at the end of the connection session, see in the status log that 4 DCS Items are connected, this is an indication that the DCS side of the connection is correct. The situation can still be that the **ksim** side of the connection has failed. I.e. if the name of the timeline is wrong, there is no warning in the SimLink GUI log. The timeline name is part of the **SubServer** keyword in the **SubServers** table. If the timeline name is wrong, the **ksim** end of the connection fails.

If you add the item under the Watch tab in the SimLink GUI, you will see that there is a problem. *With a loaded model,* the link cannot find the value of the K-Spice data item. The value of the data item is indicated with ???.

Figure 35 No value in the Watch workspace



Tag connected in the ksim end

When the **ksim** side of the connection is OK, you will see this message in the SimLink GUI log:

```
ConfigurationDetails Tag connected.  Status OK. Type:
TYPE_DOUBLE - ProcessModel/HimaRead:DisplayValue#-#
```

3.8 Missing node

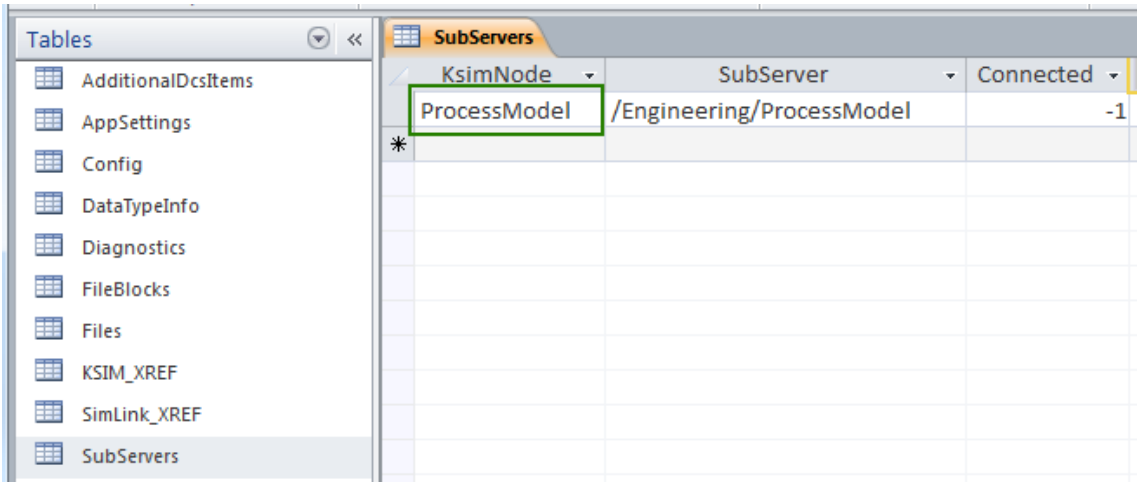
The key KsimNode in the table KSIM_XREF was not given the correct name of the KsimNode.

```
ConfigurationWarning:  Missing node, unable to connect to
HimaRead:Value #-# at 1
```

Figure 36 Missing node in the SimLink GUI log file

```
08:59:45.411.  ConfigurationWarning   Missing node, unable to connect HimaWrite:DisplayValue#-# at 6
08:59:45.411.  ConfigurationWarning   Missing node, unable to connect HimaWrite:MeasuredValue#-# at 5
08:59:45.411.  ConfigurationWarning   Missing node, unable to connect HimaRead:SignalDrift#-# at 4
08:59:45.411.  ConfigurationWarning   Missing node, unable to connect HimaRead:Value#-# at 1
08:59:45.411.  ConfigurationWarning   IO List has undefined subserver >ProsessModel<
08:59:45.380.  ConfigurationInfo      Using default SubServer of >ProcessModel< from Single SubServer
08:59:45.380.  DcsInfo                OPC connected to opcda://10.0.0.3/HIMA.PL-DA/{de044b7d-47b6
08:59:45.380.  DcsDetails              state changed to: Starting
```

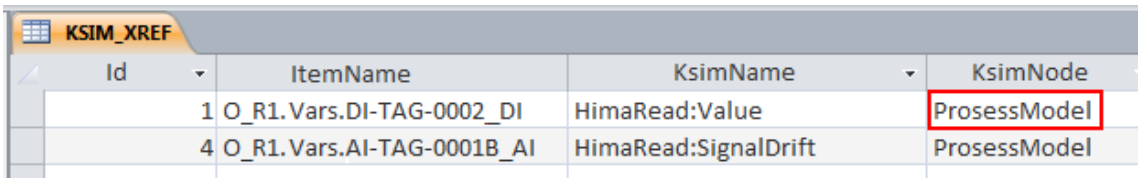
Figure 37 KsimNode in table SubServers



In the table **SubServers**, the key KsimNode was given the value **ProcessModel**. ProcessModel is correct and in agreement with the model name in K-Spice.

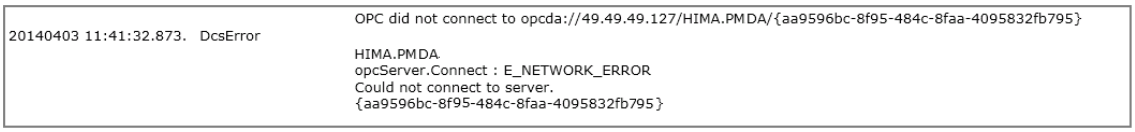
In the table **KSIM_XREF** the value of the key KsimNode was misspelled as ***ProsessModel***. The result of the configuration error was that the link was not able to connect the Hima tag with a K-Spice data item.

Figure 38 KsimNode in table KSIM_XREF



3.9 DcsError

Figure 39 The log message E_NETWORK_ERROR

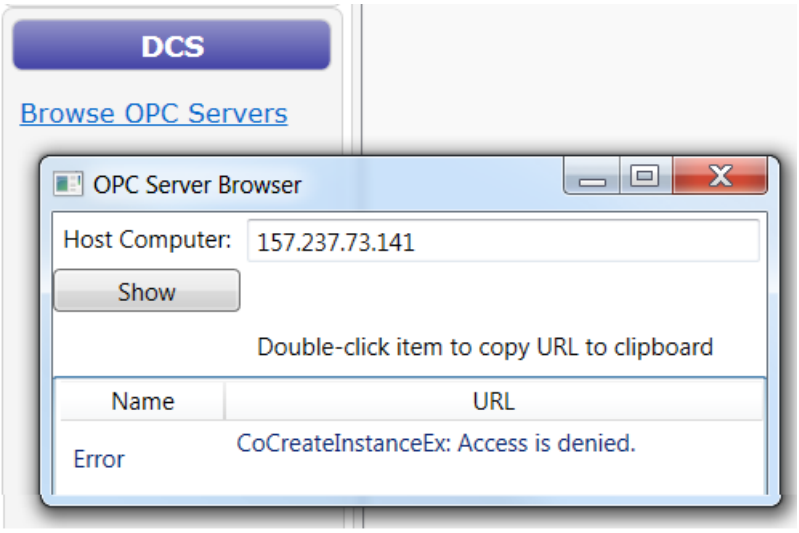


This error was caused by two control characters, Carriage Return and Line Feed, in the middle of the name of the OpcServer. The string used in the keyword OpcServer in the Config table, *was copied from a page in a text editor with ctrl-C ctrl-V.* To this simple string, the Access database *added two control characters.* Only the second part of the string, the part after the Line Feed, was seen in the Access database. When the row was made higher, the full text, two lines, was visible.

3.10 CoCreateInstanceEx

If the OPC Server is on a different machine, the option **Browse OPC Servers** can fail with the message in the bitmap below. See a description of the Browse option in the chapter The SimLink GUI. See *Browse OPC Servers.* on page 31

Figure 40 Access is denied



The first point in the checklist for DCOM configuration is: *Verify that the Model PC and the OPC Server are members of the same domain.* If not, the error message is **Access is denied**. See *DCOM configuration for AbbItsCom* on page 112

4 The Access database

The Access database chapter is about link configuration.

All the configuration details of the different links are organized as values, called **Keywords**, in **Tables** in the **Microsoft Access** database. This chapter is a description of the detailed configuration the link. The configuration is stored with the Microsoft Access database.

Some general database issues:

The Cross Reference List

An essential part of a working link is the **Cross Reference List**. All *tags received from the DCS* system ***has one line*** in the Cross Reference List. *The corresponding tag in K-Spice* is written on the same line.

*Creation and maintenance of the **Cross Reference List** for a specific project, is a task that is not covered in this manual.*

This Link Guide is written with assumption that the **Cross Reference List** for the project already exists.

The Cross Reference List will be imported into the KSIM_XREF table in the Access database and is an important part of the link configuration. See a summary in the chapter *Creation of the Cross Reference list.* on page 130

ToSubServer

ToSubServer is an important boolean keyword in the table **KSIM_XREF**. **ToSubServer** defines if K-Spice is *reading* the value *from* the DCS system, or if K-Spice is *writing* the value *to* the DCS system..

ToSubServer is in some databases implemented as a tick box, in other databases as an integer

Note

*Read: **ToSubServer** is true, **ticked on**, **—1**. K-Spice **reads** a value from the DCS system*

*Write: **ToSubServer** is false, **ticked off**, **0**. K-Spice **writes** a value to the DCS system*

Connected

Connected is another important boolean keyword in the table **KSIM_XREF** K-Spice will in normal operation read a value from the DCS system or write a value to the DCS system through the link. This is if the variable is **Connected.** (-1 is connected). Setting the keyword **Connected** in the table **KSIM_XREF** to 0 (0 is disconnected) *will take a variable out of the transfer of data.* If the variable is disconnected, the variable is still a defined variable in the Access database. But no subscription or connection is established. No data will be transferred in any direction.

Connected is in some databases implemented as a tick box, in other databases as an integer

Note

Connected: *the Connected box is* ***ticked on****, equivalent to the value* ***—1***

Disconnected: *the Connected box is* ***ticked off****, equivalent to the value* ***0***

Bucket Brigade.Int4 and Saw-toothed Waves.Real4

You will see these tagnames in many of the images in this guide. These are default tagnames in the test server MatrikonOPC Server.

Effort is made in this manual to avoid presentation of tagnames from real projects. Most of the images will be from a K-Spice demoproject. And most of the images will not present data from a real DCS system A test server, called the MatrikonOPC Server, is used to simulate a real DCS system for most of the images. The Matrikon OPC Server can be configured to send data with tagnames that are equal in form to the tagnames you will find in a real DCS system.

But the MatrikonOPC Server has also a set of *default tagnames* that are configured by the Matrikon application itself. Many of the images and bitmaps in this manual will present the default Matrikon tagnames. You will see them in the ItemName field in the KSIM_XREF table. Examples of default Matrikon tagnames are:

Random.Int4, Bucket Brigade.Real8, Bucket Brigade.Int4 and Saw-toothed Waves.Real4

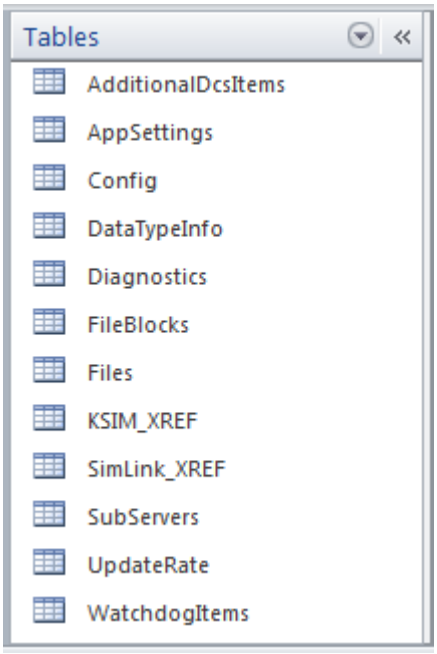
Save values

It is not necessary to explicitly save new values in the Microsoft Access database. Once a new value for a keyword is entered, the new value is immediately saved. That is at least how it should be. There is a save button, just in case.

4.1 Tables

The OpcDaCom and OpcHdaCom Access databases both have 12 tables.

Figure 41 Tables in the Access database



Tables

All the configuration details of the different links are organized as values, called **Keywords**. A **Table** in the Microsoft Access database is a group of **Keywords**.

The database used in a project is created as part of the installation and initialization of the link application. The database is actually created *during the first **Activation*** of the timeline where the link belongs. See *Creation of a new database* on page 25

The configuration in the database will vary from link to link. There will be tables in the Access database for one link, that you will not find in the Access database for another link. The tables in the Access database are defined by the installation program for the link application.

The database tables are described in this chapter, one by one in alphabetical order. The two most important tables are:

1 **Config** with general link parameters

2 **KSIM_XREF** with the description of all the tags that are sent through the link.

Standard tables vs. Specific tables.

Most tables are common to all the different DCS links. The common tables are described in the section **Standard database tables.** In addition to the Standard tables some links have **Specific tables.** I.e. the SoftLogixCom link has the two extra tables **AbTypes** and **PLC.** These two tables are **Specific tables**, and will only be described in the guide for the SoftLogixCom link.

The OPC links do not have Specific tables.

Standard keywords vs. Specific keywords

Many of the sections *in this guide*, describing one and the same table, are split in two parts. The first part describes the ***Standard keywords*** in that table. Standard keywords are common to all the different DCS links. If there is a second part, the second part describes the ***Specific keywords***. Specific keywords are unique keywords for the actual DCS link.

For example the Config table in the VirtusoCom link has a keyword **IntermediateInputClock. IntermediateInputClock** is a *Specific* keyword for the VirtusoCom link. This keyword does not exist in the other DCS links.

4.2 Standard database tables

The database tables are described in this section, one by one, in alphabetical order..

4.2.1 The **Config** table. Standard keywords

Table 2 The Config table, Standard keywords shared by all SimLink.NET applications

Keyword	Notes
ConnectCheckIntervalSeconds	• Optional • Version: 2.5.0.15 • Real value that indicates how often in seconds the SimLink checks to see if the DCS is still connected • Default is 30.0
DefaultSubServer	• Optional • Value is SubServer name to use as default for tags having blank SubServer field. • If missing *and* only a single *should-connect* SubServer is defined, SimLink.NET will use that SubServer as the default
DurationWarningMilliseconds	• Optional • Version: 2.8 2.9 • DcsServer times method call in its processing thread. It logs any method to execute that takes longer than this value • Default is 200
EpochStart	• Optional • Version: 2.9.5.0 • Base date/time to add to model time (used for real-time models) • Format is year, month, day, hour, minute, second • Default is 1970, 1, 1, 0, 0, 0

Table 2 The Config table, Standard keywords shared by all SimLink.NET applications (cont'd.)

ExitTimeoutMs	• Optional • How long the SimLink waits for DCS to disconnect when shutting down or restarting • Default is 10,000 ms, but individual SimLinks may have a different default • This **overrides** any default by the SimLink
FullUpdate	• Optional • Full Update rate for K-Spice subscriptions in seconds. • The SimLink receives an update for every tag at has changed value since the last update. • Default is the communication library default of 30 (2011/06/10).
ReconnectEnabled	• Optional • Version: 2.5.0.15 • Can be true or false. Default is false. If true, supporting SimLink will attempt to reconnect to a DCS that has lost connection
ReconnectIntervalSeconds	• Optional • Real value that indicates how often in seconds the SimLink should attempt to reconnect if ReconnectedEnabled is true • Default is 30.0
SimLinkSaveFolder	• Optional • Default is empty for disabled. Enables saving of bidirectional values in a SimLink save file (currently/typically used by OpcDaCom right now)
StepSlipMilliseconds	• Optional • How much early (in milliseconds) to ack a SimStep • Default is 200

The DefaultSubServer keyword. Example

If there are two sub models **Topsides** and **Marine**. the user can select i.e. **Marine** as the default subserver.

Topsides and **Marine** must be added as keywords **KsimNode** in the table **SubServers**. See *The **SubServers** table.* on page 78

4.2.2 The **Config** table. All OPC client links

Table 3 The Config table. Keywords shared by all OPC Client links

Keyword	Notes
OpcServer	• Required for OPC Links • OPC Server Connection URL • Example: opcda://localhost/Matrikon.OPC.Simulation/{f8582cf2-88fb-11d0-b850-00c0f0104305} • Best way to set is use Browse option from SimLink to copy to clipboard. The Browse option is described below.
EnableDateTimeStamp	• Optional • Version: 2.9.4.0 • Defaults to false • When enabled, the current **model time** is sent to the OPC DA server as the timestamp for the data value
EpochStart	• Optional • Version 2.9.4.0 • Only relevant if EnableDateTimeStamp is true • Base date/time to add to model time • Format is year, month, day, hour, minute, second • Default is 1970, 1, 1, 0, 0, 0

See the chapter **The SimLink GUI** on how to find the correct value for the keyword OpcServer. See *Browse OPC Servers.* on page 31

4.2.3 The **DataTypeInfo** table

This table gives details about the type of data at source and destination. This table can be updated by clicking on **Save Data Type info** on the main window for the link. This table is normally not used. For OPC links it can be kept as it is.

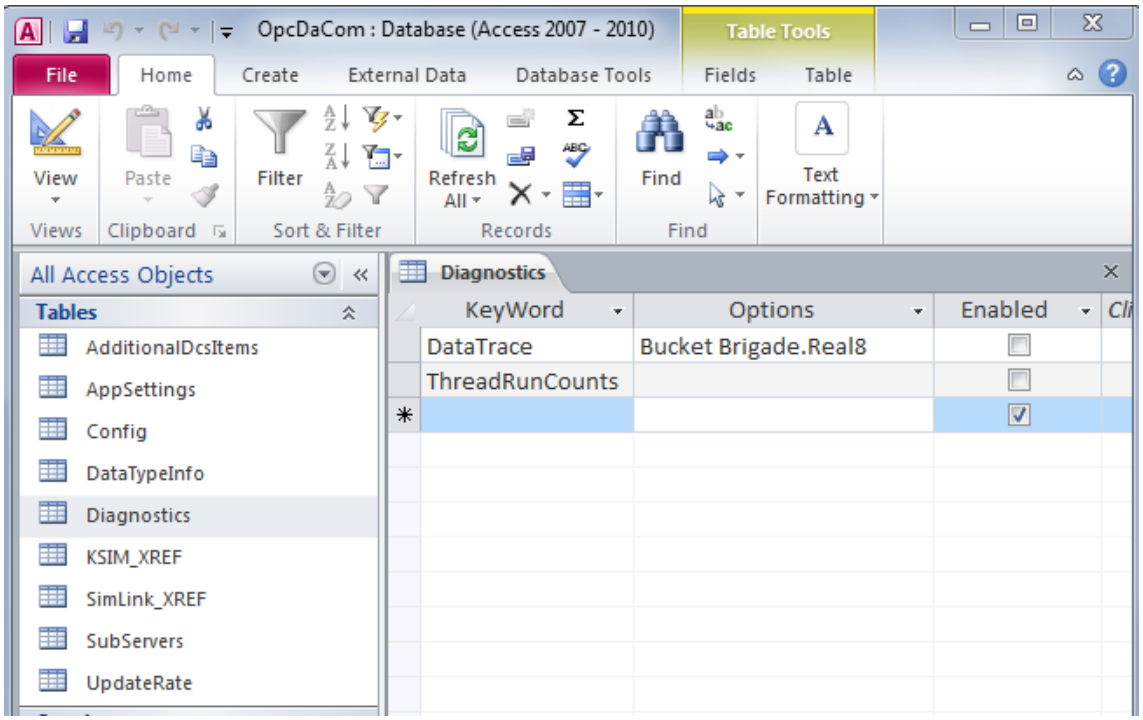
4.2.4 The **Diagnostics** table

The Diagnostics table in the SimLink.NET database can be used to configure various diagnostics available in a SimLink.

Table 4 Keywords in the Diagnostics table common to all SimLink.NET applications:

Keyword	Notes
ControlItemMonitor	• For SimLinks that support it – Every 5 minutes, will write control item values to the log file. • From version 2.11.2.: All OPC DA based links support this diagnostic. Logs control item values that are enabled for read-from-OPC-Server. • This is useful when diagnosing a problem where ControlItemTrace sends too many messages to the log file.
ControlItemTrace	• When enabled for SimLinks that support it, enhanced debug messages to log file about data updates for control items. Similar to the DataTrace messages. • From version 2.9.11.0: All OPC DA based links support this diagnostic
DataTrace	• Options : semi-colon separated list of Item names • Enhanced debug messages to log file about data updates for items in the Options list. • See *The Diagnostics table. The DataTrace keyword.* on page 61 for more information.
GroupTrace	• Options: 1) <none> indicates “all” • Options 2) semi colon separated list of rates. • Enhanced debug messages to log file about item groups based on update rate. • Support depends on specific link • From version 2.10.2.0: OPC DA based link support this. Includes Write and OnWriteComplete messages. • See *The Diagnostics table. The GroupTrace keyword.* on page 66 for more information.
ThreadRunCounts	• When enabled, adds a run count for the Manager and DCS threads running. Used to determine if a thread is freezing.

Figure 42 The Diagnostics table. An example.



4.2.5 The Diagnostics table. The DataTrace keyword.

The **DataTrace** diagnostics is a useful tool for identifying data transfer problems in a project. See Using Excel conditional formatting on SimLink log files (not written yet) for an example of using DataTrace to analyze an issue.

Use Excel

Typically it's easier to see the message if you use **Excel** to view the log file instead of a a text editor. See SimLink Log files (not written yet) and Using Excel conditional formatting on SimLink log files (not written yet) for more information.

The trace messages in the log file are of the type **Debug**, to it's easy to filter **Excel** display to only show these.

Figure 43 Select MessageType Debug in a logfile in Excel.

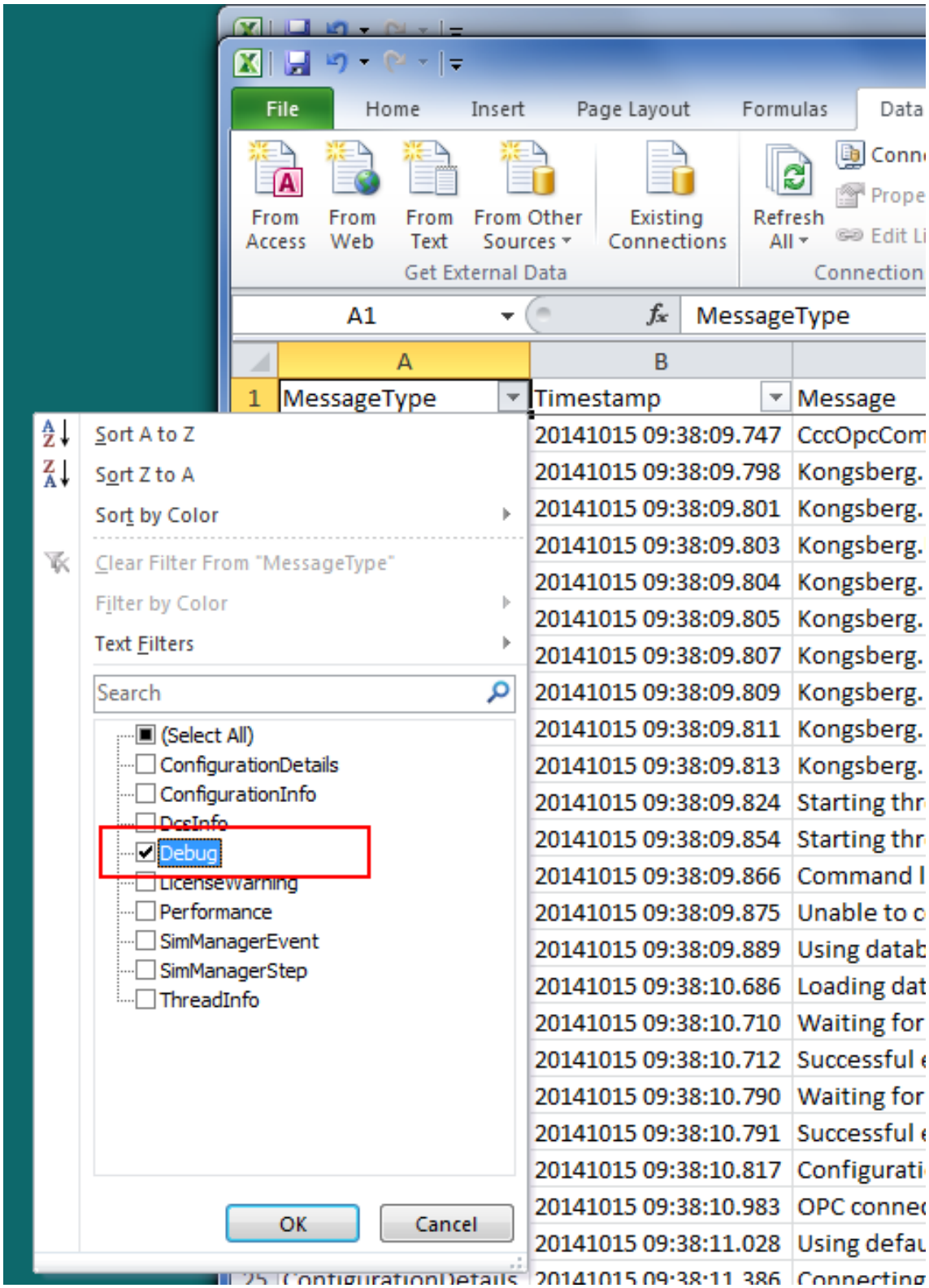
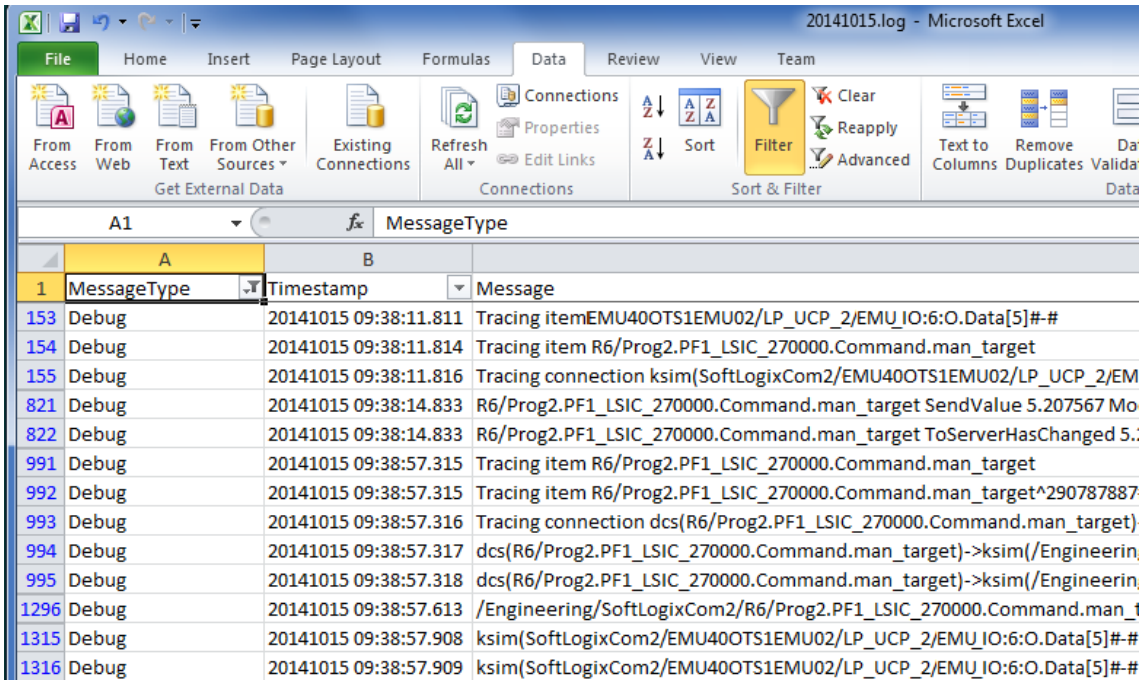


Figure 44 Debug messages sorted out



What does DataTrace do?

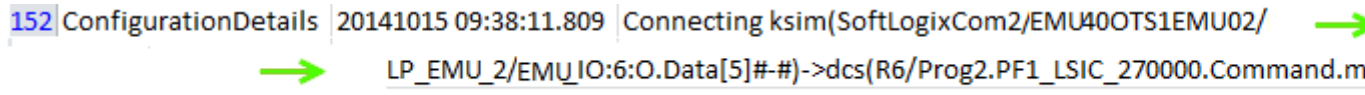
If you enable **DataTrace** on an item, it will automatically enable tracing on any item it's connected to and the connection itself. **DataTrace** enables tracing on any item whose full name begins with the one of the option strings specified. So it's possible to inadvertently enable tracing on multiple items.

For example, if you enable tracing on **Bucket Brigade**, all items that being with "Bucket Brigade" will be included in the trace. Typically, you will specify the full name of the item you wish to trace. In this example, trace was enabled on `SoftLogixCom2/EMU40OTS1EMU02/LP_EMU_2/EMU_IO:6:O.Data[5]`

How do I figure out the name to use for DataTrace?

Find the message in the log file that has the Connecting source -> destination information. This message shows the full name of the items for the SimLink.

Figure 45 The Connecting source message



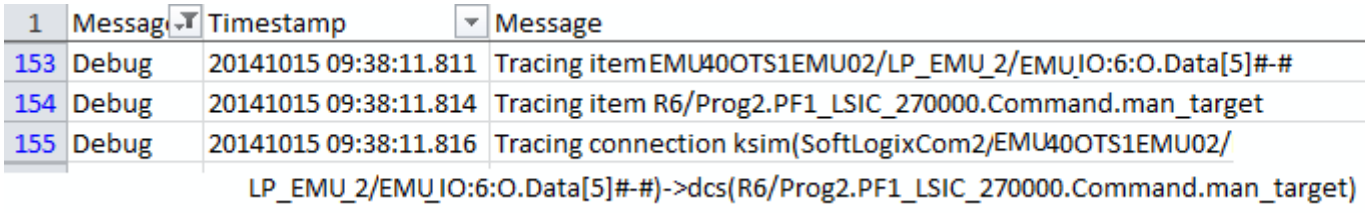
Here, the source item is
`SoftLogixCom2/EMU40OTS1EMU02/LP_EMU_2EMU_IO:6:O.Data[5]`

The destination item is `R6/Prog2.PF1_LSIC_270000.Command.man_target`

Verify DataTrace enabled for the item

When you turn on tracing, you should verify you got the name and spelling correct by looking for the tracing startup messages.

Figure 46 Verify the spelling



DataTrace was enabled for SoftLogixCom2/EMU40OTS1EMU02/LP_EMU_2/EMU_IO:6:O.Data[5] which automatically enabled tracing for R6/Prog2.PF1_LSIC_270000.Command.man_target because there is a connection defined for them.

Common Trace messages

The trace messages above are common to all SimLinks. Specific links may have addition trace messages. OPCDA-based links have additional messages, for example **DestinationUpdate**, **OnUpdateSource** and **SendValue**.

DestinationUpdate

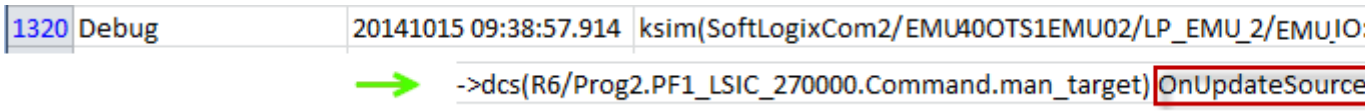
Figure 47 DestinationUpdate

Destination Update indicates that the destination item value is being updated. The value following the words **Destination Update** is the value being sent to the destination. This value is the **Source** value and does not include any inversion, gain or bias that may be applied by the destination.

Data Model Time is the model time that the **Source** item has *timestamped* for that value. If it's coming from a model subscription update, then model time was sent by the model server. If the source data update does not include a model timestamp, the SimLink will use it's current value for model time, when it saves the source value.

OnUpdateSource

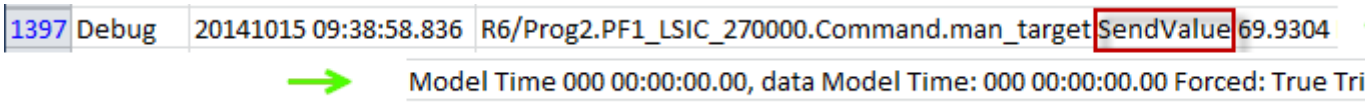
Figure 48 OnUpdateSource



OnSourceUpdate indicates the link has received an update on the source data item. The number following **SourceUpdates** is a counter of how many of these events have been received. This message is typically followed by **DestinationUpdate** message as the connection takes the source value and propogates it to the destination item.

SendValue

Figure 49 SendValue



SendValue indicates that the SimLink is getting the current value of the item so it can be transmitted to its destination (ModelServer, DCS, etc).

- The value that follows **SendValue** is the value being sent.
- **Model Time** indicates the current model time according to the link.
- data **Model Time** indicates that timestamp associated with when the value was originally updated by link.
- **Forced** is a true or false value indicating whether some component in the SimLink requested that the value be sent regardless of any tolerance value.
- **Trigger** is an internal true/false value within the link for items that should have their values sent after specific events regardless of any tolerance setting.

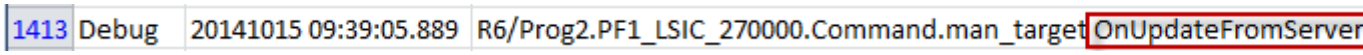
Typically, if Forced and Trigger are both false, that indicates that value is being sent because the change is outside the configured tolerance.

OPC Trace messages (OpcDaCom, CccOpcCom, DspiceCom, HimaCom, etc)

OnUpdateFromServer and ToServerHasChanged are OPC specific trace message for all OPC-DA based links.

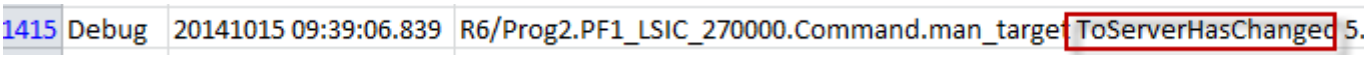
OnUpdateFromServer

Figure 50 OnUpdateFromServer



OnUpdateServer indicates that the OPC Server has sent the link a value for the OPC item. The value is displayed next in the message followed by the **Quality** for the item.

ToServerHasChanged



ToServerHasChanged indicates that the SimLink is sending a value to the OPC Server. The write to the server occurs almost immediately after this message.

TS is the timestamp that is sent to the server, where "—" indicates this feature is not enabled.

4.2.6 The Diagnostics table. The GroupTrace keyword.

GroupTrace is intended to be a common diagnostic for all SimLink for providing trace messages involving groups of tags. It will be implemented as needed over time. The first implementation is for all OPC DA based links including OpcDaCom, DspiceCom, CccOpcCom, etc.

GroupTrace is enabled by the rate number used for updating tags. You can enable it for multiple rates by separating the values by semi-colon. If the option field is blank, then the **GroupTrace** is enabled for all rates.

Groups also get trace enabled if any tag in that group has **DataTrace** enabled.

GroupTrace options

Figure 51 All rates are enabled

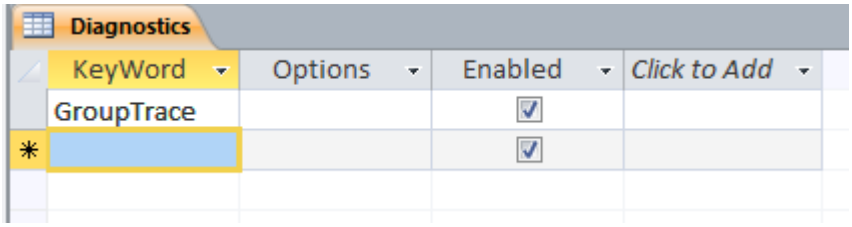


Figure 52 Just the 2000 rate is enabled

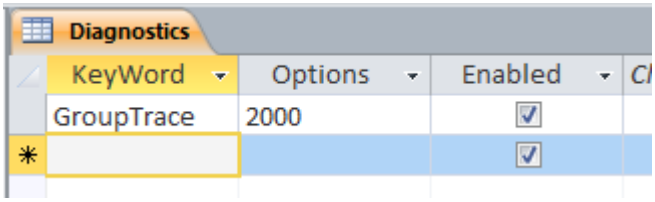
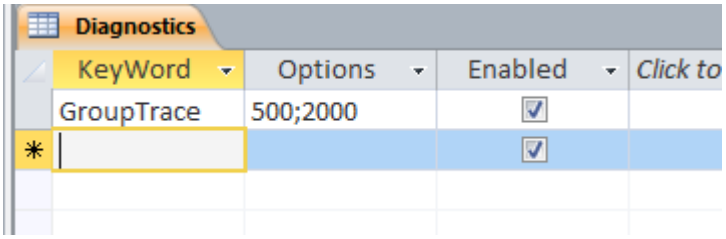


Figure 53 Groups with rate 500 and 2000 are enabled



OPC GroupTrace

OPC DA based links will get messages for **onBeforeWrite** and **onWriteComplete**.

onBeforeWrite is just before the OPC client method for write is called. **onWriteComplete** occurs when the server has indicated it has completed processing of the write.

Each write from a group includes a transaction **ID**. The **onWriteComplete** for a particular group write will have a matching **ID**

Figure 54 onBeforeWrite and onWriteComplete

	A	B	C
1	MessageType	Timestamp	Message
33	Debug	20141031 13:27:38.689	Tracing enabled for OPC KSpice_Write_2000
66	Debug	20141031 13:27:50.720	OPC KSpice_Write_2000 onBeforeWrite Transaction ID 1
67	Debug	20141031 13:28:03.058	OPC KSpice_Write_2000 onWriteComplete Transaction ID 1
68	Debug	20141031 13:28:31.061	OPC KSpice_Write_2000 onBeforeWrite Transaction ID 2
69	Debug	20141031 13:28:35.076	OPC KSpice_Write_2000 onWriteComplete Transaction ID 2
70	Debug	20141031 13:29:07.065	OPC KSpice_Write_2000 onBeforeWrite Transaction ID 3
71	Debug	20141031 13:29:15.148	OPC KSpice_Write_2000 onWriteComplete Transaction ID 3
86			

4.2.7 The **KSIM_XREF** table. Standard keywords

1 **Id** – Automatic numeric identifier for the record. Used internally

2 **KsimName** – Name of the model tag. SomeBlock:SomeVariable

3 **KsimNode** – SubServer (ModelServer) where tag exists. To be left blank if the tag is from the sub model defined in the Config table as a Default subserver. If the tag is from a different sub model then the name of the submodel has to be added here.

4 **ItemName** – Name of the DCS tag.

5 **ItemNode** – Required for all links that have a "Nodes" table. Give name corresponding to one of the entries in "Nodes". By disconnecting a node in Nodes, all variables associated with this node will be disabled.

6 **Itemtype** – Not required for all SimLinks. For Exatif, type of IO, Digital or Analog.

7 **ItemAttribute** – Alternate method of entering Attributes (BinaryCopy, etc). Optional

8 **ToSubserver** – When checked (-1), the data is read from the DCS to K-Spice model.

9 **Bidirectional** – When checked (-1), ToSubServer is ignored and data is read/write.

10 **Connected** – In order to test the IO it has to be connected. This is done by checking this cell. If an IO is to be temporarily disconnected or is a bad tag then one can uncheck (0) this cell.

11 **SourceItemName** – Source DCS item for writing data from a DCS tag to another tag. In which case, KsimName should be blank.

12 **Source Flag** – When checked (-1), ItemName is used as a source of data for a SourceItemName.

13 **Static Value** – This cell is used when a tag has to be tested on the DCS side but a corresponding K-Spice tag is not available. A static value can be entered in this cell to test if the value goes into the DCS. Static value is a good way to test unknown or bad DCS tags. It is a static value to send to the DCS instead of connecting to a ModelServer tag.

14 **DCS unit** - Optional. This indicates the unit in which the data is sent from the DCS to K-Spice. Note: The ModelServer does any required conversions

15 **Tolerance** – This is the minimum fractional change in the value of the IO required for the update to take place.

16 **Invert** – A Boolean signal can be inverted by checking (-1) this cell.

17 **KsimComment** – This cell is to be filled by the user doing the IO as the tags are added and tested. Text field ignored by the SimLink available for Project use

18 **TestDate**– This cell is to be filled by the user doing the IO as the tags are added and tested. Ignored by the SimLink

19 **Sign** – This cell is to be filled by the user doing the IO as the tags are added and tested. Ignored by the SimLink

20 **UpdateRate** – This is the rate at which the IO value updates will take place. Should match a RateName in the UpdateRate table. This can be changed to Fast/ Medium/Slow as needed.

21 **Length 0** – Not currently working. Used for automatic array connections.

22 Length 1 – Not currently working. Used for automatic array connections.

Note

Most of the names are stored in strings 256 characters long.

Note

Beginning with 2.10, the SimLink will now issue an error for an invalid unit and disable the DCS item.

4.2.8 The **KSIM_XREF** table. Examples

This table includes all the IO tag names.

Four important keywords:

- **KsimName** – Name of the K-Spice model tag. SomeBlock:SomeVariable
- **ItemName** – Name of the DCS tag.
- **ToSubserver** – When checked (-1), the data is read from the DCS to K-Spice model
- **Connected** – In order to test the IO it has to be connected. This is done by checking this cell. If an IO is to be temporarily disconnected or is a bad tag then one can uncheck (0) this cell.

See the section *The **KSIM_XREF** table. Standard keywords* on page 69 for a detailed description of the columns in the KSIM_XREF table.

The table KSIM_XREF is the implementation of the Cross Reference List in the Access database. *Creation of the Cross Reference List is outside scope of this DCS Link Guide.*

This section presents images of the KSIM_XREF from different projects, meant as a reference in troubleshooting situations. See also the chapter *Creation of the Cross Reference list.* on page 130

Figure 55 The KSIM_XREF table from a test of the new AbbItsCom link

KSIM_XREF | Config

Id	KsimName	KsimNode	ItemName	ItemNode	ItemType	It	ToSubServer
1	AbbRead:Value	Plant	Saw-toothed Waves.Real4	ABB1	AO		-1
4	AbbRead:ControlSignalOut	Plant	Bucket Brigade.Real4	ABB2	AI		0
8	23PT0001:MeasuredValue	Plant	Bucket Brigade.Real8	ABB1	AI		0
9	25HV0002:IsDefinedOpen	Plant	Bucket Brigade.Boolean	ABB1	DI		0
10	25HV0002:IsDefinedClosed	Plant	Bucket Brigade.Int1	ABB1	DI		0
11	25HV0002:ControlSignalIn	Plant	Bucket Brigade.UInt2	ABB1	AO		-1

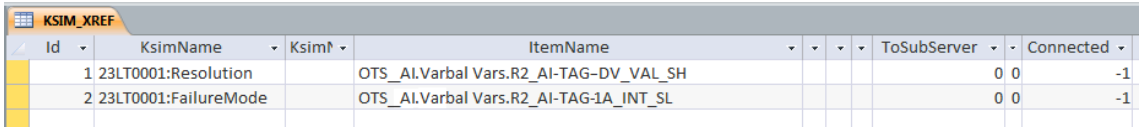
The ItemNames are from a Matrikon OPC Server set up for link test.

Figure 56 The KSIM_XREF table from a project with a OpcDaCom link

_XREF

Id	KsimName	KsimNode	ItemName	ItemNode	ToSubServer	Con
2	FT025:MeasuredValue		_In.dPo_1.ext_Sim	U1R1	☐	
3	PT021:MeasuredValue		_In.Pd_1.ext_Sim	U1R1	☐	
20	PT024:MeasuredValue		_In.Ps_1.ext_Sim	U1R1	☐	
21	PT024:MeasuredValue		_In.Ps_1.ext_Sim	U1R1	☐	
22	TT028:MeasuredValue		_In.Td_1.ext_Sim	U1R1	☐	
23	TT026:MeasuredValue		_In.Ts_1.ext_Sim	U1R1	☐	
24	ZT022:MeasuredValue		_In.Pos_1.ext_Sim	U1R1	☐	
25	ST023:MeasuredValue		_In.N.ext_Sim	U1R1	☐	

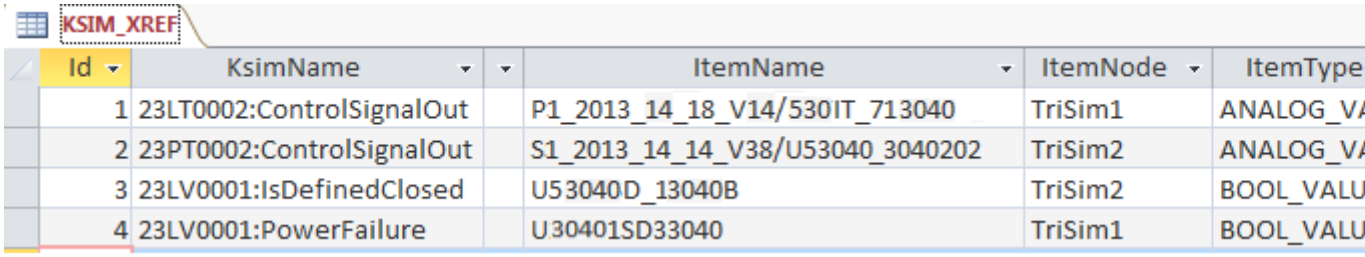
Figure 57 The KSIM_XREF table from a project with a HimaCom link



KSIM_XREF

Id	KsimName	KsimN	ItemName				ToSubServer		Connected
1	23LT0001:Resolution		OTS_AI.Varbal Vars.R2_AI-TAG-DV_VAL_SH				0	0	-1
2	23LT0001:FailureMode		OTS_AI.Varbal Vars.R2_AI-TAG-1A_INT_SL				0	0	-1

Figure 58 The KSIM_XREF table from a project with a Sim4MeCom link



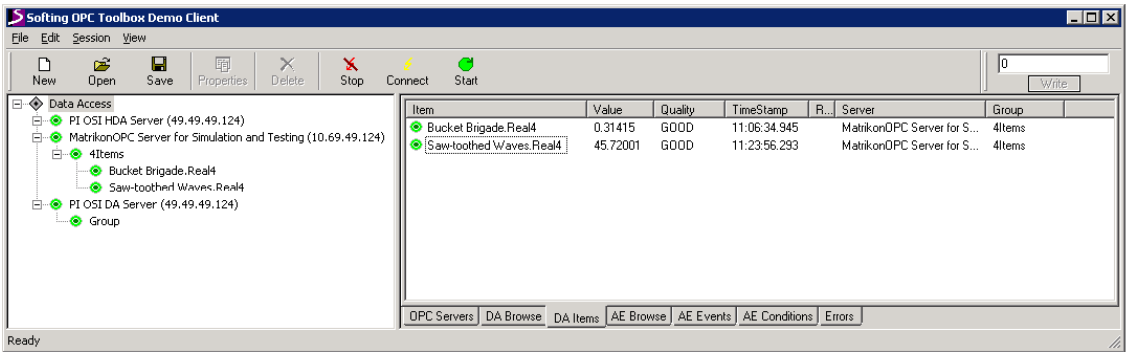
KSIM_XREF

Id	KsimName		ItemName	ItemNode	ItemType
1	23LT0002:ControlSignalOut		P1_2013_14_18_V14/530IT_713040	TriSim1	ANALOG_VA
2	23PT0002:ControlSignalOut		S1_2013_14_14_V38/U53040_3040202	TriSim2	ANALOG_VA
3	23LV0001:IsDefinedClosed		U53040D_13040B	TriSim2	BOOL_VALU
4	23LV0001:PowerFailure		U30401SD33040	TriSim1	BOOL_VALU

Please note that images from real interfaces are manipulated. It has been an issue to protect the data structure in real systems from exposure in this guide. If there are inconsistencies between tags in an image and tags in the text, this is probably not a result of inconsistencies in the real system. This is probably the result of poor text and image manipulation.

4.2.9 The **KSIM_XREF** table. An example: Two data items

Figure 59 Two data items in the SOClient



The ItemNames are from a Matrikon OPC Server used for link test. Saw-toothed Waves and Bucket Brigade are default items in the Matrikon Server. The items will not need any modification from the user's side. The items are there, and have values, when you start up any Matrikon Server.

Figure 60 Two data items defined in KSIM_XREF

Config | KSIM_XREF | UpdateRate

Id	KsimName	KsimN	ItemName	I	I	I	ToSubServer
4	AbbRead:Value	Plant	Saw-toothed Waves.Real4				-1
5	AbbRead:ControlSignalOut	Plant	Bucket Brigade.Real4				0

Figure 61 Two data items connected

```
20140411 10:51:28.192.  DcsInfo              Connecting items.
20140411 10:51:28.192.  ConfigurationDetails Tag connected. Status: OK. Type: TYPE_DOUBLE -- Plant/AbbRead:ControlSignalOut#-#
20140411 10:51:28.192.  ConfigurationInfo    Created model subscriptions /OTS_ML/Plant:1427199253/130764864 with 1 item Responses
20140411 10:51:28.192.  ConfigurationDetails Tag connected. Status: OK. Type: TYPE_DOUBLE -- Plant/AbbRead:Value#-#
20140411 10:51:28.192.  ConfigurationInfo    Created model subscriptions /OTS_ML/Plant:1427199252/130759944 with 1 item Responses
20140411 10:51:28.192.  ConfigurationInfo    Send create model subscriptions /OTS_ML/Plant:1427199253.
20140411 10:51:28.192.  ConfigurationInfo    Send create model subscriptions /OTS_ML/Plant:1427199252.
20140411 10:51:28.192.  ConfigurationDetails Connecting ksim(Plant/AbbRead:ControlSignalOut#-#)->dcs(Bucket Brigade.Real4)
20140411 10:51:28.192.  ConfigurationDetails Connecting dcs(Saw-toothed Waves.Real4)->ksim(Plant/AbbRead:Value#-#)
20140411 10:51:28.176.  ConfigurationInfo    Using default SubServer of >Plant< from Single SubServer
20140411 10:51:28.176.  DcsInfo              OPC connected to opcda://49.49.49.124/Matrikon.OPC.Simulation/{f8582cf2-88fb-11d0-b850-00c0f0104305}
```

From the SimLink GUI log.

Figure 62 Two data items in the SimLink GUI

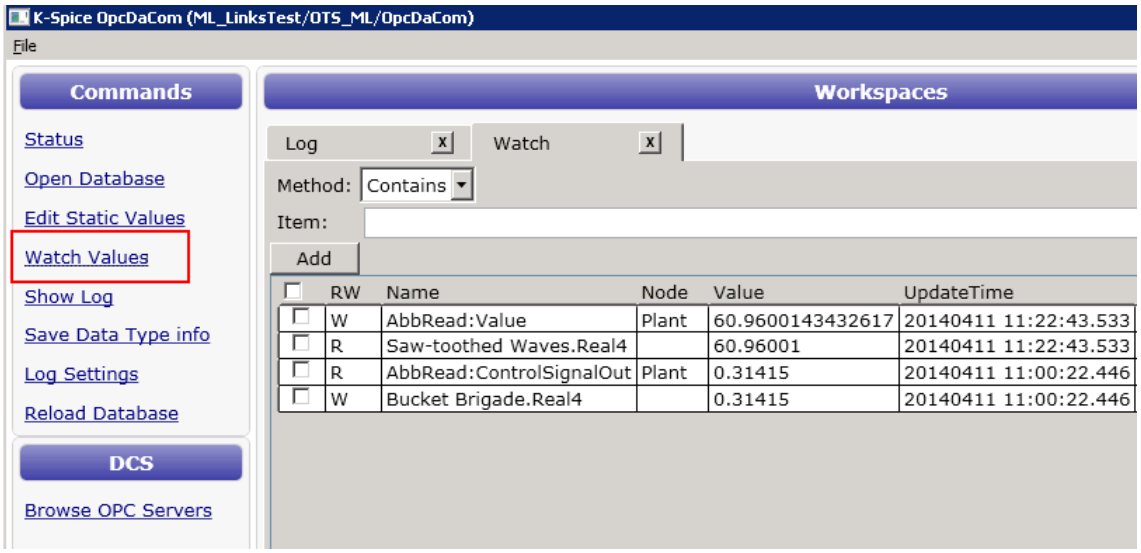
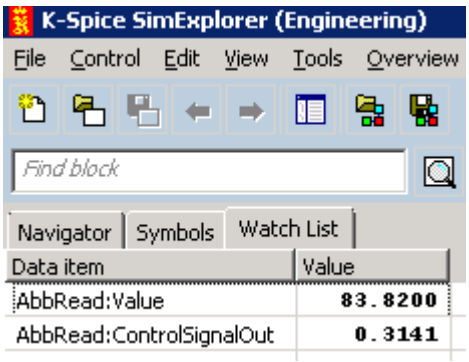


Figure 63 Finally. Two data items in K-Spice watch list



4.2.10 The **SimLink_XREF** table.

The SimLink_XREF table can be left blank for OPC links.

For SimLink_XREF table is used in link to link configuration. See *Configuring SimLink.NET to SimLink.NET* on page 81

4.2.11 The **SubServers** table.

The SubServers table.

The SubServers table must contain the name of the application containing the K-Spice model. In the example below the simulation application is called ProcessModel, part of the Engineering Timeline of K-Spice.

Figure 64 The SubServers table. An example

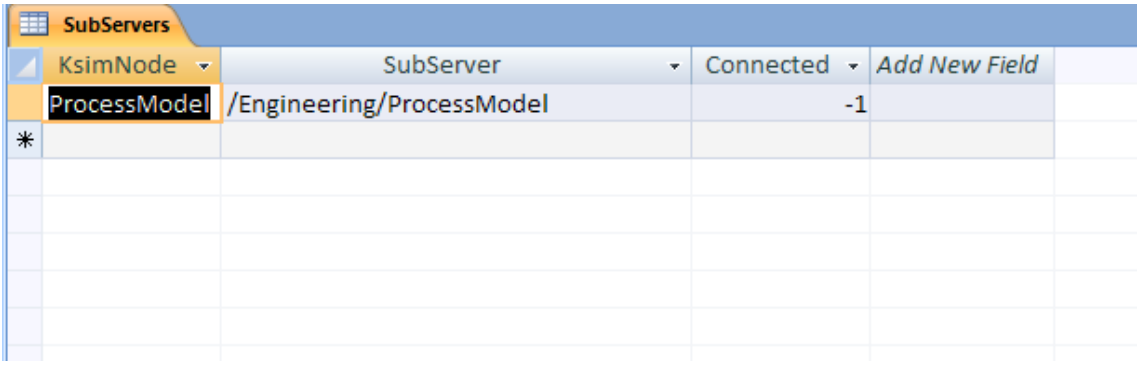
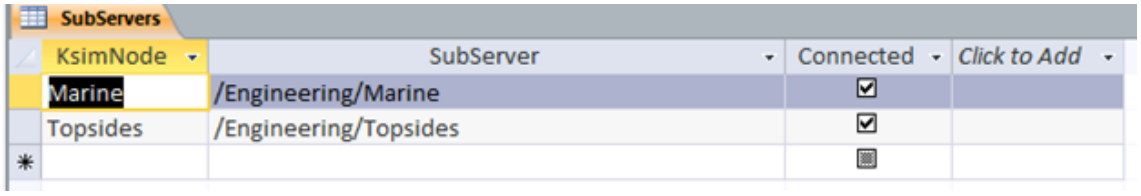
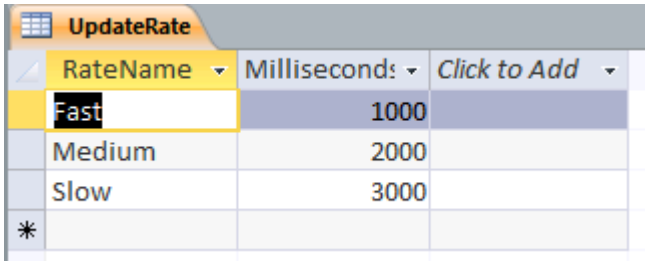


Figure 65 The SubServers table. Another example



4.2.12 The **UpdateRate** table

Figure 66 The UpdateRate table



The keyword **UpdateRate** in the KSIM_XREF table can be modified using this table.

Note

*As and when the IO tags/static tags are modified in the Access database, the database has to be reloaded into the ExatifCom link in order to do the IO connections between the K-Spice model and Yokogawa DCS. This is done using the **Reload database** tab in the SimLink GUI.*

4.2.13 The **WatchdogItems** table.

2.9.3.0 of SimLinks implemented a watchdog feature for DCS items. The user configures DCS tags that should be monitored by the link. If those tags don't change values for **TimeoutMilliseconds** while the model is running, it will treated as a watchdog expiration. **ComponentStatus.WatchdogAnyExpired** will be set to true if any configured watchdog is expired. Specific links may also have built-in watchdogs (`SoftLogixCom`, for example). **ComponentStatus.WatchdogAnyExpired** can change back to true from false if all values have changed within their timeout period.

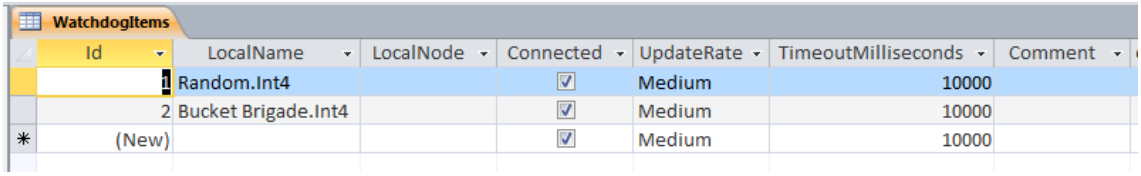
Watchdog items are configured in the database table WatchdogItems. Currently (2.9.3.0), elements of arrays are not supported.

Table 5 Configuration of Watchdog items

Id	Automatic row number
LocalName	Name of the DCS tag
LocalNode	Not required for all SimLinks. DCS node where tag exists. The PLC in SoftLogixCom, for example.
Connected	When unchecked (0), record is ignored.
UpdateRate	Update rate of item. Should match a RateName in the UpdateRate table.
TimeoutMilliseconds	Maximum time in milliseconds the value can be unchanged before a watchdog expiration. Should be at least twice as long as UpdateRate.
Comment	Text field ignored by the SimLink available for Project use.

This is a sample configured used for testing with OpcDaCom and Matrikon OPC Server for Simulation. Random.Int4 changes frequently. Bucket Brigade.Int4 does not automatically change, so it was easy to see WatchdogAnyExpired toggling between false and true

Figure 67 Sample of Watchdog Items



4.3 Configuring SimLink.NET to SimLink.NET

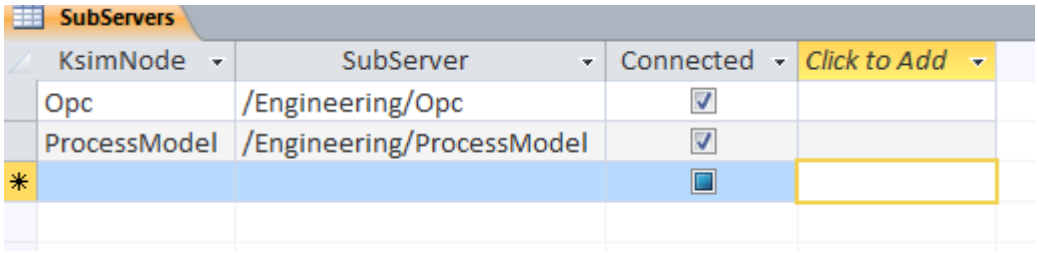
This is a section in the chapter The Access database that describes the configuration necessary for sending data in one DCS link directly to another DCS link. Several Access database tables are involved. The most essential here is the table SimLink_XREF.

4.3.1 Configure SubServers Table

In order to configure a connection from one SimLink.NET to another SimLink.NET, you must configure the target SimLink in the SubServers database.

Below is a CccOpcCom application configured to connect to an OpcDaCom application named Opc.

Figure 68 The Subservers table. Link to link.

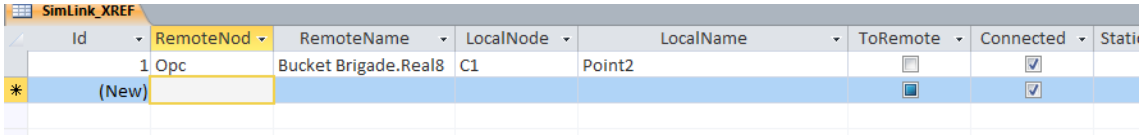


4.3.2 Configure Item Connections in SimLink_XREF

The SimLink_XREF table is similar to KSIM_XREF table. You should configure the SimLink_XREF table for tag in a SimLink that are going to receive data from the remote SimLink.

Below is a CCC example that configures C1\Point2 to receive data from Opc\Bucket Brigade.Real8.

Figure 69 The SimLink_XREF Table

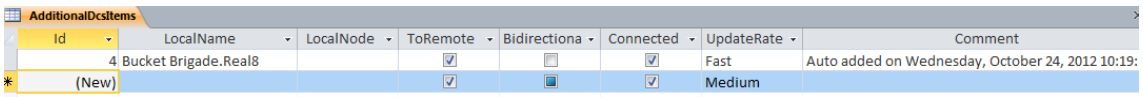


4.3.3 Activate SimLink TWICE for new connections

The first time a SimLink tries to subscribe to a remote data point, the connection may fail because the the remote SimLink may not know to make the point available. However, the remote SimLink will save the information so it can create the point the next time it starts. This information is saved in the table AdditionalDcsItems.

Below is an entry from the Opc application after the first connection attempt to Bucket Brigade.Real8.

Figure 70 The table AdditionalDcsItems



AdditionalDcsItems

Id	LocalName	LocalNode	ToRemote	Bidirectiona	Connected	UpdateRate	Comment
4	Bucket Brigade.Real8		☑	☐	☑	Fast	Auto added on Wednesday, October 24, 2012 10:19:
(New)			☑	■	☑	Medium	

4.3.4 Configuring unit conversion for link-to-link

Beginning with version 2.10.0.0, link-to-Link can do unit conversions. In order to do the conversions, you must configure the unit category and the item units. The SimLink_XREF table has 3 fields for this configuration.

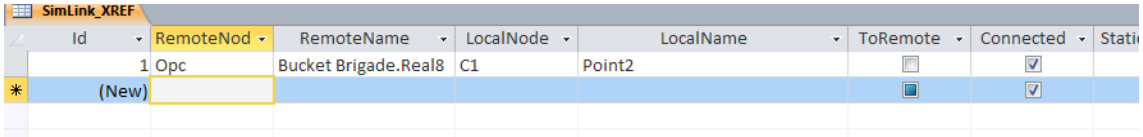
Conversion will not occur if:

- Invert is set
- Any of the unit configuration fields are blank

Table 6 Unit keywords

Field	Purpose
UnitCategory	The Unit Category for the items Example: temperature
LocalUnit	The unit of the local item Example: C
RemoteUnit	The unit of the remote item Example: F

Figure 71 Unit Keywords. Example



SimLink_XREF

Id	RemoteNod	RemoteName	LocalNode	LocalName	ToRemote	Connected	Stati
1	Opc	Bucket Brigade.Real8	C1	Point2	☐	☑	
(New)					■	☑	

5 Softing and Matrikon OPC test tools

These are two very useful test tools. The Matrikon test tools can be downloaded free from internet.

Softing test tool

Softing's SOClient is used as a OPC *client*. The program reads data from any OPC sever. The Softing OPC client reads the same data K-Spice reads. If the **SOClient** can read them, it is a good reason to believe that K-Spice can read them.

Matrikon test tool

MatrikonOPC is used to set up a test OPC *server*.

The **MatrikonOPC** test server can be set up with tags with in same format as tags in the actual DCS. The tags in the server also have values. If K-Spice can exchange values the Matrikon server, it is a good indication that K-Spice can read data from and write data to the DCS server.

There is also a program called **MatrikonOPC Explorer** that reads OPC data. The MatrikonOPC Explorer and the SOClient are two programs that do the same thing.

The company MatrikonOPC delivers OPC servers for automation vendors. From their website you can download free applications for test of OPC communication.

- MatrikonOPC Simulation Server
- MatrikonOPC Explorer
- MatrikonOPC HDA Explorer

URLs to the Matrikon download pages:

If you download the MatriconOPC Simulation Server, it is not necessary to download the Explorer. The Explorer is part of the Simulation Server download

`www.matrikonopc.com/products/opc-drivers/opc-simulation-server.aspx`

`www.matrikonopc.com/products/opc-desktop-tools/opc-explorer.aspx`

`www.matrikonopc.com/products/opc-desktop-tools/opc-hda-explorer.aspx`

Note

Softing's SOClient.exe is not available as free download. Ask a friend for a copy.

5.1 The Softing OPC client tool

Softing OPC Toolbox has an application to set up an *OPC test* ***client.*** The application is called **SOClient.exe.**

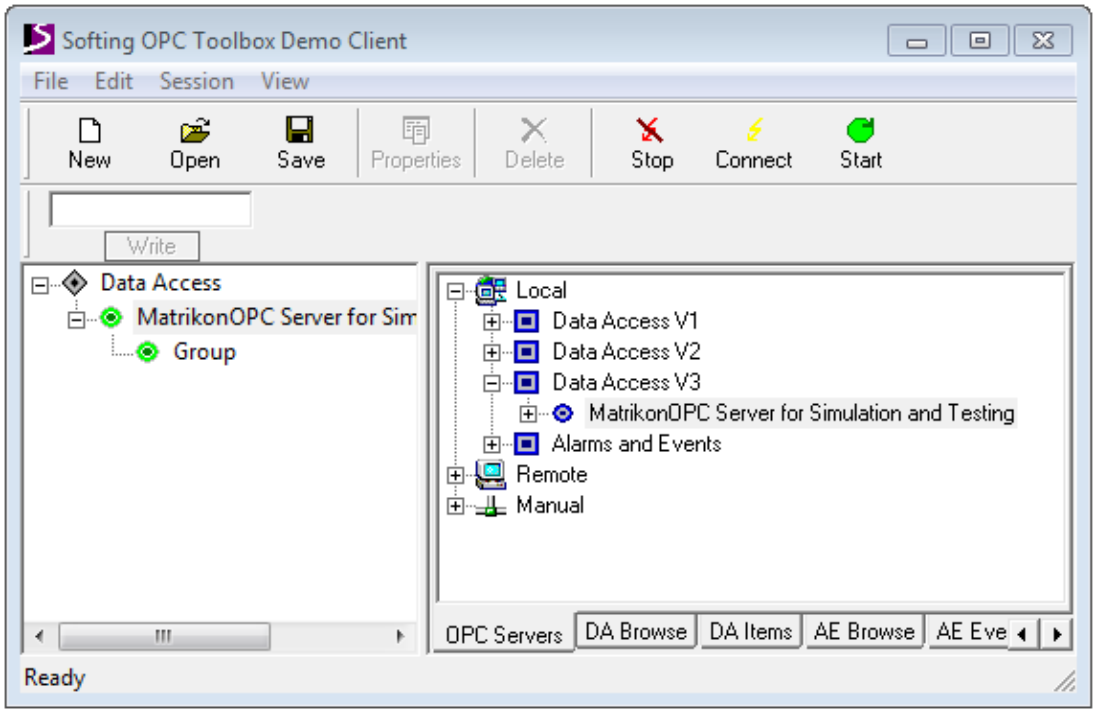
The **SOClient** can be used to read OPC values and write OPC values. The same values that K-Spice is set up to read and write. The application will be a test program, *in parallel to* K-Spice. It can be used in the initial phase to check if the Model PC, and then K-Spice, has access to the DCS values.

Setup for access to data in a Matrikon Server on the local PC.

- Run SOClient.exe
- In the SOClient Main Window

  Under tab **OPC Servers** Tree root: **Local → Data Access V3 (this is OPC format version 3) →MatrikonOPC Server for Simulation and Testing**
- Right hand click on **MatrikonOPC Server for Simulation and Testing→ Add Server**
- Verify that a default subscription opens in the left window. The default subscription is called **Group.**

Figure 72 Softing OPC Toolbox Demo Client



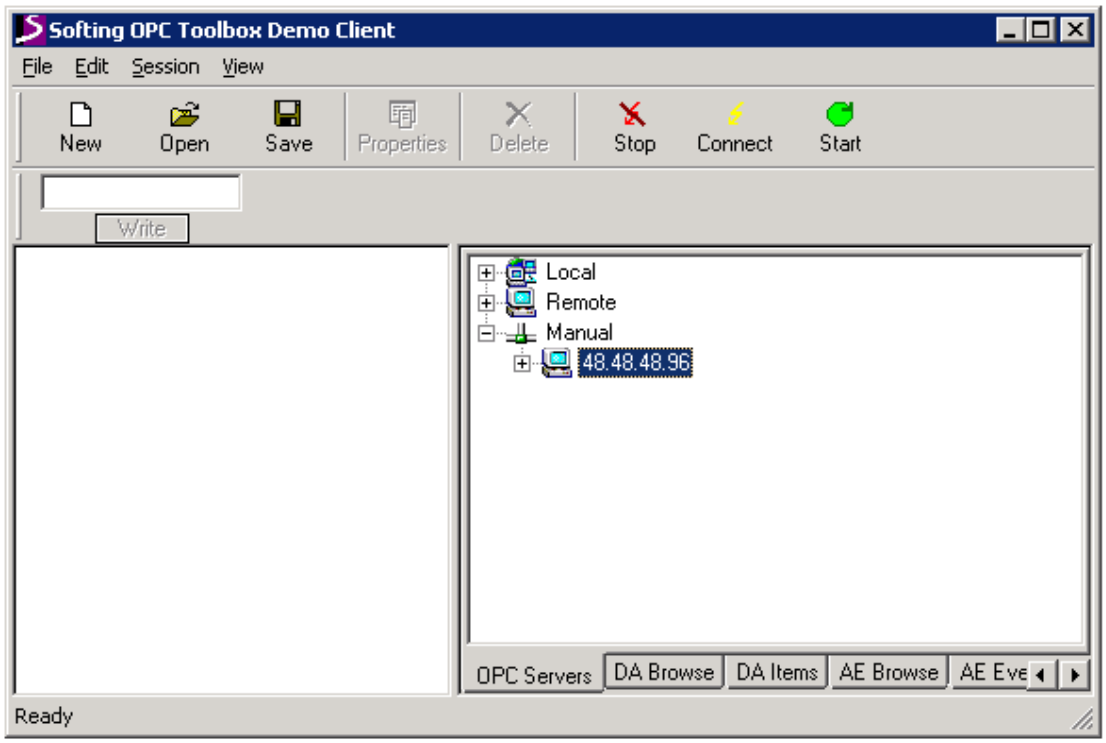
- Open tab **DA Browse**
- Right hand click on **MatrikonOPC Server for Simulation and Testing →Add Items for All tags**
- Verify that the data subscription called **Group** is filled with all the Default tags in the Matrikon Server.

- Open tab **DA Items** Both windows are filled with tags. See the figure below.

Setup for access to data in a DeltaV Server on another machine.

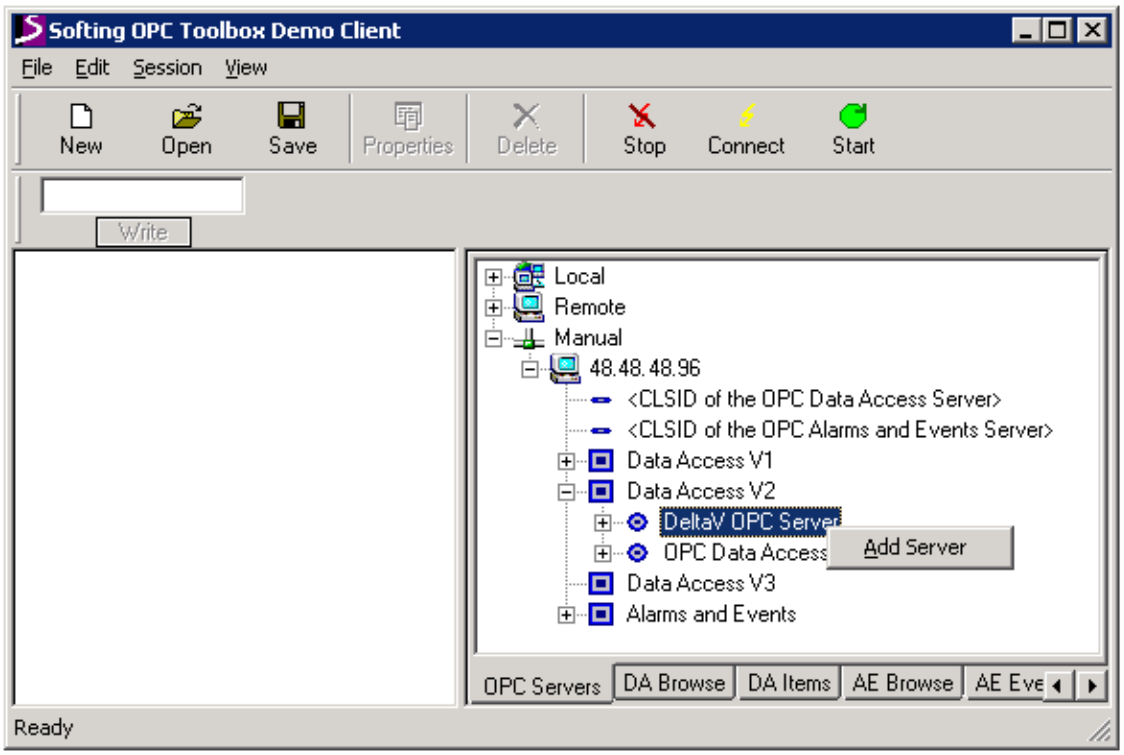
- Run SOClient.exe
- In the SOClient main window→ **Manual** Type in the IP address of the DeltaV machine. It can be tricky to write the address, click on the actual field more than once, to get into write mode for a text string.

Figure 73 Find an OPC Server with the IP address



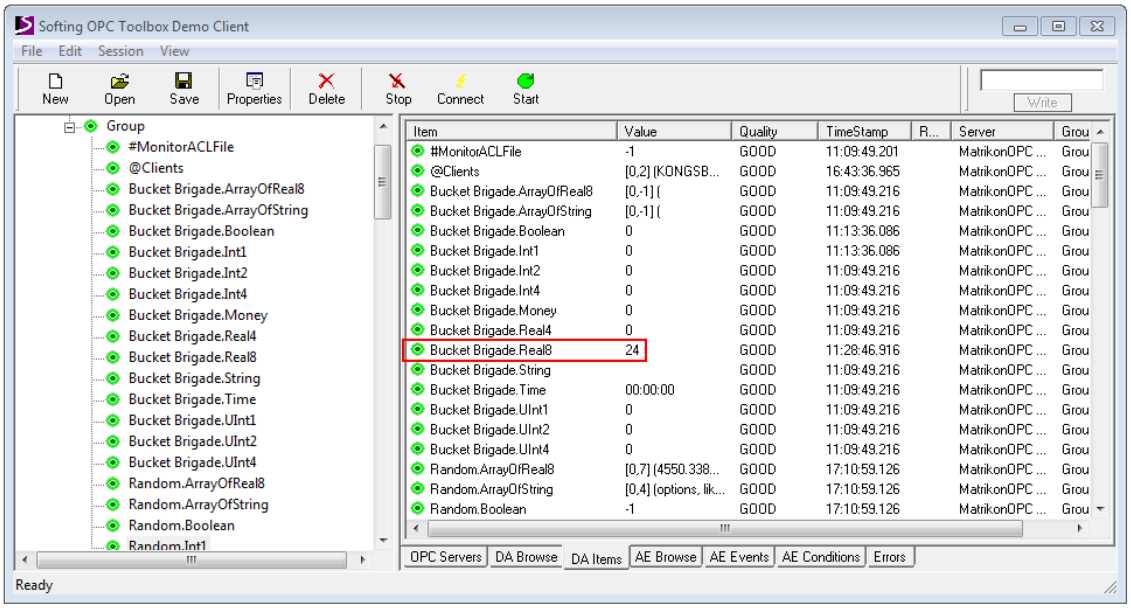
Open the tree for servers and data in servers on the OPC Server machine. Right hand click on the name on the machine→ **Add Server** to add it to list of servers in the left field of the main window.

Figure 74 Add the OPC Server to the list of servers



Check a value in the OPC Server

Figure 75 Transmitter value 24



Compare the value in the SOClient with the value in K-Spice SimExplorer

The setup is this:

- The DemoProject is opened in K-Spice
- The MatrikonOPC Server is started. No dataitem is added to the Server. The Server has only the default tags, i.e **Bucket Brigade.Real8**
- The link application **AbbItsCom** is added to the DemoProject, under the timeline **OTS_ABB**.
- A Cross Reference is entered in the Access database that is part of the AbbItsCom link

  The Cross Reference is between tag **Bucket Brigade.Real8** in the Matrikon Server and the data item **23PT0001:MeasuredValue** in the K-Spice model.
  Figure 76 The table KSIM_XREF in the Access database for AbbItsCom

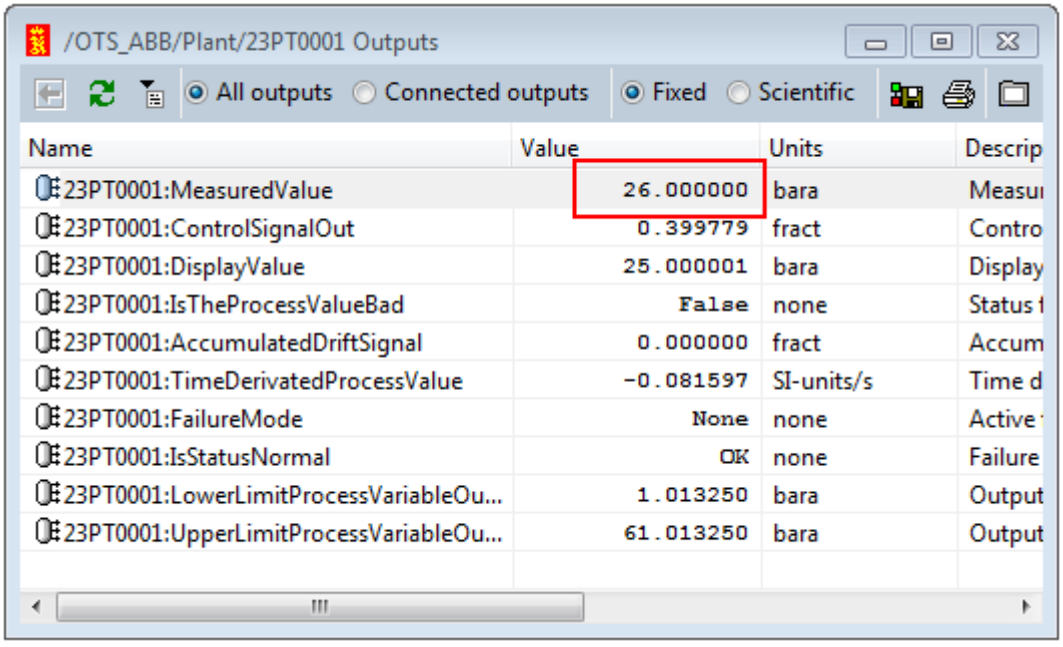
KSIM_XREF

Id	KsimName	KsimNode	ItemName	ItemTy	ToS	Bi	Con	Sou	DcsUr	Tole	In	Updatef
5	PI_602602_meas:Value	RealTime	PI-602-602.PV	Analog	-1	0	-1	0	barg	0.005	0	Medium
6	PI_602604_meas:Value	RealTime	PI-602-604.PV	Analog	-1	0	-1	0	barg	0.005	0	Medium
*	####						-1			0.005		Medium

Modify the K-Spice value

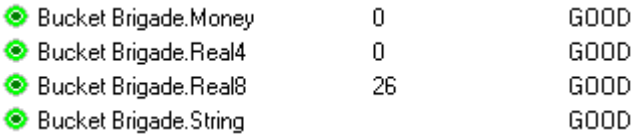
Set a new value in K-Spice, and verify that the value in SOClient, is updated.

Figure 77 New K-Spice transmitter value



The SoClient has a subscription to the tag **Bucket Brigade.Real8** in the Matrikon Server. **Bucket Brigade.Real8** is now **26.**

Figure 78 Updated value in Softing OPC Toolbox Demo Client



5.2 Error messages in Softing OPC client tool

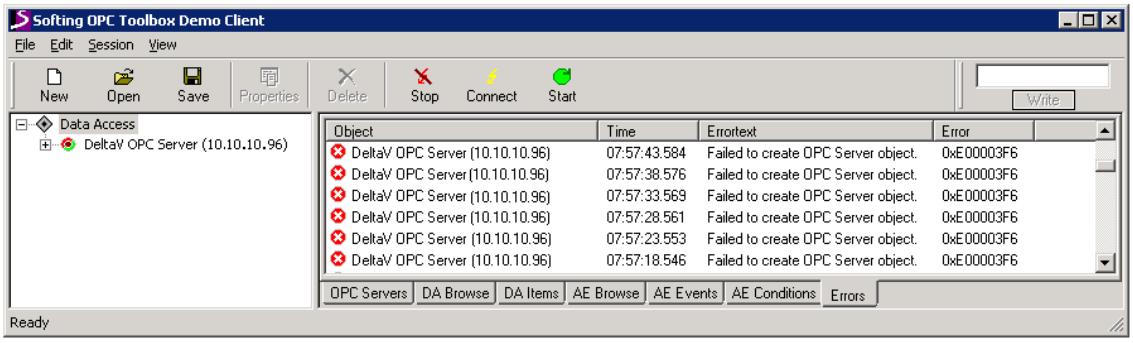
About OPCEnum

The OPC Enumerator is a browser for local or remote OPC server applications. The OPC Foundation is responsible for the application OpcEnum.exe, The application is part of resdistributable packages included in most SCADA software. This application is required by OPC Servers/Clients to list down the OPC servers installed on a computer. The application will be installed on a machine as part of the installation of any of the K-Spice OPC SimLinks.

The DCOM security is not opened up on a DeltaV Server.

There can be situations where the SOClient and the K-Spice SimLink can see an OPC Server on another machine, but no data can be transferred. The OpcEnum properties on the OPC Server machine must be opened up for read access. When an OPC Client application connects to a remote computer and attempts to browse for OPC Servers, it is actually connecting to OpcEnum.exe on the remote PC. OpcEnum retrieves the list of OPC Servers on the computer on which it resides. The inability to connect to OpcEnum is typically a result of authentication failure. See *Modify the DCOM configuration* on page 122

Figure 79 Failed to create message in Softing tool

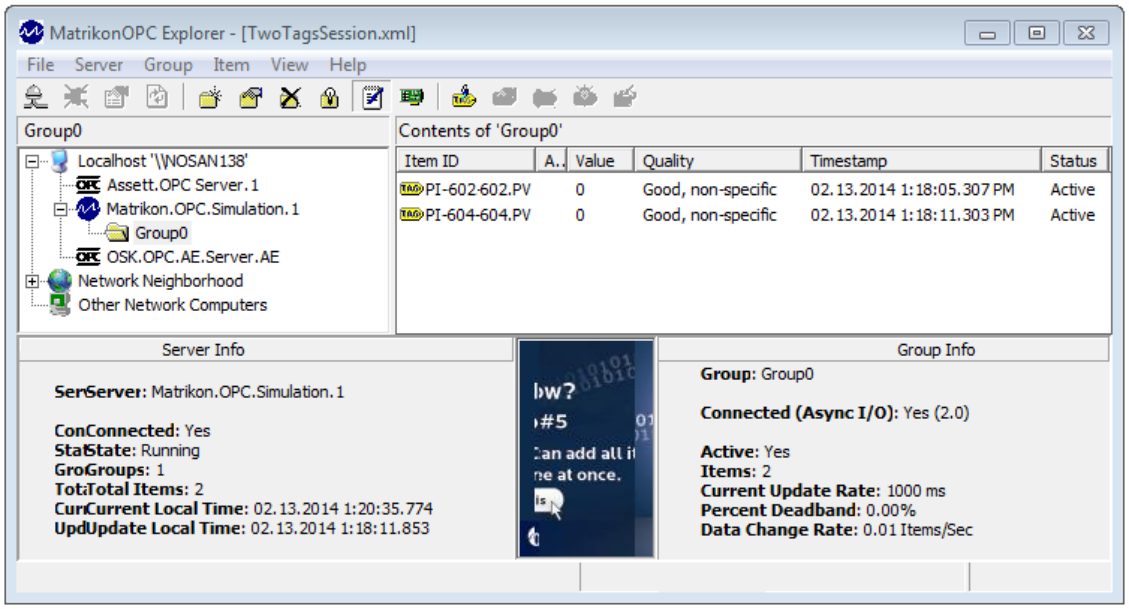


5.3 The MatrikonOPC Server

The Matrikon test tool consists of the **MatrikonOPC Simulation Server** and the **MatrikonOPC Explorer.** The **Matrikon OPCServer** ***generates simulated DCS values*** in OPC format, and is the most important of the two applications. The **MatrikonOPC Explorer** reads OPC data. The **Softing OPC Client tool** and the **MatrikonOPC Explorer** do the same thing.

See *Softing and Matrikon OPC test tools* on page 83 for URLs to the download pages.

Figure 80 The MatrikonOPC Server



The **MatrikonOPC Server** and K-Spice will normally run on the same desktop machine. There can also be situations where the DCS vendor has installed a **MatrikonOPC Server** on the DCS machine for test purposes.

The **MatrikonOPC Server** is used to verify that the DCOM settings on the Model PC are correct. The test will be if K-Spice can read and write OPC values in the **MatrikonOPC Server.** The application will be a test program, *in parallel to* the DCS OPC server. If the link can access the Matrikon values, it is an indication that it also can also access the DCS values.

The MatrikonOPC Server has two different types of data. In the next section is described how the server can be set up to send two OPC values *in Yokogawa tag format* to K-Spice. The other type of data is *a default set of tags*. You will find the default set of tags on any MatrikonOPC Server. The default tag set has characteristic names, `Random.Int4, Bucket Brigade.Real8, Bucket Brigade.Int4 Saw-toothed Waves.Real4`

Note

If you have a MatrikonOPC Server installed on your machine, you can experience the that Server starts automatically. If a link asks for data, the Server will start.

This will be the situation where the default settings of a link, I.E. the HimaCom link, has Matrikon as the default OpcServer. If you do not modify the the OpcServer in the Config table when you first create the Access database, and then activate the timeline, the MatrikonOPC Server will start. The HimaCom link will connect to the running MatrikonOPC Server, and the link will connect to 4 default Matrikon tags. At link startup, 4 default static values will be written from the HimaCom link to the Server.

5.4 Two tags in the MatrikonOPC Server

Start the server

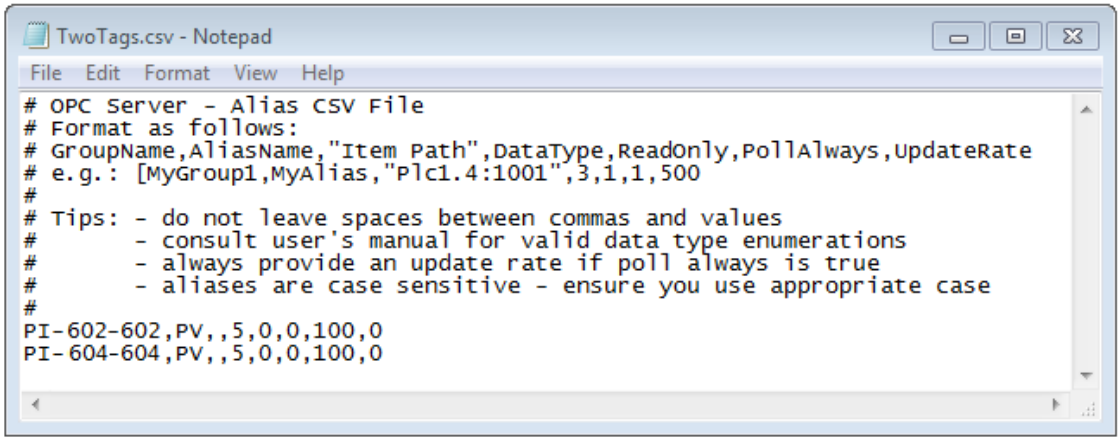
Start the **MatrikonOPC Simulation Server** from the Windows **Start** menu →**MatrikonOPC Server for Simulation**

Import Aliases

The test server will typically be configured to send a whole array of values. The starting point will then be an Excel sheet with all the tags in the actual Yokogawa server. To simplify the documentation, we will only handle two tags here. The starting point is the file **TwoTags.csv**, a comma separated file, with two tags in it. A comma separated file can easily be exported from an Excel sheet.

DataType **5** is double. DataType **11** is boolean.

Figure 81 TwoTags.csv



In the the **MatrikonOPC Server** main window **File →Import Aliases TwoTags.csv**

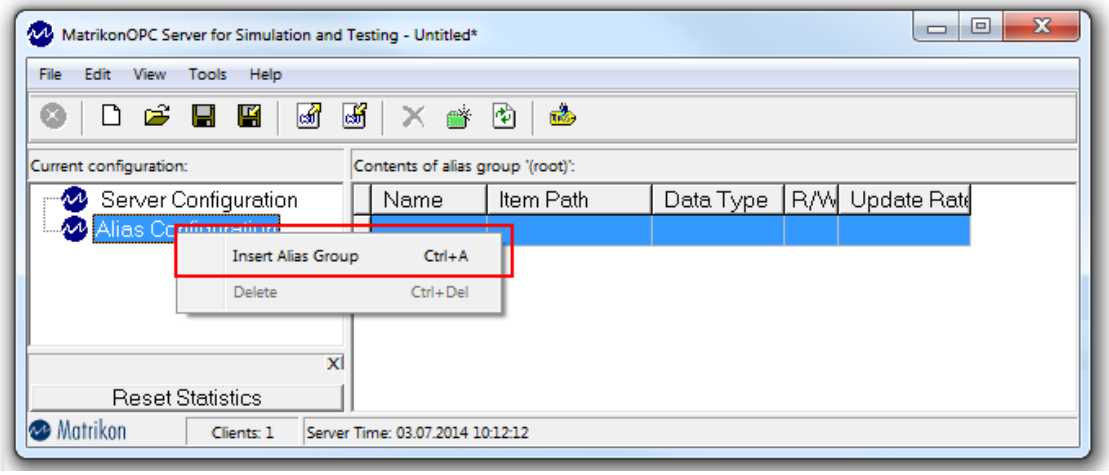
Save the Server configuration as **TwoTags.xml**

Create Aliases manually

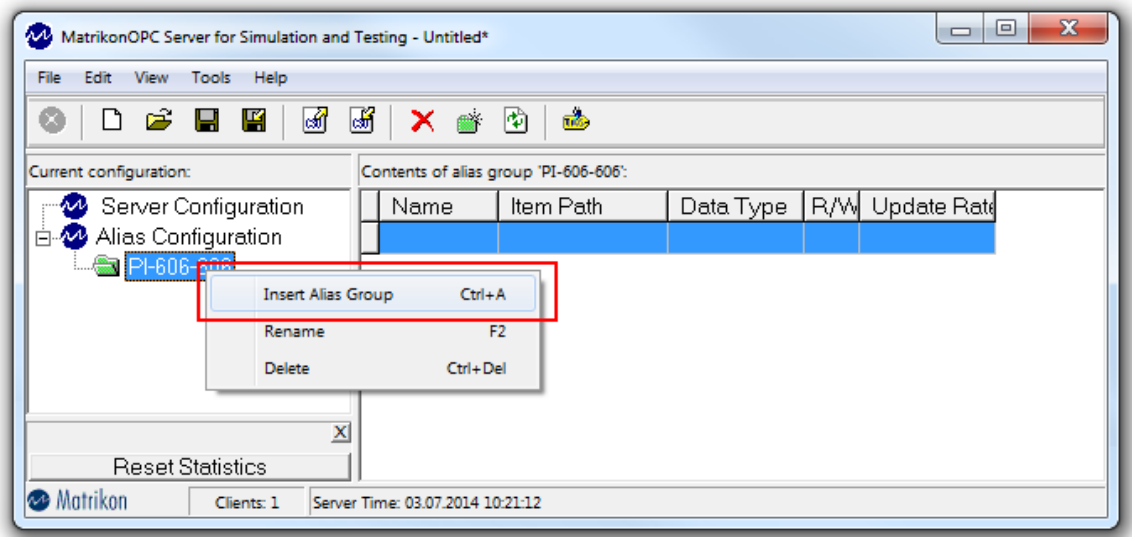
One alternative is to import a comma separated file, the other alternative is to configure the MatrikonOPC Server manually. With manual configuration the starting point is an empty MatrikonOPC Server.

In the pane on the left hand side, called **Current configuration,** right hand click on menu item **Alias Configuration**. Select **Insert Alias Group**.

Figure 82 Insert Alias Group



Rename the new alias group to **PI-602–602.** Right hand click on the new menu item **PI-602–602**. Select **Insert Alias Group**

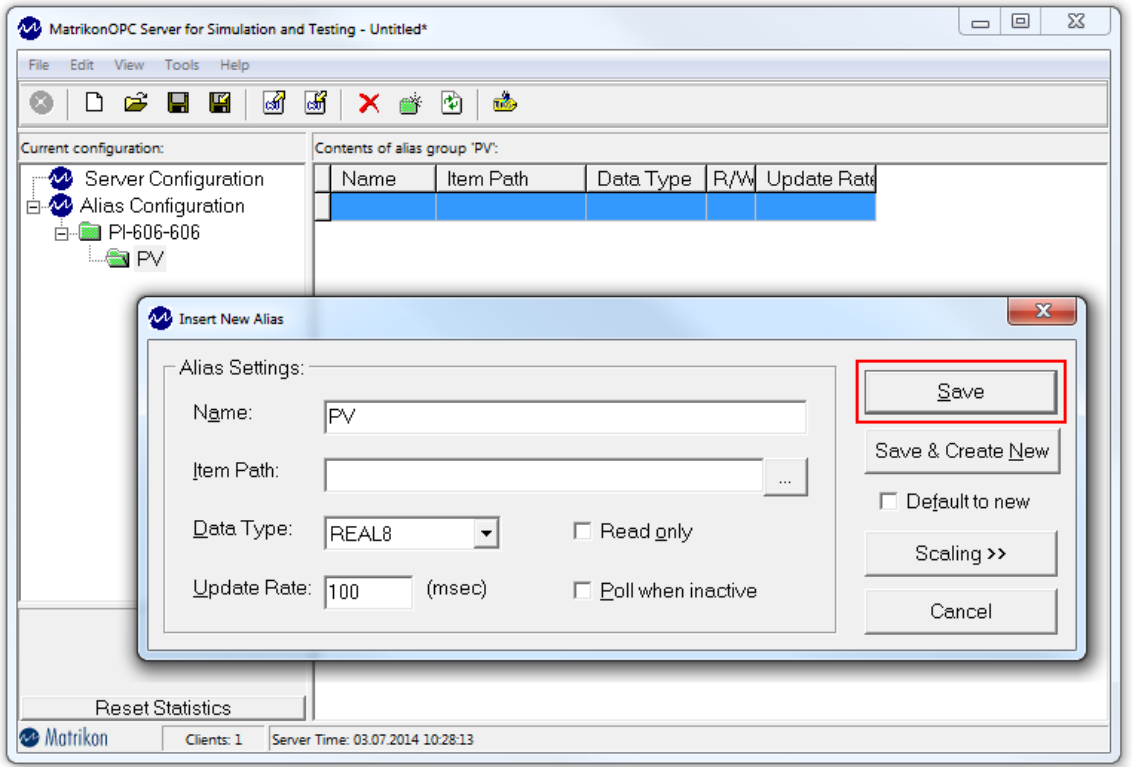


Rename the new alias subgroup to **PV.**

Double click on the new menu item **PV**. In pane on the right hand side, called **Contents of alias group 'PV',** double click on the empty line. The dialog **Insert New Alias** pops up.

In the dialog **Insert New Alias** fill in Name: **PV** and Data Type: **REAL8** and Update Rate: **100**. *Save* the settings.

Figure 83



Configure the second alias **PI-604–604** in the same way.

Save the Server configuration as **TwoTags.xml**

Import Aliases vs. Create Aliases manually.

There is no difference between the two ways of setting up the configuration for the MatrikonOPC Server. The result is the same. The files **TwoTags.xml** created with a comma seaparated file, and **TwoTags.xml** created manually, are identical.

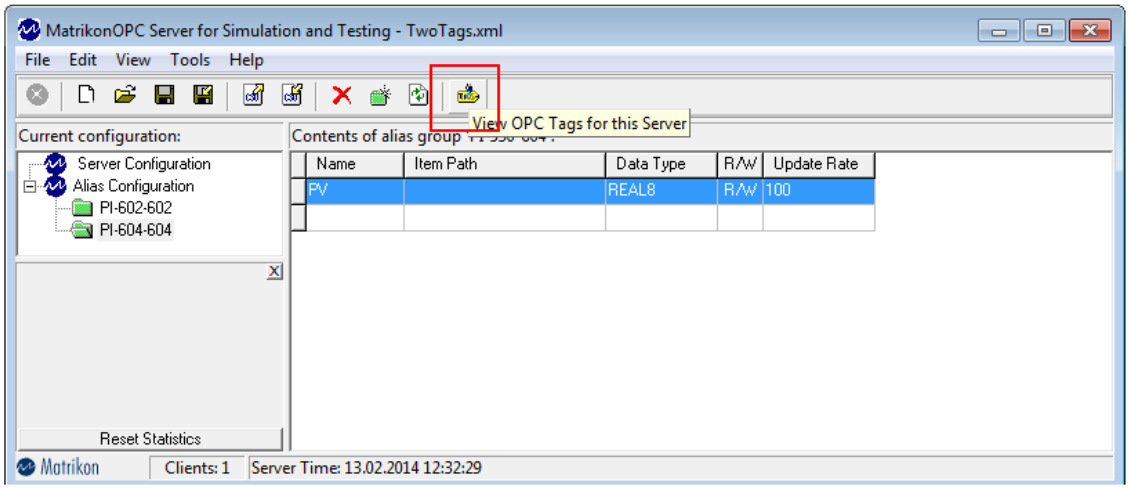
Figure 84 The file TwoTags.xml saved from MatrikonOPC Server

```
<?xml version="1.0"?>
- <Matrikon.OPC.Simulation>
    <CSimRootDevLink description="Sim Server Root"/>
  - <PSTAliasGroup>
    - <PSTAliasGroup name="PI-602-602">
        <PSTAlias name="PV" updateRate="100" dataType="5" itemPath=""/>
      </PSTAliasGroup>
    - <PSTAliasGroup name="PI-604-604">
        <PSTAlias name="PV" updateRate="100" dataType="5" itemPath=""/>
      </PSTAliasGroup>
    </PSTAliasGroup>
</Matrikon.OPC.Simulation>
```

The MatrikonOPC Explorer

In the **MatrikonOPC Server** main window click the button **View OPC Tags for this Server**.

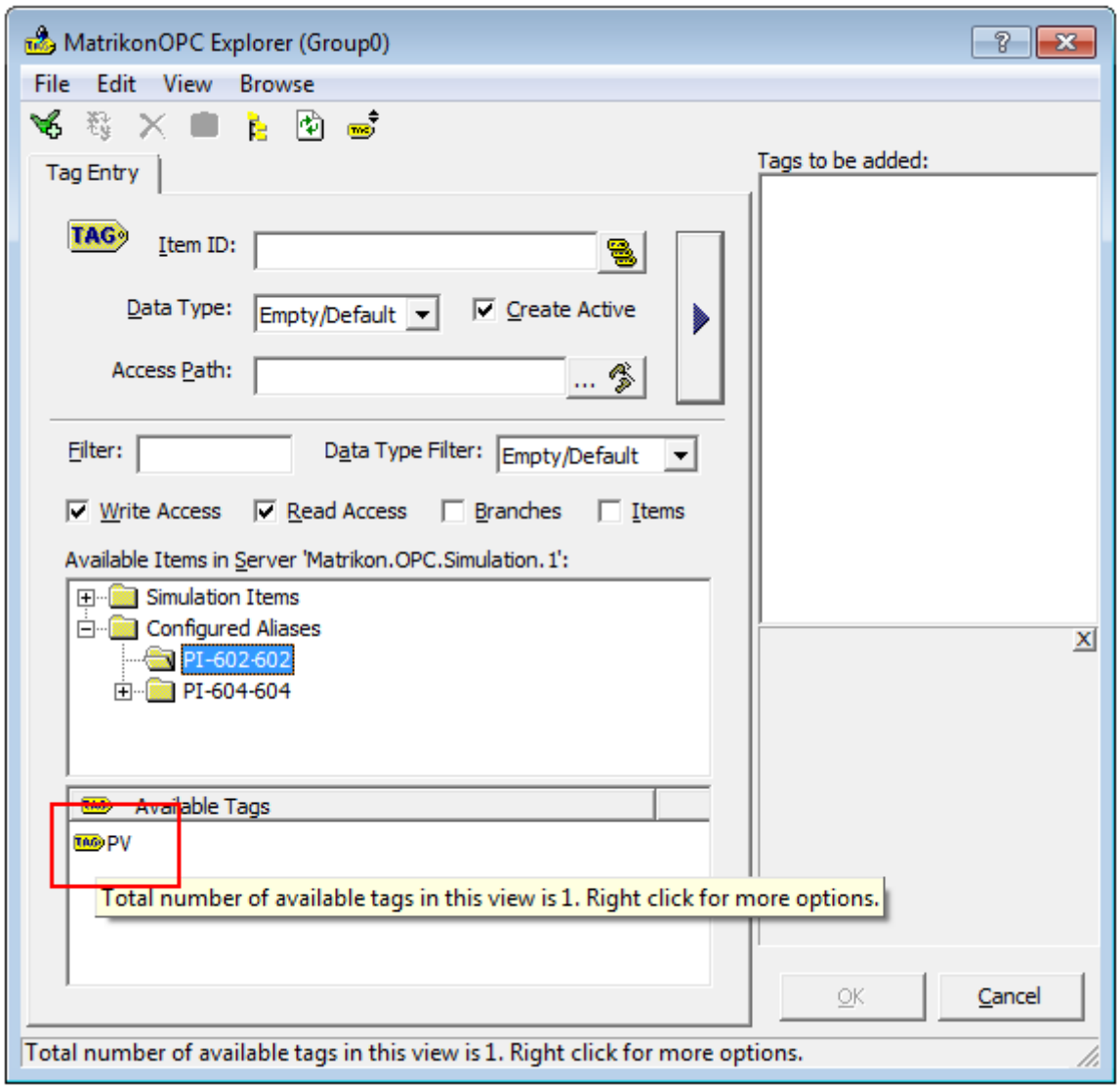
Figure 85 Start the MatrikonOPC Explorer from the Server



In dialog **MatrikonOPC Explorer (Group 0)** add tags by clicking on **PI-602–602** in the window **Available Items**

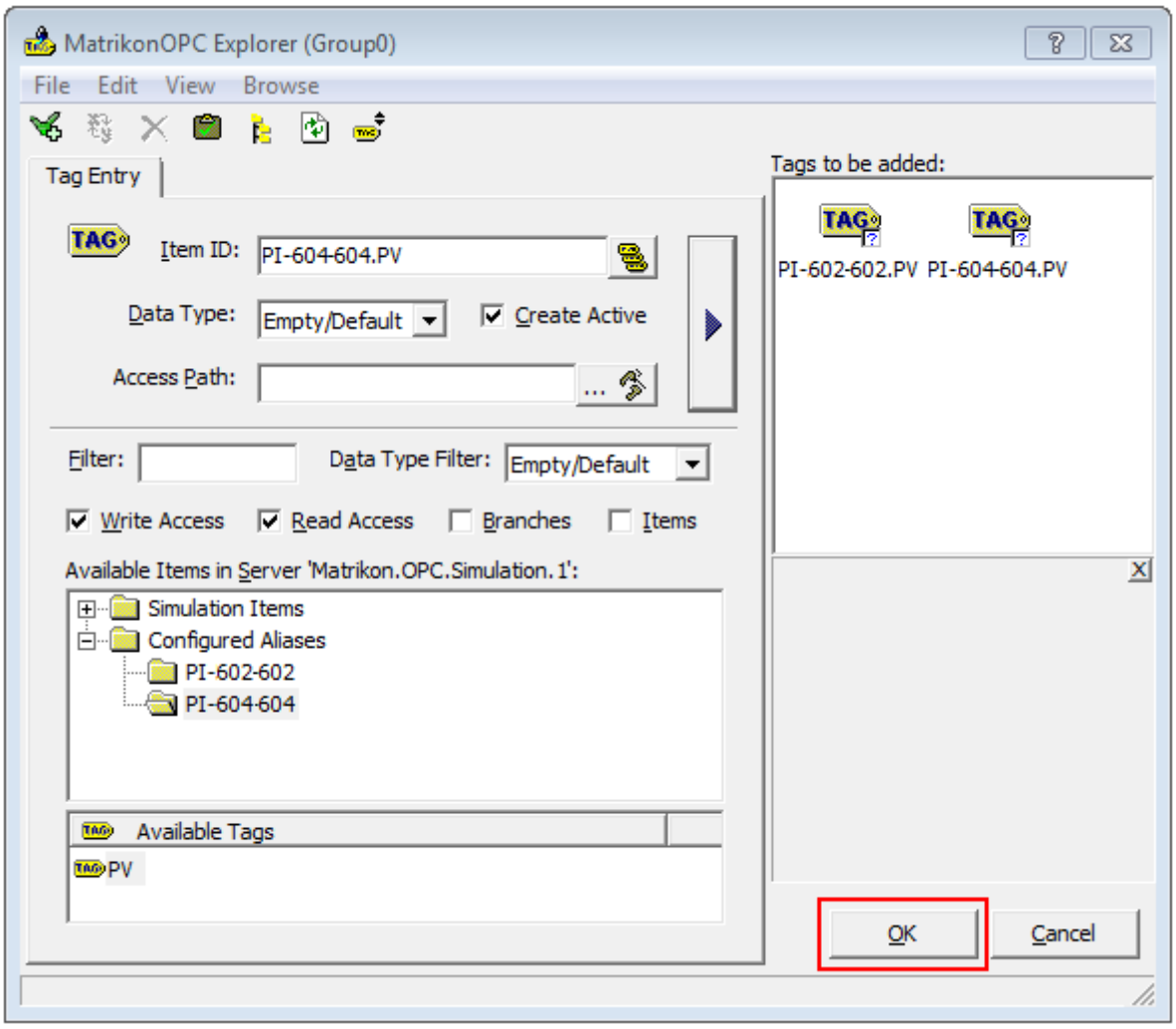
In the window **Available Tags** double click on **PV.**

Figure 86 Available tags in the Explorer



Verify that **PI-602–602.PV** is written in the window **Tags to be added.** Add both tags.

Figure 87 Two tags in MatrikonOPC Explorer (Group0)



Confirm the list of added tags with **OK.**

The **MatrikonOPC Explorer** main window opens. You may save the Explorer configuration as **TwoTagsSession.xml**

Figure 88 Group0 in TwoTagsSession.

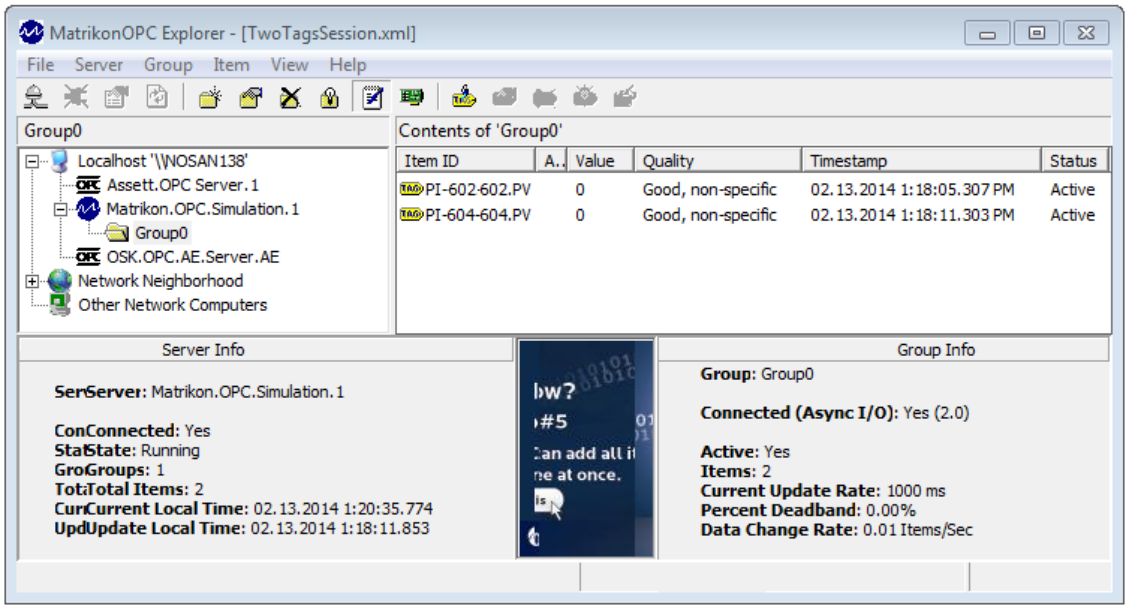


Figure 89 The file TwoTagsSession.xml saved from MatrikonOPC Explorer

```
<?xml version="1.0"?>
- <Session>
  - <Hostname RemoteHost="\\NOSAN138" Remote="0">
    - <Server GroupCount="1" Connected="1" Name="Matrikon.OPC.Simulation.1">
      - <Group Connected="2" Name="Group0" ItemCount="2" PercentDeadband="0.00" TimeBias="0" ReqUpdateRate="1000" Active="-1">
          <Item Active="-1" ReqDataType="0" AccessPath="">PI-602-602.PV</Item>
          <Item Active="-1" ReqDataType="0" AccessPath="">PI-604-604.PV</Item>
        </Group>
      </Server>
    </Hostname>
</Session>
```

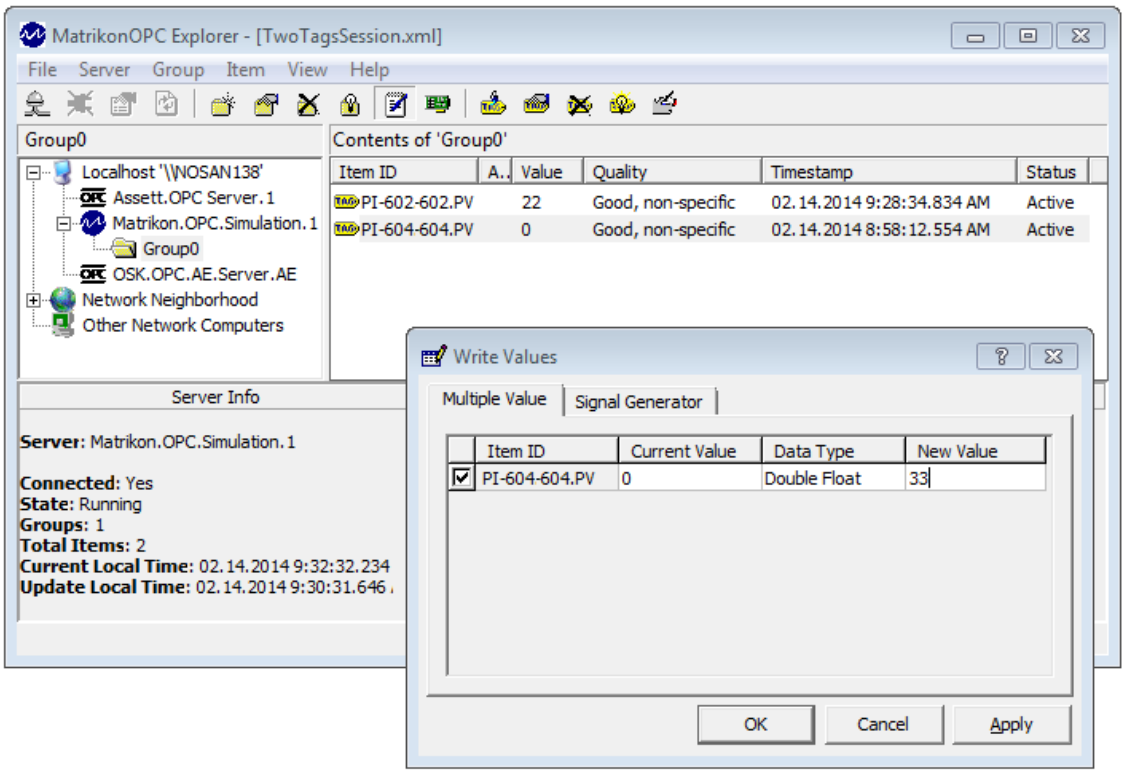
Set values in the MatrikonOPC Server

In the **MatrikonOPC Explorer** main window, double click on **PI-602–602.PV**, in window **Contents of Group0**. Verify that the dialog **Write Values** opens.

In the dialog **Write values** set **PI-602–602.PV** to Value **22**

Do the same for **PI-604–604.PV** and set that Value to **33**.

Figure 90 Dialog Write Values in MatrikonOPC Explorer



5.5 Two tags in the OpcDaCom link

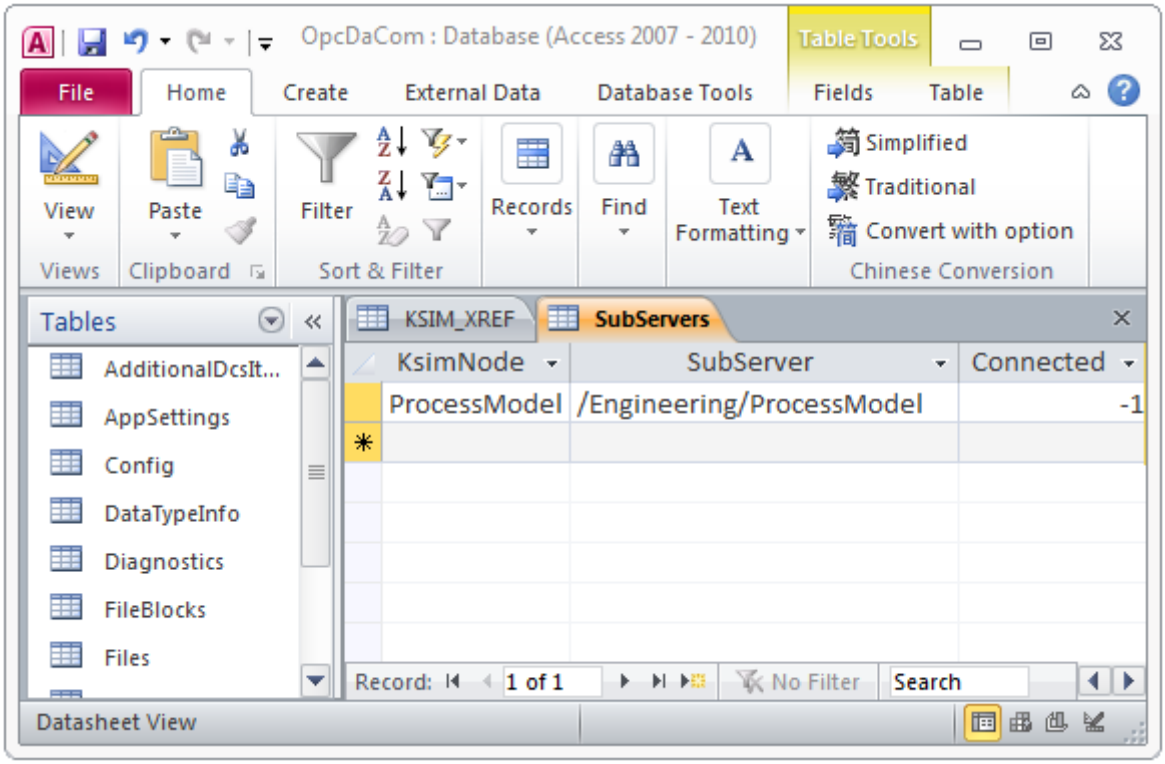
This section continues the example with Two tags in the previous section. The Two tags configured in the MatrikonOPC Server, is read in K-Spice via the link OpcDaCom.

Configuration of the OpcDaCom link in K-Spice

See *Create a link application in the K-Spice project* on page 18 for a more detailed description.The following is a short description of the three relevant tables in the Access database.

The first table connects the variables with the RealTime timeline, via the keyword **KsimNode** in the two tables **SubServers** and **Config.**

Figure 91 The SubServers table defines the timeline of the variables in the Access database for the OpcDaCom link



The Cross Reference List is set up as different keywords in the KSIM_XREF table

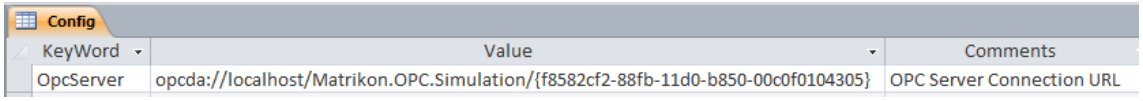
Figure 92 The KSIM_XREF table in the Access database

KSIM_XREF

Id	KsimName	KsimNode	ItemName	ItemType	T	B	C	S	Static	DcsUnit	Tol	I	UpdateR
1	20PI01:DisplayValue	ProcessModel	PI-602-602.PV	Analog	-1	0	-1	0		bara	0.005	0	Medium
2	20PI02:DisplayValue	ProcessModel	PI-604-604.PV	Analog	-1	0	-1	0		bara	0.005	0	Medium

K-Spice connects to the MatrikonOPC Server with the information in the Config table.

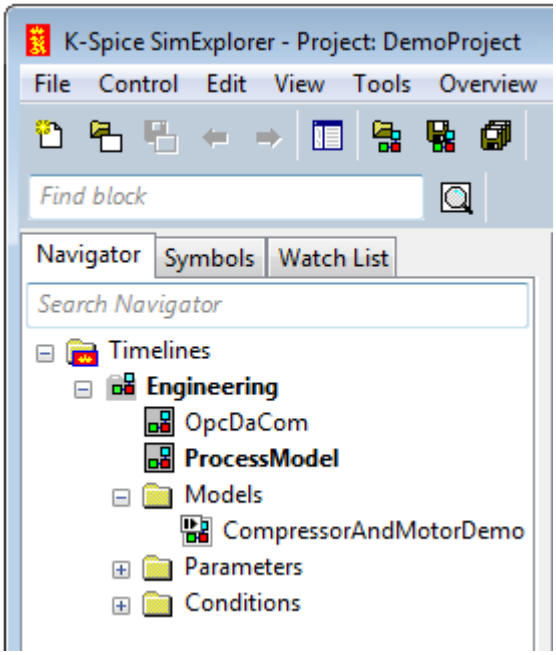
Figure 93 The Config table with the URL to the Matrikon server



Start K-Spice and read the Two tags

The next step is to open the project in K-Spice. Activate the RealTime timeline, and load the model.

Figure 94 Active timeline with an OpcDaCom link

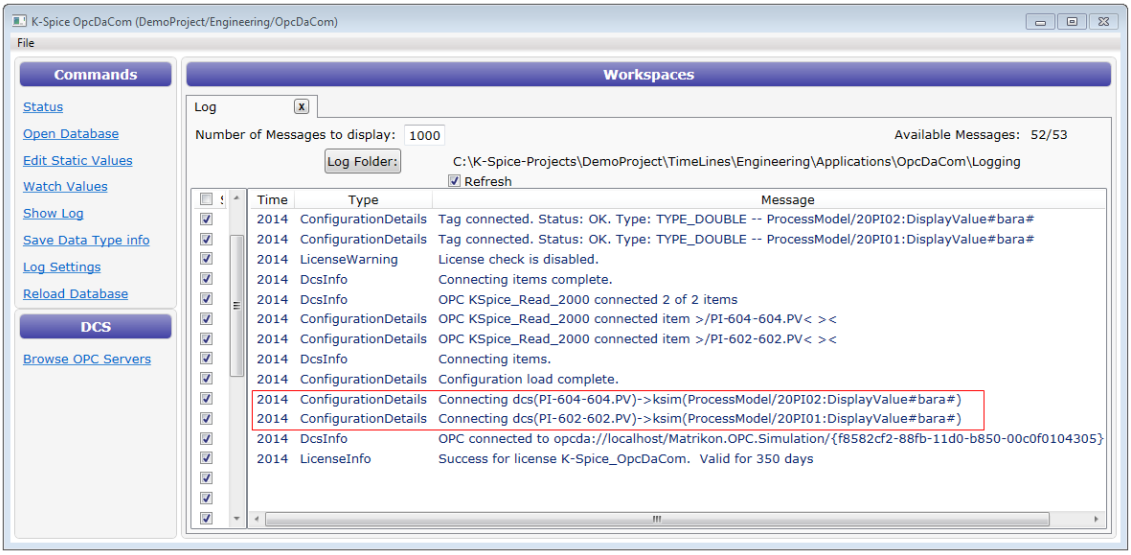


Watch the **SimManager** console, and look for error messages. The SimLink GUI, the main window for the **OpcDaCom** opens.

Open the log file for the link. Verify that the OPC tags and the K-Spice tags are connected.

In the log file below the messages without interest, are filtered out.

Figure 95 The log file for the OpcDaCom link

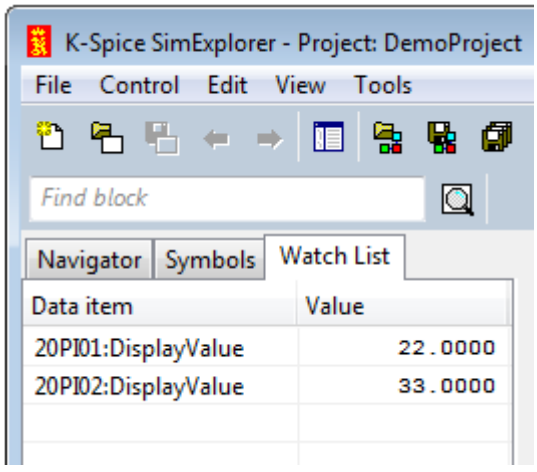


Set a new value in the OPC Server

Double click on one of the two tags in the **MatrikonOPC Explorer**, and enter a new value in the dialog **Write Values**, as explained above. The K-Spice model can be in **Pause.**

Verify that the new values for the Two tags in the OPC server come through to K-Spice.

Figure 96 The new OPC Server values in K-Spice



5.6 The MatrikonOPC HDA Explorer

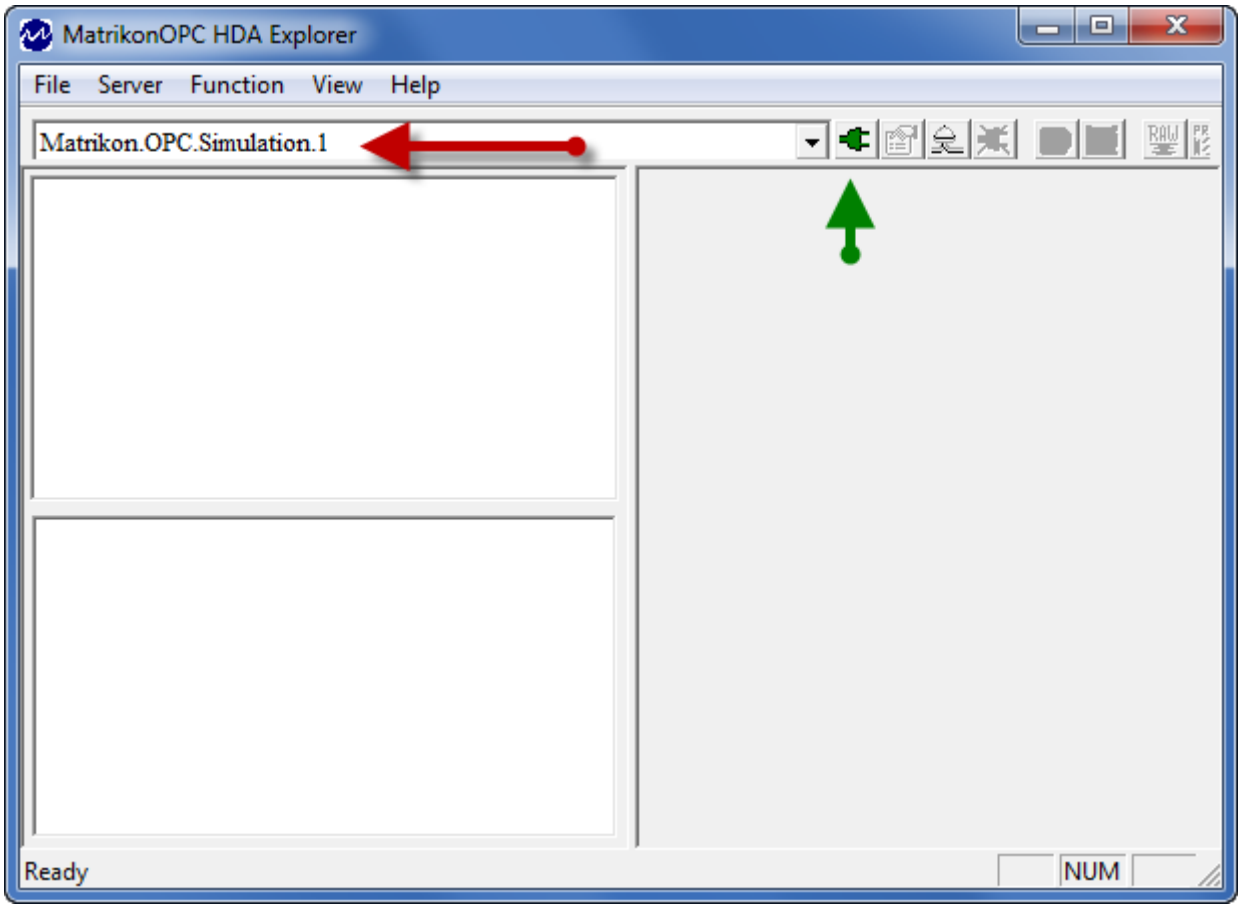
Verification of transfer of data from an OPC Historical database with MatrikonOPC HDA Explorer

Url to Matrikon:
`www.matrikonopc.com/products/opc-desktop-tools/opc-hda-explorer.aspx`

Direct link to Matrikon

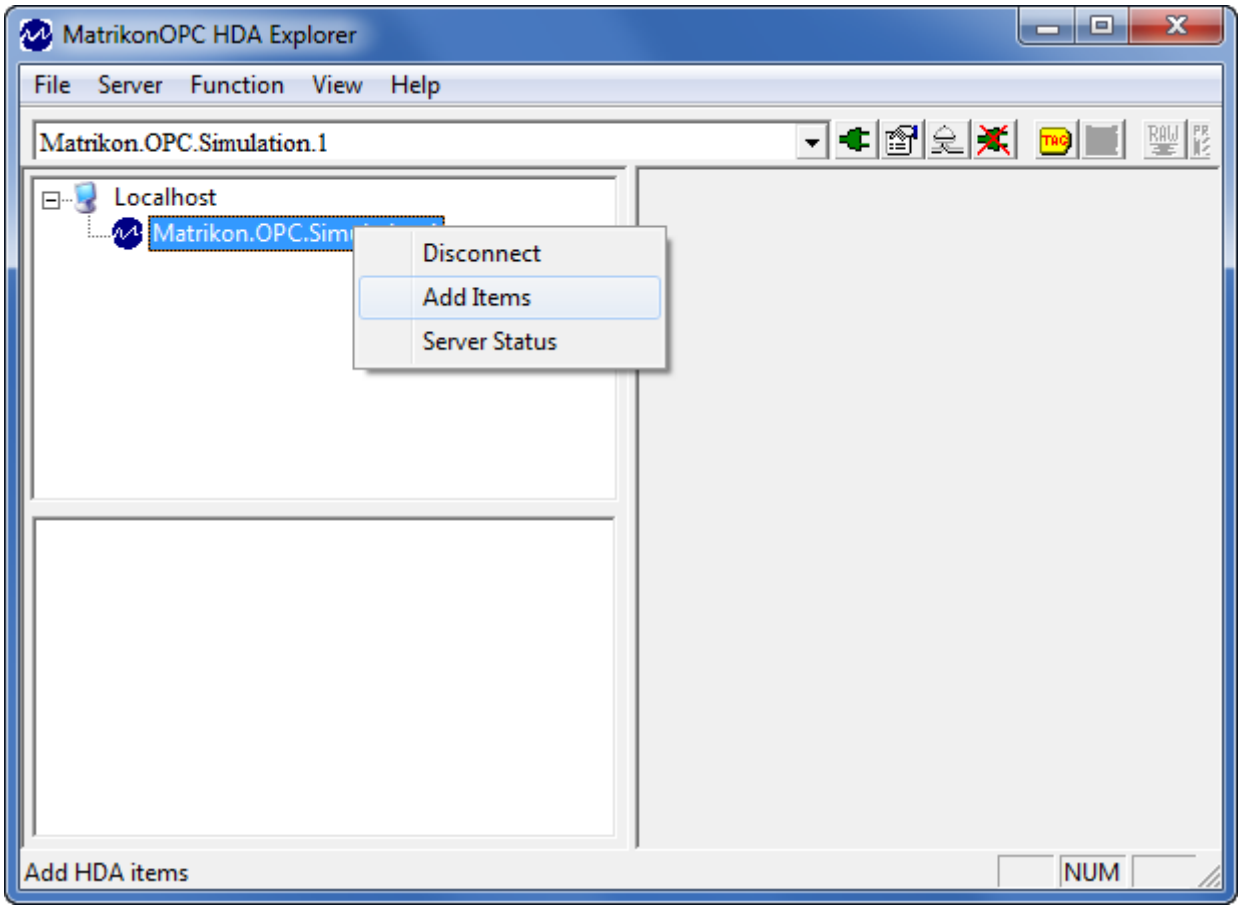
After running MatrikonOPC HDA Explorer, select your target server in the drop down and then click the connect icon.

Figure 97 Connect to server



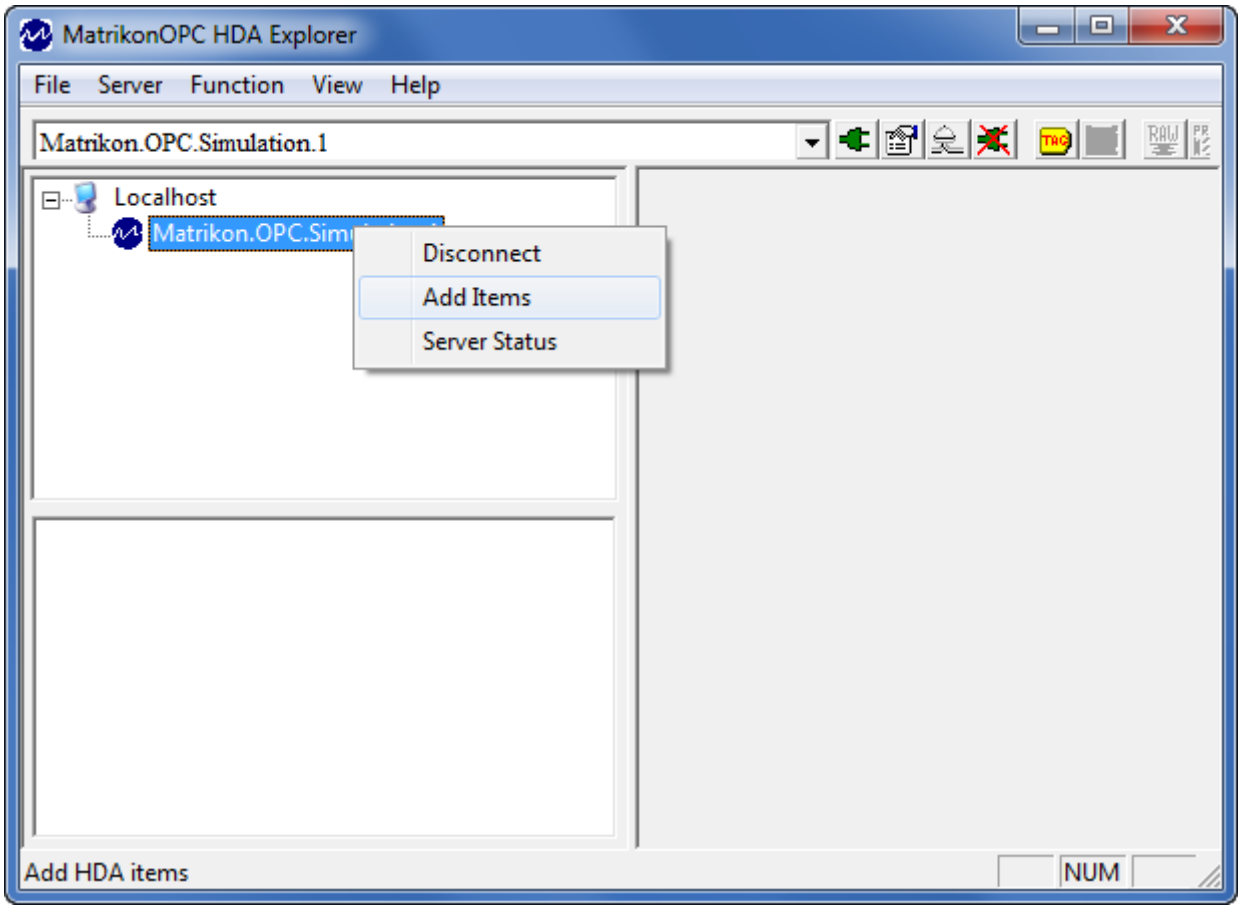
Then right-click on the server in the connection tree. Then left click on Add Items

Figure 98 Add items in server



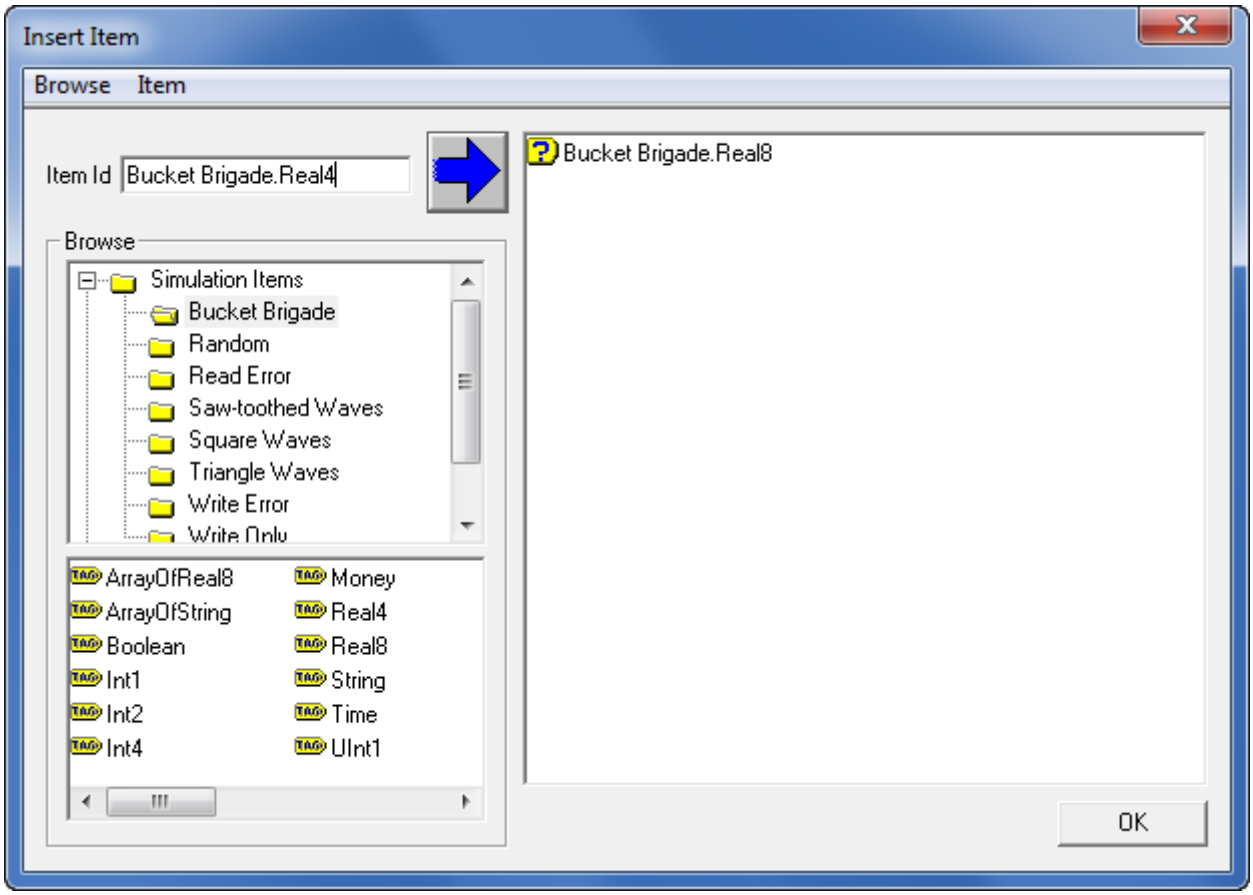
You can navigate to the desired item then the right click on it and then left click "Add To Tag List"

Figure 99 Add items to tag list



You can also type the name in the Item Id text box. Then click the blue arrow.

Figure 100 Add first item to the MatrikonOPC HDA Explorer



Click the OK button to add the items to the MatrikonOPC HDA Explorer.

Figure 101 Add more items to the MatrikonOPC HDA Explorer

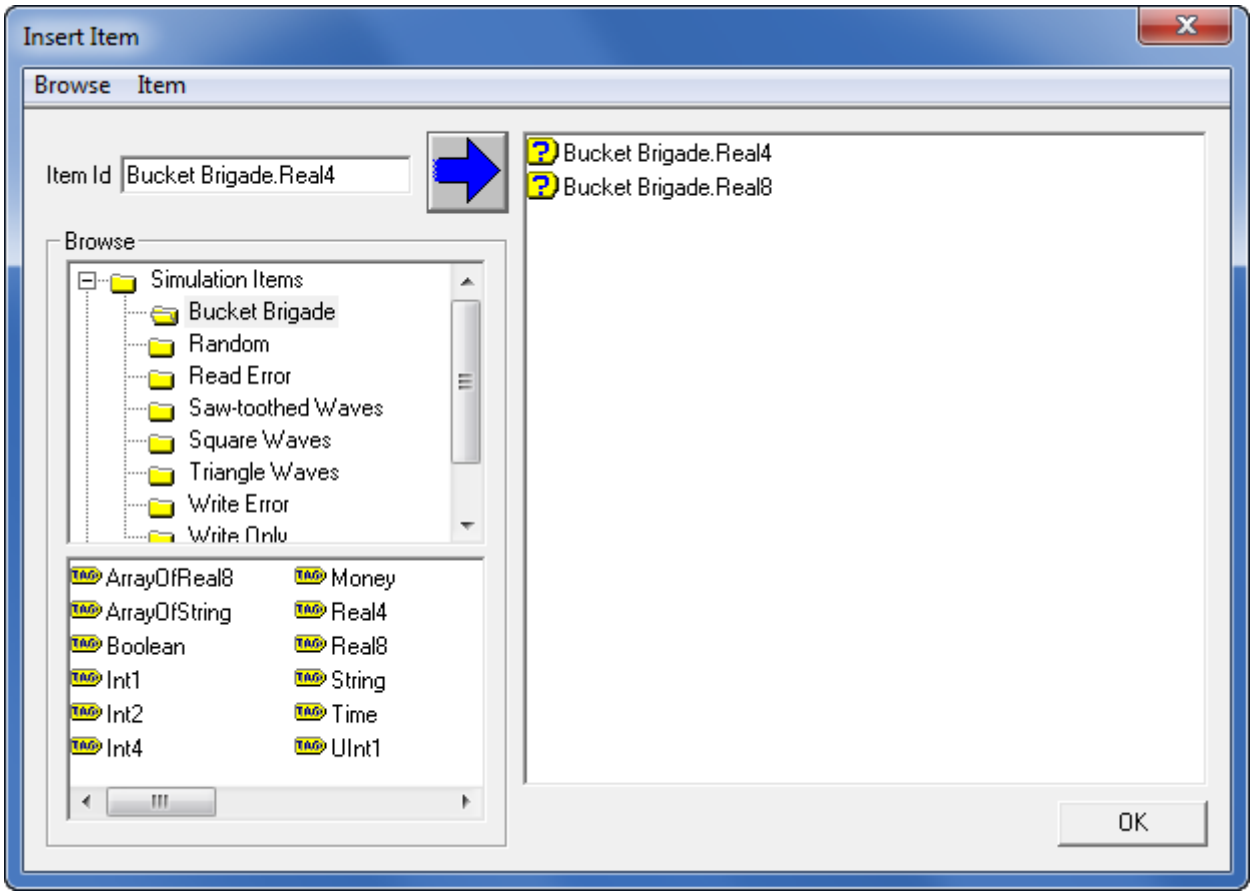
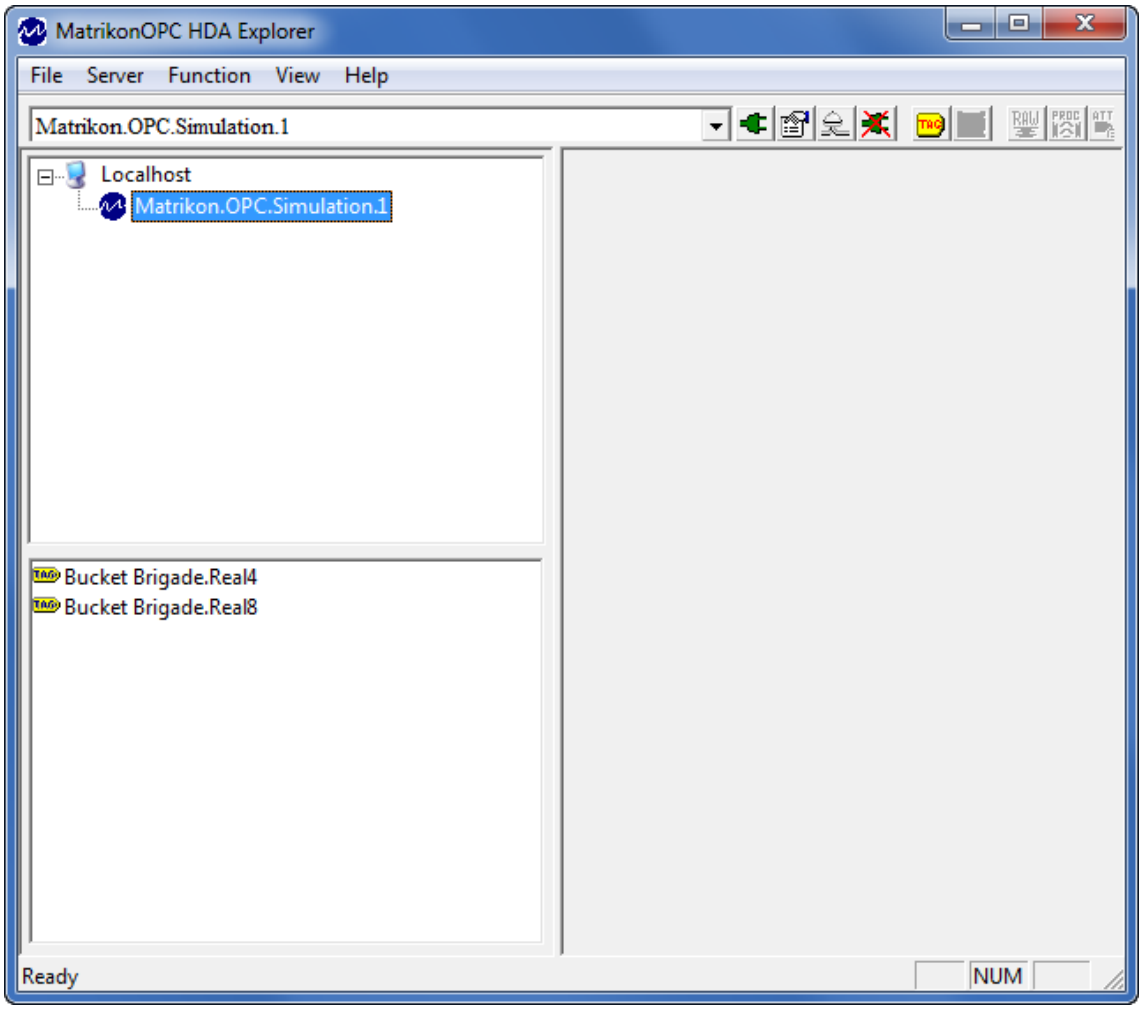
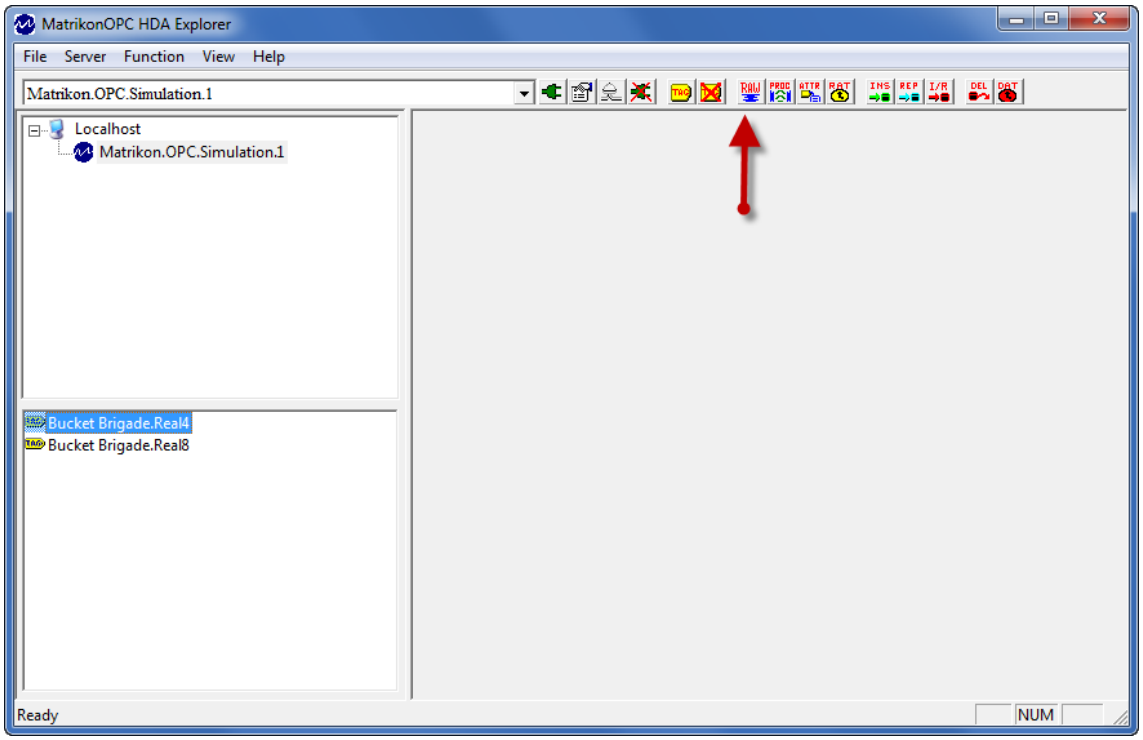


Figure 102 Items added to the MatrikonOPC HDA Explorer



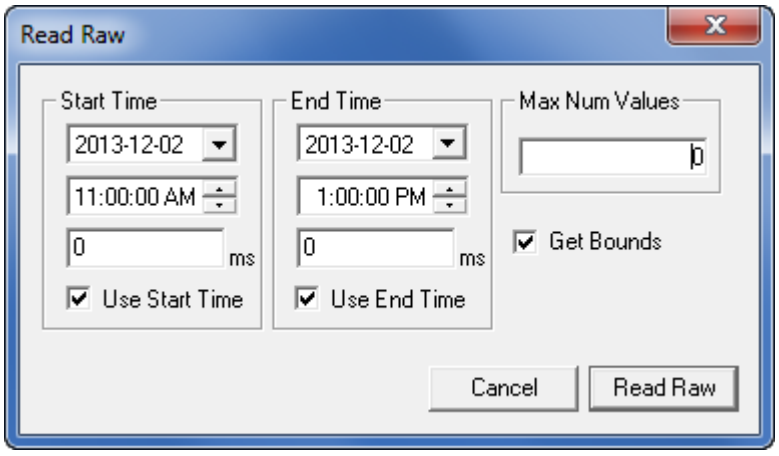
OpcHdaCom uses the **ReadRaw** call to get data from the server. To do this with the **MatrikonOPC HDA Explorer**, click on the desired tag to select it. Then click the **RAW** button in the ribbon.

Figure 103 Use Read raw to get data



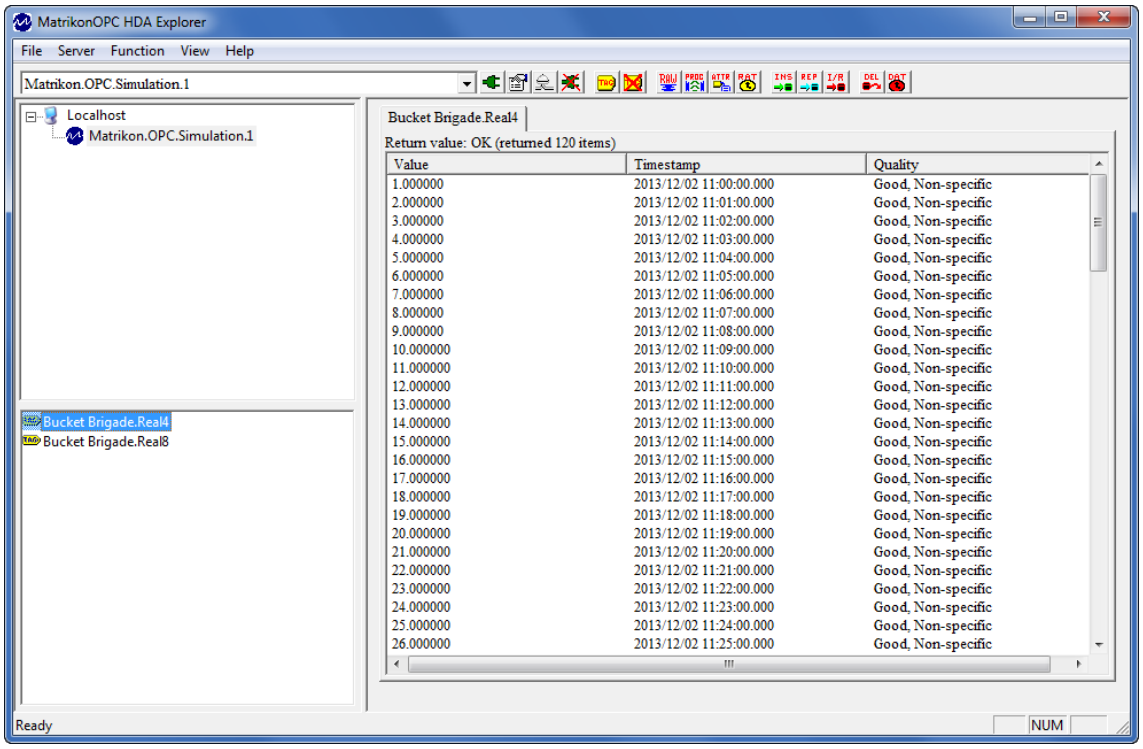
Enter a start time and an end time in the dialog box. **Max Num Values** of 0 instructs the server to return all data within time range. Click **Read Raw** to get the data from the server.

Figure 104 Set interval in the Read Raw dialog



If successful, **MatrikonOPC HDA Explorer** will show the results.

Figure 105 Successful reading in the MatrikonOPC HDA Explorer



6 OPC DCOM configuration

About this chapter.

There are three important sections in this chapter.

DCOM configuration for AbbItsCom on page 112. The names are specific for the ABB ITS system. The steps should be relevant for configuring DCOM communication on any system.

Check the local security policy settings on page 121 and *Modify the DCOM configuration* on page 122 These two sections are more or less an alternative to the section DCOM configuration for AbbItsCom. These two sections have focus on opening up any security setting that can block the communication.

6.1 DCOM configuration for AbbItsCom

About this section

This is a description specific for the the AbsItsCom link. If you are working with other links, there are things in this section, that can be relevant. So this section is not profiled to be part of the link guide for the AbsItsCom link, only. This section is part of all the link guides.

The OPC server name `ITSOPCDASERVER.ITSOPCDASurrogate.1` is a standard name on ABB's Aspect Servers. On a HIMA Server the corresponding name of the OPC Server can be `HIMA.PLC_1.DA.1`

For general description of installation of a link, See *Create a link application in the K-Spice project* on page 18

Installation of ABB software

The ABB software package called "800xA Simulator Link Standalone Client" must be installed on the Model PC. Note that in older versions of 800xA, this software is called "ITS Link Standalone Client". The package is also referred to as the "ITS link"

The text below refer to the user account **800xAServiceAccount.** On other installations the user account is called **800xAServAcc!**You can open **Services** on the Aspect Server to find the name of the relevant user account.

On some plants the OPC Data Server can be installed on a third machine, the Engineering PC. In this situation the OPC Data Server on the Engineering PC is a replica of the OPC Data Server on the Aspect Server.

There are no figures in the checklists below. In this way the lists are compact and easy to read. You will find the relevant figures at the end of the section.

DCOM configuration for AbbItsCom

Checklist on both machines

1 Verify that both the Model PC and the Aspect Server are members of the same domain.

You can verify the computer's domain in a cmd.exe window. Write `set u` and return. You will get the **USERDOMAIN**

The computer's domain is also verified in the **Control panel →User Accounts→Manage User Accounts**

Domain on Windows Server 2003 and 2008

Windows Servers can be set up with Active Directory and Domains. The Servers can alternatively be set up with only Users and Groups. If there are no domain, the `set u` in the command window will show **USERDOMAIN** and the machine name.

2 Verify that the clocks are more or less synchronized.

3 The Windows user must have *Administrator rights* on both machines.

This is necessary when the AbbItsCom link, installed on the Model PC, starts the ITS OPC server, on the Aspect Server. The ITS OPC server is the application ITSOPCDASurrogate.exe.

You can set the *Administrator rights* by opening **Administrative Tools →Computer Management** and find **Local Users and Groups →Groups**

DCOM configuration for AbbItsCom continued

Checklist on the Aspect Server

4 Verify that the user or the user's group is part of local group **Administrators.**

You can do that by opening **Computer Management** and find **Local Users and Groups →Groups**

5 If the Windows Firewall is enabled, make an **Exception** for `ITSOPCDASurrogate.exe`.

Exception on a Windows 7 and Server 2008:

This is done in the **Control panel** Open **Administrative Tools →Windows Firewall →Allow a program or a feature through Windows Firewall**

In dialog **Allowed Programs: Change settings Allow another program.** In dialog **Add a program: Browse.** Locate`ITSOPCDASurrogate.exe`. Add the program to the list of allowed programs with the button **Add.** Leave the Browser with **OK** and verify that **ITSOPCDASurrogate** is listed in the window **Programs.**

Verify that **ITSOPCDASurrogate** is in the list **Allowed program and features** and that the button **Domain** is checked. See figure below: **K-Spice ABB ITS Com in the list Allowed Programs.**

Exception on Windows XP and Server 2003:

Make an Exception in the Windows Firewall. See the figure below: **Excpetions on a Windows Server 2003 machine.**

6 **ITSOPCDASurrogate** must be set to run as user **800xAServiceAccount.**

Before you can do this, you must know the *password* for the user **800xAServiceAccount.**

To set the user **800xAServiceAccount:**

Start with the **Control panel.** Open **Administrative Tools →Component Services.**

In dialog **Component Services:** Go to **Console Root→ Component Services →Computers → My Computer→ DCOM Config** You will see a field of icons here. You can change the icons to a list view with a right click on **DCOM Config. →View → Details**

Find `ITSOPCDASurrogate`

Right hand click on the icon `ITSOPCDASurrogate` or the corresponding list item, then **→Properties**

In dialog **ITSOPCDASurrogate Properties:** Open tab **Identify** Select **This user** Enter User: **800xAServiceAccount** and Password.

DCOM configuration for AbbItsCom continued.

Checklist on the Model PC

7 Add the key `HKEY_LOCAL_MACHINE\Software\Classes\ITSOPCDASERVER.ITSOPCDASurrogate.1` in the Registry.

A way to do this is to start `regedit.exe` on the Aspect Server, and export the actual key as a **.reg** file. Then import the **.reg** file to the registry on the Model PC by copying the file over to the Model PC and double click on it.

This is an example of a .reg file:

OPC/Vendor: `ABB AS`

CLSID:`{023CD2A3-43BA-4435-BBF6-ECBC2D6BDDCE}`

See the figure below, **Registry file for a Hima link.**

8 If the Windows Firewall is enabled, it may be necessary to open TCP port **135.**

9 If the Windows Firewall is enabled, make an **Exception** for `AbbItsCom.exe`.

Exception on a Windows 7 and Server 2008:

This is done in the **Control panel** Open **Administrative Tools →Windows Firewall →Allow a program or a feature through Windows Firewall**

In dialog **Allowed Programs: Change settings Allow another program.** In dialog **Add a program: Browse.** Locate `AbbITSCom.exe`. Add the program to the list of allowed programs the button **Add.** Leave the Browser with **OK** and verify that **K-Spice ABB ITS Com** is listed in the window **Programs.**

Verify that **K-Spice ABB ITS Com** is in the list **Allowed program and features** and that the button **Domain** is checked. See figure below: **K-Spice ABB ITS Com in the list Allowed Programs.**

Exception on Windows XP and Server 2003:

Make an Exception in the Windows Firewall. See the figure below: **Excepetions on a Windows Server 2003 machine.**

10 It might be necessary to add the user **800xAServiceAccount** under **Default Access permissions** and **Launch and Activation Permissions.**

However, this is often the standard if both computers are in the same domain.

This is done in the **Control panel** (In **Control panel** with 8 items, open **System and security.**) Open **Administrative Tools →Component Services.**

In dialog **Component Services:** Go to **Console Root→ Component Services →Computers → My Computer**

The settings for adding a user is found in **COM Security:**

Right hand click on **My Computer →Properties →COM Security**.

In dialog **My Computer Properties** select tab **COM Security**

11 Check that the name of the ABB OPC server is correct in the config table in the Access database.

When AbbItsCom is running on the Model PC and connecting to the OPC server running on the Aspect Server, the correct OPC name is `ITSOPCDAServer.ITSOPCDASurrogate.1`

When both programs are set up to run on the same machine, the correct OPC name is `ITSOPCDAServer.ITSOPCDAHandler.1`

An example of how to set the name of the OPC server is shown in a figure below:
Name of the OPC Server in the Config table

Two dialogs for setting Exception:

Figure 106 Allowed Programs with K-Spice ABB ITS Com on a Windows 7 machine

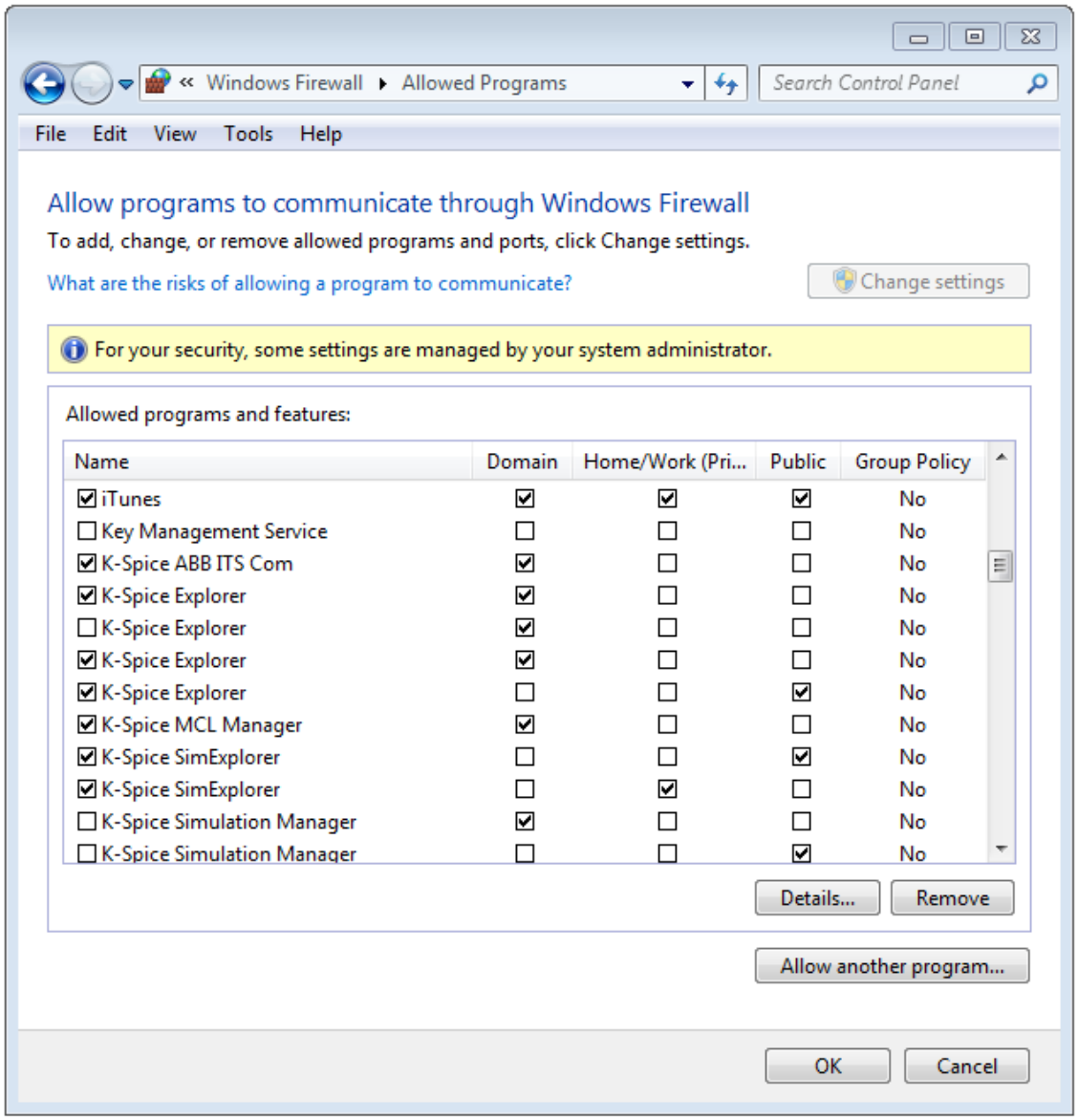
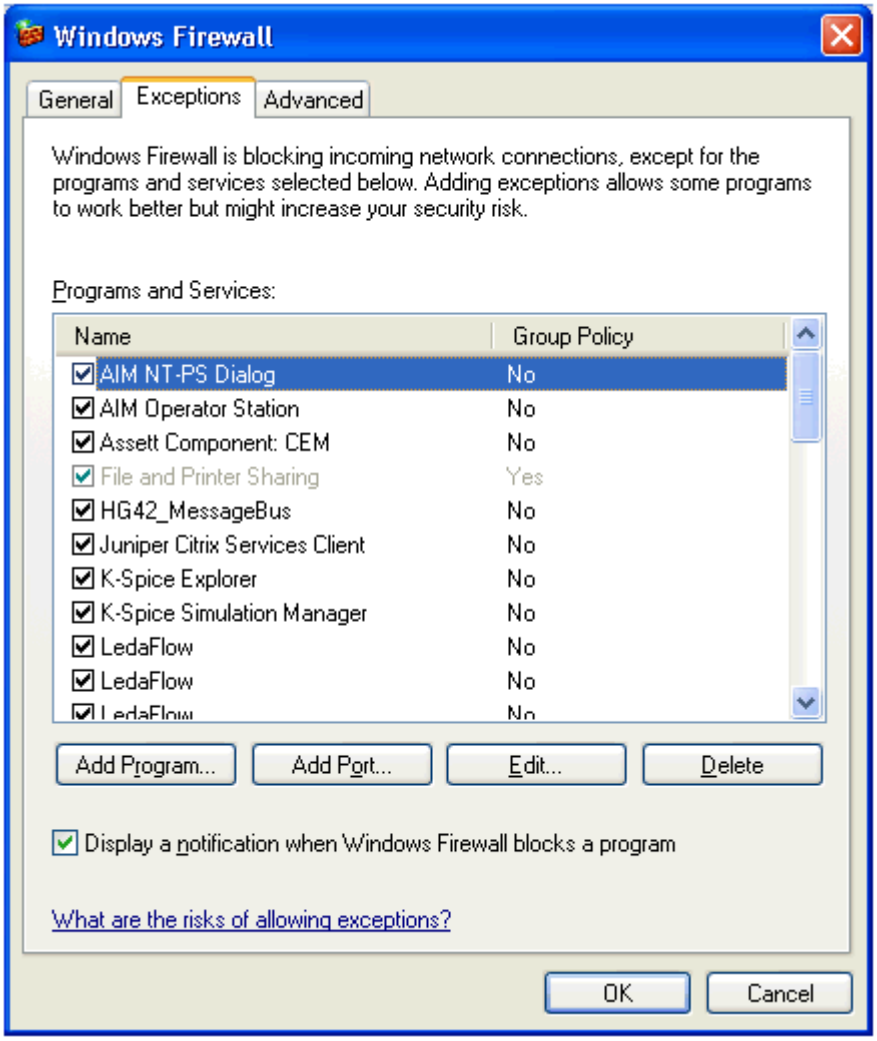


Figure 107 Excpetions on a Windows Server 2008



Name of the OPC Server in the Config table.

This table is from the old C++ version of the AbbItsCom link.

Figure 108 Name of the OPC Server when the Server is running on the Model PC

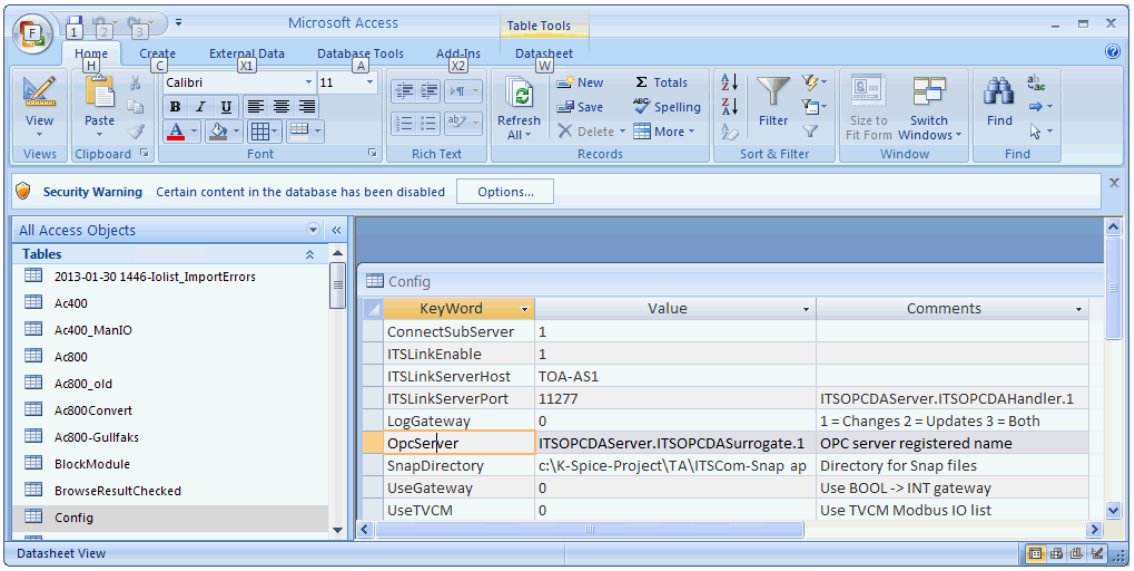


Figure 109 Registry file for a HIMA link

```
Windows Registry Editor Version 5.00

[HKEY_LOCAL_MACHINE\SOFTWARE\Classes\HIMA.PLCA.1]
@="HIMA X-OTS DA (PLCA)"

[HKEY_LOCAL_MACHINE\SOFTWARE\Classes\HIMA.PLCA.1\CLSID]
@="{AA8326BC-8FAA-8FAA-8FAA-8325832FB733}"
```

File: `PLCA.reg`

Please notice that many *screen shots* in this guide are ***manipulated***. The policy has been to disguise the real names and structures and the real urls's of the DCS databases. Data items from a K-Spice demo project are normally untouched. Data items names from real projects are normally manipulated.

HIMA.PLCA.1 and {AA8326BC-8FAA-8FAA-8FAA-8325832FB733} are both manipulations. The format is the same as the format for the real Hima database. Use the functionality **Browse OPC Servers** in the SimLink GUI to find the real values.

6.2 Check the local security policy settings

This section is first part of a Wiki page called **OPC DCOM configuration**

See the second part of the Wiki page in the next section *Modify the DCOM configuration* on page 122

The following setting may be used to open up OPC communications

- Go to the **Start** menu →**Control Panel**
- Open **Administrative Tools**
- Open **Local Security Policy**
- In dialog **Local Security Policy:**
- Browse to **Security Settings** →**Local Policies**→ **Security Options**

In dialog Local Security Policy, page Policy

Right click on DCOM: Machine Access Restrictions and select →Properties

- Click on **Edit Security**
- In dialog **Access permission:**
- Add **Anonymous, Everyone, Interactive, Network, System** with all local and remote permissions set to **Allow**.
- Close the dialog with **OK**

Right click on DCOM: Machine Launch Restrictions and select →Properties

- Click on **Edit Security**
- In dialog **Launch and Activation Permission:**
- Add **Anonymous, Everyone, Interactive, Network, System** with all local and remote permissions set to **Allow**
- Close the dialog with OK

Right click on Network access: Let Everyone permissions apply to anonymous users and select →Properties

- In dialog **Network access: Let Everyone permissions apply to anonymous users:**
- Select the **Enabled** radio button
- Close the dialog with OK.

6.3 Modify the DCOM configuration

This section is the second part of a Wiki page called **OPC DCOM configuration**

See the first part of the Wiki page in the previous section *Check the local security policy settings* on page 121

Modify the DCOM configuration.

Goto the **Start** menu and run **dcomcnfg.exe:**

- In the bottom field with the text *Search programes and files* enter **dcomcnfg** and **enter.** This will bring up the **Component Services** browser window.

My computer Properties

- In dialog **Component Services:**
- Browse to **Console Root →Component Services →Computers →My Computer**
- Right click on **My Computer** and select **→Properties**

In the dialog My Computer Properties:

The tab Default Properties

- Enable **Distributed COM** on this computer
- Set **Default Authentication Level** to **Connect**
- Set **Default Impersonation Level** to **Identify**

The tab Default Protocols

- **Connection-orientated TCP/IP** should be the only one listed.

The tab Com Security

In frame with subtitle Access Permissions

- Click on **Edit Default**
- In dialog **Access Permission:**
- Add **Anonymous, Everyone, Interactive, Network, System** with all local and remote permissions set to **Allow**.
- Close the dialog with **OK**
- Click on **Edit Limits**
- In dialog **Access Permission:**
- Add **Everyone** with all local and remote permissions set to **Allow**.
- Close the dialog with **OK**

In frame with subtitle Launch and Activation Permissions

- Click on **Edit Default**
- In dialog**Launch and Activation Permission:**

- Add **Anonymous, Everyone, Interactive, Network, System** with all local and remote permissions set to **Allow**.
- Close the dialog with **OK**
- Click on **Edit Limits**
- In dialog **Access Permission:**
- Add **Everyone** with local and remote permissions for both launch and activation set to **Allow**.
- Close the dialog with **OK**

OpcEnum Properties

- In dialog **Component Services:**
- Browse to **Console Root →Component Services →Computers →My Computer →DCOM Config**

  You can right hand click on **DCOM Config →View →Details** You will see the objects in a list, not object icons
- Find the icon/entry **OpcEnum**. Right click on **OpcEnum** icon/entry and select **→Properties**

In the dialog OpcEnum Properties:

The tab General

- Set **Authentication level** to **→Default**

The tab Location

- Set **Run application on this computer** to **→On**

The tab Security

- Set **Launch and Activation Permissions** to **→Use Default**
- Set **Access Permissions** to **→Use Default**
- Set **Configuration Permissions** to **→Customize**

The tab Security in the frame with the subtitle Configuration Permissions

- Click on **Edit**
- In dialog **Change Configuration permissions:**
- Add **Anonymous, Everyone, Interactive, Network, System** with all local and remote permissions set to **Allow**.
- Close the dialog with **OK**

The tab Identity

- Set **The system account** to **→On**

RSLinx Properties

RSLinx is the name of a specific OPC Server. Do this configuration for the actual OPC Server.

You will find RSLinx on the same page that you found OpcEnum in the previous section. If you start from the top, you will find RSLinx here:

- In dialog **Component Services:**
- Browse to **Console Root →Component Services →Computers →My Computer→DCOM Config**

  You can right hand click on **DCOM Config →View →Details** You will see the objects in a list, not object icons.
- Find the icon/entry **RSLinx**. Right click on the **RSLinx** icon/entry and select **→Properties**

In the dialog RSLinx Properties:

The tab General

- Set **Authentication level** to **→Default**

The tab Location

- Set **Run application on this computer** to **→On**

The tab Security

- Set **Launch and Activation Permissions** to **→Use Default**
- Set **Access Permissions** to **→Use Default**
- Set **Configuration Permissions** to **→Customize**

The tab Security in the frame with the subtitle Configuration Permissions

- Click on **Edit**
- In dialog **Change Configuration permissions:**
- Add **Anonymous, Everyone, Interactive, Network, System** with all local and remote permissions set to **Allow**.
- Close the dialog with **OK**

The tab Identity

- Set **The system account** to **→On**
- Close the dialog with **OK**

Note

Reboot the computer

6.4 ODBC Data Source Administrator

For the old links, the *directory for the Access database* must be added as a computer system configuration. This is the case for the old AbbItsCom link, and other old links. This configuration must be modified, if next time another link shall run on the computer.

On a Windows 7 machine, run: `C:\Windows\SysWOW64\odbcad32.exe`

On a 32 bit machine, run: `C:\Windows\System32\odbcad32.exe`

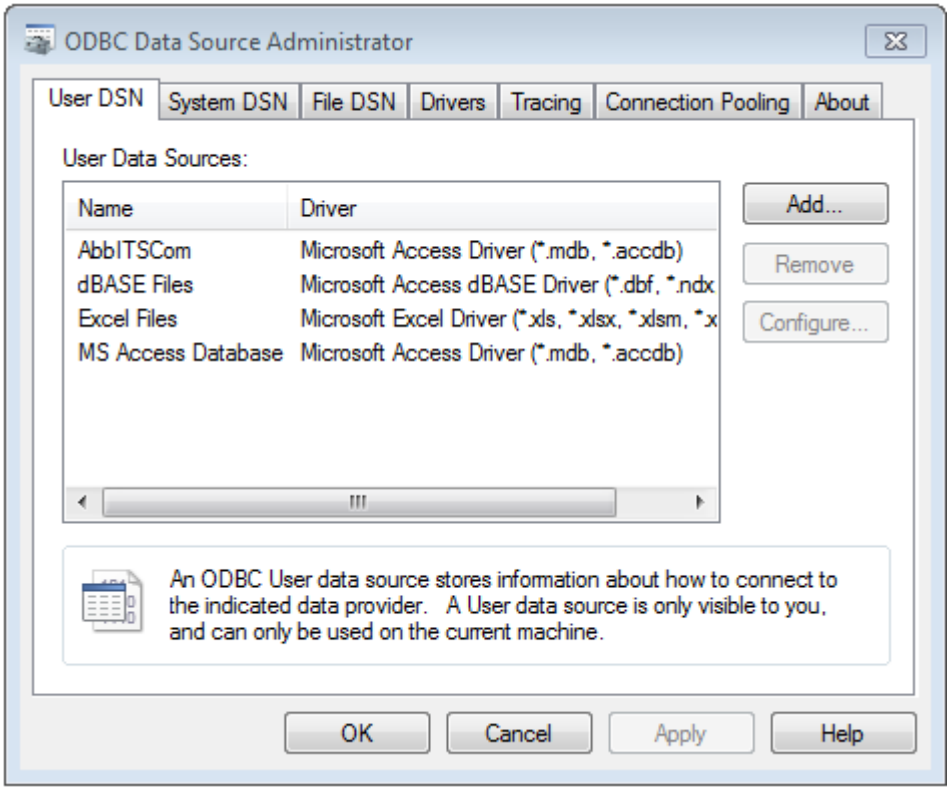
In dialog **ODBC Data Source Administrator** button **Add**

In dialog **Create a new Data Source:** select →**Microsoft Access Driver (*.mdb, *.accdb)** and **Finish**

In dialog **ODBC Microsoft Access Setup:** Data Source Name: **AbbItsCom** and Database: **Select**

In dialog **Select Database** browse to `C:\K-Spice-Projects\Demo_ABB_CPP\TimeLines\OTS_ABB_CPP\Applications\ABB\AbbItsCom.mdb` Exit with **OK**

Figure 110 ODBC configured to read the AbbItsCom database



7 Variable transfer between DCS and K-Spice

This is chapter covers the basic data transfer functionality in the links. Not covered in this manual are issues like Timing, Firewall fighting, one way versus two way communication, historical values, IP addresses, Hand shake protocols, Ports and more.

7.1 DCS connection philosophy

The idea of connecting a K-Spice dynamic simulation model to a “soft” controller implementation of a DCS is to stimulate (provide equivalent signal representations) the system so it operates like it is actually connected to a real plant. This situation allows the model to run under DCS control with model variables being connected to the field inputs and outputs of the system.

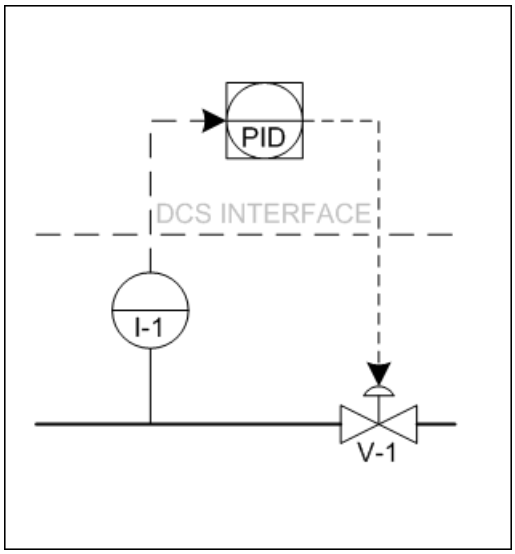
The DCS link should include the following functionality:

- Configuring the frequency (sampling rate) of a model variable.
- Configuring the individual tolerance of a variable to determine if there has been significant changes in the model that requires the transfer of a variable.
- Mapping of variables between the K-Spice model and K-Spice®.
- Scaling of variables.
- Sending static values to selected DCS inputs.

The control signals required to be mapped are usually between transmitters, valves, electric motors, local control loops and emulated sequences. The safety signals from K-Spice® are also usually connected to shutdown inputs on valves and electric motors.

The below figure shows a simple diagram depicting two common connections between a K-Spice model and a DCS system, where the transmitter signal in the model is sent to the DCS control function block. The output of the DCS function block shown, a valve control signal, is sent back to the model to regulate the control valve.

Figure 111 A K-Spice/DCS interface



A typical K-Spice and DCS connection for a simple control loop (I-1 is a transmitter in K-Spice). The transmitter signal is sent to the function block in the external DCS system. The DCS sends a valve control signal back to K-Spice.

7.2 Interfaces to the DCS Servers

The K-Spice communication with the DCS Server consists of data transfer two types. One type of data transfer is getting process data from the system, temperatures, pressures etc. The other type of data transfer is two directional, sending commands like Run, Pause, Save Conditions, Load Conditions.

The different DCS vendors offer different interfaces to their DCS data.

- OPC. This is a standard interface for getting process data. This interface may exist in parallel to a propritary standard for two directional transfer of commands between the DCS system and K-Spice
- Propritary interfaces.

DCS to DCS links

The SimLinks can be set up to *get data* from one DCS Server, and *send it to another DCS Server*. This is used when the process has two different control systems that cover two different sections of the process. In the actual situation one of the control systems will normally send data of some kind to the other control system. It can be transfer of process data like temperatures etc. Link-to-link is typically used for data between control systems to the latency of bringing it to a model tag first. It also saves on having to create those model tags.

This data transfer can be simulated with a K-Spice SimLink sending data from one part of the simulated process, to another part of the simulated process.

AutoswitchCom utilizes link-to-link to manage automatically switching between historical data and current data for real-time systems.

7.3 Model variable subscriptions

Model variable subscriptions perform general tasks common to all DCS connections, which include:

- organising values to be exchanged into groups according to transfer direction, value type and sample rate (model variable tolerance related).
- allowing each group to be connected to a defined external system.
- defining handling characteristics (for each value within a group) for that particular variable by setting values on attributes associated with each variable.
- conveying commands issued in the simulator to the external DCS so it is possible to act in syncronisation with the model (commands include `Run`, `Freeze`, `Change Speed`, `Save` and `Load Snapshot`).

The groups containing exchange variables for a particular process model instance are organised in an Access database, which is defined in a later section.

7.4 DCS driver functionality

The DCS driver for a particular DCS is a functional block with a defined interface towards the model variable subscriptions and internal logic implementing the system specific communication tasks.

The tasks solved by a driver are:

- Transfer values between the external system and the model. Tag names used to identify a value in the external system are found as parameters in the entry for a variable in the IO list.
- If applicable, convert commands received from the model variable subscriptions and forward them to the external system. Once a command has finished, return the proper acknowledgement to the model variable subscription.

The appropriate driver for a defined external DCS system is added through the Access database by selecting the appropriate DCS link for the control system.

7.5 Packing and unpacking variables

The DCS driver is also responsible for packing and unpacking data in cases where multiple, individual model variables are sharing a common DCS variable (for example, a mask of limit switches.)

8 The Cross Reference List

Scope of this guide

The Cross Reference List is essential when the link is up and running. One line in the list has two main items. One item is a DCS value. The other item is the K-Spice data item in the K-Spice simulator. The link will read the DCS value and write the value in the K-Spice data item. Or the read and write is the other way round.

Setting up the Cross Reference List is often done with queries and tables *in a separate Access database*, **such project specific tables and queries should not be entered in the link database itself**. After the Cross Reference List is created, the Cross Reference List will be copied into the link's Access database. The final list is copied to the table KSIM_XREF.

Setting up the Cross Reference List is outside scope of this guide. Making a Cross Reference List is a difficult topic to document. Much of the work is based on experience and personal skills. Not much is written down.

Note

The text in this guide is based on the assumption that the Cross Reference List already exists.

8.1 Creation of the Cross Reference list.

This guide will not cover the creation of the Cross Reference list. But 4 figures on the topic are shown below, as a reference to the work with the lists. The chapter called The Access database have examples of the table KSIM_XREF from different link tests.

See *The **KSIM_XREF** table. Examples* on page 73

and *Configuration of the OpcDaCom link in K-Spice* on page 100

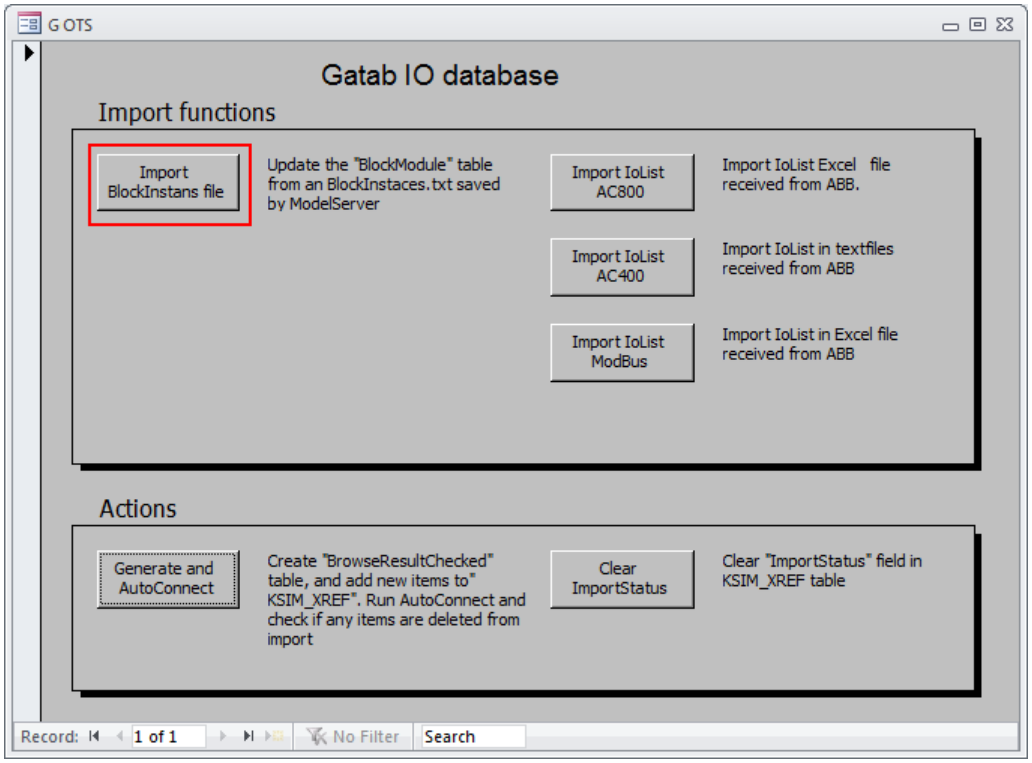
The following 4 figures, are extracts of essential parts of the creation of the Cross Reference list.

Import block instances

Each K-Spice simulator application contains a BlockInstances.txt file. This file contains a list of all the tags in the application.

Click on the Import BlockInstanse file button. This will fill the BlockModule table with valid simulator tags.

*Figure 112 An Access form with the button **Import BlockInstance file.** in the database Cross Reference List*

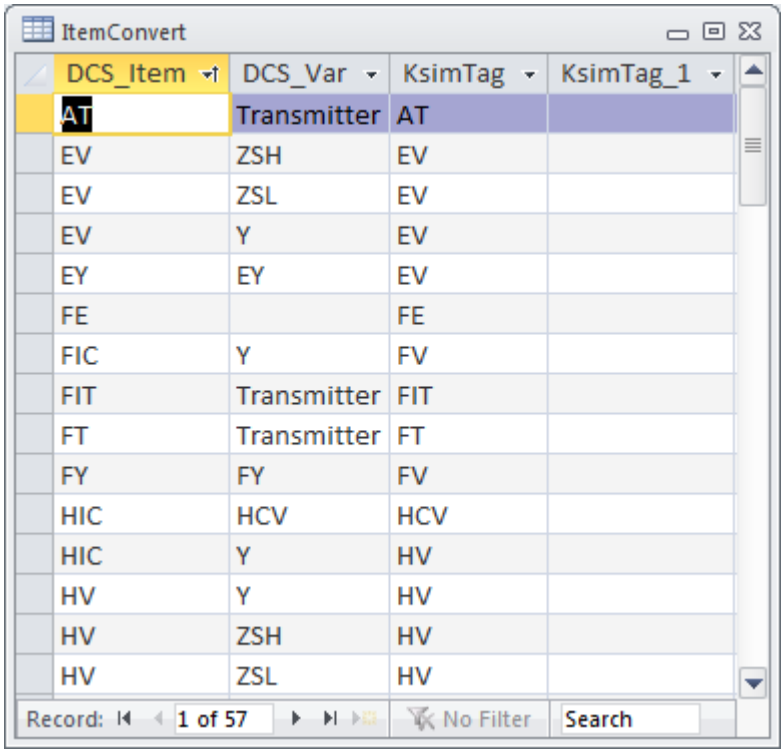


The table ItemConvert

ItemConvert is a table used to match the DCS tag to the relevant K-Spice tag.

The DCS_Item column is a handmade list of text strings that have been extracted from the DCS IO tags. The DCS_Var column is a handmade list of text strings that denote the function of the signal. The KsimTag column is a handmade list of text strings that will be used by the script for matching with relevant K-Spice tags.

Figure 113 The first part of the table ItemConvert in the database Cross Reference List



ItemConvert

DCS_Item	DCS_Var	KsimTag	KsimTag_1
AT	Transmitter	AT	
EV	ZSH	EV	
EV	ZSL	EV	
EV	Y	EV	
EY	EY	EV	
FE		FE	
FIC	Y	FV	
FIT	Transmitter	FIT	
FT	Transmitter	FT	
FY	FY	FV	
HIC	HCV	HCV	
HIC	Y	HV	
HV	Y	HV	
HV	ZSH	HV	
HV	ZSL	HV	

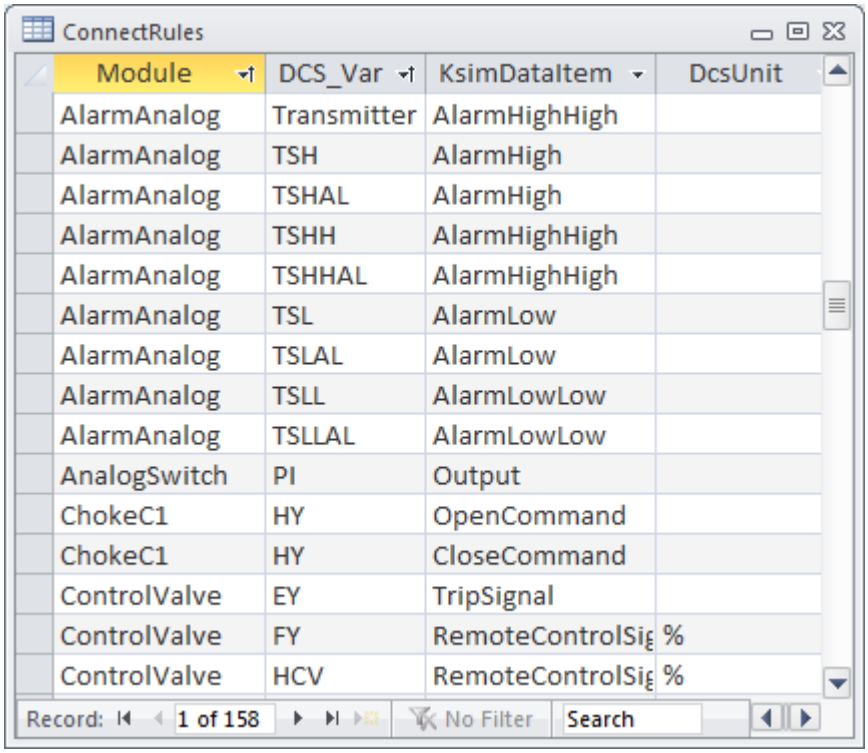
Record: 1 of 57 No Filter Search

The table ConnectRules

ConnectRules is a table used to convert the DCS tag to the relevant K-Spice tag.

The Module column is a handmade list of K-Spice modules. The DCS_Var column is a handmade list of text strings that denote the function of the signal. The KsimDataItem column is a handmade list of K-Spice module variables that will be used by the script for matching the DCS IO signal with the relevant variable.

Figure 114 A part of the table ConnectRules in the database Cross Reference List



ConnectRules

Module	DCS_Var	KsimDataItem	DcsUnit
AlarmAnalog	Transmitter	AlarmHighHigh	
AlarmAnalog	TSH	AlarmHigh	
AlarmAnalog	TSHAL	AlarmHigh	
AlarmAnalog	TSHH	AlarmHighHigh	
AlarmAnalog	TSHHAL	AlarmHighHigh	
AlarmAnalog	TSL	AlarmLow	
AlarmAnalog	TSLAL	AlarmLow	
AlarmAnalog	TSLL	AlarmLowLow	
AlarmAnalog	TSLLAL	AlarmLowLow	
AnalogSwitch	PI	Output	
ChokeC1	HY	OpenCommand	
ChokeC1	HY	CloseCommand	
ControlValve	EY	TripSignal	
ControlValve	FY	RemoteControlSig	%
ControlValve	HCV	RemoteControlSig	%

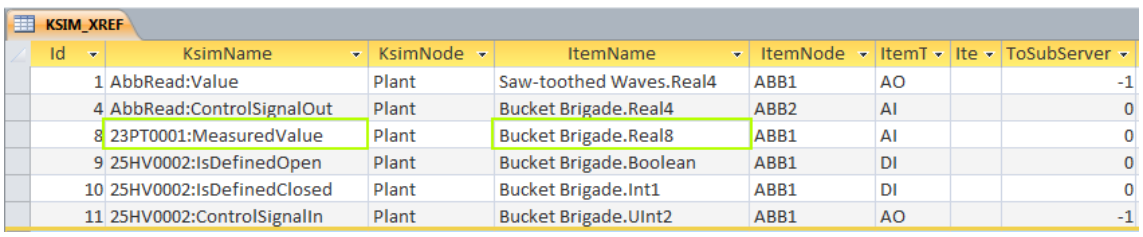
Record: 1 of 158 No Filter Search

A Cross Reference list for an OpcDaCom link

See *Configuration of the OpcDaCom link in K-Spice* on page 100

This figure is from the chapter **MatrikonCom desktop tools** in the Dcs Link Guide for OPC links:

Figure 115 The KSIM_XREF table in the Access database for a OpcDaCom link



KSIM_XREF

Id	KsimName	KsimNode	ItemName	ItemNode	ItemT	Ite	ToSubServer
1	AbbRead:Value	Plant	Saw-toothed Waves.Real4	ABB1	AO		-1
4	AbbRead:ControlSignalOut	Plant	Bucket Brigade.Real4	ABB2	AI		0
8	23PT0001:MeasuredValue	Plant	Bucket Brigade.Real8	ABB1	AI		0
9	25HV0002:IsDefinedOpen	Plant	Bucket Brigade.Boolean	ABB1	DI		0
10	25HV0002:IsDefinedClosed	Plant	Bucket Brigade.Int1	ABB1	DI		0
11	25HV0002:ControlSignalIn	Plant	Bucket Brigade.UInt2	ABB1	AO		-1

8.2 Item Connections

The link configuration is finished when any item in the list of DCS tags has been coupled to a K-Spice tag in the list KSIM_XREF.

This will be confirmed in the link's logfile. There are no more warnings **Not connecting**.

8.2.1 Source Item Connections (DCS loopback)

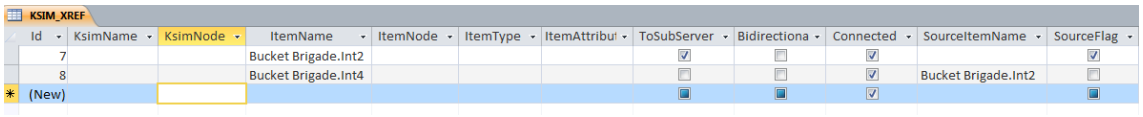
You configure the source of the data by:

- Specify the DCS connection info in ItemNode and ItemName
- Check the ToSubServer setting
- Check the SourceFlag setting
- Leave KsimName and KsimNode blank

You configure a connection from the source item to another item in the same DCS.

- Configure the target item with ItemNode and ItemName
- ToSubServer should be unchecked
- Put the full ItemNode/ItemName as SourceItemName
- Leave KsimName and KsimNode blank

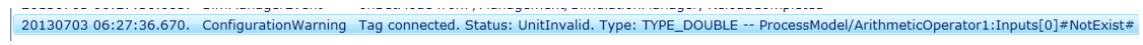
Figure 116 Dcs Loopback



8.2.2 DcsUnit

The unit conversion is done by the ModelServer. If it doesn't support the requested unit, there is a warning in the log file but the subscription completes and the unconverted value is used.

Figure 117 Warning unit does not exist



Link-to-Link (SimLink_XREF) does not currently (2013/07/03) support unit conversion.

Note

Beginning with 2.10, the SimLink will now issue an error for an invalid unit and disable the DCS item.

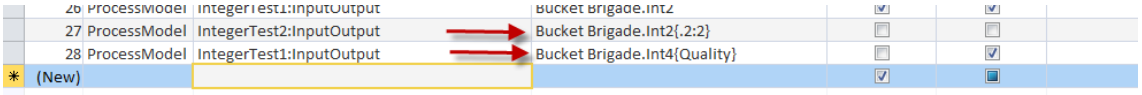
8.2.3 Attributes

This section describes the configuration options for the **item names** in KSIM_XREF and SimLink_XREF. It applies to the following fields:

- KSIM_XREF.ItemName
- SimLink_XREF.LocalName
- SimLink_XREF.RemoteName

Item names can include attributes for modifying the connection. These attributes appear immediately after the name (no spaces) within curly braces. An item may have multiple attributes if they are compatible with each other by separating them with a ;. {A1;A2;A3} for example

Figure 118 Itemnames with attributes



8.2.4 Binary Copy

Syntax: `BinaryCopy(ConversionType,NumberDcsElements)`

Where:

ConversionType is the desired type for the conversion (ModelServer or other SimLink). Note that this is not necessarily the data type of the tag in the partner. For example, if the DCS is storing `Float` values in a buffer of `Int16` values, you use `Float` as the **ConversionType** even if you are connecting to a `Double` in the partner application.

Valid values are:

- Double
- Float
- Int32

NumberDcsElements is the number of elements required in the DCS target.

`NumberDcsElements*sizeof(DCS-Type)` should equal the `sizeof(ConverstionType)`.

For example, connection a `Float` to an `INT` in **SoftLogixCom** requires 2 DCS Elements.

Examples:

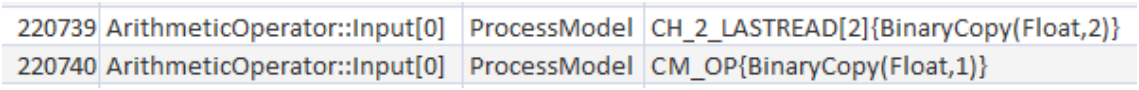
```
Int32Tag{BinaryCopy(Float,1)}
```

This connects a 4-byte `Float` from the partner to a 4-byte `integer` in the DCS.

```
Int16Array[2]{BinaryCopy(Float,2)}
```

This connects a 4-byte `Float` to 2 elements in a `Int16` array beginning at element #2. That is, the binary contents of a `float` are copied to/from `Int16Array[2]` and `Int16Array[3]`.

Figure 119 Binary copy



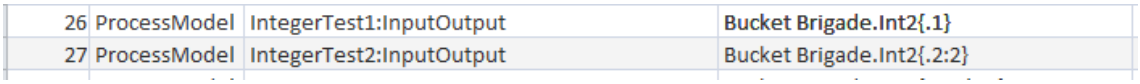
220739	ArithmeticOperator::Input[0]	ProcessModel	CH_2_LASTREAD[2]{BinaryCopy(Float,2)}
220740	ArithmeticOperator::Input[0]	ProcessModel	CM_OP{BinaryCopy(Float,1)}

8.2.5 Bit Addressing

The syntax for *bit address* is `{.N[:S]}` where N is the starting bit number and S is the number of bits.

Bits are number from 0 beginning with the low-order bit.

Figure 120 Bit addressing



26	ProcessModel	IntegerTest1:InputOutput	Bucket Brigade.Int2{.1}
27	ProcessModel	IntegerTest2:InputOutput	Bucket Brigade.Int2{.2:2}

`{.1}` indicates that the model is connecting to Bit #1 of the Item. This item will have a value of 0 or 1.

`{.2:2}` indicates that the model is connecting to Bits 2 and 3. This item will have a value of 0-3.

8.2.6 Item connection Quality

To connect to the **quality** associated with the data, use the attribute **Quality**. Not all SimLinks provide a quality. See *Are the data quality flags supported?* on page 163

Figure 121 Data quality

28	ProcessModel	IntegerTest1:InputOutput	Bucket Brigade.Int4{Quality}

9 Tips and troubleshooting

The Cross Reference list

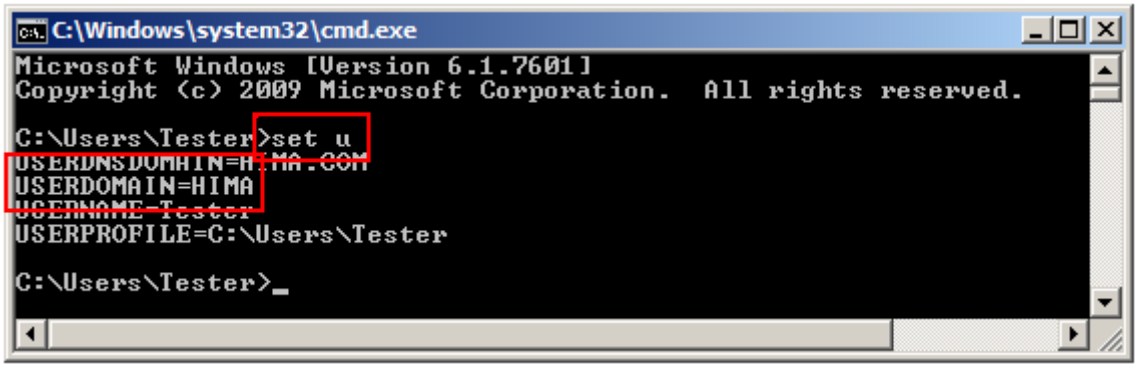
The Cross Reference List is essential when the link is up and running. One line in the list has two main items. One item is a DCS value. The other item is the K-Spice tag where this DCS value is received in the K-Spice simulator. Setting up the Cross Reference List is normally done with queries and tables in an Access database. After the Cross Reference List is created, it will be copied into the link's Access database. *Setting up the Cross reference list is not documented in this guide.* Creation of a Cross Reference List will be documented later in a separate guide.

9.1 DOS commands

set u

Will display the user and the domain.

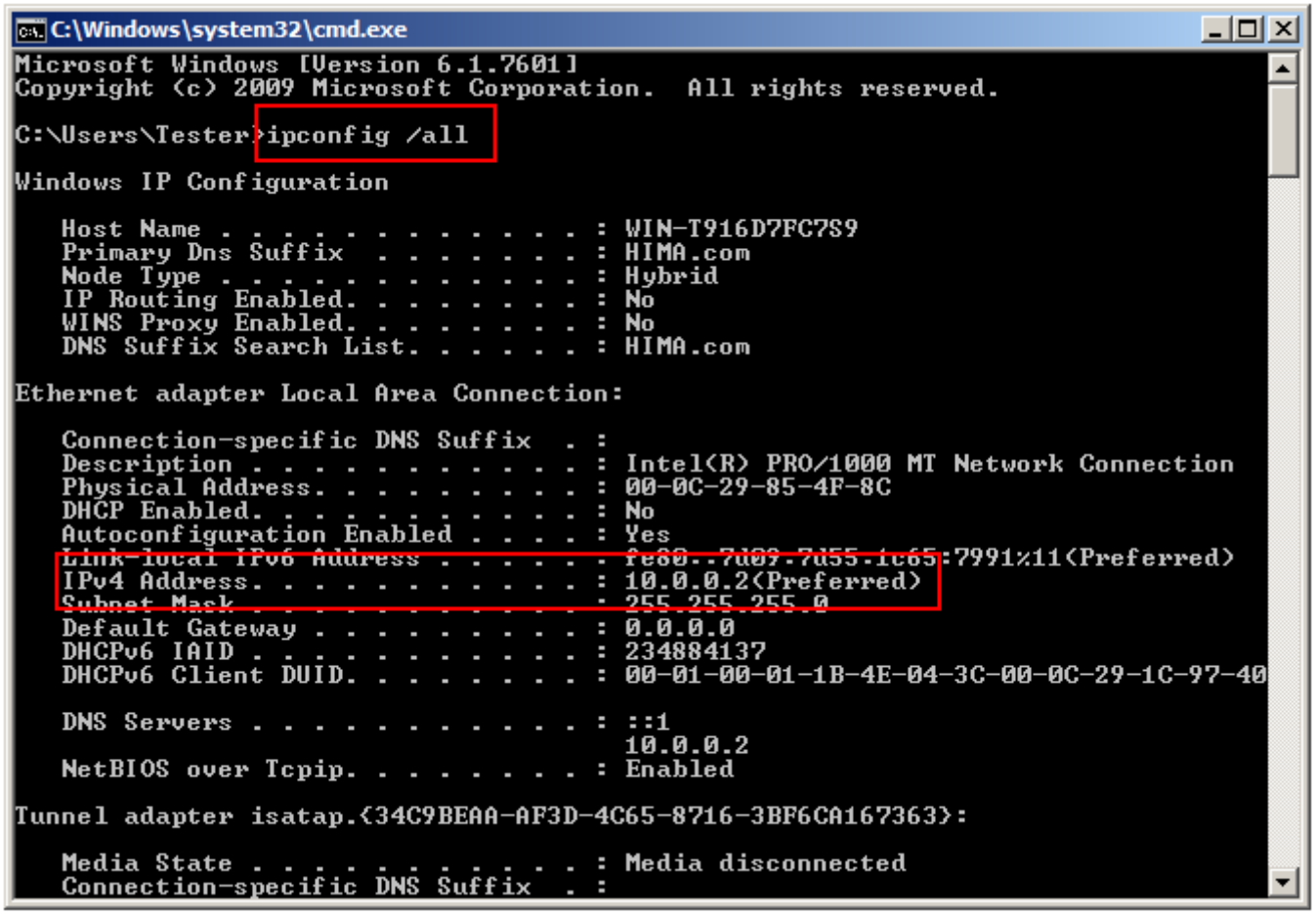
Figure 122 set u



ipconfig /all

Used to find the IP address of the machine. This is a virtual machine `10.0.0.3`, on a test net.

Figure 123 ipconfig /all



9.2 System Values.

This is a list of "system values" supported by the .NET SimLinks that are available for subscription.

rw is read from the SimLink and/or write to the SimLink

Component status

Table 7 Component status

Name	Notes
ComponentStatus.ExternalConnected	• boolean • read • Version: 2.5.0.15

Table 7 Component status (cont'd.)

	• boolean true indicates the DCS is connected
ComponentStatus.LinkHealthGood	• boolean • read • Version: 2.9.3.0 • boolean true if Link Health is good. Incorporates connection status, watchdog status and other items set by specific link
ComponentStatus.WatchdogAnyExpired	• boolean • read • Version: 2.9.3.0 • boolean true if any watchdog is currently expired. Can go back to false if a monitored value changes

Bucket

Table 8 Bucket

Name	Notes
SimLink.Bucket.TestBool	• boolean • read/write • Version: 2.5.0.15 • Bucket boolean value
SimLink.Bucket.TestDouble	• double • read/write • Version: 2.5.0.15 • Bucket boolean value
SimLink.Bucket.TestInt	• ont • read/write • Version: 2.5.0.15 • Bucket boolean value

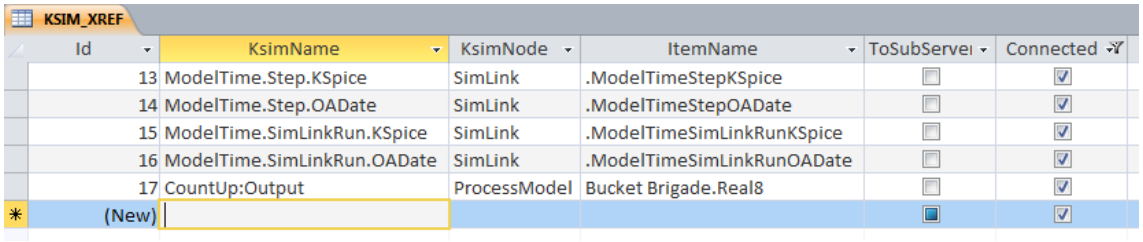
ModelTime

OADate values are calculated by first converting the K-Spice time to a date/time and then converting that to OADate. The K-Spice conversion uses the EpochStart configuration found in*The **Config** table. Standard keywords* on page 52

Table 9 ModelTime

Name	Notes
ModelTime.Step.KSpice	• double • read • Version: 2.9.5.0 • Last step time (K-Spice seconds) received from SimManager
ModelTime.Step.OADate	• double • read • Version: 2.9.5.0 • Last step time (OADate) received from SimManager
ModelTime.SimLinkRun.KSpice	• double • read • Version: 2.9.5.0 • Last step time (K-Spice seconds) acknowledged to SimManager
ModelTime.SimLinkRun.OADate	• double • read • Version: 2.9.5.0 • Last step time (OADate) acknowledged to SimManager

Figure 124 System Values. An Example.



In this example, they are being sent to DCS items

9.3 Disable ExplicitReadAfterConnect

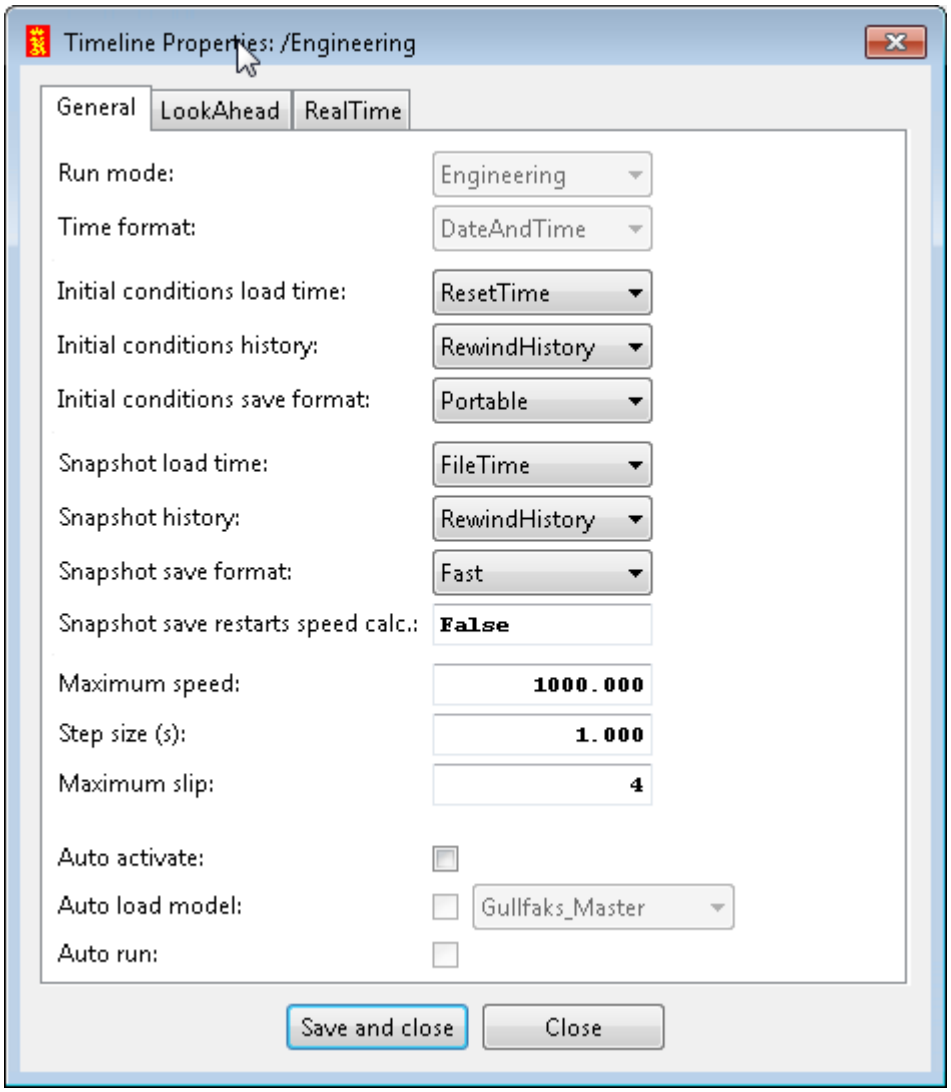
After testing for some days we have seen that some signals are not transfered correctly from ABB and into the link/model. One example is a signal sent from ABB with the value 255, looking at the same in the link/model shows the value FALSE, not even an Integer but a Boolean.

After trying to put the trouble signals in it's own UpdateRate group it looks better. But more and more signals failed and new UpdateRate groups were added. All trouble signals were put in it's own group.

After a lot of testing and checking a function in our link handling something called "ExplicitReadAfterConnect" was disabled, and it looks liks this have solved the problem. This function is defined in a common dll for all the links and there are obviously some links who require this function, but not the ABB link.

9.4 Changing Date and Time setting

Figure 125 Time format in Time line properties



The link was sending the date 01.01.1970 to ABB and this caused problems with the Alarms on the ABB side. We tried to acknowledge alarms and then save IC, but after reload of IC the alarms were unacked again.

The Time format was then changed from Engineering to DateAndTime to solve this. Right click and select properties on the Timeline Engineering. Only possible to change Time format when the Timeline is deactivated.

9.5 Reconnect support

SimLinks supporting Reconnect

Table 10 Reconnect

Name	Version
OpcDaCom	2.5.0.15
OpcHdaCom	2.5.0.15

9.6 K-Spice PID Controller connection

Connecting OPC server output to K-Spice PID Controller Output

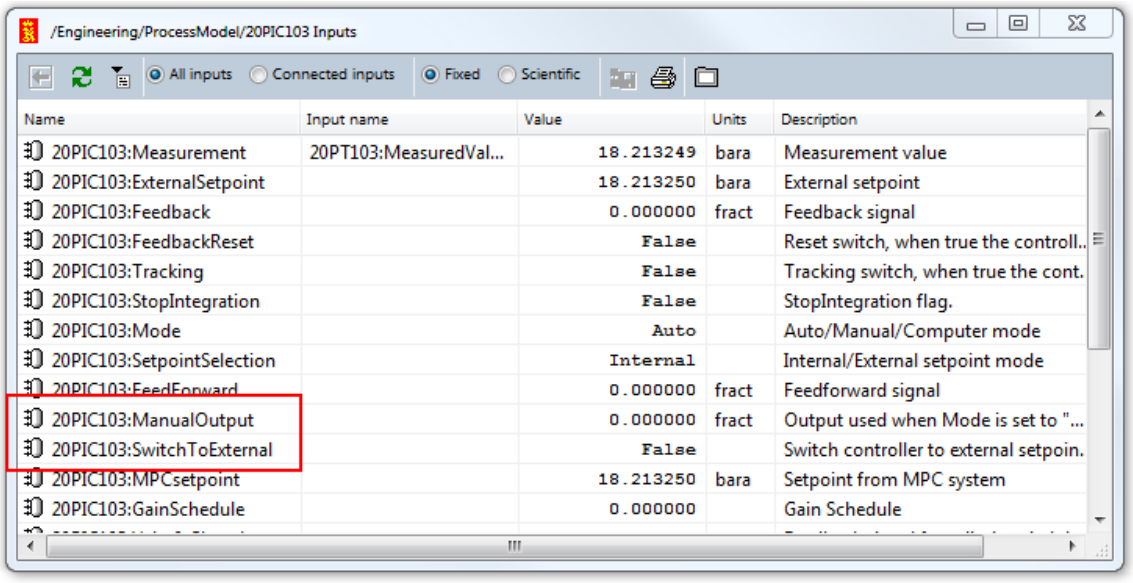
Go to start of metadata In some cases we have to set values of PID controller output from the remote control system via OPC server. An example of such a case is when the K-Spice model is used for optimization purpose by connecting it to an external optimizer using an OPC server

In such a case if we try to turn the K-Spice PID controller into manual mode and write to the PID ControllerOutput from the OPC server, it does not allow it as the "ControllerOutput" is an Output dataitem and the internal code prevents it from getting written through an OPC server value.

Inorder to make this work, the user needs to do the following.

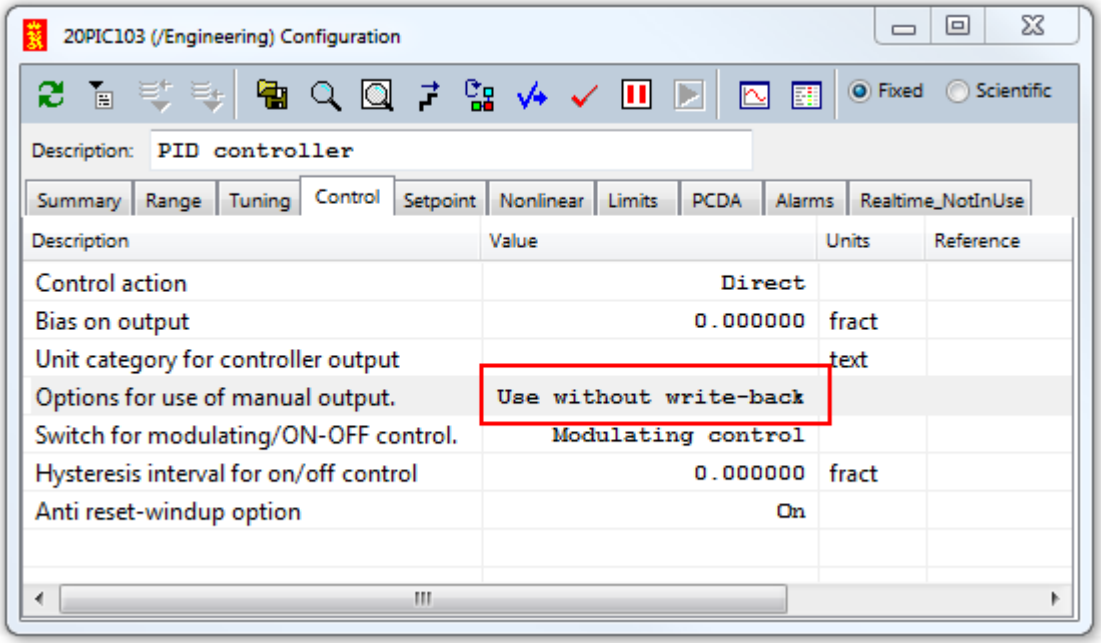
The OPC server output has to be mapped to PID Controller "ManualOutput" in the OpcDaCom database.

Figure 126 Pid controller output to connect to



In the K-Spice model we need to set the parameter "Options for use of manual output" to "Use not writeback"

Figure 127 PidController Control setting



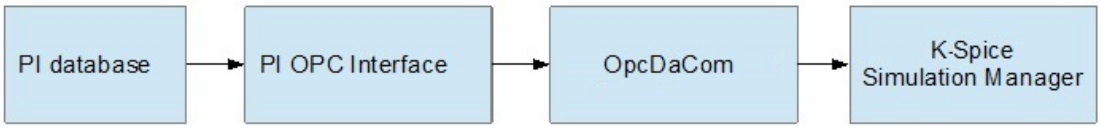
Please refer to OpcDaCom set up K-Spice for more information on OpcDaCom link set up.

This can also be done by setting the controller to **Computer mode** and connect the OPC signal to **ComputerOutput** of the controller.

9.7 PI Integration

The K-spice model will communicate to the PI historian through an OPC:

Figure 128 Dataflow from the PI database to K-Spice



Note

The client is responsible to ensure that the PI Database is configured in such a way so the values in PI are true reflections of the process state in real-time. OPC interface (or OPC server) provided by Client should have all requested data tags configured. Tag name list to be provided to Kongsberg to configure OpcDa and OpcHdaComs. The PI OPC Interface must support connection from external clients.

If PI is populated batch-wise, then ensure that complete set is populated with correct timestamp – partial update will create inconsistencies in the model where some data is realtime and some 30 minutes older for example. the K-Spice simulator assumes PI data is real time and needs a complete set of consistent data to run the simulator. See the figure below.

The following parameters are configurable aspects of the PI historical backfill service:

Figure 129 PI configuration parameters

Parameter	Type	Description	Default Value
Timer update interval	Integer	The update interval for processing of inserted OE records	60 seconds
Maximum number of records	Integer	The maximum number of records per batch for batch processing	100
Timestamp UTC logging	Boolean	Are timestamps sent through to the PI server in UTC or local time?	True
PI Server name	String	Name of the machine where the PI server resides. If this field is blank, then the PI SDK will automatically use the default server on the network.	N / A
Record insertion type	Enum	Insert, replace or discard. Determines the methodology used to add records to the PI database.	Insert
Timestamp tolerance	Integer	Timestamp tolerance when correlating related database records.	1 second
Insert missing PI points	Boolean	Allows the service to add pi-points if found to be missing from the PI database	True
New PI point compression	Boolean	If the service creates new PI points, is compression enabled?	False
Integer as float	False	Log integer values as float types	False

The PI server should be synchronized based on UTC time, this is done by configuring a scan class in the PI OPC Interface. The correct unit for each measurement must be provided in the Historian.

It should be checked to see if it is possible to get some kind of status signal/time tag from PI showing when the last batch-wise update was done in PI. KOGT recommends to configure this if possible. (This will be useful for user to see which dataset the model is working with)

If a data quality code tag is available as an independent OPC object, it is recommended to configure this in PI so it can be transferred to the K-Spice model.

If the K-Spice simulator should follow realtime data (with soft realtime constraints), there will be a certain delay between the data are transferred from the DCS and populated into PI. If the simulator is run in real time model, the link OpcDaCom must be configured to request data after PI has collected the full batch of tags. This can be done by increasing OpcDaCom reading interval (From 15 to 60s e.g.).

K-Spice will be configured to read data with a given timestamp. For the system that will read data online it is important to set a timestamp sufficiently high to allow transfer of a full data batch from DCS to PI database. The data points available in PI must reflect the real process signals, data filtering and averaged values should be handled with care.

For historical runs and offline calibration the OpcHdaCom is used. The PI OPC interface should have OPCHDA server for this to work.

9.8 Remotely Starting Controllers

In an Operator Training Simulator (OTS) setting, it is preferable to to launch all processes from a single instructor station. There are a few requirements to launching FCS and SCS test functions remotely. The user must have sufficient privileges, the instruction must use a suite of tools called **PSTools**, and the Yokogawa machine must share the `$admin` folder with the instructor.

The first task is to ensure the instructor machine has sufficient privileges. The account must be an administrator on the instructor machine and either be an administrator on the Yokogawa machine or the user must have access to an administrative account. If you do not want the user to have administrative privileges on the Yokogawa machine, you can create a text document which uses Windows PowerShell to create an encrypted string that contains the username and password of the account with administrative privileges. See also the guide to creating the text document which contains the secure account password under Tips and troubleshooting in section *Saving Credentials Securely* on page 151. Place the file in the `Kongsberg`directory of the Yokogawa machine for use by the final **PowerShell** script.

Next, download the **PSTools** suite from the Microsoft website:
`http://technet.microsoft.com/en-us/sysinternals/bb897553`
Place these in the `Kongsberg` folder of the instructor machine. The **PSExec** program is what will allow you to run a script on a remote machine. Check the web for more details on how to use **PSExec** For this procedure, it is easiest to copy/paste the following text into a batch file (i.e. `RunControllers.bat`):

`RunControllers.bat`

```
D:

cd Kongsberg

psexec \\computer [-u user] [-p password][-i [session][-w
directory] powershell [arguments]
```

Finally, you can ensure that the Yokogawa machine is sharing the `$admin` folder . See *Sharing the $Admin folder* on page 150 It requires you to edit the registry

After completing the steps above, the task is as simple as using a batch script to run **PSExec** (i.e. `RunControllers.bat`) which will in turn run a batch script (i.e `ControllersBegin.bat`) that refers to a **Powershell** script that (i.e. `ControllersRunAs.ps1`) contains information about the credentials and the final batch script (`FCSnSCS.bat`) that contains the commands that will run the various test functions.

9.9 Data organization on the OPC server

The data server has three divisions:

- Server - Contains all of the group objects
- Group - Maintains information about itself and contains and organizes the OPC items
- Item - Contains a unique identifier held within the group. The identifier acts as a reference for the individual data source, as well as value, quality, and timestamp information. The value is the data from the source. The quality status gives information about the device. The timestamp is the time that the data was retrieved

An OPC application accesses all items through the OPC group rather than through the item itself. The group also contains a specific update rate for itself, which tells the server at what rate to make data changes available to the OPC client. A deadband specific for each group tells the server to reject values if they have changed by less than a specified deadband percentage

Client software developers and users of these applications have greater flexibility in implementing a solution that is tailored to their needs because data is organized into groups and the naming, or tagging, of data points is determined by the client software. Grouping is beneficial in dealing with large sets of data sources because it provides greater organization of the data as well as easy reference to similar sets of data. In an OPC application, a tag gives a unique identifier to an I/O point. Based on the OPC specification, the client or server software is responsible for naming tags. The software can programmatically name tags or specify that the user name tags. This flexibility is a significant factor in the ability of client software to provide solutions that are tailored for high-channel-count applications.

Client software also specifies the rate at which the server supplies new data to the client. Because the server is responsible for data publication, the client software does not need to perform time-consuming data polling, which frees up more time for analysis and data logging. Moreover, the client software instead becomes a reactive object that waits for new data to arrive. Therefore, the client becomes event-driven and handles large sets of data much more efficiently.

The client also specifies deadbands on the server, which allows the client to determine which data is important and then disregard data that is insignificant. Deadband percentages reject values that do not change more than a certain percentage from the previous value recorded. By establishing moderate deadband values, the client receives only information about channels which the client deems essential. This prevents the client from being flooded with superfluous information. In this way, the client can monitor a much greater number of channels.

9.10 User accounts in Windows

9.10.1 Sharing the $Admin folder

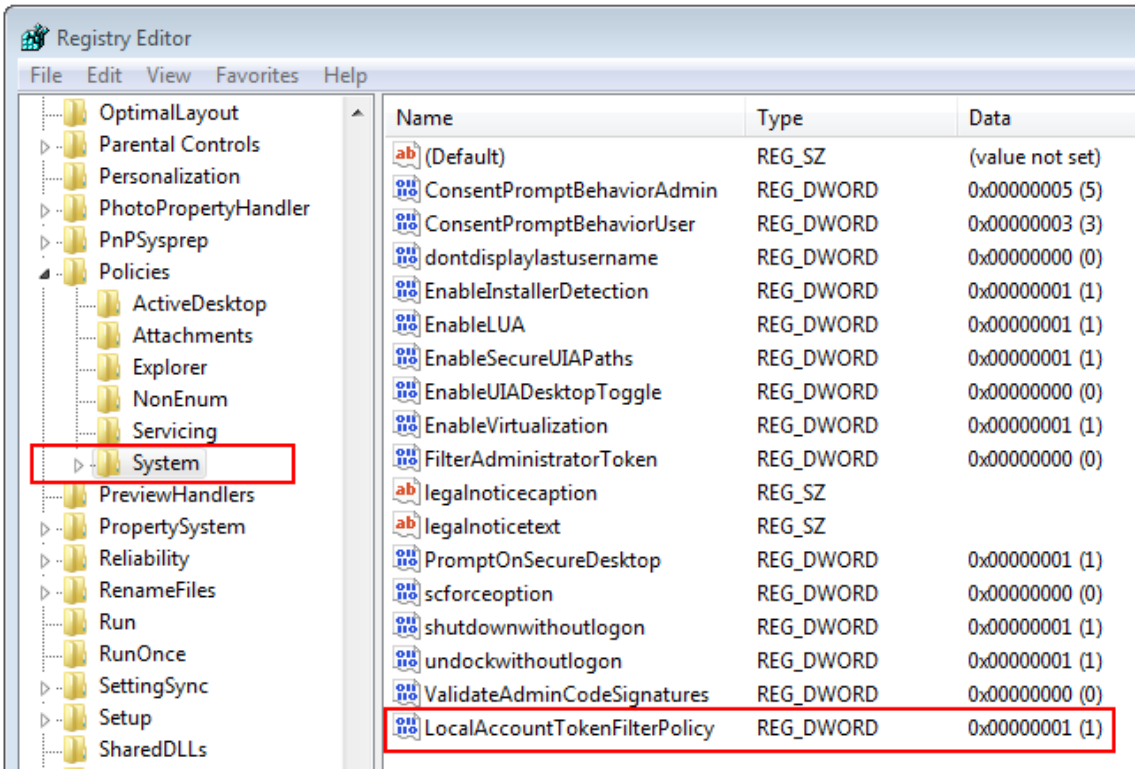
Access to some files and programs are required when performing tasks across multiple servers, such as remotely running a script or executable using PSExec. Even when sufficient credentials are provided, the $Admin folder is not shared.

1. Open the start menu
2. Search for RegEdit.exe
3. Open the program, it may ask for elevated privileges.
4. Once in the registry, locate the following directory:

   ```
   HKLocalMachine\Software\Microsoft\Windows
   \CurrentVersion\Policies\System
   ```

5. Create a new DWord called **LocalAccountTokenFilterPolicy**.
6. Double-click on the newly created DWord to edit its properties
7. Enter a value of 1 (Hexadecimal)
8. Click OK and exit RegEdit

Figure 130 Registry setting LocalAccountTokenFilterPolicy



For further information, please see this article: from which all information was taken.

9.10.2 Saving Credentials Securely

Various Windows tasks require administrative privileges but there are instance when the user may not have access to a sufficient account. It is possible to automate some procedures using a saved credential ensuring that all users can perform required tasks without exposing the password.

Creating the Secure String

The encrypted password must be saved on the desired server or PC. It cannot be created on a different machine and then transferred to another server or PC.

- Open **Powershell** as an **Administrator**

  Please note, this means using the **Run as Administrator** option, opening **Powershell** when logged in as an administrator is not sufficient.

- **Copy/Paste** the following command:

  ```
  read-host -assecurestring | convertfrom-securestring | out-file
  D:\Kongsberg\securestring.txt
  ```

  You may change the save directory in the final part of the command

- Enter the password that you would like to save.

  The text will appear as asterisks

- After pressing **Enter,** there should be a new text file called `securestring.txt`

When creating a PowerShell script (*.ps1), you can use the following variables to store credentials:

```
$username = "domain01\admin01"

$password = cat D:\Kongsberg\securestring.txt |
convertto-securestring

$cred = new-object -typename
System.Management.Automation.PSCredential -argumentlist
$username, $password
```

Using the `-credential` argument along with the newly created `$cred` variable should allow a script to run with sufficient privileges without prompt for a password. Be sure to change the `securestring.txt` directory if it was not saved in `D:\Kongsberg`

Example of the use of a file for credentials

The following example shows the use of the "securestring.txt" file for credentials when trying to use the Restart-Computer cmdlet on a remote computer.

RestartModel.ps1

```
$username = "Kongsberg\admin"

$password = cat D:\Kongsberg\securestring.txt |
convertto-securestring
```

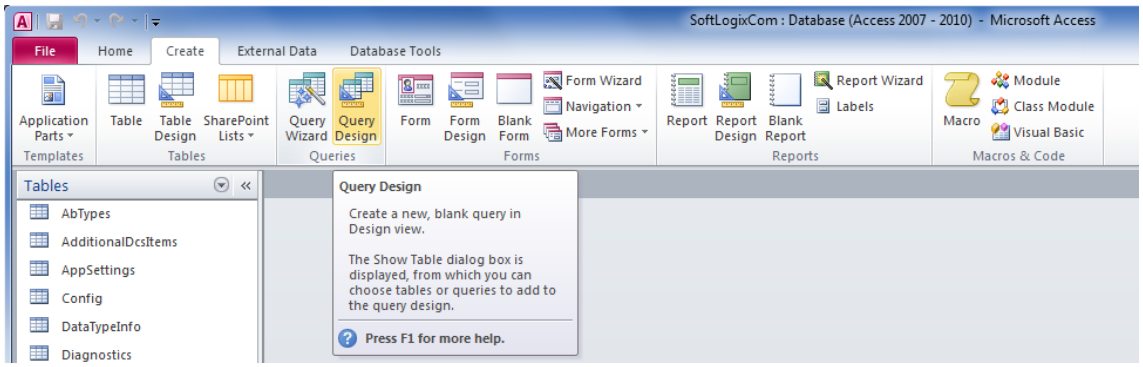
```
$cred = new-object -typename
System.Management.Automation.PSCredential -argumentlist
$username, $password

Restart-Computer -computer BGFOTS1MDL01 -credential $cred -force
```

9.11 SQL tips and tricks

9.11.1 Creating a SQL query in the SimLink database.

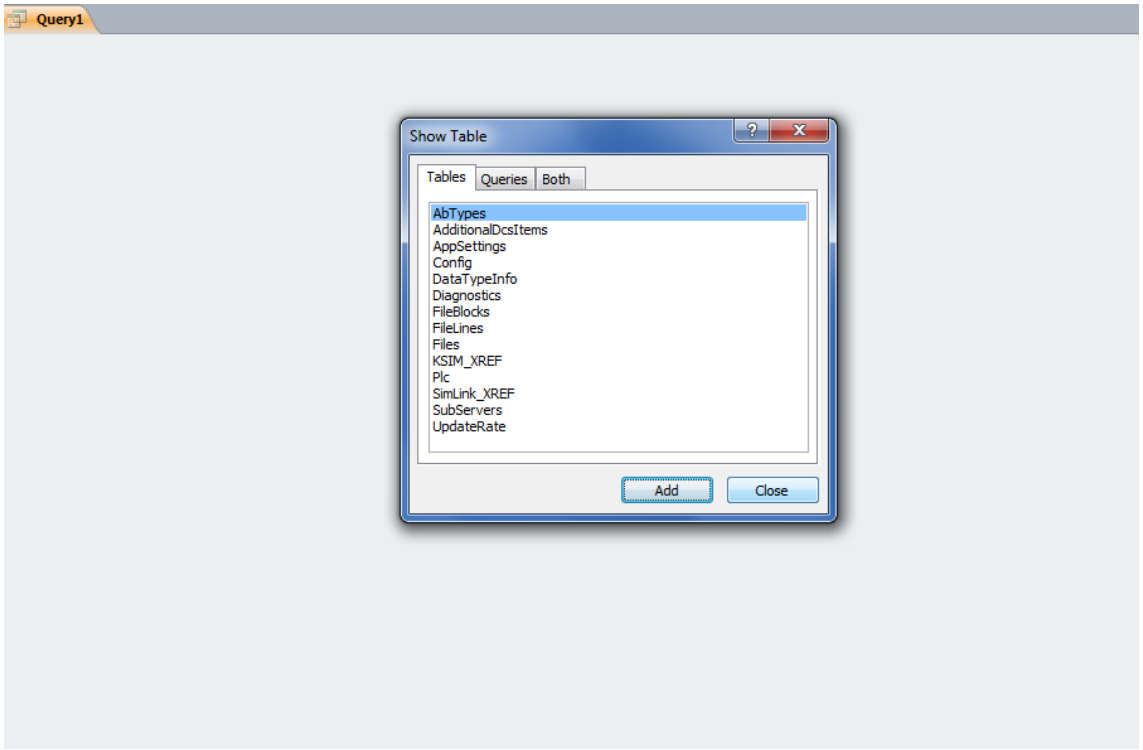
With the SimLink database open in Access, click on the Main Menu Create and then the button Query Design.

Figure 131 Create a new query



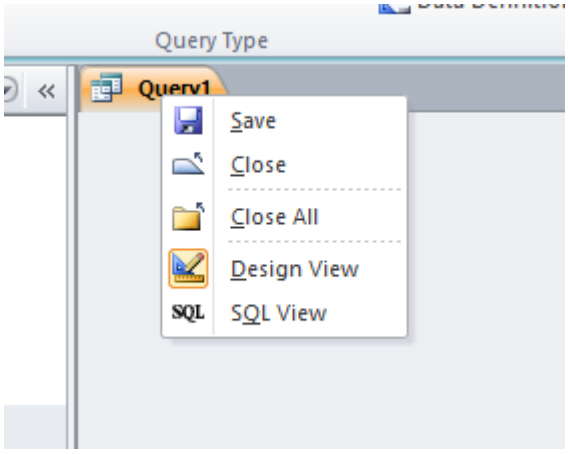
When the Query1 editor displays the Show Table dialog. Click Close

Figure 132 The Query1 editor



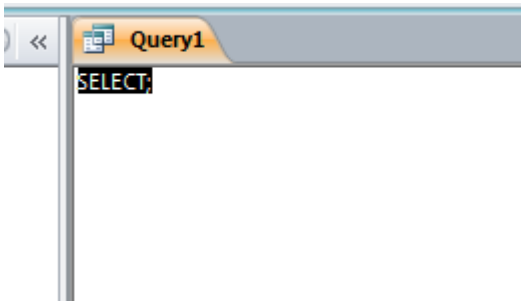
Right click on the Query1 tab and then click on SQL View.

Figure 133 Query1 SQL View



Your Query1 edit screen should look like this:

Figure 134 The query1 edit screen

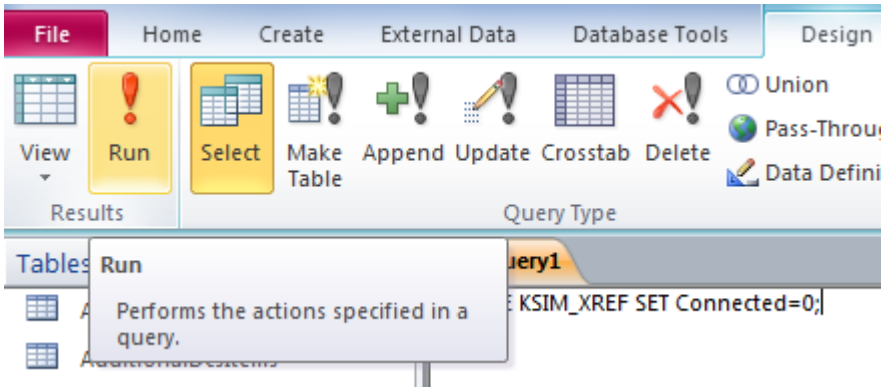


You can type in the desired SQL statement.

9.11.2 Running Query

Press the Run button to run the SQL query.

Figure 135 Run the query



9.11.3 Clear connected on all items

```
UPDATE KSIM_XREF SET Connected=0
```

9.11.4 Set connected for all items

Yes/No values (booleans) in Access typically have a value of 0 or -1.

```
UPDATE KSIM_XREF SET Connected=-1;
```

9.11.5 Clear connected for a particular item node

Notice that the ItemNode name is in single quotes

```
UPDATE KSIM_XREF SET Connected=0 WHERE
ItemNode='SIM1_AA_AA2/VRR_Train_1';
```

9.11.6 Finding Duplicates in KsimName

Use Query Wizard from Create Menu to find duplicates in KsimName. This can be useful to find two signals connected to valve control signal input or a motor start/stop signal.

Figure 136 Select table in Find duplicates query wizard

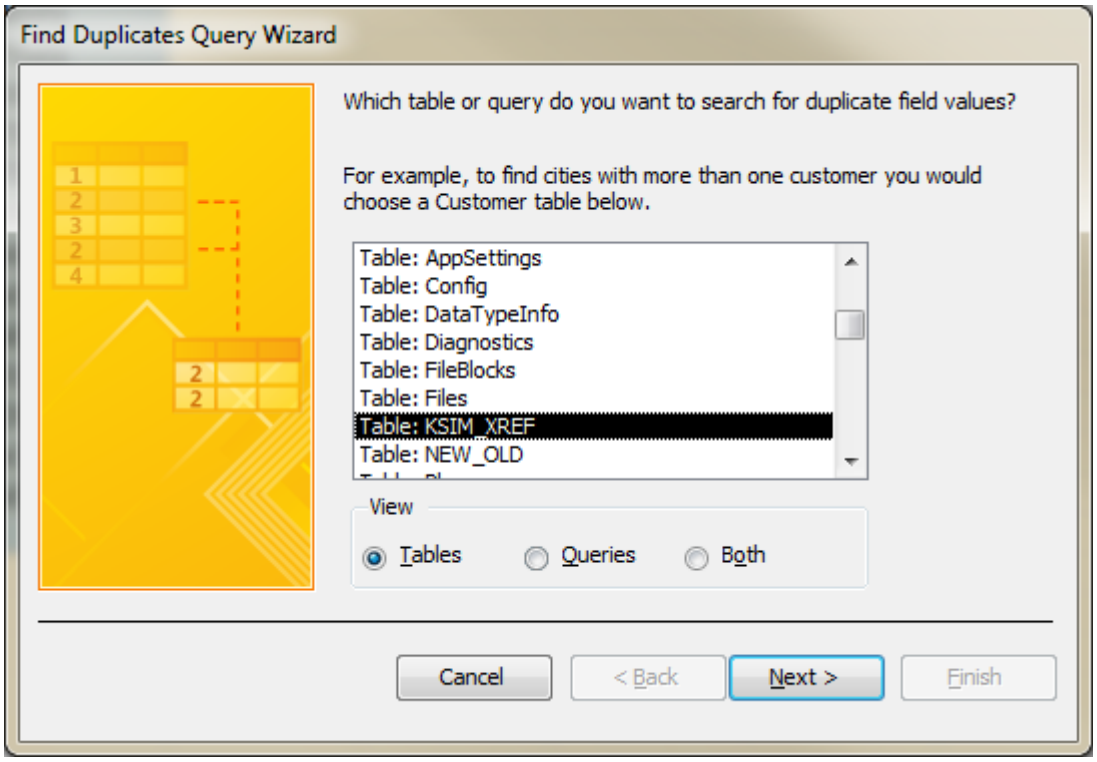
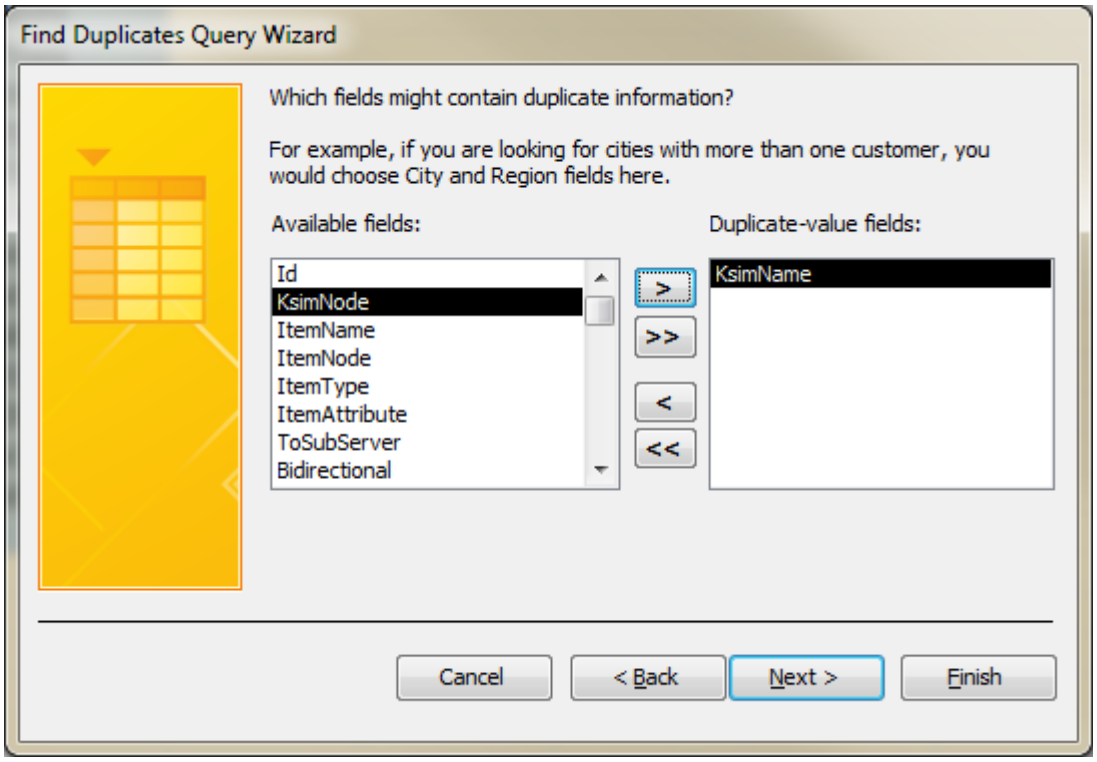


Figure 137 Select field in Find duplicates query wizard



Continue with next to add any more fields with the duplicate data. This will generate a table with duplicate values in KsimName.

9.12 Basic Terminology

Short explanations of the basic concepts in DCS interfacing

9.12.1 DCS

Distributed Control System

Distributed Control System is a computerized control system used to control the production line in the industry The entire system of controllers is connected by networks for communication and monitoring. The entire system of controllers is connected by networks for communication and monitoring.

9.12.2 OPC

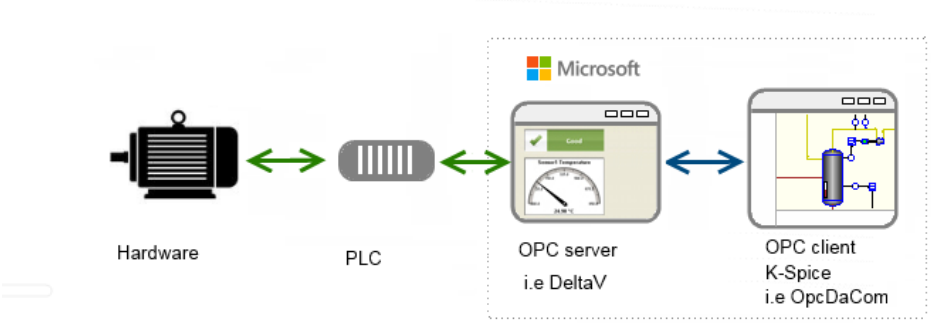
OPC = OLE for Process Control

OPC is a software interface standard that allows Windows programs to communicate with industrial hardware devices.

OPC was designed to provide a common bridge for Windows based software applications and process control hardware. Standards define consistent methods of accessing field data from plant floor devices.

OPC is implemented in server/client pairs. The OPC server is a software program that converts the hardware communication protocol used by a PLC into the OPC protocol. The OPC client software is any program that needs to connect to the hardware, such as an HMI. The OPC client uses the OPC server to get data from or send commands to the hardware.

OPC is designed to be an abstraction layer between industrial networks and proprietary PLC drivers. The OPC standard specifies the behavior that the interfaces are expected to provide to their clients; and the clients receive the data from the interfaces using standard function calls and methods. Consequently, as long as a computer analysis or data acquisition program contains an OPC client protocol, and an industrial device driver has an associated OPC interface, the program can communicate with the device. The specification also includes a protocol for working with data control systems and application databases, as well as online data access, alarm and event handling, and historical data access for all of these data sources.



OPC Data Access

OPC DA is a part of the OPC standard that handles realtime data. Other parts of the OPC standard handles historical data and alarms and events. There are three attributes associated with OPC DA. These are (1) a value, (2) the quality of the value, and (3) a timestamp. The OPC DA specification states that these three attributes have to be returned to an OPC client making a request. Therefore, if the data source is not capable of providing a timestamp, for example, the OPC DA server must create a timestamp.

OPC Historical Data

In addition to the OPC DA specification , the OPC Foundation also maintains the OPC HDA (Historical Data Acscess) specification. In contrast to the real time data that is accessible with OPC DA, OPC HDA allows access and retrieval of archived data.

OPC Alarm and Event handling

The OPC server also provides alarm and event handling to clients. Within a server, an alarm is an abnormal condition of special significance to the client - a condition associated with the state of the server, a group or an item within the server. For example, if a data source value that represents the real-world temperature of a mixer drops below a certain temperature, the OPC server will send an alarm to the application, so that the application will properly handle the low temperature. Events are detectable occurrences that are important to the server and client, such as system errors, system configuration changes, and operator actions.

9.12.3 OLE and COM

OPC = OLE for Process Control

OLE = Object Linking and Embedding

COM = Component Object Model

OLE

Object linking and embedding (OLE) is a Microsoft technology that facilitates the sharing of application data and objects written in different formats from multiple sources. Linking establishes a connection between two objects. OLE is a defined standard for the interface between the client and the server. The server has a set of routines it will answer. The client uses these routines to get data from the server. OLE is a part of more general standard for inter process communication, called Component Object Model, COM,.

9.12.4 DCOM

Distributed Component Object Model is a proprietary Microsoft technology for communication among software components distributed across networked computers. DCOM, which originally was called "Network OLE", extends Microsoft's COM, and provides the communication substrate under Microsoft's COM+ application server infrastructure.

9.12.5 PLC and PAC

Programmable Logic Controllers and Programmable Automation Controllers

Most suppliers of industrial data acquisition and control devices, such as Programmable Logic Controllers (PLCs) and Programmable Automation Controllers (PACs), are designed to work with the OPC Foundation standard.

9.12.6 Sockets

Two applicatons that interchange data, use sockets, also called ports, to set up the communication.

A socket is a physical, existing area inside a computer. The application that send data will set up electrical signals on a defined socket. The application that reads data will listen for electrical signals on the same socket. The two applications share the socket. Other applications can not access the socket. The socket is also referred to as the *port number* used for write or read. For two way communication, another socket is assigned for signals going in the other direction. If the signal is coming from another machine, the electrical signal is transferred via wires to a socket in the machine that has an application that read the data. The standard for Internet communication from socket on one machine

to socket on another machine, is TCP/IP. OPC does not use TCP/IP. A direct connection is established between the OPC server and the OPC client. Data is transferred with standard network protocols. The communication is based on the OPC standards.

The K-Spice applications and sockets

The different K-Spice applications communicate via sockets. The Simulation Manager will listen for logon requests from any application on socket 16000. If SimExplorer is started with Instance 2, SimExplorer will send a logon request to the Simulation Manager. Then SimExplorer will listen for logon accept from the Simulation Manager on port 16202. If SimExplorer is started with Instance 3, SimExplorer will listen for logon accept from the Simulation Manager on port 16203. If two instances of SimExplorer is running on the same machine, they must listen on two different sockets. The two instances of SimExplorer must be started with two different instance numbers.

9.12.7 SCADA

Supervisory control and data acquisition

The term SCADA usually refers to centralized systems which monitor and control entire sites, or complexes of systems spread out over large areas (anything from an industrial plant to a nation). Most control actions are performed automatically by RTUs or by PLCs. Host control functions are usually restricted to basic overriding or supervisory level intervention. For example, a PLC may control the flow of cooling water through part of an industrial process, but the SCADA system may allow operators to change the set points for the flow, and enable alarm conditions, such as loss of flow and high temperature, to be displayed and recorded. The feedback control loop passes through the RTU or PLC, while the SCADA system monitors the overall performance of the loop

9.13 FAQs

9.13.1 FAQs about all SimLinks

Does it run/has it been tested on Windows 7?

Yes

Are there any software environment restrictions?

SimLinks have the same installation and OS requirements as the rest of K-Spice.

How is the cross referencing done - is this a database, spreadsheet, flat file ?

SimLinks use an Access database to store configuration information

Can we have different tag groups running at different update times?

Yes

Does it support units conversion both ways or do the units need to be consistent?

SimLinks support unit conversion between a link and a ModelServer. Unit conversion between links (Link-to-Link) is not supported as of 2.8

Beginning with 2.10, the SimLink will now issue an error for an invalid unit and disable the DCS item.

Are there options on link start for synchronising the model tags with the DCS tags?

There are no user options for synchronising data. When a condition file or snapshot is loaded, all items send their data to any subscriber.

Can we edit the tag list dynamically or does this need a KSpice restart?

Dynamically, but you should understand the impact

If the SimLink has been started as an interactive process, there is a option to Reload Database. This option disconnects from the DCS and then reconnects in order to use any updated configuration information in the database. If the SimLink is for a DCS with a *persistent instance* (SoftLogixCom, OpcDaCom), then there should be no impact from the reload.

However, if the SimLink is for *a DCS that is started by the connection* (ProSimCom), then a fresh instance of the DCS is created. This instance may not have any values from a condition file load or from the results of recent model execution.

Does the SimLink run as a separate process or is this integrated in to model server/simulation manager?

Each instance of a SimLink is a separate process that can be run on the same machine as the model or on a different machine.

Is the communications task synchronized with the model execution?

As a separate process, DCS communication does not block ModelServer execution. SimManager does ensure that timing remains synchronized among all KSpice applications. That is, SimManager will not allow the model to take another step until all application (SimLinks, ModelServer, etc) acknowledge completion of the model step. (KSpice does support a configuration for how many steps may be pending.)

If the SimLink runs as a separate process, can this be put on a different core?

Yes, but . . .

Unless you've seen a performance whitepaper or Kongsberg has conducted extensive testing, it's recommended that you do not assign applications to specific cores. Microsoft has spent thousands of hours optimizing the operating system for automatic assignment.

If the communications task is synchronised, will delayed DCS response slow down the model execution speed?

As a separate process, data communication with the DCS does not block model execution. If the SimLink actively uses DCS timing to acknowledge steps back to SimManager, then a delayed response for timing will slow down model execution speed. Links that actively monitor DCS timing include ProSimCom and HimaCom.

Is it possible to measure the DCS response time?

Currently there is no diagnostic to do this.

What diagnostics are available in event that we get problems with the link? Are these accessible within the model?

The SimLink has Log, Watch and Status tabs that can be used to monitor some aspects of SimLink. For additional diagnostics See *The **Diagnostics** table* on page 60.

Other than link status, SimExplorer has no diagnostics for a SimLink. A model can subscribe to system values described in *System Values.* on page 137 and configure watchdogs according to *The **WatchdogItems** table.* on page 80

Does the link automatically reconnect after a disconnection? If so will it automatically poll all tags after a reconnection?

See also *Reconnect support* on page 142

9.13.2 FAQs about OPC DA based SimLinks

What data types are supported

- Integer
- Boolean
- Floating Point
- String

Are the data quality flags supported?

Yes.You may subscribe to the quality flag coming from the OPC server. See also the quality section in *Item connection Quality* on page 135

Do the systems need to be in the same domain/Workgroup? Would it be advisable to use a tunneller to prevent problems having to set up DCOM?

Preferred installation is OpcDaCom is on the same machine as OPC Server you are connecting to. This allows OpcDaCom to act as the OPC tunneller for K-Spice. If this is not feasible, the project will need to figure out DCOM security issues or purchase a tunneller.

Does it support exception based updates or are all tags polled?

Exception based

9.13.3 FAQs about OpcDaCom

How is client/server functionality handled? We need only client support. Is this the default?

This is client only

10 Switchover functionality

In K-Spice, common modules that receive signals from the DCS have the functionality to switch over from local control (i.e., controllers internal in K-Spice) to remote control (signals from the DCS). This switchover functionality is easily accessed by left-clicking on a symbol in K-Spice and bringing up the default faceplate that contains these switches.

10.1 DCS connection philosophy

The idea of connecting a K-Spice dynamic simulation model to a "soft" controller implementation of a DCS is to stimulate (provide equivalent signal representations) the system so it operates like it is actually connected to a real plant. This situation allows the model to run under DCS control with model variables being connected to the field inputs and outputs of the system.

The DCS link should include the following functionality:

1. Configuring the frequency (sampling rate) of a model variable.
2. Configuring the individual tolerance of a variable to determine if there has been significant changes in the model that requires the transfer of a variable.
3. Mapping of variables between the K-Spice model and DCS Links.
4. Scaling of variables.
5. Sending static values to selected DCS inputs.

The control signals required to be mapped are usually between transmitters, valves, electric motors, local control loops and emulated sequences. The safety signals from DCS Links are also usually connected to shutdown inputs on valves and electric motors. The below figure shows a simple diagram depicting two common connections between a K-Spice model and a DCS system, where the transmitter signal in the model is sent to the DCS control function block. The output of the DCS function block shown, a valve control signal, is sent back to the model to regulate the control valve.

10.2 DCS connectivity requirements

Connecting to controllers via Object Linking and Embedding for Process Control (OPC) supports the following commands:

1

2 Run

3 Freeze

4 Step I.e., step one time step, not necessary for most systems

5 Load snapshot/initial condition

6 Save snapshot

7 Set time I.e., if K-Spice needs to synchronize with an external clock

All the commands are reserved OPC commands within K-Spice. If the external system does not support these commands via OPC, the commands can be sent ‘outside’ the OPC link. The external soft controllers will also need to Set and Get model variables.

Note

Not all links use OPC

10.3 Common DCS interfacing modules

K-Spice modules which commonly have DCS I/O:

Table 11 The Common DCS interface module table

Module	to DCS	from DCS
Field Transmitter	MeasuredValue ControlSignalOut	
ControlValve	IsDefinedOpen IsDefinedClosed ValveStemPosition	RemoteControlSignalIn
MotorOperatedValve	IsDefinedOpen IsDefinedClosed ValveStemPosition	RemoteOpen RemoteClose RemoteStop
PulseControlledValve	IsDefinedOpen IsDefinedClosed ValveStemPosition	RemoteSetOpen RemoteSetClosed
TrueClosesValve	IsDefinedOpen IsDefinedClosed ValveStemPosition	RemoteOffOn
TrueOpensValve	IsDefinedOpen IsDefinedClosed ValveStemPosition	RemoteOnOff
PulseControlledAsynchronousMachine	MachineStatus	RemoteControlSignalIn RemoteSetOn RemoteSetOff

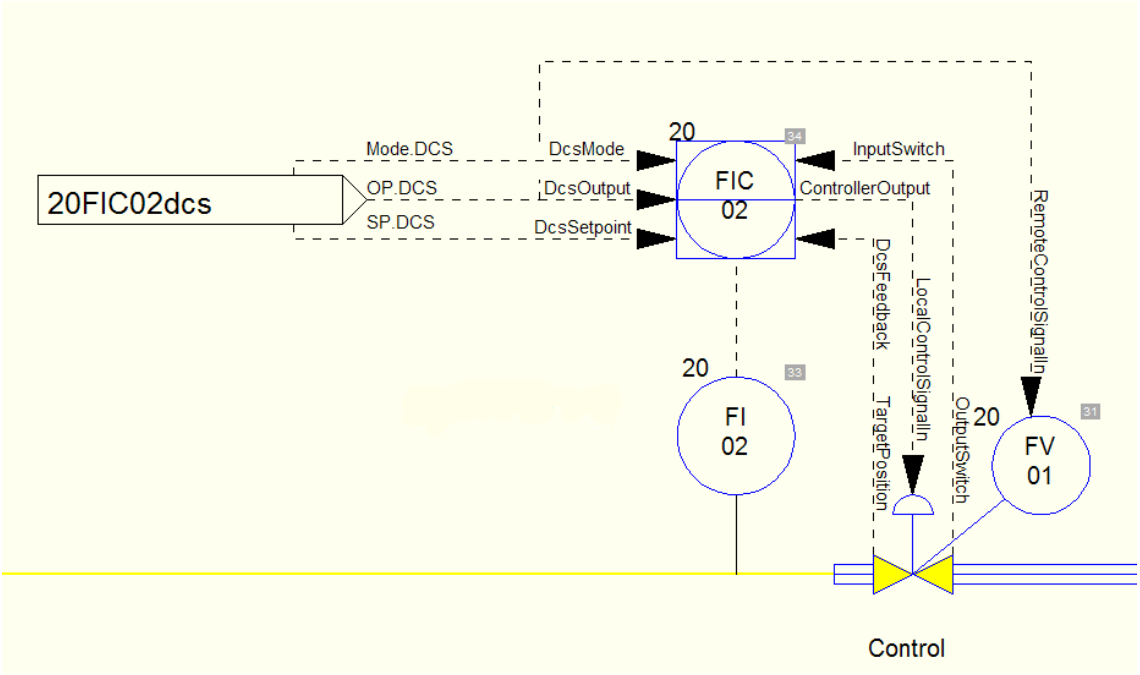
Table 11 The Common DCS interface module table (cont'd.)

Module	to DCS	from DCS
TrueStartsAsynchronousMachine	MachineStatus	RemoteOnOff RemoteControlSignalIn
TrueStopsAsynchronousMachine	MachineStatus	RemoteOffOn RemoteControlSignalIn
DcsPidController		DcsMode DcsSetpoint DcsOutput

10.4 Basic DCS Configurations

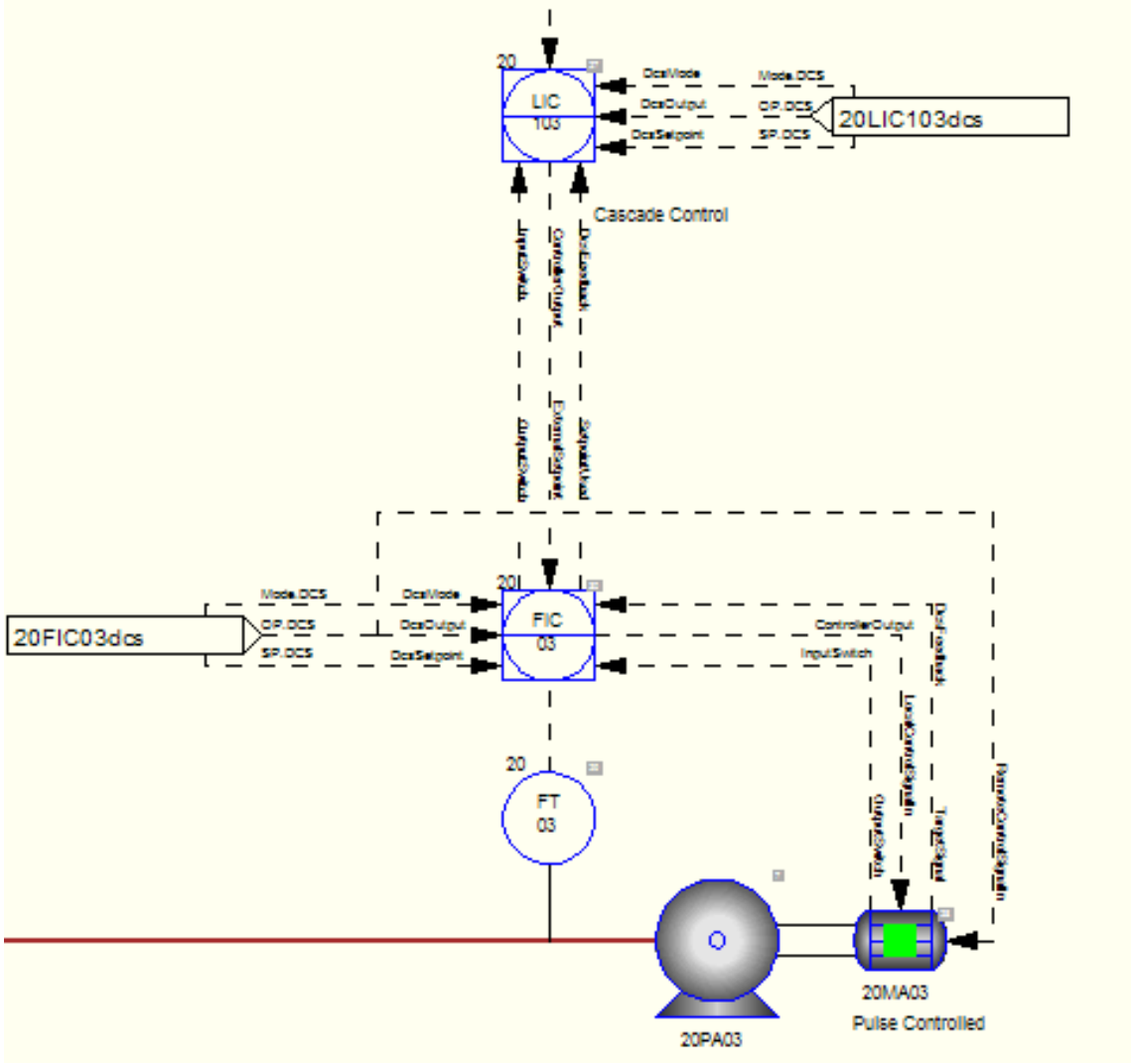
Here are keyword configurations for DCS used in K-Spice:

Figure 138 A basic control loop



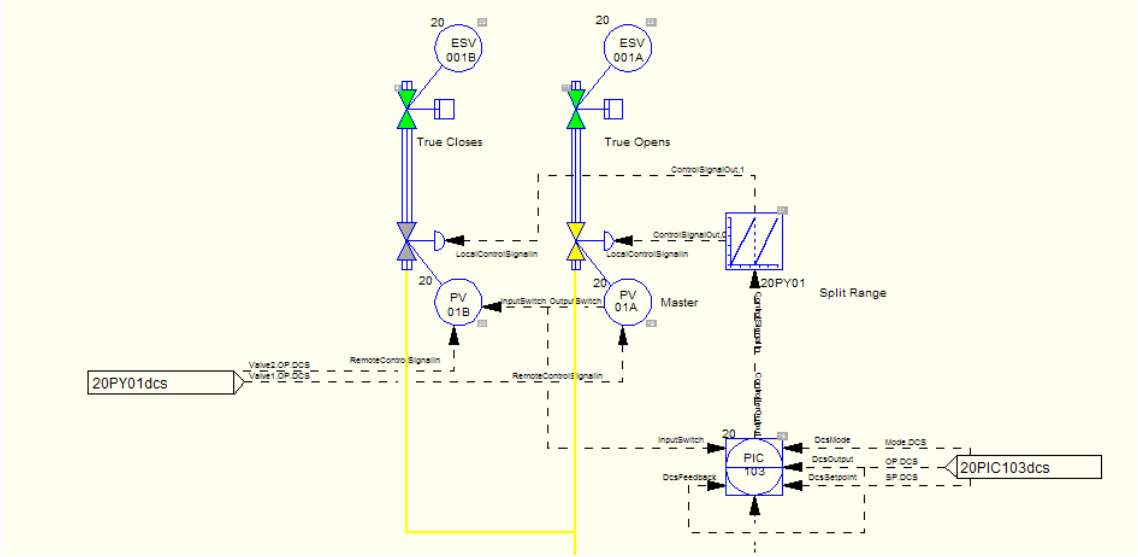
For configuration bring up the faceplate for the ControlValve (20FV01), the Controller (20FIC02) and the DCS Controller (20FIC02dcs).

Figure 139 A cascade control configuration



For configuration bring up the faceplate for the PulseControlledAsynchronousMachine (20MA03), the Master Controller (20LIC103), the slave controller (20FIC03), the DCS Master Controller (20LIC03dcs) and the DCS Slave Controller (20FIC03dcs).

Figure 140 A split range control configuration



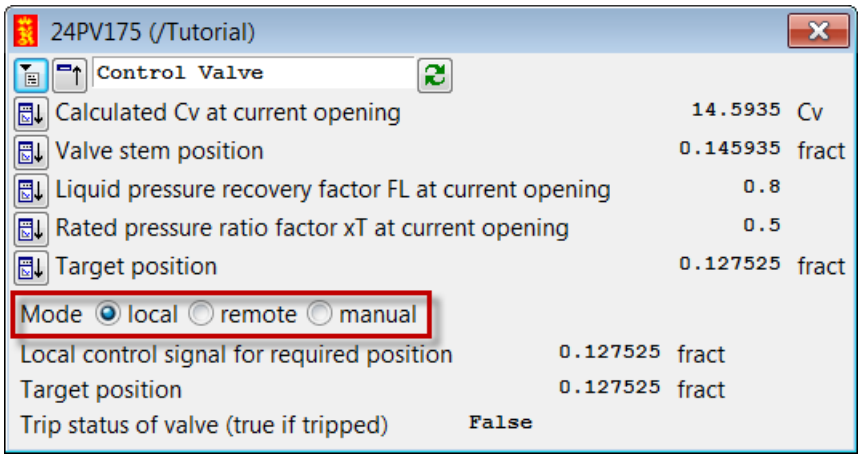
For configuration bring up the faceplate for the Master ControlValve (20PV01A), the Slave ControlValve (20PV01B), the Split Range Relay (20PY01), the Controller (20PIC103) and the DCS Controller (20PIC103dcs).

10.5 Typical switchover procedure

When the process model is first built, simple control logic is implemented to keep the model stable when not connected to the DCS. When connecting to a DCS, first all the I/O variable mappings are completed, i.e., matching the DCS tags to the process model variables.

Using a control valve as an example, to switch over the valve to remote DCS control, open the valve's faceplate and select the remote radio button, as shown in the below figure. If the DCS connection is successful, the valve will follow the DCS signal being sent to that valve.

Figure 141 A K-Spice valve faceplate with switching options



K-Spice valve faceplate with local, remote and manual signal switching options highlighted.