Wonderware

ABCIP DAServer User's Guide

3/4/13

Invensys Systems, Inc.
26561 Rancho Parkway South
Lake Forest, CA 92630 U.S.A.
(949) 727-3200

http://www.wonderware.com

For comments or suggestions about the product documentation, send an e-mail message to ProductDocumentationComments@invensys.com.

Contents

Welcome

The Wonderware ABCIP DAServer is a Microsoft Windows application that allows client applications direct and indirect access to Allen-Bradley families of ControlLogix, GuardLogix, FlexLogix, CompactLogix, SoftLogix 5800, MicroLogix, PLC-5, and SLC500 controllers.

The DAServer does not require any Rockwell Software RSLinx™ package.

Documentation Conventions

This documentation uses the following conventions:

Convention	Used for
Initial Capitals	Paths and file names.
Bold	Menus, commands, dialog box names, and dialog box options.
`Monospace`	Code samples and display text.

Technical Support

Wonderware Technical Support offers a variety of support options to answer any questions on Wonderware products and their implementation.

Before you contact Technical Support, refer to the relevant section(s) in this documentation for a possible solution to the problem. If you need to contact technical support for help, have the following information ready:

- The type and version of the operating system you are using.
- Details of how to recreate the problem.
- The exact wording of the error messages you saw.
- Any relevant output listing from the Log Viewer or any other diagnostic applications.
- Details of what you did to try to solve the problem(s) and your results.
- If known, the Wonderware Technical Support case number assigned to your problem, if this is an ongoing problem.

Chapter 1

Getting Started

The DAServer is one component of a software system that connects your software application with information on the factory floor.

This DAServer documentation covers only the information you need to configure and run the DAServer component. See the documentation that comes with the related components for details on their operation. You can find installation instructions in a help file on the distribution CD.

You use the DAServer Manager to configure, activate, and troubleshoot the DAServer. The DAServer Manager is located in the System Management Console (SMC).

This documentation describes some of the features of the DAServer Manager. See the *DAServer Manager User's Guide* to find more information on:

- Global parameters
- Configuration sets
- Time zone features
- Icon definitions
- Activation/deactivation
- Configuring as a service
- Importing/exporting device items
- Standard diagnostics

You can troubleshoot problems with the DAServer using the ArchestrA Log Viewer, a snap-in to the SMC. See the Log Viewer help file to find information on:

- Viewing error messages.
- Determining which messages are shown.
- Bookmarking error messages.

You may also be able to troubleshoot problems using your client application, such as the Wonderware InTouch HMI software. The client application can use system device items to determine the status of nodes and the values of some parameters.

Before You Begin

Before configuring the DAServer, verify the following items:

- A PC is set up with the necessary network cards, and is connected to the necessary networks.
- The Windows administration account is created or identified.
- The DAServer and any other Wonderware software such as the DAServer Manager is installed with the proper licenses. For more information, see the License Utility documentation on the distribution CD.
- The client software is installed.
- The device(s) is/are connected (networked) and, if necessary, programmed.

Before configuring the DAServer, you should know:

- The device network configuration and addresses.
- Which data items are needed for the client application.
- The device name/topic name/group name.
- The desired update intervals.

Supported Client Protocols

Client applications connect to the ABCIP DAServer using:

- OPC
- SuiteLink
- DDE/FastDDE

Important: On Windows Vista and later operating systems, Local DDE is supported only when the DAServer is configured as "Not a Service" and activated from its executable file or launched from InTouch. Local DDE is not supported when the DAServer is activated from the System Management Console (SMC)

Supported Device Protocols

The ABCIP DAServer connects to supported controllers across an Ethernet/IP network using the Common Industrial Protocol (CIP).

- The **Ethernet Industrial Protocol (EtherNet/IP)** is an open industrial networking standard using CIP on top of a TCP/IP suite as its lower-level transport and data-link vehicle.
- The **Common Industrial Protocol (CIP)** is a common application-layer protocol used by EtherNet/IP, ControlNet, and DeviceNet. The following network communication protocols use CIP as their top application layer:
  - **ControlNet** - The ControlNet protocol is a real-time deterministic control-layer networking protocol using CIP as its top application layer.
  - **DeviceNet** - The DeviceNet protocol is an open lower-level networking standard using CIP on top of Controller Area Network (CAN) to connect industrial field devices to controllers and computers.

Note: The ABCIP DAServer does not implement the DeviceNet protocol internally. Device data from the DeviceNet network needs to be mapped to the appropriate controllers before the ABCIP DAServer can access them.

For supported non-Logix controllers, Allen-Bradley uses the Programmable Controller Communications Commands (PCCC) application-layer protocol.

This is accomplished by using the DF1 protocol in its lower data-link layer. The DF1 protocol supports features for both D1 (data transparency) and F1 (two-way simultaneous transmission with embedded responses) subcategories of ANSI x3.28 specifications.

This protocol is encapsulated in CIP when used by the DAServer to communicate with the following supported Allen-Bradley controllers:

- PLC-5 controllers
- SLC500 controllers
- MicroLogix controllers

Supported Device Networks

The ABCIP DAServer communicates with supported devices either directly or indirectly across the following device networks:

- ControlNet
- Data Highway 485 (DH485)
- Data Highway Plus (DH+)
- DeviceNet
- Ethernet

Supported Devices

The ABCIP DAServer will provide direct and indirect connectivity to the following Allen-Bradley controllers:

- ControlLogix Controllers
- GuardLogix Controllers
- SoftLogix 5800 Controllers
- CompactLogix Controllers
- FlexLogix Controllers
- MicroLogix Controllers
- PLC-5 Controllers
- SLC500 Controllers

Note: The Optimize for Startup mode is not supported on Logix Controllers using firmware version 21 or above. For more information, see "Logix5000 Optimization Mode" on page 111.

ControlLogix Controllers

- All ControlLogix processors (1756-series processors) directly accessible from the Ethernet using the ControlLogix Ethernet or EtherNet/IP Bridge module (1756-ENET, 1756-ENBT, 1756-EN2T, or 1756-EWEB) through the backplane.
- All ControlLogix processors (1756-series processors) accessible through the ControlLogix Gateway from the ControlNet network by means of the ControlLogix ControlNet Bridge module (1756-CNB/CNBR or 1756-CN2/CN2R).

- ControlLogix 1756-RM and 1757-SRM processors directly accessible from the Ethernet using the ControlLogix Ethernet or EtherNet/IP Bridge module (1756-EN2T for the 1756-RM or 1756-ENBT for the 1757-SRM) or accessible through the ControlLogix Gateway from the ControlNet Bridge module (1756-CN2R).

The 1756-EWEB enhanced Web-server module provides both CIP communications and Internet browser web-services. ABCIP DAServer supports ONLY CIP communications.

The ABCIP DAServer is capable of accessing multiple ControlLogix processors in a single chassis.

GuardLogix Controllers

- All GuardLogix Integrated Safety processors (1756-LSP & 1756-L6xS) directly accessible from the Ethernet using the ControlLogix Ethernet or EtherNet/IP Bridge module (1756-ENBT or 1756-EWEB) through the backplane.
- All GuardLogix Integrated Safety processors (1756-LSP & 1756-L6xS) accessible through the ControlLogix Gateway from the ControlNet network by means of the ControlLogix ControlNet Bridge module (1756-CNB/CNBR or 1756-CN2).

SoftLogix 5800 Controllers

- All SoftLogix 5800 controllers (1789-series) directly accessible from the Ethernet on an industrial or desktop PC.

CompactLogix Controllers

- All CompactLogix processors (1769/1768-series) directly accessible from the Ethernet using the integrated EtherNet/IP port.
- All CompactLogix processors (1769/1768-series) accessible from the Ethernet via the EtherNet/IP interface module for CompactLogix/MicroLogix (1761-NET-ENI).

FlexLogix Controllers

- All FlexLogix processors (1794-series) accessible from the Ethernet using the EtherNet/IP communications daughter-card (1788-ENBT).

- All FlexLogix processors (1794-series) accessible through the ControlLogix Gateway from the ControlNet network by means of the ControlNet daughter-card (1788-CNC).

MicroLogix Controllers

- All MicroLogix 1100 processors (1763-series) and 1400 processors (1766-series) directly accessible from the Ethernet using the integrated EtherNet/IP port.
- All MicroLogix 1000/1200/1500 processors accessible from the Ethernet via the Ethernet/IP interface module for CompactLogix/MicroLogix (1761-NET-ENI) series B or higher.
- All MicroLogix 1000/1200/1500 processors accessible from the DH485 network using the RS-232C-to-DH485 Advanced Interface Converter module (1761-NET-AIC) to connect to the Data Highway Plus network through a DH+-to-DH485 Bridge module (1785-KA5) and routed through the ControlLogix Gateway by means of the ControlLogix DH+/RIO Bridge module (1756-DHRIO) to Ethernet.

PLC-5 Controllers

- All PLC-5 processors (1785-series) accessible through the ControlLogix Gateway from the Data Highway Plus network by means of the ControlLogix DH+/RIO Bridge module (1756-DHRIO)
- All ControlNet-capable PLC-5 processors (1785-series) accessible through the ControlLogix Gateway from the ControlNet network by means of the ControlLogix ControlNet Bridge module (1756-¬CNB/CNBR).

SLC500 Controllers

- All SLC 5/05 processors accessible from the Ethernet using the built-in EtherNet/IP interface.
- All SLC 5/03, /04 processors (1747-series) accessible from the Ethernet using the EtherNet/IP interface module (1761-NET-ENI).
- All SLC 5/04 processors (1747-series) accessible through the ControlLogix Gateway from the Data Highway Plus network by means of the ControlLogix DH+/RIO Bridge module (1756-DHRIO).

- All SLC 5/03, /04, /05 processors (1747-series) linked to the SLC500 ControlNet RS-232 interface module (1747-KFC15) accessible through the ControlLogix Gateway from the ControlNet network by means of the ControlLogix ControlNet Bridge module (1756-CNB).
- All SLC 5/03, /04, /05 processors (1747-series) accessible from the DH485 network using the RS-232C-to-DH485 Advanced Interface Converter module (1761-NET-AIC) to connect to the Data Highway Plus network through a DH+-to-DH485 Bridge module (1785-KA5) and routed through the ControlLogix Gateway by means of the ControlLogix DH+/RIO Bridge module (1756-DHRIO) to the Ethernet.

While primarily intended for use with Wonderware InTouch®, the DAServer may be used by any Microsoft Windows program capable of acting as a DDE, FastDDE, SuiteLink™, or OPC client.

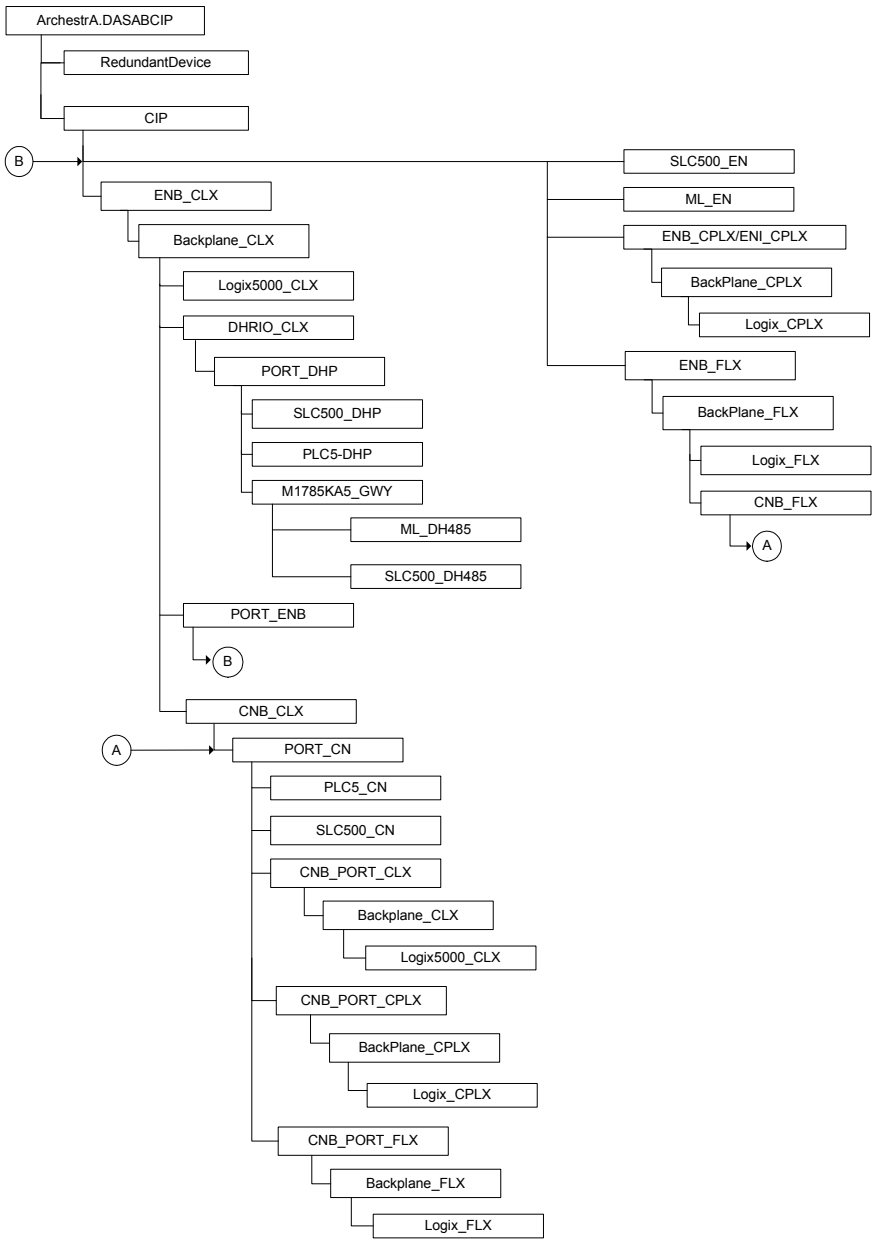
Supported Topologies

This ABCIP DAServer communicates with supported Allen-Bradley families of CompactLogix, ControlLogix, FlexLogix, GuardLogix, MicroLogix, PLC-5, SLC500, and SoftLogix 5800 controllers across:

- ControlNet
- Data Highway 485 (DH485)
- Data Highway Plus (DH+)
- DeviceNet
- Ethernet

Before attempting to configure your DAServer, you should determine the hierarchical structure of your network/controller environment.

See the following ABCIP DAServer Software Topology Diagram:.



Dual ENB Routing Topology

As of version 5.0, DASABCIP is able to connect to a ControlLogix rack via an ENB module on the same subnet, and route from the backplane to a second ENB module on a different subnet.

The ControlLogix rack becomes a router between two subnets. DASABCIP will be able to connect to the following controllers and devices on a second Ethernet subnet:

- ControlLogix

- CompactLogix
- FlexLogix
- MicroLogix
- SLC500

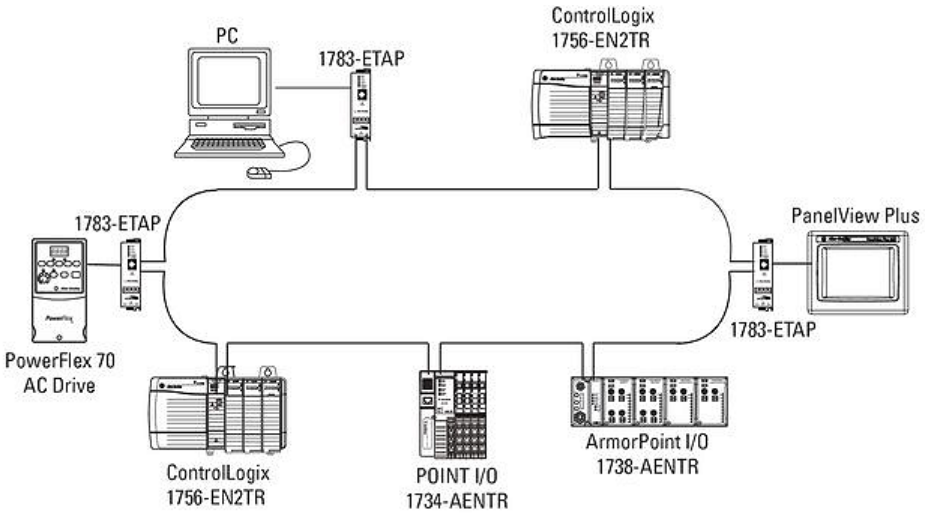
Note: Only one Ethernet subnet hop is supported.

Device-Level Ring (DLR) Topology

DLR is network technology provided by Rockwell Automation to enable Ethernet ring network topologies at the device level. The DLR protocol enables Ethernet devices to connect directly to neighboring nodes through dual network ports to form a ring topology.

When a DLR detects a break in the ring, it provides alternate routing of the data to help recover the network.

As of version 5.0, DASABCIP provides data connectivity to other supported controllers connected on the same DLR network. The computer where DASABCIP is installed must be connected to the DLR network via the 1783-ETAP device. Following is a sample DLR topology:



Windows Firewall Considerations

If the DAServer runs on a computer with a firewall enabled, a list of application names or port numbers must be put in the firewall exception list so the DAServer can function correctly. The DAServer installation program makes the required entries in the firewall exception list for you.

The following applications are added in to the firewall exception list on the computer where the DAServer run-time application is installed:

- DASABCIP.exe
- aaLogger.exe
- DASAgent.exe
- dllhost.exe
- mmc.exe
- OPCEnum.exe
- Slssvc.exe

The following port numbers are added to the firewall exception list on the computer where the DAServer run-time application is installed:

- 5413 - TCP port for slssvc.exe
- 445 - TCP port for file and printer sharing
- 135 - TCP port for DCOM

The following applications are added in to the firewall exception list on the computer where the DAServer Manager (configuration part) is installed:

- aaLogger.exe
- dllhost.exe
- mmc.exe

The following port numbers are added in to the firewall exception list on the computer where the DAServer Manager (configuration part) is installed:

- 445 - TCP port for file and printer sharing
- 135 - TCP port for DCOM

Un-installing the DAServer does not remove the firewall exception list entries. You must delete the firewall exception list entries manually. For more information on how to do this, see your firewall or Windows security documentation.

Chapter 2

Setting Up Your DAServer

This section describes the procedures required to set up the ABCIP DAServer for use with the supported device gateways and communication-interface modules.

Many high-level functions and user-interface elements of the DAServer Manager are universal to all DAServers. These universal functions are described in detail in the *DAServer Manager User's Guide*.

See the *DAServer Manager User's Guide* to find more information on:

- Global parameters
- Configuration sets
- Time zone features
- Icon definitions
- Activation/deactivation
- Configuring as a service
- Importing/exporting device items
- Standard diagnostics

The ABCIP DAServer uses the Common Industrial Protocol (CIP) to communicate with all devices across an Ethernet network.

The ABCIP Hierarchy in the DAServer starts with the PORT_CIP Object, followed by selected supported EtherNet/IP communication bridge/interface modules. The communication bridge module indirectly allows the DAServer to access the following networks to which the devices are connected:

- ControlNet Network
- Data Highway 485 (DH485) Network
- Data Highway Plus (DH+) Network
- DeviceNet Network
- Ethernet Network

Note: Before attempting to configure your DAServer, you should determine the hierarchical structure of your network/controller environment.

Checklist for Setting up the ABCIP DAServer

If you are setting up a DAServer for the first time, perform the following tasks in the order listed:

1. Review the items described in "Before You Begin" on page 12.
2. Locate the DAServer in the System Management Console (SMC). See "Finding Your DAServer in the SMC" on page 24.
3. Configure the global parameters. See the *DAServer Manager User's Guide*.
4. Add a Port. See "Adding a Port" on page 28.
5. Add and configure ports and devices. See applicable configuration set-up in "Configuring ABCIP DAServer Ports" on page 31.
6. Add one or more device groups. See "Device Group Definitions" on page 88.
7. Add device items. See "Device Item Definitions" on page 90.
8. Activate the DAServer. See "Configuring the DAServer as Service" on page 100.
9. Access data from the client, see "Accessing the Data in Your DAServer" on page 107.
10. Troubleshoot any problems. See "Troubleshooting" on page 195.

Finding Your DAServer in the SMC

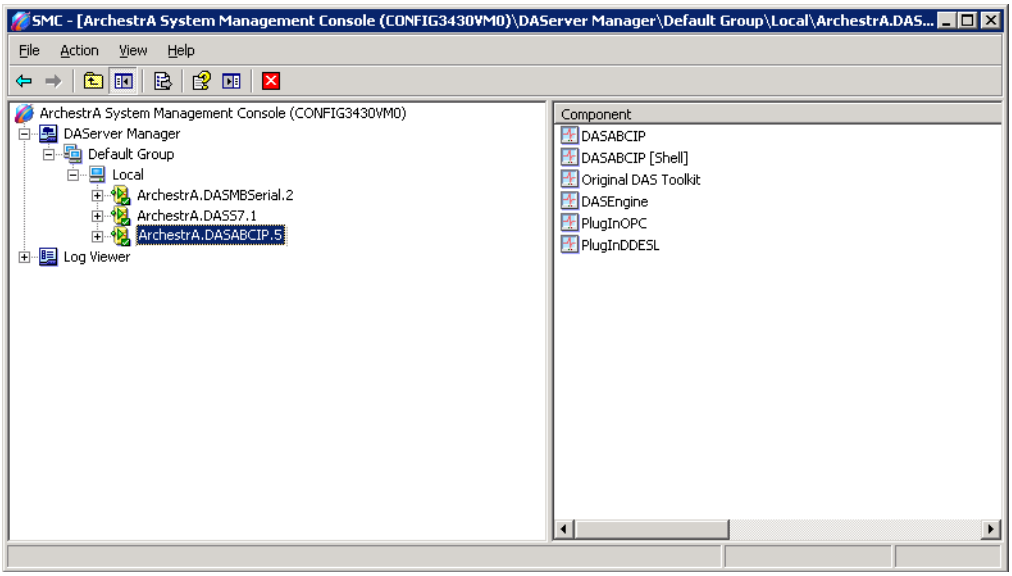
Each DAServer is identified by a unique program name (ProgID) under the SMC. The ProgID for this ABCIP DAServer is: **ArchestrA.DASABCIP.5**.

On the computer where the DAServer is installed, it can be found in the local node of the default group of the DAServer Manager.

You do not need to install the DAServer Manager on the same computer as the DAServer. When you access the DAServer remotely, you will not find the DAServer node under the local node. You must locate and identify the DAServer on a computer in one of the node groups.

To find the DAServer

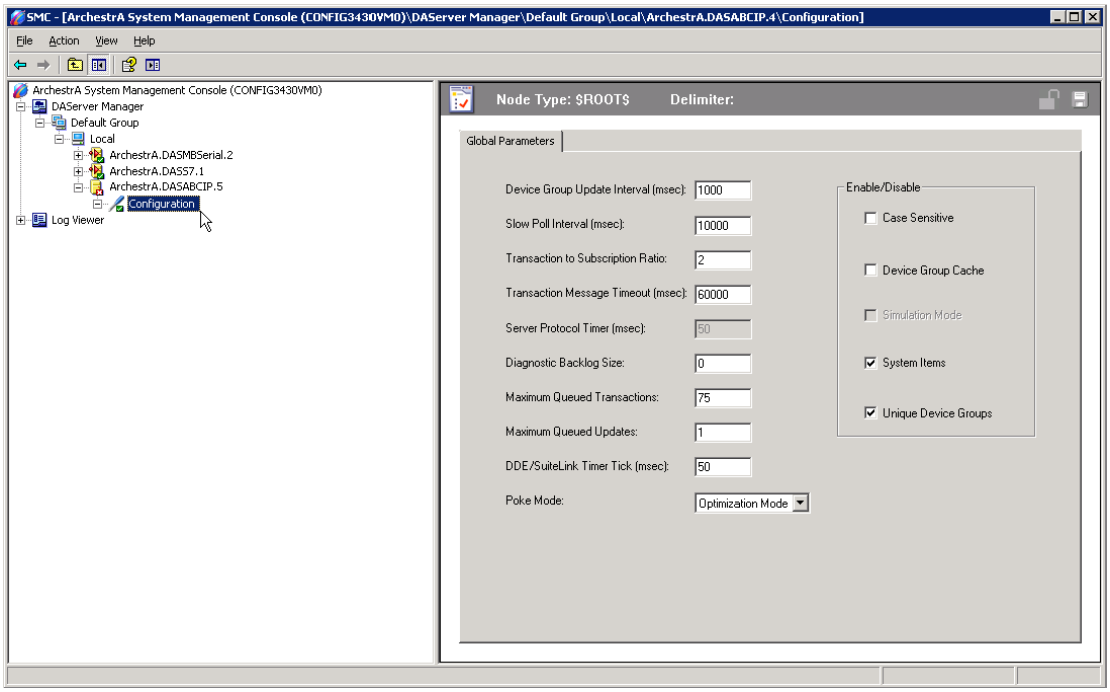
1. On the system **Start** menu, click **Programs**. Navigate to the Wonderware folder that contains the System Management Console and then click **System Management Console**.
2. In the **System Management Console**, expand **DAServer Manager**.
3. Locate the group with the node **ArchestrA.DASABCIP.5**



4 Expand the **ArchestrA.DASABCIP.5** node to display the global parameters.

5 Select the **configuration** node to display the global parameters.

To view global parameter configuration and settings in detail, see the *DAServer Manager User's Guide*.



Chapter 3

Configuration

Network Communication Bridge/Interface Modules are the communication links between the ABCIP DAServer and its supported Allen-Bradley controllers. You must create these links within the DAServer Manager hierarchy to bridge/route control and information data between different networks to target controllers.

This is accomplished by creating Port Objects. These Port Objects simulate the physical hardware layout and must be built to establish communications to each of the controllers. Once you have built the ABCIP hierarchy, you can configure the respective devices for communications. Finally, you can create the desired Device Groups for each controller.

Before you add these Ports in the SMC, you need to identify your hardware topology to the devices being connected.

Once you have established this hierarchy you will then add, rename, or delete Port objects to accurately represent how your network is organized.

Adding, Renaming, Deleting Port Objects

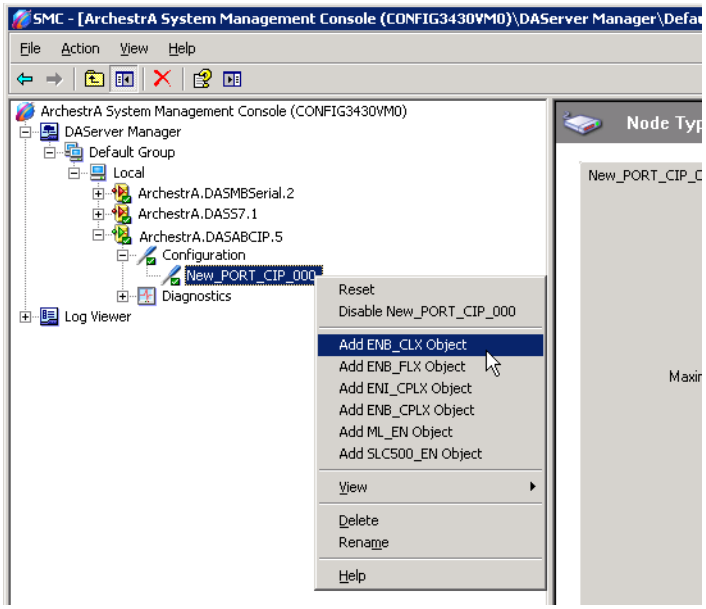
Use the procedures described in this section to add, rename, or delete port objects.

Adding a Port

The first step in specifying the network between the DAServer and a device is to add Port objects. After you add the necessary Ports depicting your network, you will then be able to add and communicate with your devices.

To add a port

1. Open the **DAServer Manager** in the SMC.
2. Locate and expand the target DAServer group hierarchy you wish to add ports to.
3. Right-click the default **Configuration** node and select the applicable **Add Port Object**. The console tree will now show the new port with its default port name selected.
4. Edit the name as needed and press Enter.

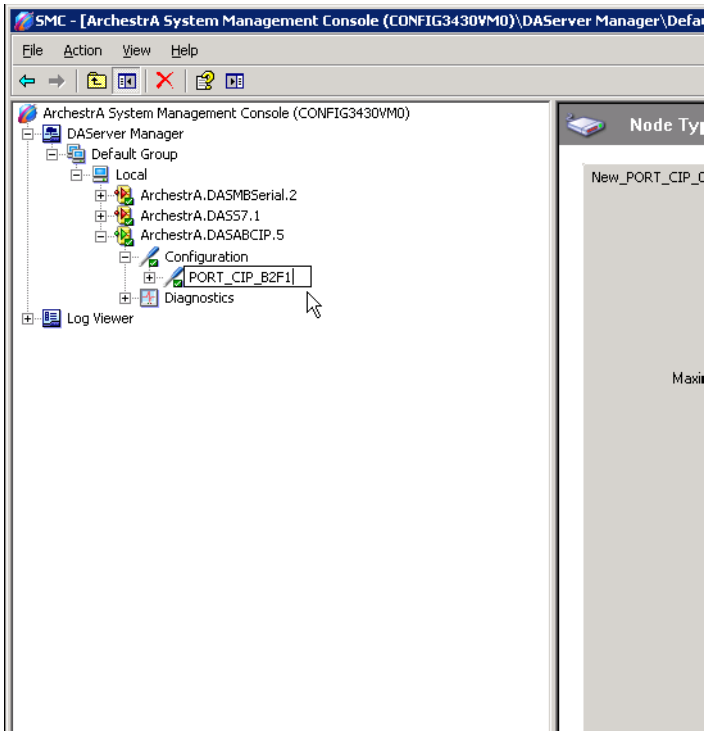


Renaming a Port

After you create ports in the DAServer Manager, it may be necessary to rename them to work with your client applications.

To change an existing port object name

1. In the DAServer Manager, expand the DAServer hierarchy tree to display the target port object node to display the port you wish to rename.
2. Select and right-click the port object's name (or <F2>). Click **Rename.**
3. Type the new name and press **Enter**.



Note: Changing the port name prevents clients from registering data using the old name. Data for existing queries is set to bad quality. Try not to make changes to parameters like the Port name after you develop a large client application.

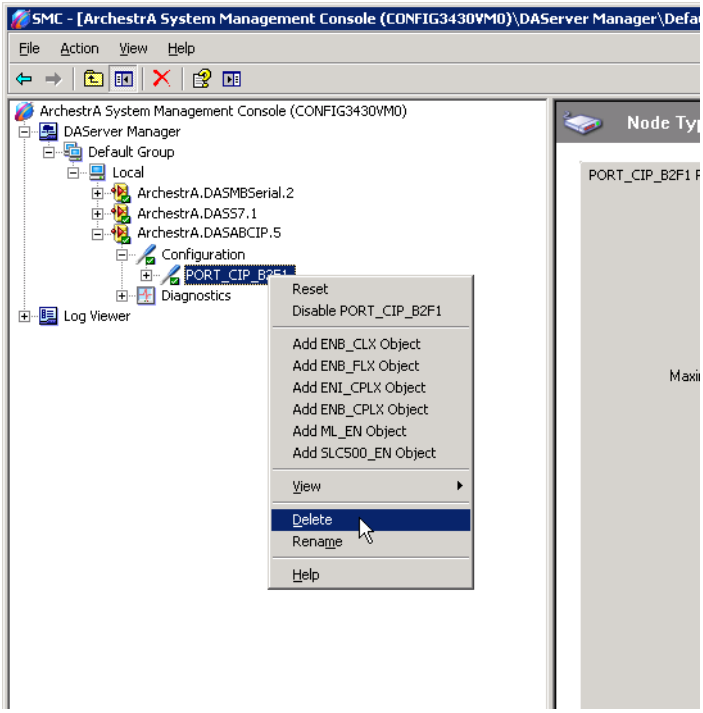
Deleting a Port

If your hardware network topology is changed you may need to delete a port object.

When you delete a port, all nodes below the port in its hierarchy (child nodes) are also deleted. If a client application requests new data from a deleted port or from a node on a deleted port, the request is rejected. Data for existing queries is set to bad quality.

To delete a port

1. In the DAServer Manager, expand the DAServer hierarchy tree to display the target port object node you wish to delete.
2. Right-click the port object node to be deleted and click **Delete**.
3. Read the warning and then click **Yes**. The port object and all nodes (devices) below it in the hierarchy are deleted.



Configuring ABCIP DAServer Ports

The ABCIP Hierarchy in the DAServer starts with the **PORT_CIP** Object, followed by the supported communication-interface/gateway modules that allow the DAServer to access the supported networks and devices.

The following sections detail the steps necessary to configure your DAServer Port Objects according to your network type.

The logical endpoint for each branch of the ABCIP hierarchy tree is always a Processor Type node which represents the controller device.

Note: Before attempting to configure your DAServer, you should determine the hierarchical structure of your device/network environment.

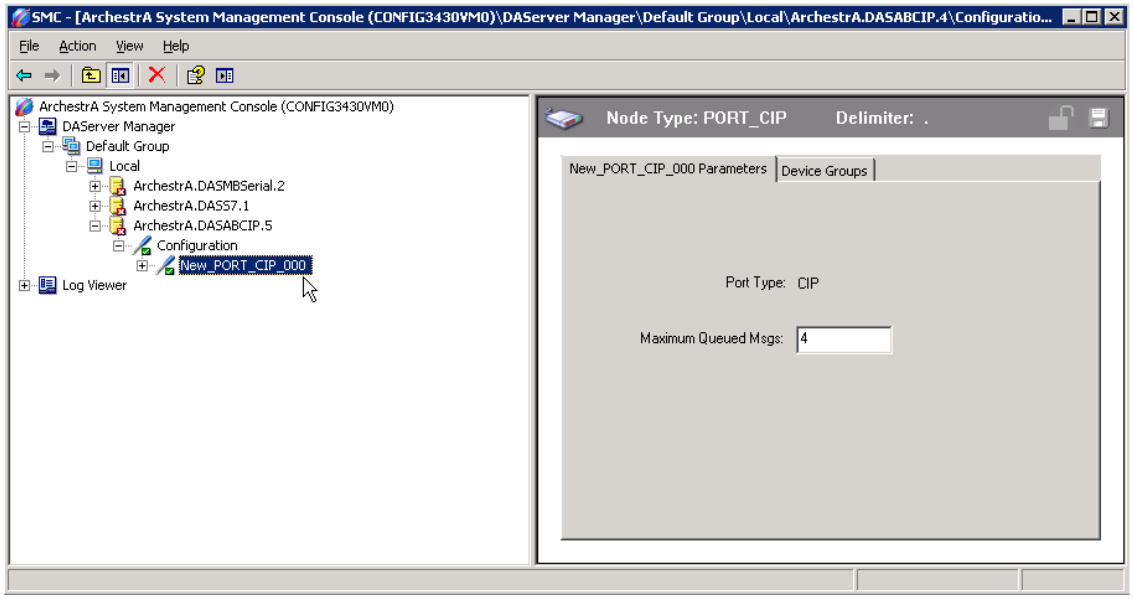
CIP Port Object Set-up

The DAServer hierarchy tree under the DAServer Manager starts at the **PORT_CIP** port object. It is a logical representation of the Ethernet port for CIP communications in a computer.

Note: Only one PORT_CIP object is allowed per ABCIP DAServer.

To create PORT_CIP objects from the Configuration branch

1. Select and right-click on **Configuration**.
2. Select **Add PORT_CIP Object** from the shortcut menu. An object called **New_PORT_CIP_000** is created.
3. Rename the newly created object as appropriate. The **Port_CIP_000 Parameters** configuration view is displayed in the **Configuration** branch of the hierarchy.



This configuration view has two parameters, one of which is configurable:

Port Type: The information is provided automatically by the DAServer Manager (CIP).

Maximum Queued Msgs: The default number of unconnected messages that the DAServer can send to a device before a reply is received.

- When this number is reached, the DAServer queues messages until a reply is received from the device.
- Valid range is 1 - 40.
- The default value is 4.

The Ethernet Network

Through the **PORT_CIP** object, the ABCIP DAServer accesses data from the ControlLogix, CompactLogix, FlexLogix, GuardLogix, MicroLogix, SLC500, and SoftLogix 5800 controllers on the Ethernet network that uses the EtherNet/IP protocol.

ENB_CLX Object

The **ENB_CLX** object represents the physical Allen-Bradley EtherNet/IP Communications module within a ControlLogix chassis.

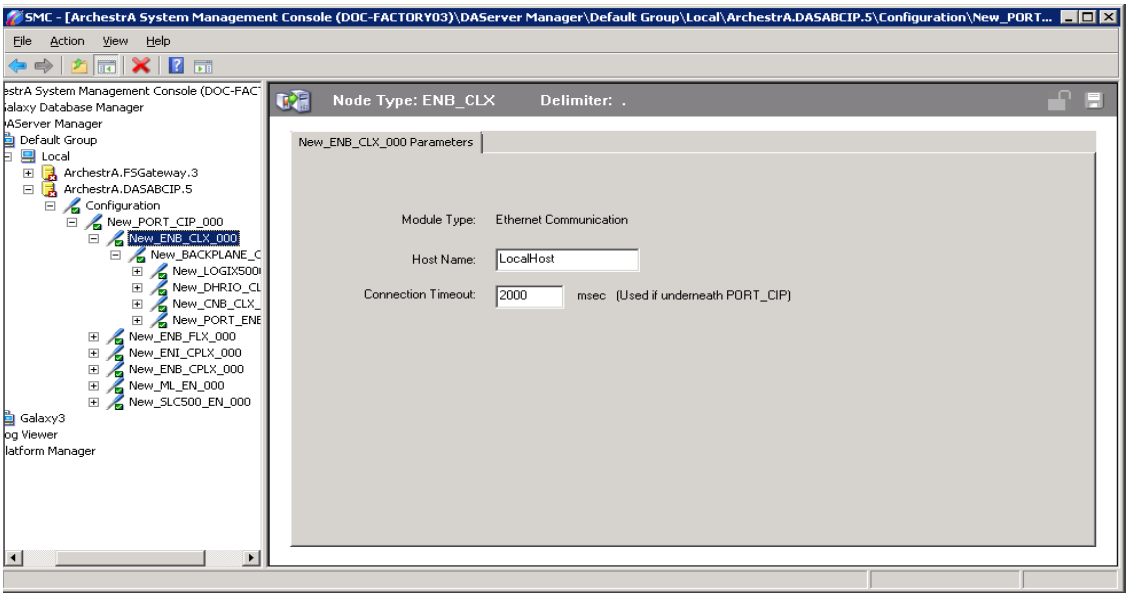
- 1756-ENET
- 1756-ENBT
- 1756-EN2T
- 1756-EWEB

The ENB_CLX object is hosted by **CIP.**

Note: A maximum of 65535 ENB_CLX objects can be created for the DAServer.

To add ENB_CLX objects to your ABCIP hierarchy

1. Select and right-click on the **New_PORT_CIP_000** object.
2. Select **Add ENB_CLX Object** from the shortcut menu. A **New_ENB_CLX_000** object is created.
3. Rename the newly created object as appropriate. The **ENB_CLX** parameters view is displayed.



This configuration view has three parameters, two of which are configurable:

Module Type: Information provided automatically by the DAServer Manager (Ethernet Comm).

Host Name: Host Name or IP Address of the destination 1756-ENET/ENBT/EN2T/EWEB module.

- The Host Name is defined in the system Host file, usually found in: \Windows\system32\drivers\etc\hosts folder.
- Type in the network address where the PLC is located (for example, “10.11.12.13”) or type in a host name if one is defined in the LocalHost list. The number of characters cannot be more than 255. The field cannot be blank.

Note: The Host Name defaults to the LocalHost. If the **LocalHost** is selected and deleted, resulting in a blank **Host Name** box, and you apply the changes, this will result in an error message.

Important: If setting up a SoftLogix or GuardLogix device, the host or IP address of the corresponding SoftLogix or physical GuardLogix device must be entered in the Ethernet/IP Bridge Module (ENB) node within the ABCIP DAServer hierarchy to establish communications with the device. For more information, see "SoftLogix 5800 Controllers" on page 15 or "GuardLogix Controllers" on page 15.

Connection Timeout: Time (in milliseconds) allowed for establishing a socket connection to a target device.

- The valid range is 10 - 10000 milliseconds.
- The default value is 2000.
- The connection timeout is used if the object is underneath PORT_CIP.

BACKPLANE_CLX Object

The **BackPlane_CLX** object represents the physical backplane of an Allen-Bradley ControlLogix controller chassis.

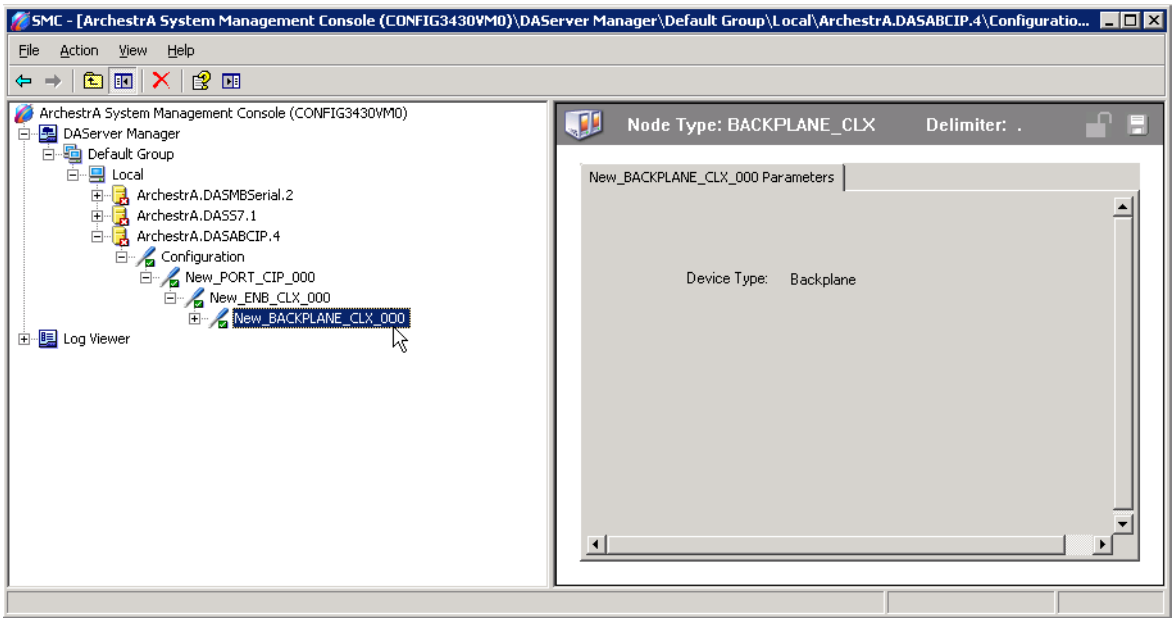
This object is hosted by the **ENB_CLX** and **CNB_Port_CLX** objects.

Note: Only one instance of the **BACKPLANE_CLX** object can be created per **ENB_CLX** and **CNB_Port_CLX** branch.

Note: The DAServer is capable of operating with multiple ControlLogix processors in a single backplane.

To add the BACKPLANE_CLX object to your ABCIP hierarchy

1. Select and right-click on the **New_ENB_CLX_000** object.
2. Select **Add BACKPLANE_CLX Object** from the shortcut menu. The **New_BACKPLANE_CLX_000** object is created.
3. Rename the newly created object as appropriate. The **BACKPLANE_CLX Parameters** view is displayed.



This configuration view has one element:

Device Type: The information is provided automatically by the DAServer Manager (BackPlane).

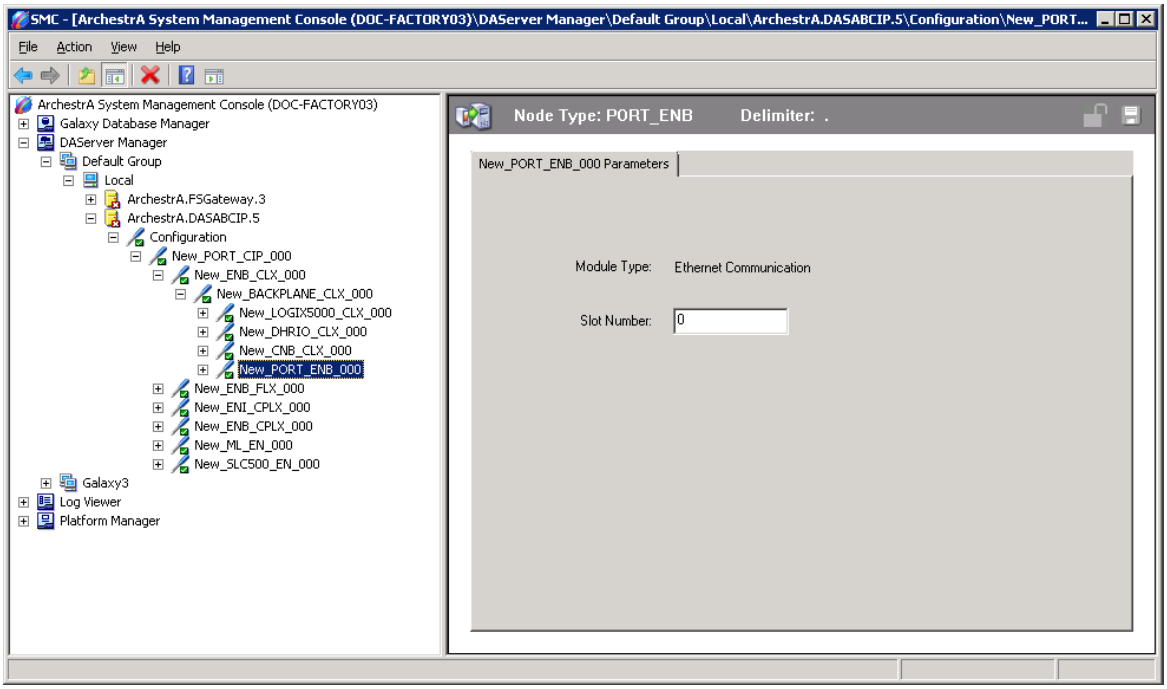
PORT_ENB Object

The Port_ENB object represents the physical Ethernet port for the Allen-Bradley Ethernet Network bridge module.

This object is hosted by the **BACKPLANE_CLX** object.

To add the PORT_ENB object to your ABCIP hierarchy

1. Select and right-click on the **New_BACKPLANE_CLX_000** object.
2. Select **Add PORT_ENB Object** from the shortcut menu. The **New_PORT_ENB_000** object is created.
3. Rename the newly created object as appropriate. The **PORT_ENB Parameters** view is displayed.



This configuration view has one element:

Slot Number: A sequential number beginning with 0 (zero) assigned to each slot in a ControlLogix chassis.

- The slot number indicates where the module resides in the parent backplane.
- The valid range is 0 - 16.
- The default value is 0 (zero).

LOGIX5000_CLX Object

The Logix5000_CLX object is a logical representation of the Allen-Bradley ControlLogix processor modules within a ControlLogix chassis.

- 1756-L1
- 1756-L55
- 1756-L6x
- 1756-L7x

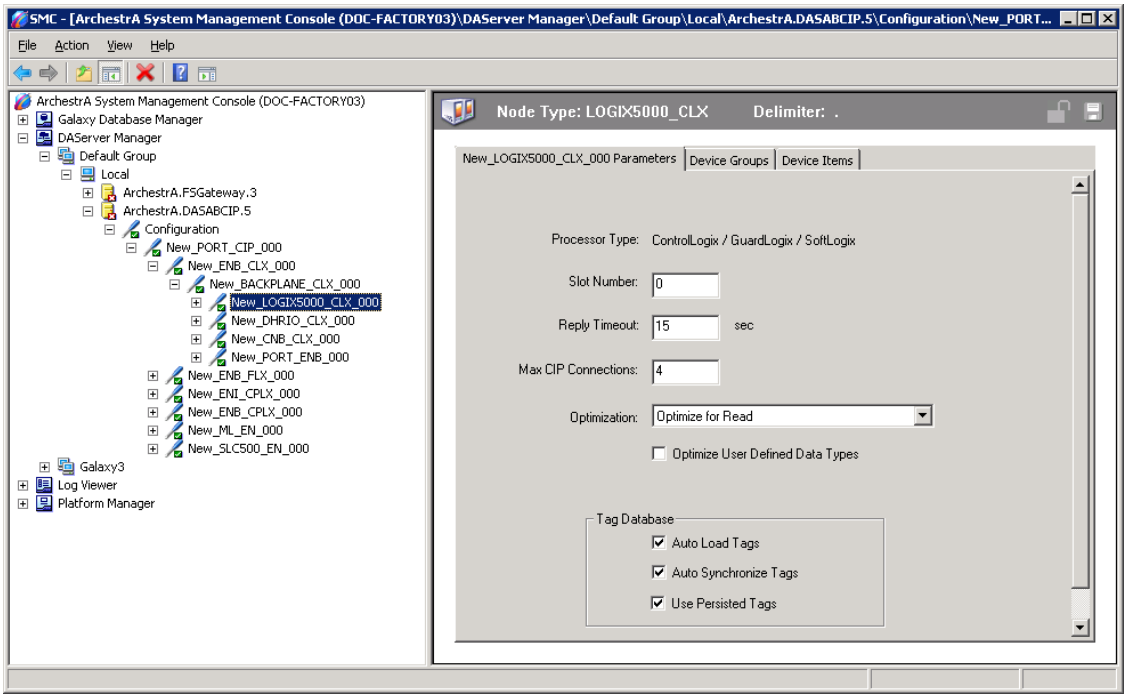
The Logix5000_CLX object is also a logical representation of the following Allen-Bradley processor modules:

- SoftLogix 5800
- GuardLogix 1756-L6xS

This object is hosted by **BackPlane_CLX**.

To add the LOGIX5000_CLX object to your ABCIP hierarchy

1. Select and right-click on the **New_BACKPLANE_CLX_000** object.
2. Select **Add LOGIX5000_CLX Object** from the shortcut menu. The **New_LOGIX5000_CLX_000** object is created.
3. Rename the newly created object as appropriate. The **Logix5000_CLX** Parameters view is displayed.



This configuration view has nine parameters, eight of which are configurable:

Processor Type: Information provided automatically by the DAServer Manager (ControlLogix /GuardLogix /SoftLogix).

Slot Number: A sequential number beginning with 0 (zero) assigned to each slot in a ControlLogix chassis.

- The slot number indicates where the module resides in the parent backplane.
- The valid range is 0 - 16.
- The default value is 0 (zero).

Reply Timeout: Time (in seconds) the DAServer will wait for the acknowledgement after it sends out a message. The message will be resent when time-out occurs.

- The valid range is 1 - 300 seconds.
- The default value is 15.

Max CIP Connections: The maximum number of CIP connections which can be originated from the DAServer to this device.

- The valid range is 1 - 31.
- The default value is 4 (four).

Optimization Mode (For detailed information, see "Logix5000 Optimization Mode" on page 111)**:**

- **No optimization**: The server uses the most basic communication method available by using the tag name for each communication with the controller. The tag database will be uploaded from the processor to validate the tag names.
- **Optimize for read** (*Default*): All tags are accessed by predefining messages in the controller, thus optimizing blocks of information from the controller. Initialization of this mode requires that these message blocks are built when connecting to the controller, therefore startup time will require more time. This mode is most effective with large number of tags on continuous scan.

- **Optimize for startup time**: This option provides the best overall performance. All tags are accessed from the Logix processor using the device's memory location table. If this option is checked, the 'Auto Synchronize Tag' option is checked automatically and cannot be unchecked.

Optimize User Defined Data Types: The optimization for reading structures is enabled when selected (Default is unchecked). For more detailed information, see "UDT Optimization" on page 113.

If selected, the server will retrieve the whole structure in one packet provided the size of the structure is 488 bytes or less.

Tag Database Options: Three options are selectable to implement manual or automated updates of the Logix processor's tag database. For more information, see "Logix5000 Online Tag Management" on page 116.

- Auto Load Tags on Startup (Default)
- Auto Synchronize Tags
- Use Persisted Tags (Default)

Note: If the Optimization setting is selected for "Optimize for startup time", the "Auto Synchronize Tags" option is automatically selected and unchangeable (dimmed). The DAServer will need to synchronize physical address tags from the device.

Important: Support for secured Logix5000 controllers will affect the way the 'Auto Synchronize Tags' and 'Persisted Tags' behave. For detailed information, see "Accessing Secured Logix5000-series Controllers" on page 123.

ENB_FLX Object

The **ENB_FLX** object represents the physical Allen-Bradley FlexLogix Ethernet Communication Daughter Card.

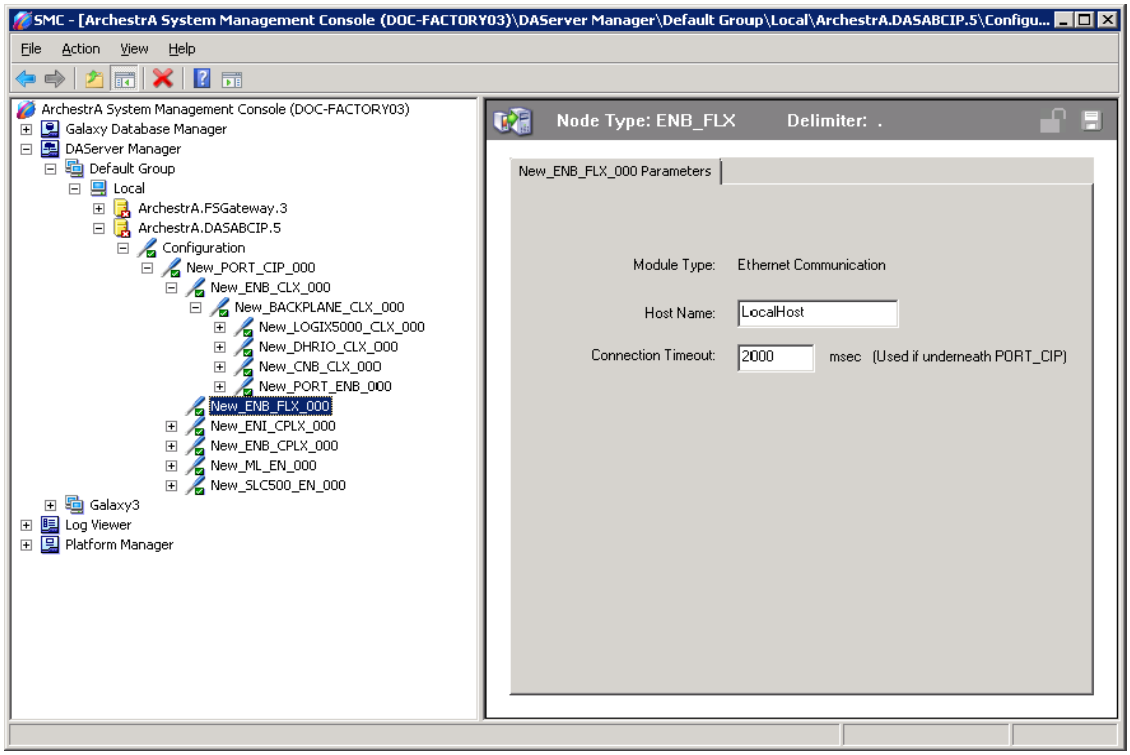
- 1788-ENBT

This object is hosted by the **CIP** Network Object.

Note: A maximum of 65535 **ENB_FLX** objects can be created for the DAServer.

To add ENB_FLX objects to your ABCIP hierarchy

1. Select and right-click on the **New_PORT_CIP_000** object.
2. Select **Add ENB_FLX** Object from the shortcut menu. A **New_ENB_FLX_000** object is created.
3. Rename the newly created object as appropriate. The **ENB_FLX** Parameters view is displayed.



This configuration view has three parameters, two of which are configurable:

Module Type: Information provided automatically by the DAServer Manager (Ethernet Communication).

Host Name: The Host Name or IP Address of the destination 1788-ENBT module.

- The Host Name is defined in the system Host file, usually found in: \Windows\system32\drivers\etc\hosts folder.
- The number of characters cannot be more than 255. The field cannot be blank.

Note: The Host Name defaults to the LocalHost. If the **LocalHost** is selected and deleted, resulting in a blank **Host Name** box, and you apply the changes, this will result in an error message.

Connection Timeout: Time (in milliseconds) allowed for establishing a socket connection to a target device.

- The valid range is 10 - 10000 milliseconds.
- The default value is 2000.
- The connection timeout is used if the object is underneath PORT_CIP.

BACKPLANE_FLX Object

The **BackPlane_FLX** object represents the physical backplane of an Allen-Bradley FlexLogix controller assembly.

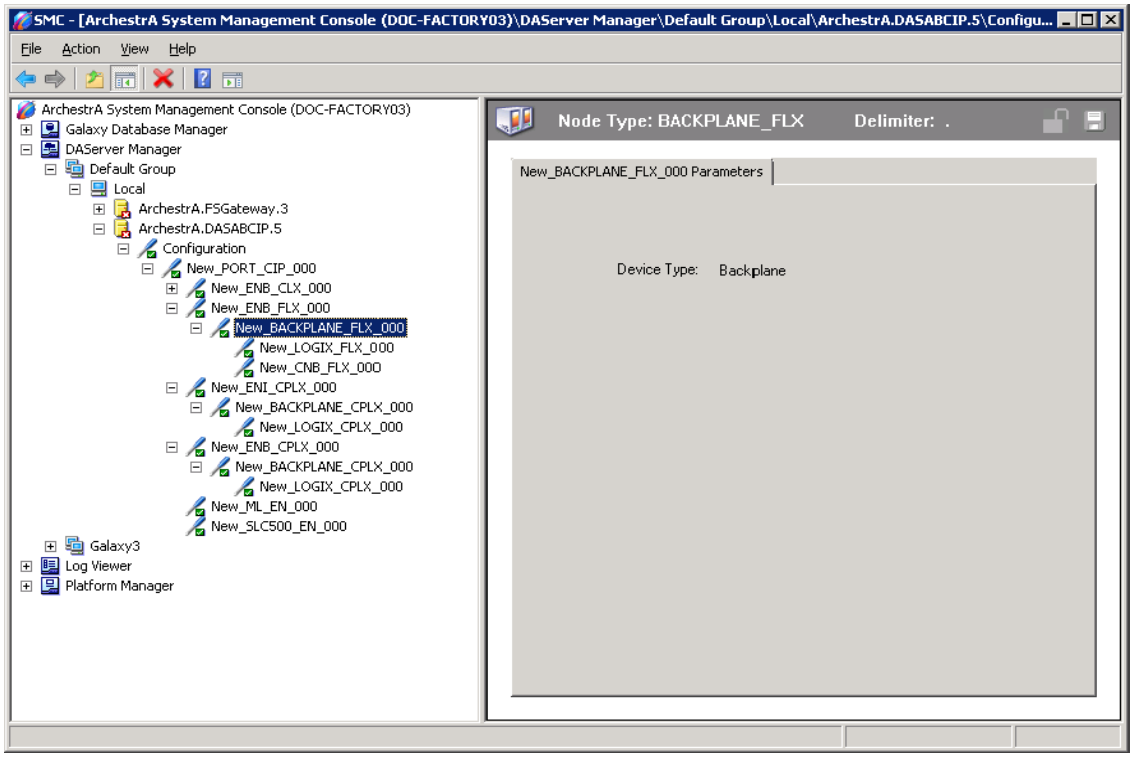
This object is hosted by **ENB_FLX** and **CNB_Port_FLX**.

Note: Only one instance of the BACKPLANE_FLX object can be created per ENB_FLX branch.

Note: The DAServer is capable of operating with multiple FlexLogix processors in a single backplane.

To add the BACKPLANE_FLX object to your ABCIP hierarchy

1. Select and right-click on the **New_ENB_FLX_000** object.
2. Select **Add BACKPLANE_FLX Object** from the shortcut menu. A **New_BACKPLANE_FLX_000** object is created.
3. Rename the newly created object as appropriate. The **BACKPLANE_FLX** Parameters view is displayed.



This configuration view has one element:

Device Type: The information is provided automatically by the DAServer Manager (Backplane).

LOGIX_FLX Object

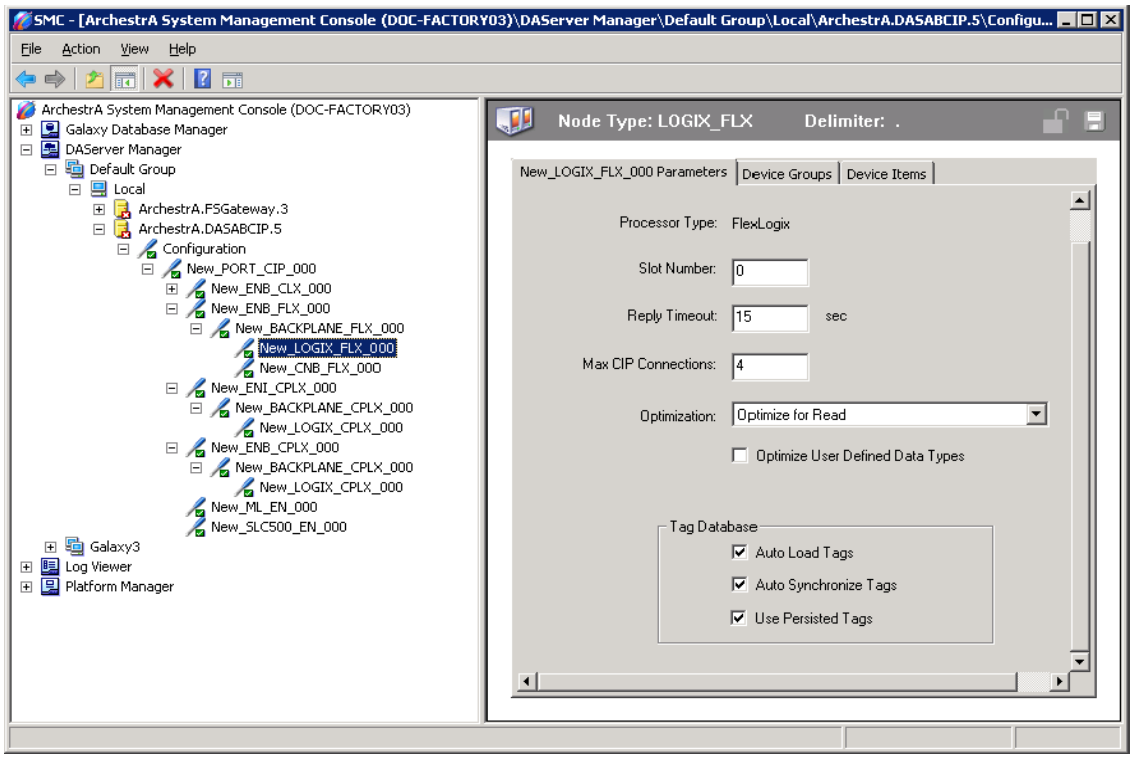
The **Logix_FLX** object represents the physical Allen-Bradley FlexLogix processor module.

- 1794-Lxx

This object is hosted by **BACKPLANE_FLX**.

To add the LOGIX_FLX object to your ABCIP hierarchy

1. Select and right-click on the **NEW_BACKPLANE_FLX_000** object.
2. Select **Add LOGIX_FLX Object** from the shortcut menu. The New_LOGIX_FLX_000 object is created.
3. Rename the newly created object as appropriate. The **LOGIX_FLX** Parameters view is displayed.



This configuration view has nine parameters, eight of which are configurable:

Processor Type: Information provided automatically by the DAServer Manager (FlexLogix).

Slot Number: A sequential number beginning with 0 (zero) assigned to each slot in a FlexLogix chassis.

- The slot number indicates where the module resides.
- The valid range is 0 - 16.
- The default value is 0.

Reply Timeout: Time (in seconds) the DAServer will wait for the acknowledgement after it sends out a message. The message will be resent when time-out occurs.

- The valid range is 1 - 300.
- The default value is 15.

Max CIP Connections: The maximum number of CIP connections which can be originated from the DAServer to this device.

- The valid range is 1 - 31.
- The default value is 4 (four).

Optimization Mode (For detailed information, see "Logix5000 Optimization Mode" on page 111)**:**

- **No optimization**: The server uses the most basic communication method available by using the tag name for each communication with the controller. The tag database will be uploaded from the processor to validate the tag names.
- **Optimize for read** (*Default*): All tags are accessed by predefining messages in the controller, thus optimizing blocks of information from the controller. Initialization of this mode requires that these message blocks are built when connecting to the controller, therefore startup time will require more time. This mode is most effective with large number of tags on continuous scan.
- **Optimize for startup time**: This option provides the best overall performance. All tags are accessed from the Logix processor using the device's memory location table. If this option is checked, the 'Auto Synchronize Tag' option is checked automatically and cannot be unchecked.

Optimize User Defined Data Types: The optimization for reading structures is enabled when selected (Default is unchecked). For more detailed information, see "UDT Optimization" on page 113.

- If selected, the server will retrieve the whole structure in one packet provided the size of the structure is 488 bytes or less.

Tag Database Options: Three options are selectable to implement manual or automated updates of the Logix processor's tag database. For more information, see "Logix5000 Online Tag Management" on page 116.

- Auto Load Tags on Startup (Default)

- Auto Synchronize Tags
- Use Persisted Tags (Default)

Note: If the Optimization setting is selected for "Optimize for startup time", the "Auto Synchronize Tags" option is automatically selected and unchangeable (dimmed). The DAServer will need to synchronize physical address tags from the device.

Important: Support for secured Logix5000 controllers will effect the way the 'Auto Synchronize Tags' and 'Persisted Tags' behave. For detailed information, see "Accessing Secured Logix5000-series Controllers" on page 123.

ML_EN Object

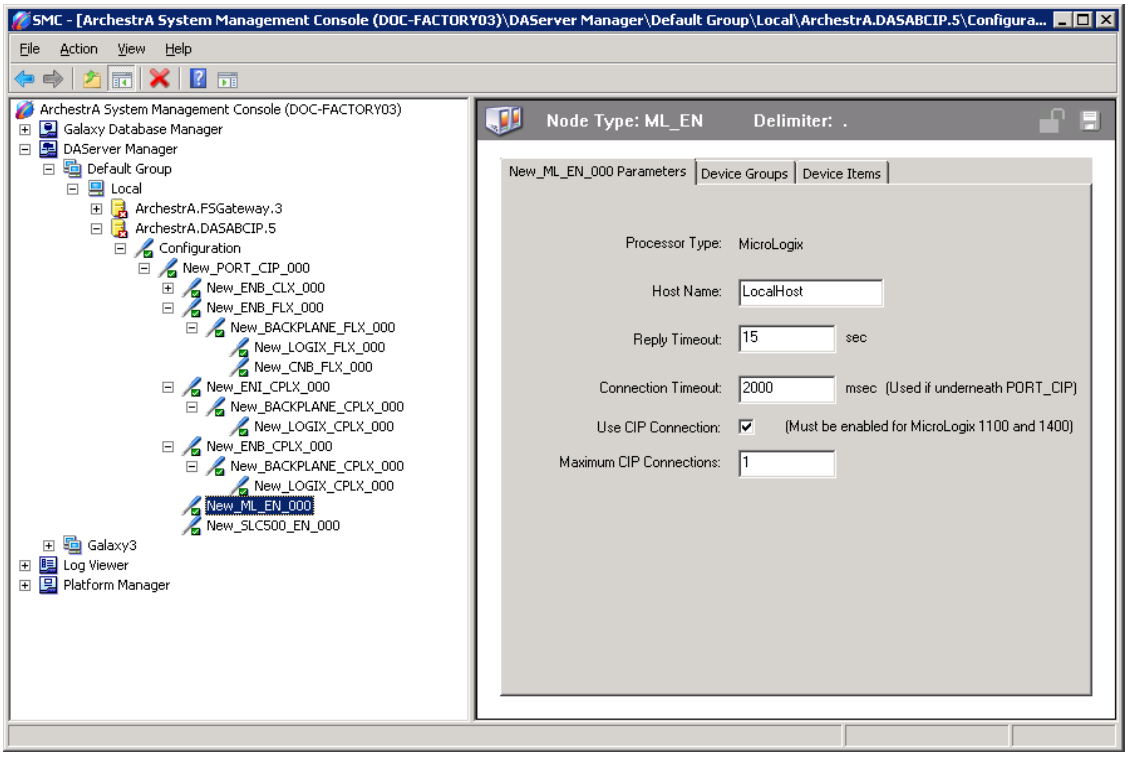
The **ML_EN** object represents the physical Allen-Bradley MicroLogix processor with the built-in EtherNet/IP port or coupled with the Ethernet Interface module for MicroLogix and CompactLogix (1761-NET-ENI).

- 1763-L16xxx
- 1761-L10xxx, 1761-L16xxx, 1761-L20xxx, 1761-L32xxx
- 1762-L24xxx, 1762-L40xxx
- 1764-LSP, 1764-LRP

This object is hosted by **CIP** Network Object

To add ML_EN objects to your ABCIP hierarchy

1. Select and right-click on the **New_PORT_CIP_000** object.
2. Select **Add ML_EN Object** from the shortcut menu. A **New_ML_EN_000** object is created.
3. Rename the newly created object as appropriate. The **ML_EN** Parameters view is displayed.



This configuration view has six parameters, five of which are configurable:

Processor Type: Information provided automatically by the DAServer Manager (MicroLogix).

Host Name: The Host Name or IP Address of the destination MicroLogix processor or 1761-NET-ENI module connected to a MicroLogix processor.

- The Host Name is defined in the system Host file, usually found in: \Windows\system32\drivers\etc\hosts folder.
- The number of characters cannot be more than 255. The field cannot be blank.

Reply Timeout: Time (in seconds) the DAServer will wait for the acknowledgement after it sends out a message. The message will be resent when time-out occurs.

- The valid range is 1 - 300 seconds.
- The default value is 15.

Connection Timeout: Time (in milliseconds) allowed for establishing a socket connection to a target device.

- The valid range is 10 - 10000 milliseconds.
- The default value is 2000.
- The connection timeout is used if the ML_EN object is beneath PORT_CIP.

Use CIP Connection: This option specifies if the CIP connection should be used to communicate with the MicroLogix controller. It must be selected to support MicroLogix model 1100/1400-series controllers with direct CIP connection. It is optional for all other MicroLogix models.

- The default value is True.

Note: The number of CIP connections in the controllers are limited (See Max CIP Connections below).

Max CIP Connections: The maximum number of CIP connections which can be originated from the DAServer to this device.

- The valid range is 1 - 31.
- The default value is 1 (one).

Note: Max CIP Connections setting available only if the CIP connection is selected.

The logical endpoint for each branch of the ABCIP hierarchy tree is always a Processor Type node, which represents the controller device.

ENB_CPLX Object

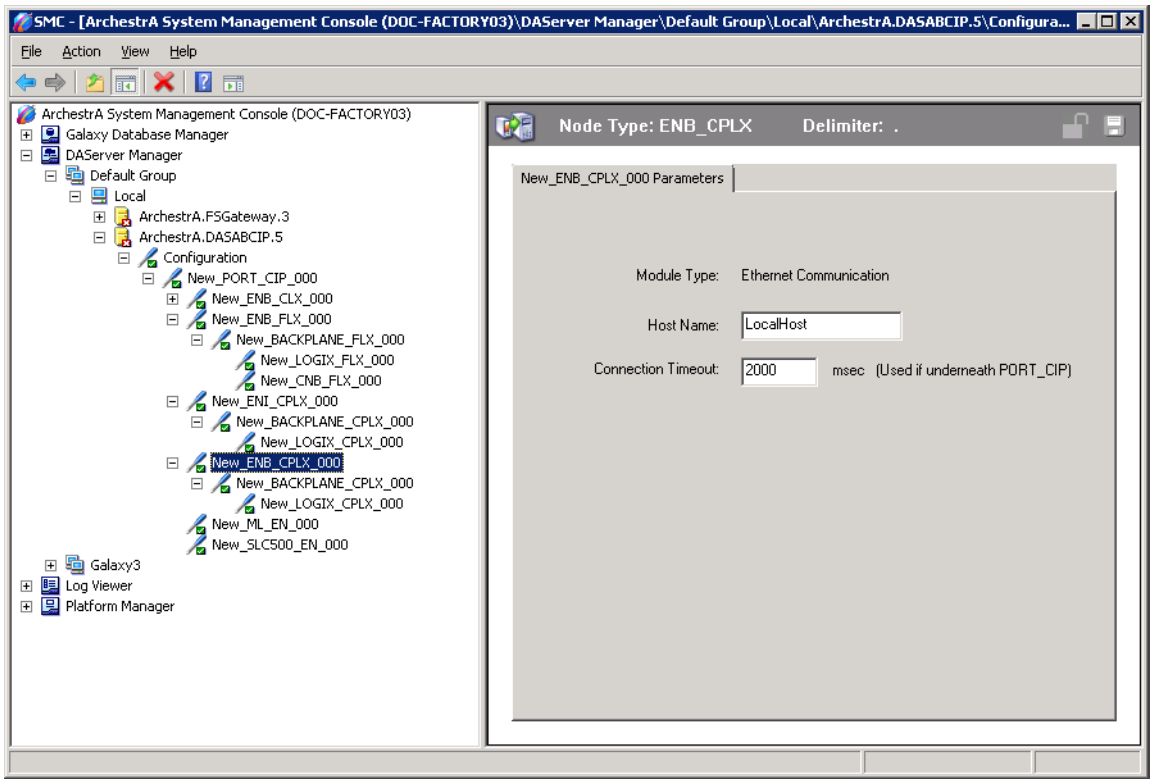
The **ENB_CPLX** object represents the physical integrated EtherNet/IP port on the Allen-Bradley CompactLogix Ethernet processor.

This object is hosted by **CIP** Network Object

Note: A maximum of 65536 ENB_CPLX objects can be created for the DAServer.

To add ENB_CPLX objects to your ABCIP hierarchy

1. Select and right-click on the **New_PORT_CIP_000** object.
2. Select **Add ENB_CPLX_Object** from the shortcut menu. A **New_ENB_CPLX_000** object is created.
3. Rename the newly created object as appropriate. The **ENB_CPLX** Parameters configuration view appears.



This configuration view has three parameters, two of which are configurable:

Module Type: Information provided automatically by the DAServer Manager (Ethernet Communication).

Host Name: The Host Name or IP Address of the destination Ethernet-capable CompactLogix processor.

- The Host Name is defined in the system Host file, usually found in: \Windows\system32\drivers\etc\hosts folder.
- The number of characters cannot be more than 255. The field cannot be blank.

Note: The Host Name defaults to the LocalHost. If the **LocalHost** is selected and deleted, resulting in a blank **Host Name** box, and you apply the changes, this will result in an error message.

Connection Timeout: Time (in milliseconds) allowed for establishing a socket connection to a target device.

- The valid range is 10 - 10000 milliseconds.
- The default value is 2000.
- The connection timeout is used if the ENB_CPLX object is beneath PORT_CIP.

ENI_CPLX Object

The **ENI_CPLX** object represents the physical Allen-Bradley Ethernet Interface module for MicroLogix and CompactLogix (1761-NET-ENI).

- 1761-NET-ENI Module

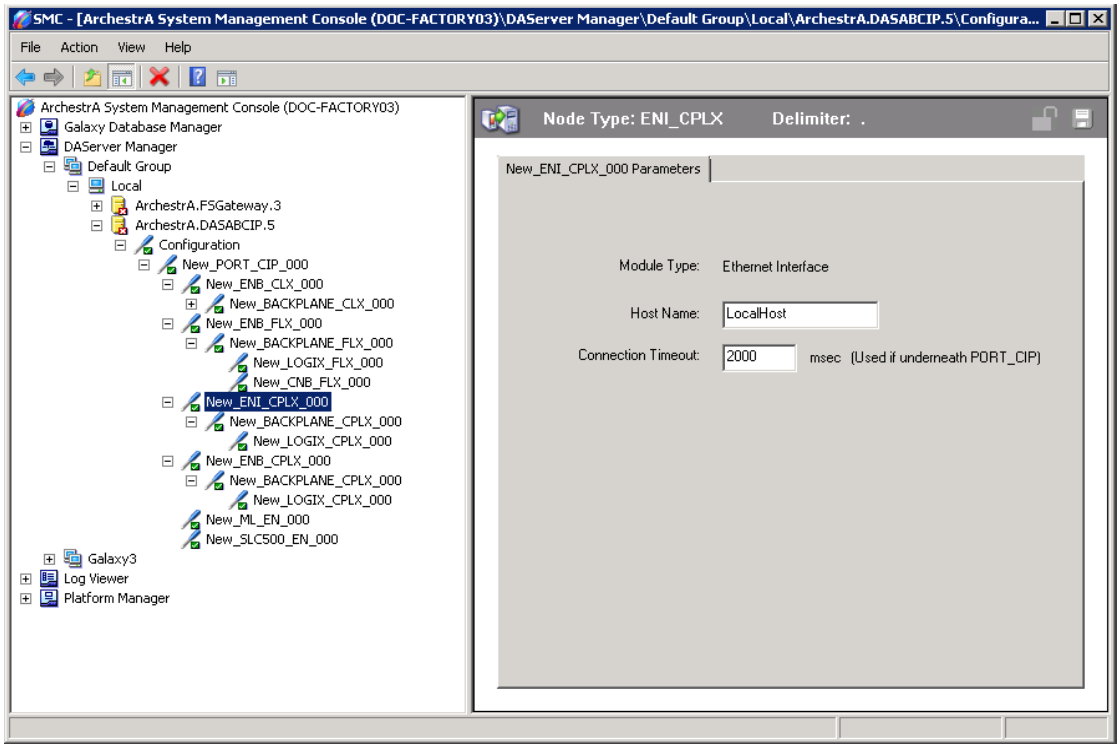
This object is hosted by **CIP** Network Object

Note: A maximum of 65535 ENI_CPLX objects can be created for the DAServer.

To add ENI_CPLX objects to your ABCIP hierarchy

1. Select and right-click on the **New_PORT_CIP_000** object.
2. Select **Add ENI_CPLX_Object** from the shortcut menu. A **New_ENI_CPLX_000** object is created.

3 Rename the newly created object as appropriate. The **ENI_CPLX** Parameters configuration view is displayed.



This configuration view has three parameters, two of which are configurable:

Module Type: Information provided automatically by the DAServer Manager (Ethernet Interface).

Host Name: The Host Name or IP Address of the destination 1761-NET-ENI module.

- The Host Name is defined in the system Host file, usually found in: \Windows\system32\drivers\etc\hosts folder.

Note: The Host Name defaults to the LocalHost. If the **LocalHost** is selected and deleted, resulting in a blank **Host Name** box, and you apply the changes, this will result in an error message.

Connection Timeout: Time (in milliseconds) allowed for establishing a socket connection to a target device.

- The valid range is 10 - 10000 milliseconds.
- The default value is 2000.
- The connection timeout is used if the ENI_CPLX object is beneath PORT_CIP.

BACKPLANE_CPLX Object

The **BACKPLANE_CPLX** object represents the physical backplane of a CompactLogix controller assembly.

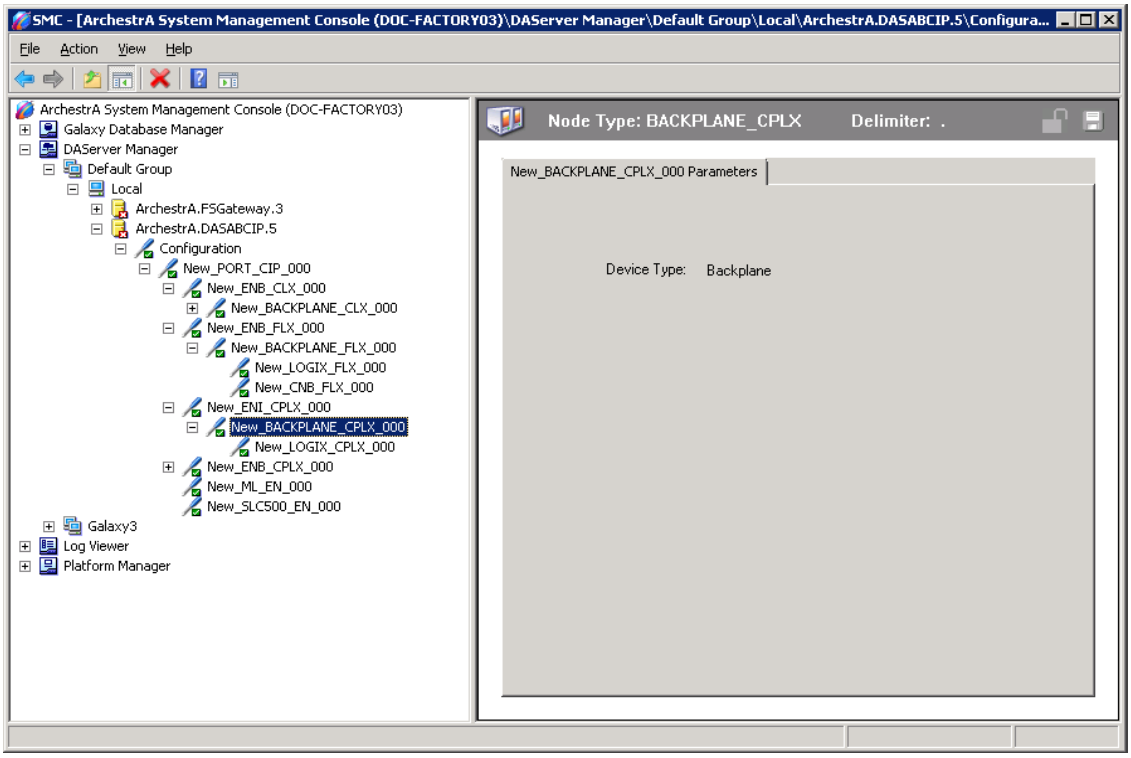
This object is hosted by **ENB_CPLX** and **ENI_CPLX**.

Note: Only one instance of the BACKPLANE_CPLX object can be created per ENB_CPLX or ENI_CPLX branch.

Note: The DAServer is capable of operating with multiple CompactLogix processors in a single backplane.

To add the BACKPLANE_CPLX object to your ABCIP hierarchy

1. Select and right-click on the **New_ENB_CPLX_000** or **New_ENI_CPLX_000** object.
2. Select **Add BACKPLANE_CPLX Object** from the shortcut menu. The **New_BACKPLANE_CPLX_000** object is created.
3. Rename the newly created object as appropriate. The **BACKPLANE_CPLX** Parameters view appears.



This configuration view has one element:

Device Type: The information is provided automatically by the DAServer Manager (Backplane).

LOGIX_CPLX Object

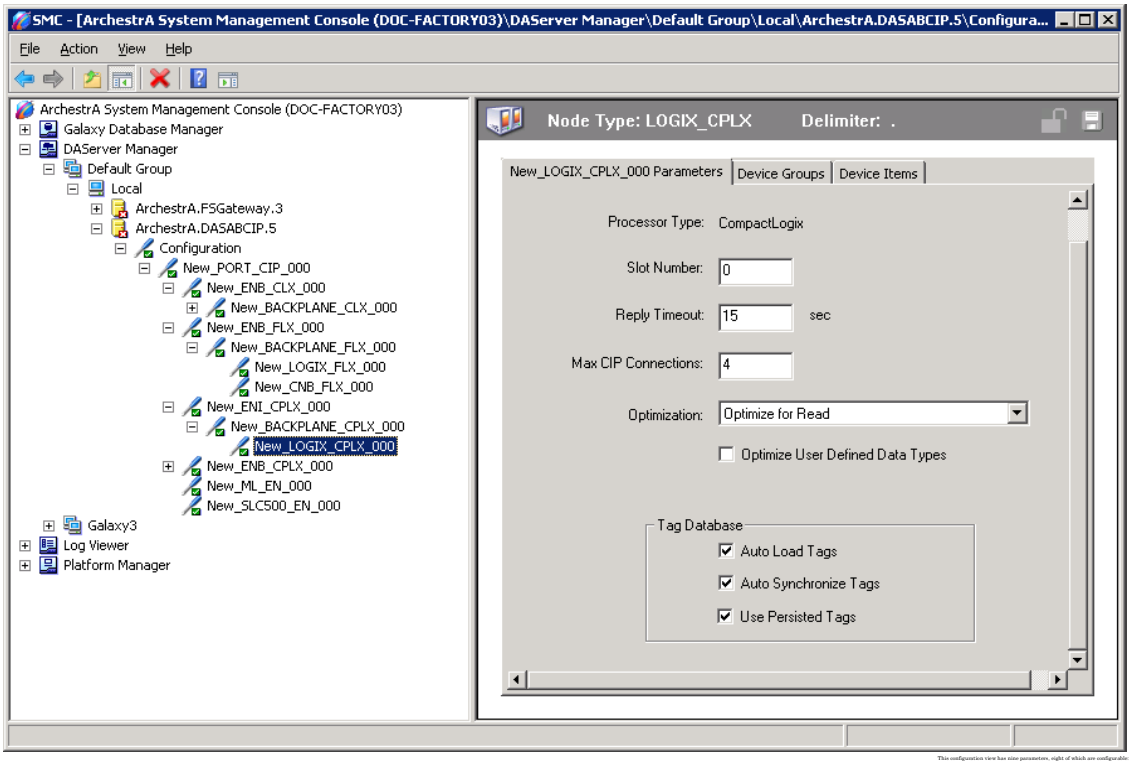
The **LOGIX_CPLX** object represents the physical CompactLogix processor module.

- 1768-Lxx
- 1769-Lxx

This object is hosted by **BACKPLANE_CPLX**.

To add the LOGIX_CPLX object to your ABCIP hierarchy

1. Select and right-click on the **NEW_BACKPLANE_CPLX_000** object.
2. Select **Add LOGIX_CPLX Object** from the shortcut menu. The **New_LOGIX_CPLX_000** object is created.
3. Rename the newly created object as appropriate. The **LOGIX_CPLX** Parameters view is displayed.



Processor Type: Information provided automatically by the DAServer Manager (LOGIX5000).

Slot Number: A sequential number beginning with 0 (zero) assigned to each slot in a Logix5000 chassis.

- The slot number indicates where the module resides.
- The valid range is 0 - 16.
- The default value is 0 (zero).

Reply Timeout: Time (in seconds) the DAServer will wait for the acknowledgement after it sends out a message. The message will be resent when time-out occurs.

- The valid range is 1 - 300.
- The default value is 15.

Max CIP Connections: The maximum number of CIP connections which can be originated from the DAServer to this device.

- The valid range is 1 - 31
- The default value is 4 (four)

Optimization Mode (For detailed information, see "Logix5000 Optimization Mode" on page 111)**:**

- **No optimization**: The server uses the most basic communication method available by using the tag name for each communication with the controller. The tag database will be uploaded from the processor to validate the tag names.
- **Optimize for read** (*Default*): All tags are accessed by predefining messages in the controller, thus optimizing blocks of information from the controller. Initialization of this mode requires that these message blocks are built when connecting to the controller, therefore startup time will require more time. This mode is most effective with large number of tags on continuous scan.
- **Optimize for startup time**: This option provides the best overall performance. All tags are accessed from the Logix processor using the device's memory location table. If this option is checked, the 'Auto Synchronize Tag' option is checked automatically and cannot be unchecked.

Optimize User Defined Data Types: The optimization for reading structures is enabled when selected (Default is unchecked). For more detailed information, see "UDT Optimization" on page 113.

- If selected, the server will retrieve the whole structure in one packet provided the size of the structure is 488 bytes or less.

Tag Database Options: Three options are selectable to implement manual or automated updates of the Logix processor's tag database. For more information, see "Logix5000 Online Tag Management" on page 116.

- Auto Load Tags on Startup (Default)

- Auto Synchronize Tags
- Use Persisted Tags (Default)

Note: If the Optimization setting is selected for "Optimize for startup time", the "Auto Synchronize Tags" option is automatically selected and unchangeable (dimmed). The DAServer will need to synchronize physical address tags from the device.

Important: Support for secured Logix5000 controllers will affect the way the 'Auto Synchronize Tags' and 'Persisted Tags' behave. For detailed information, see "Accessing Secured Logix5000-series Controllers" on page 123.

SLC500_EN Object

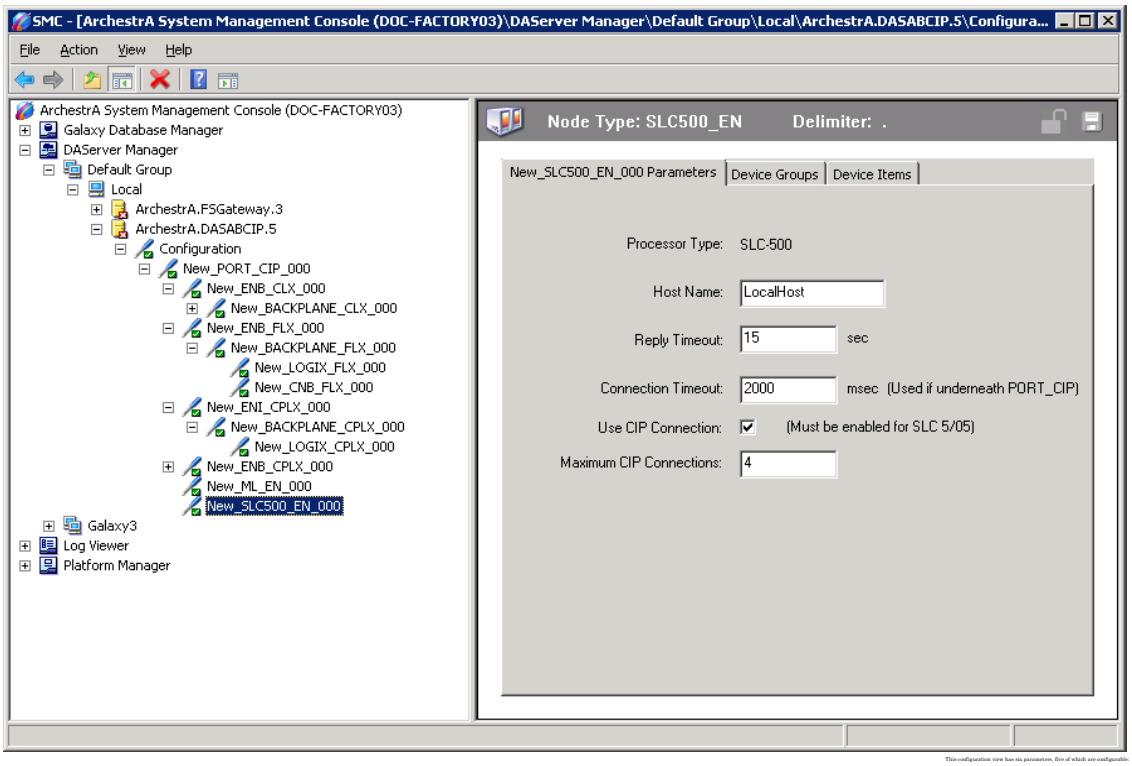
The **SLC500_EN** object represents the physical Allen-Bradley SLC500 processor connected to an Allen-Bradley Ethernet Interface for MicroLogix and CompactLogix (1761-NET-ENI).

- 1747-L5xx with 1761-NET-ENI

This object is hosted by **CIP** Network Object

To add SLC500_EN objects to your ABCIP hierarchy

1. Select and right-click on the **New_PORT_CIP_000** object.
2. Select **Add SLC500_EN Object** from the shortcut menu. A **New_SLC500_EN_000** object is created.
3. Rename the newly created object as appropriate. The **SLC500_EN** Parameters view is displayed.



Processor Type: Information provided automatically by the DAServer Manager (SLC500).

Host Name: The Host Name or IP Address of the destination 1761-NET-ENI Module.

- The Host Name is defined in the system Host file, usually found in: \Windows\system32\drivers\etc\hosts folder.

- The number of characters cannot be more than 255. The field cannot be blank.

Reply Timeout: Time (in seconds) the DAServer will wait for the acknowledgement after it sends out a message. The message will be resent when time-out occurs.

- The valid range is 1 - 300 seconds.
- The default value is 15.

Connection Timeout: Time (in milliseconds) allowed for establishing a socket connection to a target device.

- The valid range is 10 - 10000 milliseconds.
- The default value is 2000.
- The connection timeout is used if the SLC500_EN object is beneath PORT_CIP.

Use CIP Connection: Must be selected to support SLC 5/05-series controllers with direct CIP connection.

- The default value is True.

Max CIP Connections: The maximum number of CIP connections which can be originated from the DAServer to this device.

- The valid range is 1 - 31.
- The default value is 4 (four).

Note: Max CIP Connections setting available only if the direct CIP connection is selected.

The ControlNet Network

Routing through the CNB_CLX or the CNB_FLX object from Ethernet network, the ABCIP DAServer accesses data from ControlLogix, GuardLogix, CompactLogix, FlexLogix, PLC-5, and SLC500 processors over the ControlNet network.

CNB_CLX Object

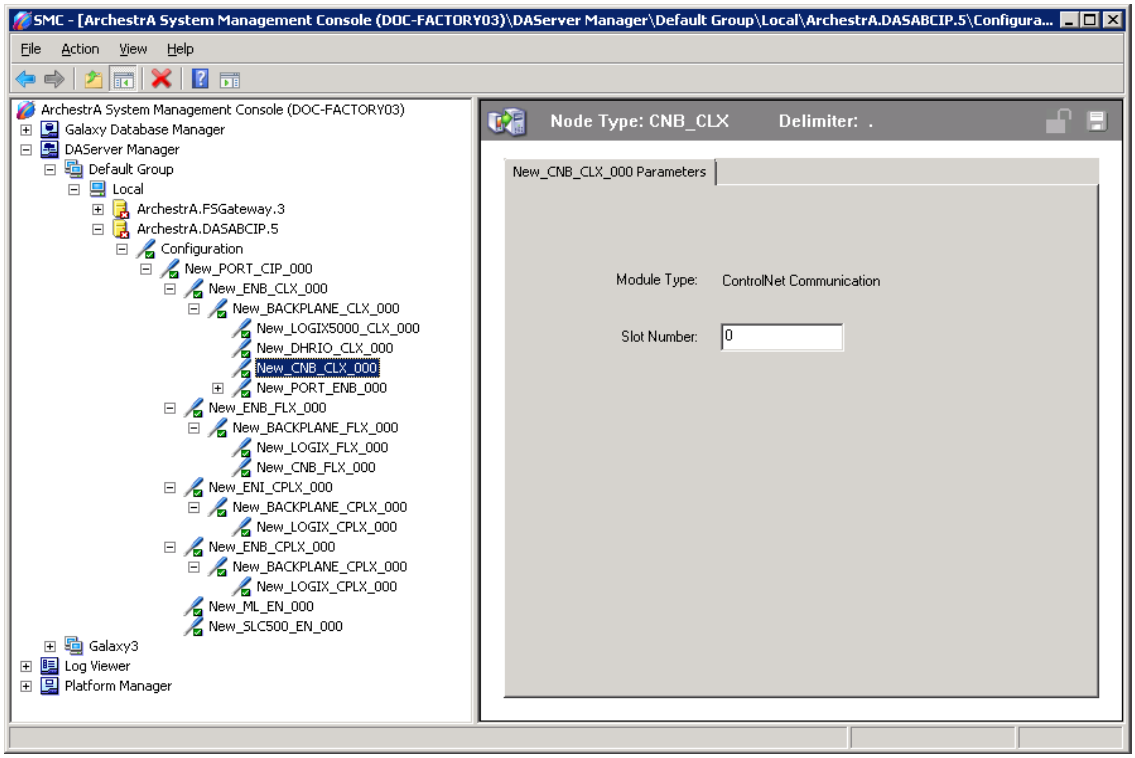
The **CNB_CLX** object represents the physical Allen-Bradley ControlLogix ControlNet Bridge module.

- 1756-CNB(R)
- 1756-CN2(R)

This object is hosted by **BACKPLANE_CLX.**

To add the CNB_CLX object to your ABCIP hierarchy

1. Select and right-click on the **New_BACKPLANE_CLX_000** object.
2. Select **Add CNB_CLX Object** from the shortcut menu. The **New_CNB_CLX_000** object is created.
3. Rename the newly created object as appropriate. The **CNB_CLX** Parameters view is displayed.



This configuration view has two parameters, one of which is configurable:

Port Type: Information provided automatically by the DAServer Manager (ControlNet Communication)

Slot Number: A sequential number beginning with 0 (zero) assigned to each slot in a ControlNet communications interface module.

- The slot number indicates where the sub-module resides.
- The valid range is 0 - 16.
- The default value is 0 (zero).

CNB_FLX Object

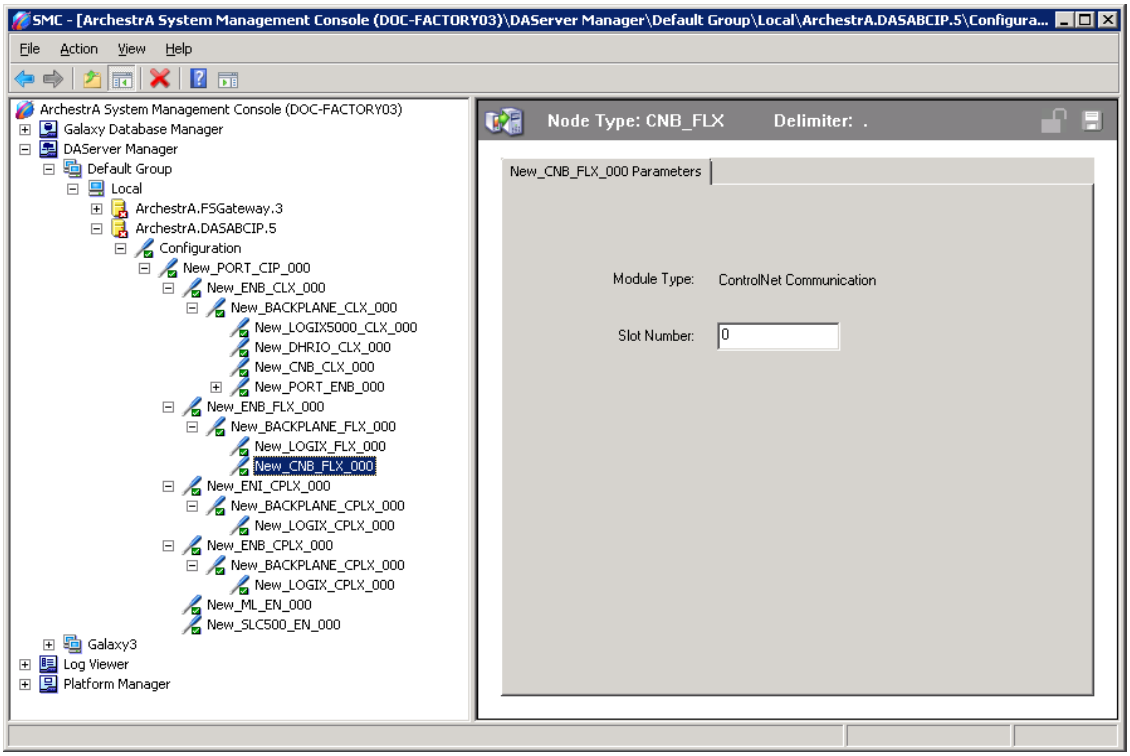
The **CNB_FLX** object represents the physical Allen-Bradley FlexLogix ControlNet Communication Daughter Card.

- 1788-CNC(R)
- 1788-CNF(R)

This object is hosted by **BACKPLANE_FLX.**

To add the CNB_FLX object to your ABCIP hierarchy

1. Select and right-click on the **NEW_BACKPLANE_FLX_000** object.
2. Select **Add CNB_FLX Object** from the shortcut menu. The **New_CNB_FLX_000** object is created.
3. Rename the newly created object as appropriate. The **CNB_FLX** Parameters view is displayed.



This configuration view has two parameters, one of which is configurable:

Port Type: The information is provided automatically by the DAServer Manager (ControlNet Comm.)

Slot Number: A sequential number beginning with 0 (zero) assigned to each slot in a FlexLogix chassis.

- The slot number indicates where the module resides.
- The valid range is 0 - 16.
- The default value is 0.

Note: ABCIP DAServer supports single hops from one ControlNet link to another for accessing data in the target ControlLogix or FlexLogix processor. That is, an additional level of Logix_CLX or Logix_FLX object can be populated under the respective BACKPLANE_CLX_000 or BACKPLANE_FLX_000 object along the CNB_CLX or CNB_FLX hierarchy branch.

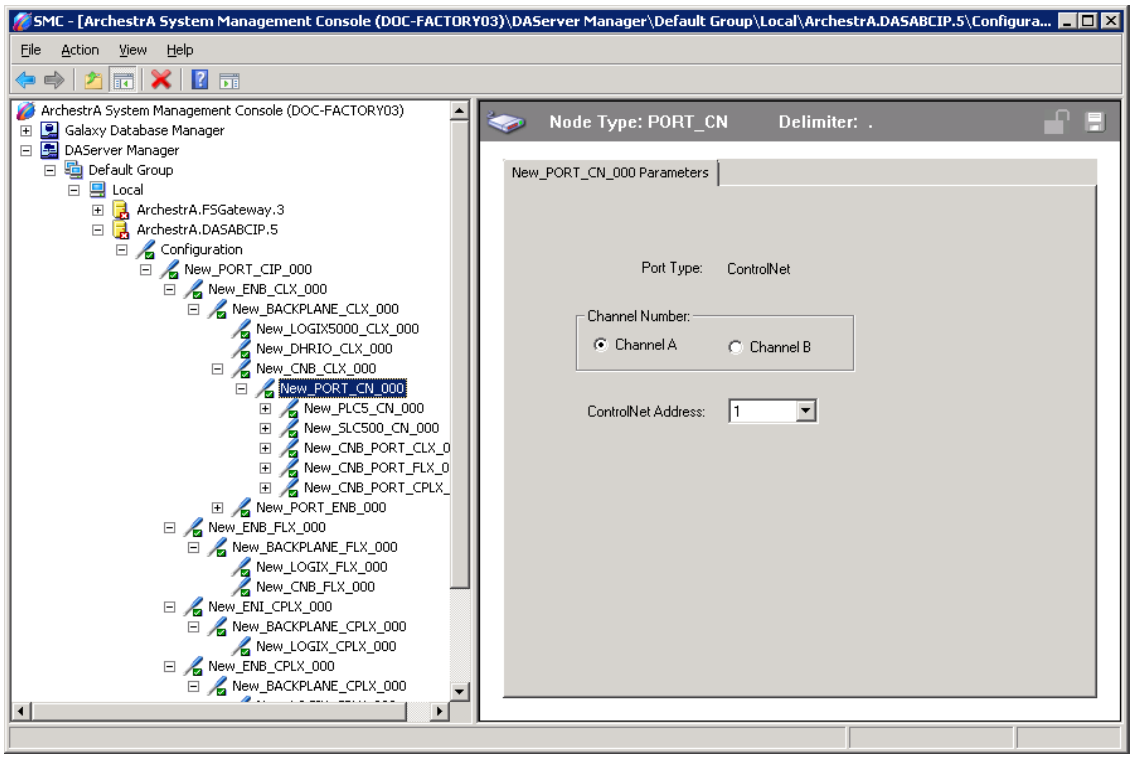
PORT_CN Object

The **Port_CN** object represents the physical ControlNet port for the Allen-Bradley ControlNet Bridge module.

This object is hosted by **CNB_CLX** and **CNB_FLX.**

To add the PORT_CN object to your ABCIP hierarchy

1. Select and right-click on the **New_PORT_CN_000** object.
2. Select **Add PORT_CN Object** from the shortcut menu. The **New_PORT_CN_000** object is created.
3. Rename the newly created object as appropriate. The **PORT_CN** Parameters view is displayed.



This configuration view has three parameters, two of which are configurable:

Port Type: Information provided automatically by the DAServer Manager (ControlNet).

Channel Number: The number of physical channels/ports used on the ControlNet network.

- Select Channel A or Channel B.

ControlNet Address: The node address on the ControlNet network.

- The valid range is 1 - 99 decimal.
- The default value is 1 (one).

PLC5_CN Object

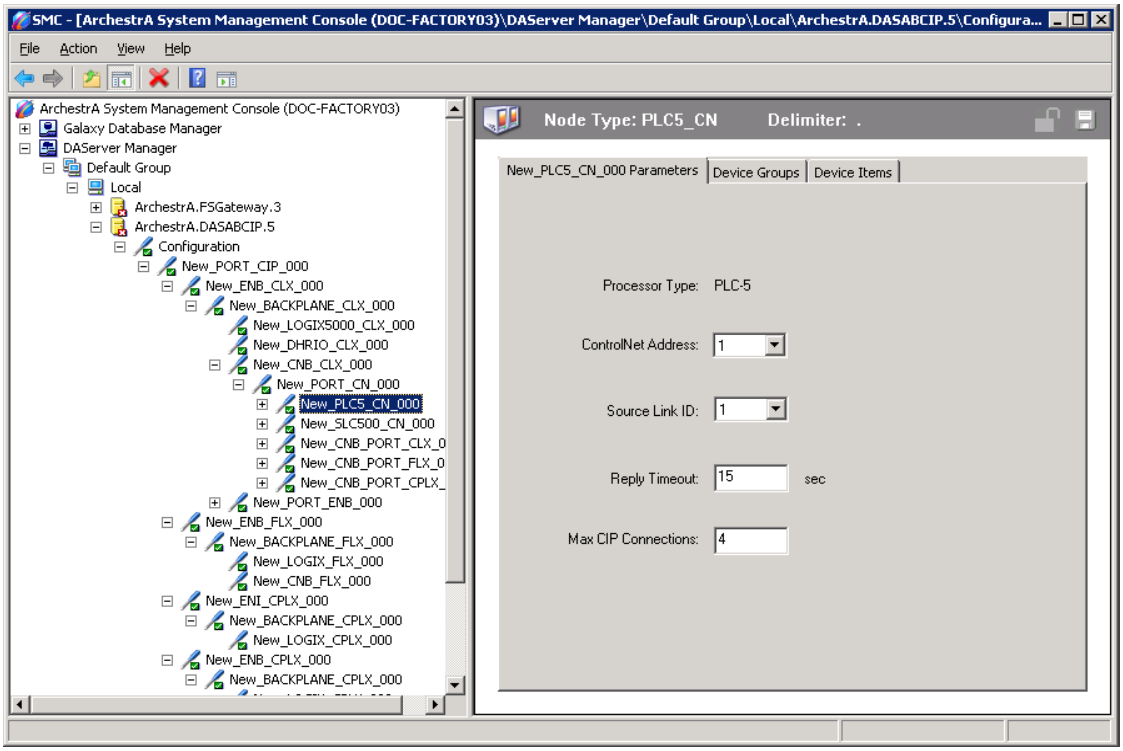
The **PLC5_CN** object represents the physical Allen Bradley ControlNet-capable PLC-5 processor.

- 1785-LxxC

This object is hosted by **PORT_CN.**

To add the PLC5_CN object to your ABCIP hierarchy

1. Select and right-click on the **New_PORT_CN_000** object.
2. Select **Add PLC5_CN Object** from the shortcut menu. The **New_PLC5_CN_000** object is created.
3. Rename the newly created object as appropriate. The **PLC5_CN** Parameters view is displayed.



The configuration view contains five parameters, four of which are configurable:

Processor Type: The information is automatically provided (PLC-5).

ControlNet Address: The node address on the Control Net network (Octal).

- The valid range is 1 - 99 decimal.
- The default value is 1 (one).

Source Link ID: The source link ID of the module. This link ID has to match what has been defined in the ControlNet Routing table for the ControlNet network.

- The valid range is 1 - 199.
- The default value is 1 (one).

Reply Timeout: Enter the maximum amount of time (in seconds) that the DAServer will wait for a response from the controller.

- The valid range is 1 - 300 seconds.
- The default value is 15 seconds.

Max CIP Connections: The maximum number of CIP connections which can be originated from the DAServer to this device.

- The valid range is 1 - 31.
- The default value is 4 (four).

SLC500_CN Object

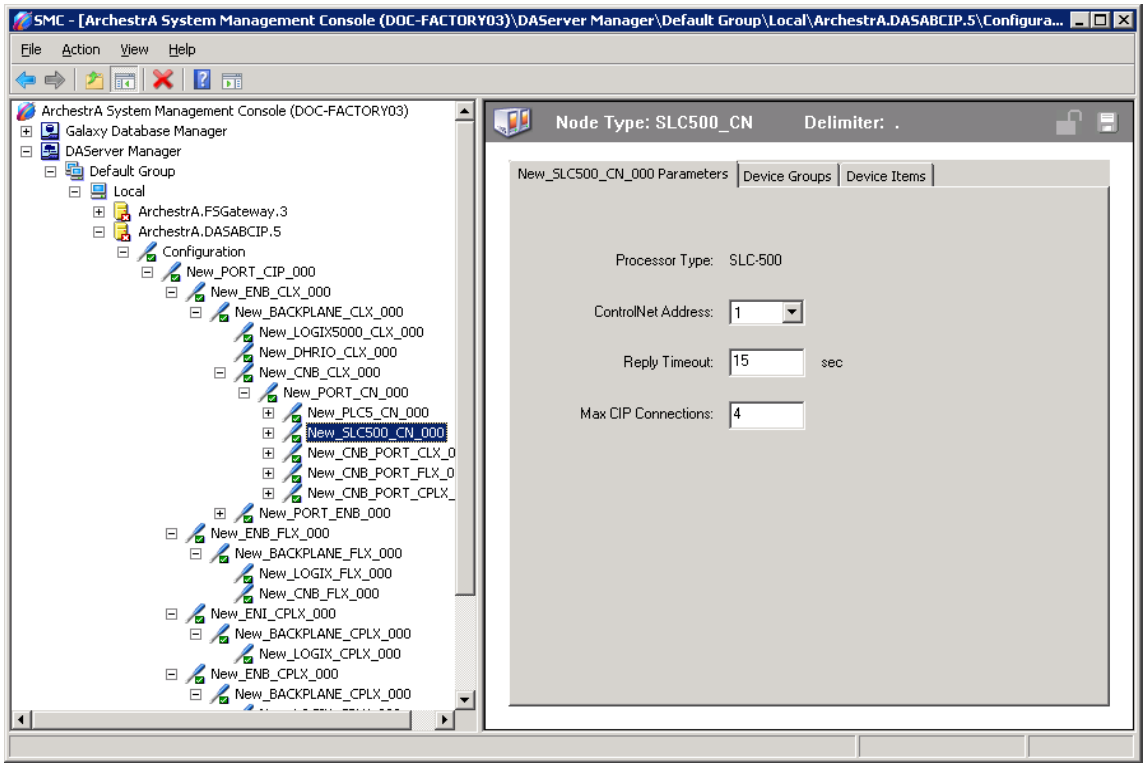
The **SLC500_CN** object represents the physical Allen Bradley SLC500 processor coupled with the Allen-Bradley SLC500 ControlNet RS-232 Interface module (1747-KFC15).

- 1747-L5xx with 1747-KFC15

This object is hosted by **PORT_CN.**

To add the SLC500_CN object to your ABCIP hierarchy

1. Select and right-click on the **New_PORT_CN_000** object.
2. Select **Add SLC500_CN Object** from the shortcut menu. The **New_SLC500_CN_000** object is created.
3. Rename the newly created object as appropriate. The **SLC500_CN** Parameters view is displayed.



The configuration view contains four parameters, three of which are configurable:

Processor Type: The information is automatically provided (SLC500).

ControlNet Address: The node address on the ControlNet Network (Octal).

- The valid range is 1 - 99 decimal.
- The default value is 1 (one).

Reply Timeout: Enter the maximum amount of time (in seconds) that the DAServer will wait for a response from the controller.

- The valid range is 1 - 300 seconds.
- The default value is 15 seconds.

Max CIP Connections: The maximum number of CIP connections which can be originated from the DAServer to this device.

- The valid range is 1 - 31.
- The default value is 4 (four).

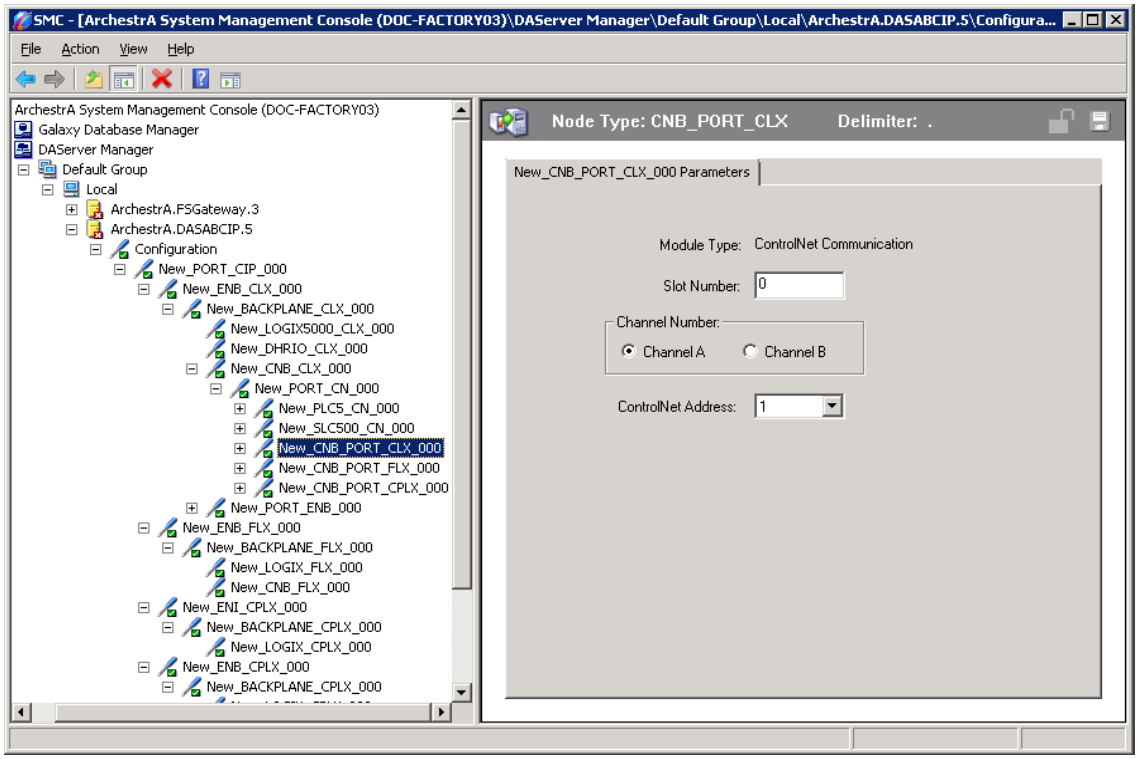
CNB_PORT_CLX Object

The **CNB_Port_CLX** object is a logical representation of the ControlNet port for the Allen-Bradley ControlNet bridge module.

This object is hosted by **PORT_CN.**

To add the CNB_PORT_CLX object to your ABCIP hierarchy

1. Select and right-click on the **New_PORT_CN_000** object.
2. Select **Add CNB_PORT_CLX Object** from the shortcut menu. The **New_CNB_PORT_CLX_000** object is created.
3. Rename the newly created object as appropriate. The **CNB_PORT_CLX** Parameters view is displayed.



This configuration view has four parameters, three of which are configurable:

Port Type: The information is provided automatically by the DAServer Manager (ControlNet).

Slot Number: A sequential number beginning with 0 (zero) assigned to each slot in the ControlLogix ControlNet Bridge module.

- The slot number indicates where the sub-module resides.
- The valid range is 0 - 16.
- The default value is 0 (zero).

Channel Number: The number of physical channels/ports used on the ControlLogix ControlNet interface module.

- Select Channel A or Channel B.

ControlNet Address: The node address on the ControlNet network.

- The valid range is 1 - 99 decimal.
- The default value is 1 (one).

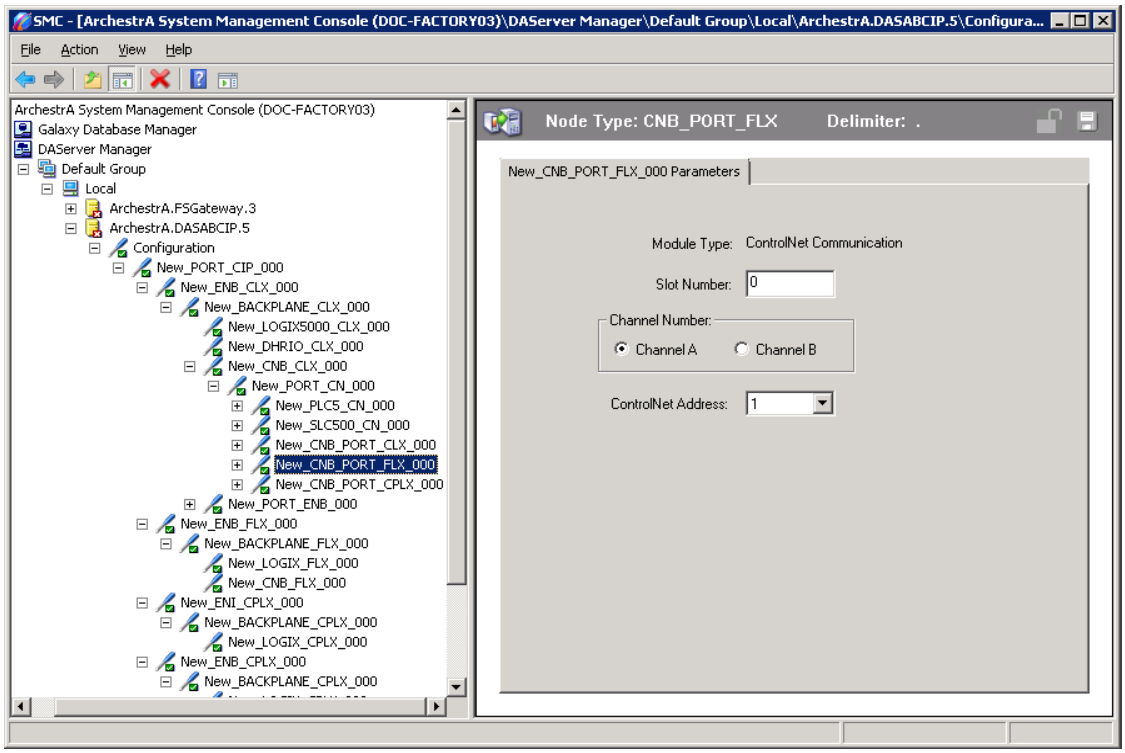
CNB_PORT_FLX Object

The **CNB_Port_FLX** object represents the physical ControlNet port for the Allen-Bradley FlexLogix ControlNet Communication Daughter Card.

This object is hosted by **PORT_CN.**

To add the CNB_PORT_FLX object to your ABCIP hierarchy

1. Select and right-click on the **New**_PORT_CN_000 object.
2. Select **Add CNB_PORT_FLX Object** from the shortcut menu. The **New_CNB_PORT_FLX_000** object is created.
3. Rename the newly created object as appropriate. The **CNB_PORT_FLX** Parameters view is displayed.



This configuration view has four parameters, three of which are configurable:

Port Type: Information provided automatically by the DAServer Manager (ControlNet).

Slot Number: A sequential number beginning with 0 (zero) assigned to each slot in the ControlLogix ControlNet Bridge module.

- The slot number indicates where the sub-module resides.
- The valid range is 0 - 16.
- The default value is 0.

Channel Number: The number of physical channels/ports used on the ControlLogix ControlNet interface module.

- Select Channel A or Channel B.

ControlNet Address: The node address on the ControlNet network.

- The valid range is 1 - 99 decimal.
- The default value is 1 (one).

CNB_PORT_CPLX Object

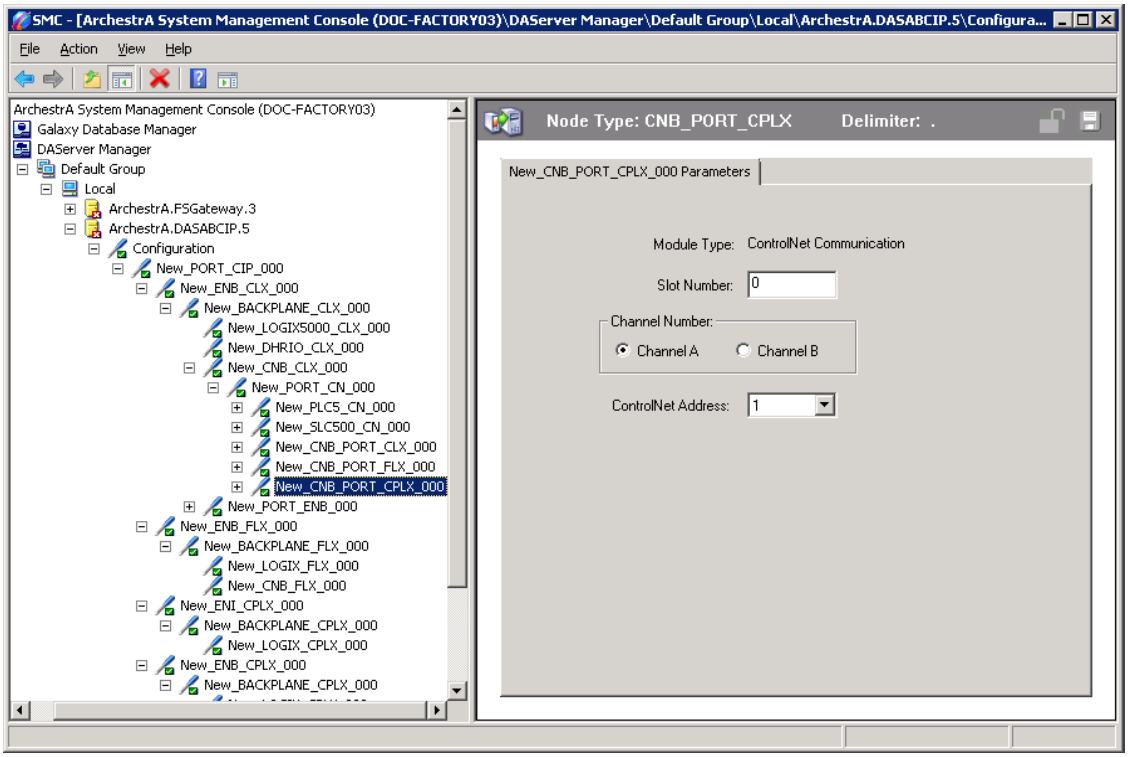
The **CNB_Port_CPLX** object represents the physical ControlNet port for the Allen-Bradley CompactLogix ControlNet processor module.

This object is hosted by **PORT_CN.**

To add the CNB_PORT_CPLX object to your ABCIP hierarchy

1. Select and right-click on the New_PORT_CN_000 object.
2. Select Add **CNB_PORT_CPLX** Object from the shortcut menu. The **New_CNB_PORT_CPLX_000** object is created.

3 Rename the newly created object as appropriate. The **CNB_PORT_CPLX** Parameters view is displayed.



This configuration view has four parameters, three of which are configurable:

Port Type: The information is provided automatically by the DAServer Manager (ControlNet Communication).

Slot Number: A sequential number beginning with 0 (zero) assigned to each slot in the ControlLogix ControlNet Bridge module.

- The slot number indicates where the sub-module resides.
- The valid range is 0 - 16.
- The default value is 0 (zero).

Channel Number: The number of physical channels/ports used on the ControlLogix ControlNet interface module.

- Select Channel A or Channel B.

ControlNet Address: The node address on the ControlNet network.

- The valid range is 1 - 99 decimal.
- The default value is 1 (one).

The DeviceNet Network

The following Allen-Bradley controllers can be configured to communicate with I/O data from the DeviceNet devices for the DAServer to access using the following methods:

- ControlLogix controller by means of its DeviceNet Bridge module.
- FlexLogix controller by means of its DeviceNet daughter-card.
- CompactLogix controller by means of its DeviceNet scanner.
- PLC-5 controller by means of its DeviceNet scanner.
- SLC500 controller by means of its DeviceNet scanner.
- MicroLogix controller by means of its DeviceNet scanner.

Important: The DeviceNet connectivity is achieved with the DeviceNet scanner attached to the corresponding controller. The ABCIP DAServer does not internally implement the DeviceNet protocol.

The Data Highway Plus Network

Routing through the **DHRIO_CLX** object, the ABCIP DAServer accesses data from the PLC-5 and SLC500 processors on the Data Highway Plus network, as well as the MicroLogix processors on the DH485 network via the DH+/DH485 Bridge module (1785-KA5).

DHRIO_CLX Object

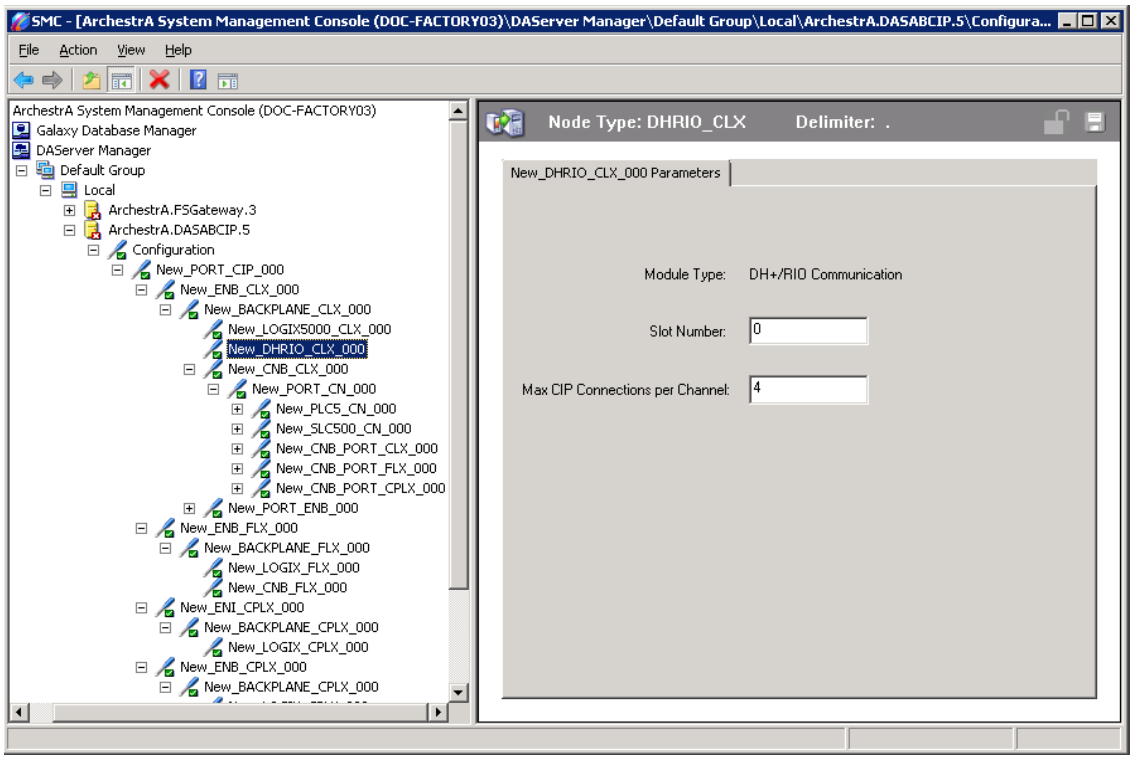
The **DHRIO_CLX** object represents the physical Allen-Bradley ControlLogix DH+/RIO Communication Interface module.

- 1756-DHRIO

This object is hosted by **BACKPLANE_CLX**.

To add the DHRIO_CLX object to your ABCIP hierarchy

1. Select and right-click on the **New_BACKPLANE_CLX_000** object.
2. Select **Add DHRIO_CLX Object** from the shortcut menu. The **New_DHRIO_CLX_000** object is created.
3. Rename the newly created object as appropriate. The **DHRIO_CLX** Parameters view is displayed.



This configuration view has three parameters, two of which are configurable:

Module Type: Information provided automatically by the DAServer Manager (DH+/RIO Communication).

Slot Number: A sequential number beginning with 0 (zero) assigned to each slot in a ControlLogix DH+/RIO Bridge module.

- The slot number indicates where the sub-module resides.
- The valid range is 0 - 16.
- The default value is 0 (zero).

Max CIP Connections per Channel: The maximum number of CIP connections allowed per channel.

- The valid range is 1- 31.
- The default value is 4 (four).

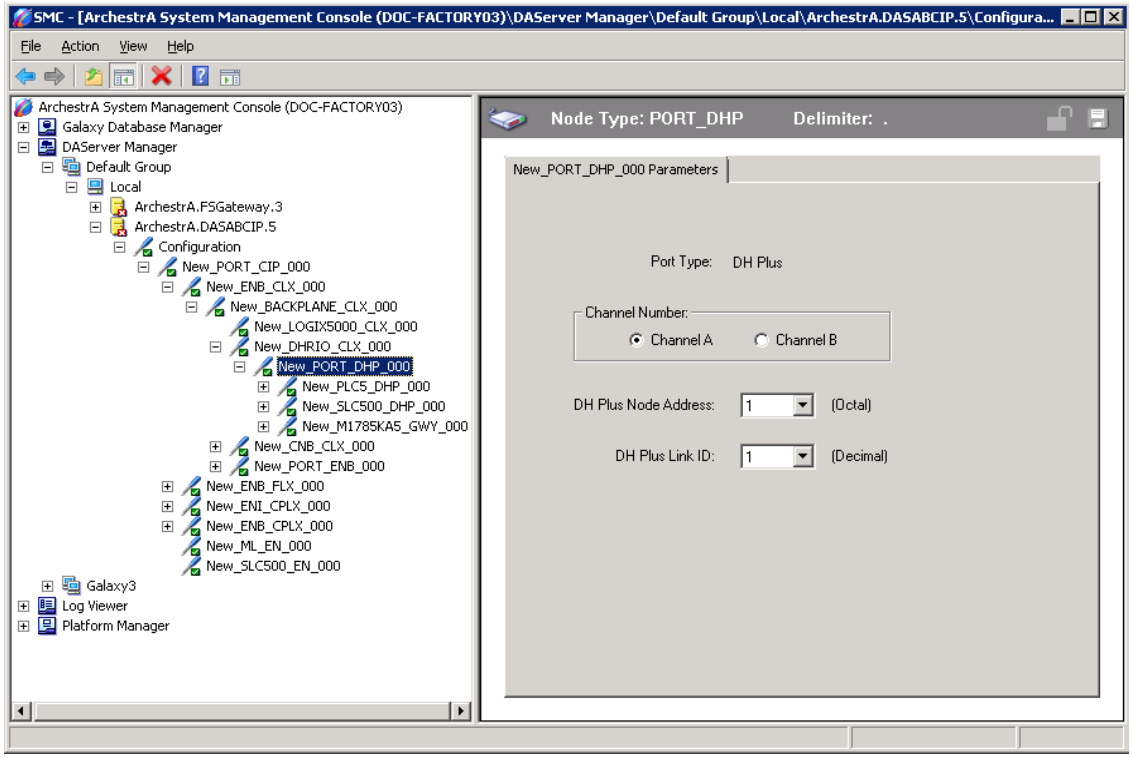
PORT_DHP Object

The **PORT_DHP** object represents the physical DH+ port for the Allen-Bradley DH+/RIO Communication Interface e module.

This object is hosted by **DHRIO_CLX**.

To add the PORT_DHP object to your ABCIP hierarchy

1. Select and right-click on the **New_DHRIO_CLX_000** object.
2. Select **Add PORT_DHP Object** from the shortcut menu. The New_PORT_DHP_000 object is created.
3. Rename the newly created object as appropriate. The **PORT_DHP** Parameters view is displayed.



4. This configuration view has four parameters, three of which are configurable:

   Port Type: Information provided automatically by the DAServer Manager (DH Plus).

   Channel Number: The number of physical channels/ports used on the ControlLogix DH+/RIO Bridge module.

   - Select Channel A or Channel B.

   DH Plus Node Address: The node address on the DH+ network (Octal).

- The valid range is 0 - 77 octal.
- The default value is 1 (one) octal.

DH Plus Link ID: The DH+ link ID of the channel.

- The link ID is defined in the DHRIO routing table for the channel.
- The valid range is 1 - 199.
- The default value is 1 (one).

PLC5_DHP Object

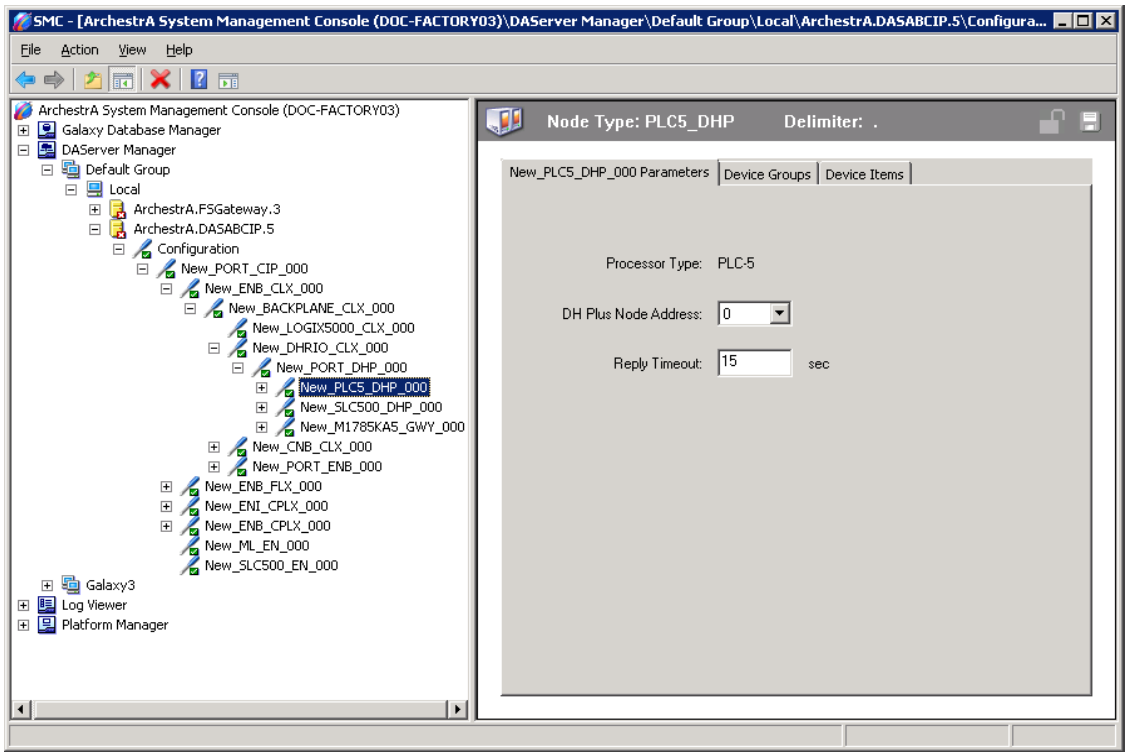
The **PLC5_DHP** object is represents the physical Allen-Bradley PLC-5 processor on the Data Highway Plus network.

- 1785-Lxx(B)
- 1785-LxxC
- 1785-LxxE
- 1785-LxxL

This object is hosted by **PORT_DHP.**

To add the PLC5_DHP object to your ABCIP hierarchy

1. Select and right-click on the **New_PORT_DHP_000** object.
2. Select **Add PLC5_DHP Object** from the shortcut menu. The New_PLC5_DHP_000 object is created.
3. Rename the newly created object as appropriate. The **PLC5_DHP** Parameters view is displayed.



The configuration view contains three parameters, two of which are configurable:

Processor Type: The information is automatically provided (PLC-5).

DH Plus Node Address: The node address on the DH+ network (Octal).

- Select the DH+ node number from the drop-down box.
- The valid range is 0 - 77 octal.
- The default value is 0 (zero) octal.

Reply Timeout: Enter the maximum amount of time (in seconds) that the DAServer will wait for a response from the controller.

- The valid range is 1 - 300 seconds.
- The default value is 15 seconds.

SLC500_DHP Object

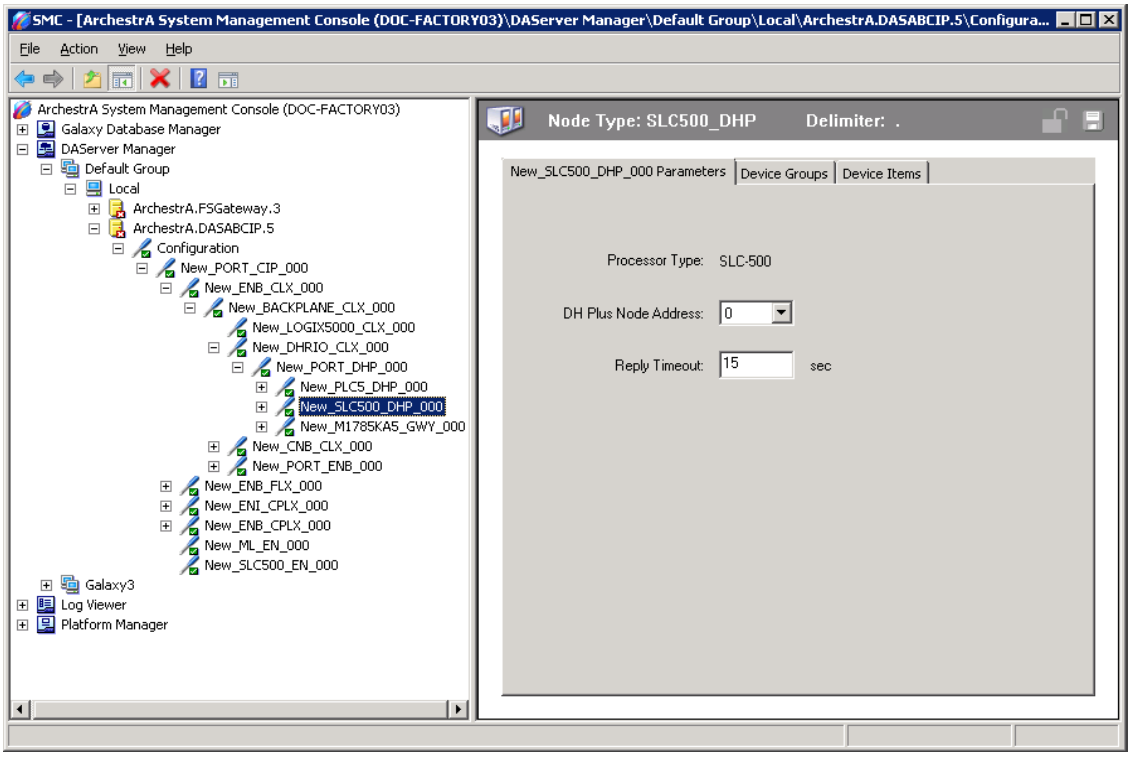
The **SLC500_DHP** object represents the physical Allen-Bradley SLC500 processor on the Data Highway Plus network.

- 1747-L54x

This object is hosted by **PORT_DHP**.

To add the SLC500_DHP object to your ABCIP hierarchy

1. Select and right-click on the **New_PORT_DHP_000** object.
2. Select Add **SLC500_DHP** Object from the shortcut menu. The **New_SLC500_DHP_000** object is created.
3. Rename the newly created object as appropriate. The **SLC500_DHP** Parameters view is displayed.



The configuration view contains three parameters, two of which are configurable:

Processor Type: The information is automatically provided (SLC500).

DH Plus Node Address: The node address on the DH+ network (Octal).

- Select the DH+ node number from the drop-down box.
- The valid range is 0 - 77 octal.
- The default value is 0 (zero) octal.

Reply Timeout: Enter the maximum amount of time (in seconds) that the DAServer will wait for a response from the controller.

- The valid range is 1 - 300 seconds.
- The default value is 15 seconds.

M1785KA5_GWY Object

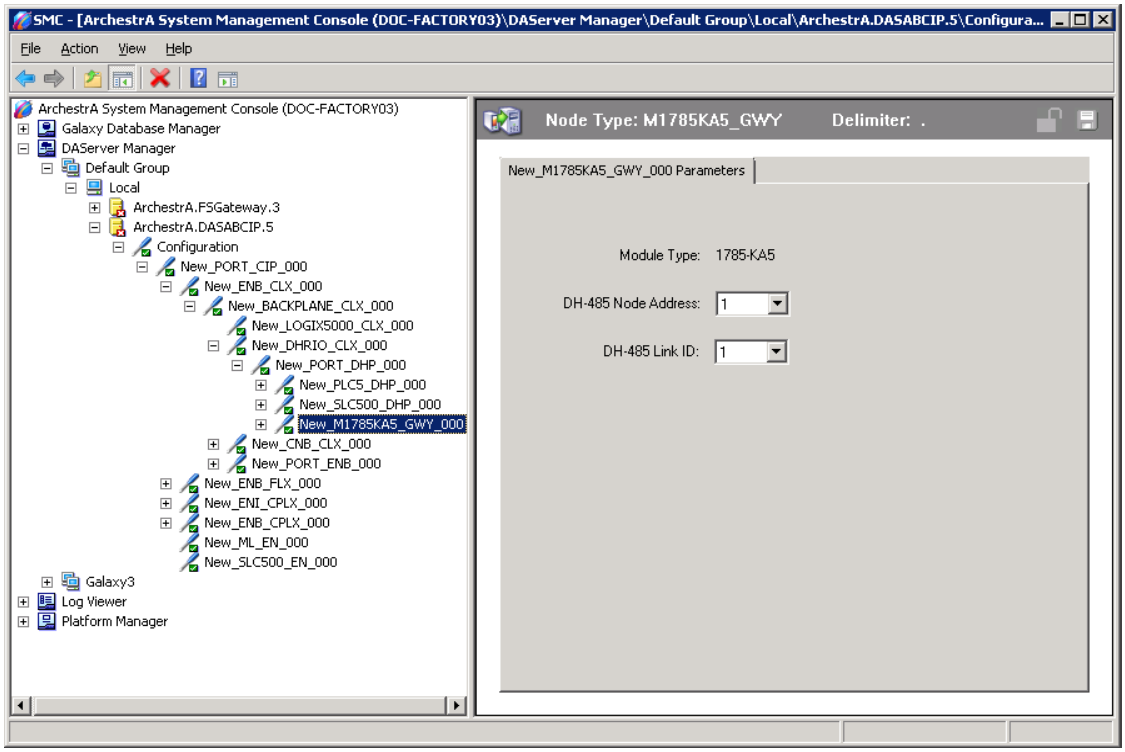
The **M1785KA5_GWY** object represents the physical Allen-Bradley DH+/DH485 Gateway (1785-KA5) Communication module.

- 1785-KA5

This object is hosted by **PORT_DHP**.

To add the M1785KA5_GWY object to your ABCIP hierarchy

1. Select and right-click on the **New_PORT_DHP_000** object.
2. Select **Add M1785KA5_GWY Object** from the shortcut menu. The New_M1785KA5_GWY_000 object is created.
3. Rename the newly created object as appropriate. The **M1785KA5_GWY** Parameters view is displayed.



There are three parameters in this configuration view, two of which are configurable:

Module Type: The information is automatically provided (1785-KA5).

DH485 Node Address: The node address on the DH485 network.

- Select the DH485 node number from the drop-down box.
- The valid range is 1 - 31.
- The default value is 1 (one).

DH485 Link ID: The DH485 link ID of the module.

- This link ID has to match what has been defined in the DHRIO Routing table for the DH+ Bridge.
- The valid range is 1 - 199.
- The default value is 1 (one).

ML_DH485 Object

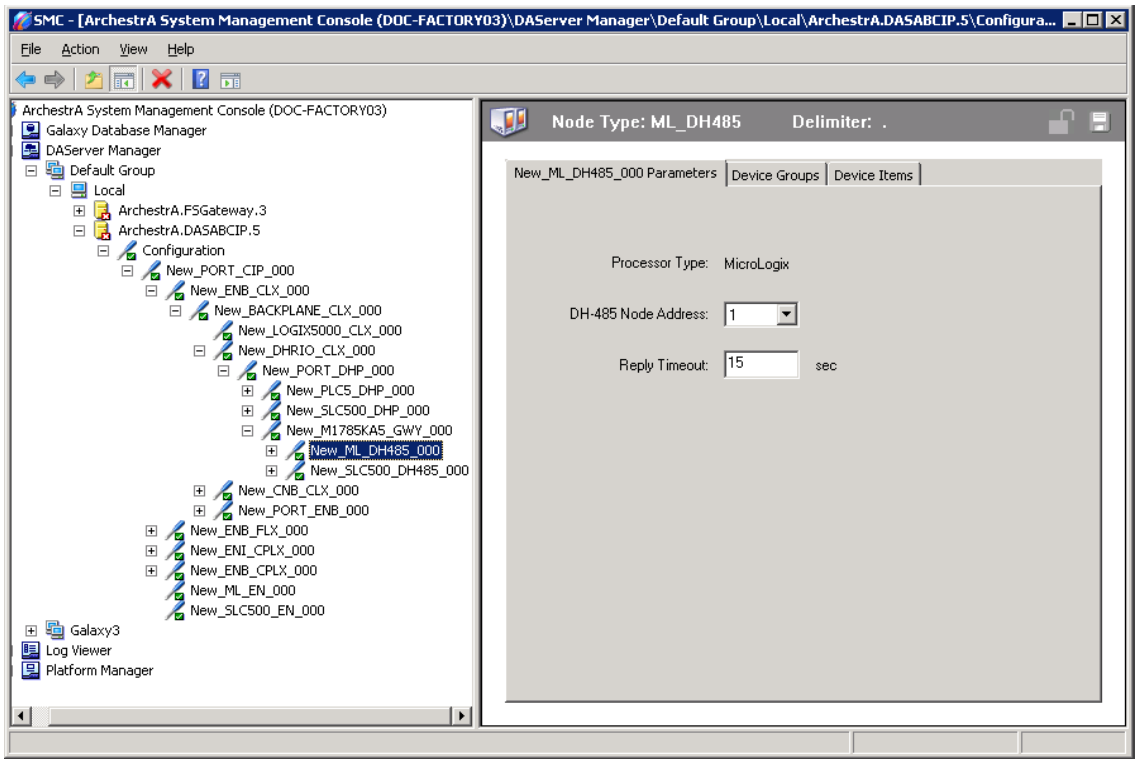
The **ML_DH485** object represents the physical Allen-Bradley MicroLogix processor coupled with the Allen-Bradley Advanced Interface Converter for DH485 (1761-NET-AIC).

- 176x-Lxxx with 1761-NET-AIC

This object is hosted by **M1785KA5_GWY**.

To add the ML_DH485 object to your ABCIP hierarchy

1. Select and right-click on the **New_M1785KA5_GWY_000** branch.
2. Select **Add ML_DH485 Object** from the shortcut menu. The **New_ML_DH485_000** object is created.
3. Rename the newly created object as appropriate. The **ML_DH485** Parameters view is displayed.



The configuration view contains three parameters, two of which are configurable:

Processor Type: The information is automatically provided (MicroLogix).

DH485 Node Address: The node address on the DH485 network.

- Valid range is 0 - 31.
- The default value is 1 (one).

Reply Timeout: The maximum amount of time (in seconds) that the DAServer will wait for a response from the controller.

- The valid range is 1 - 300 seconds.
- The default value is 15 seconds.

SLC500_DH485

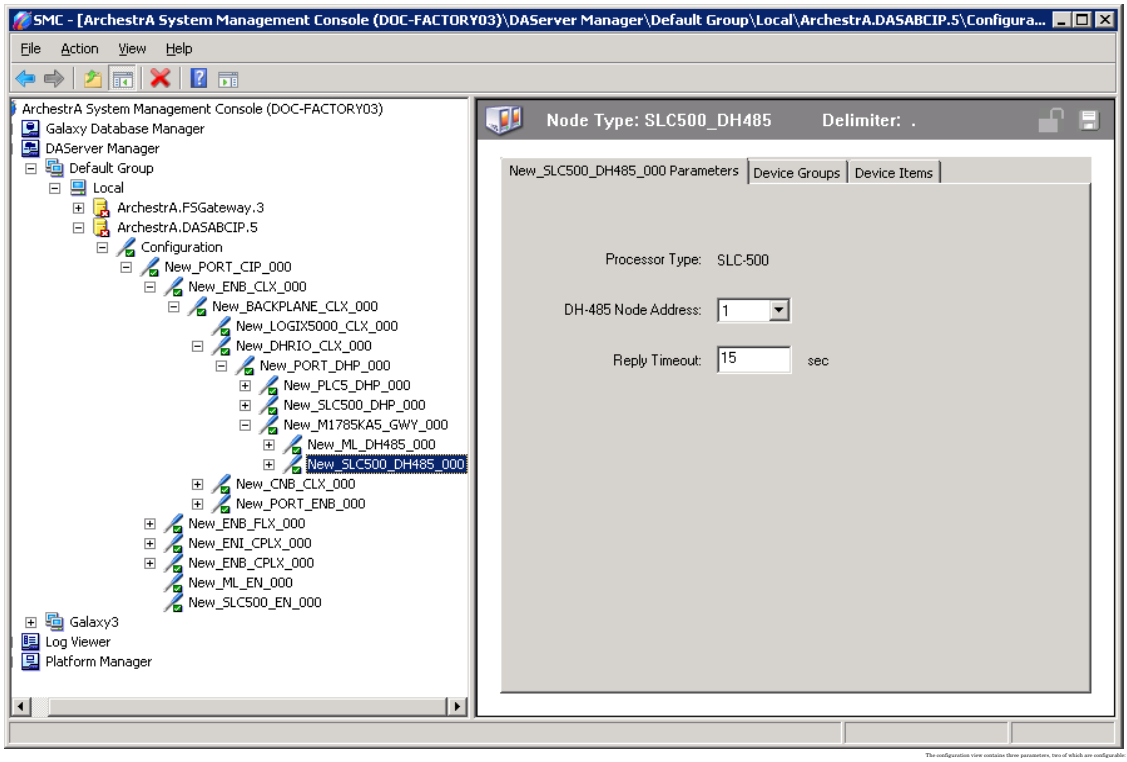
The **SLC500_DH485** object represents the physical Allen-Bradley SLC500 processor coupled with the Allen-Bradley Advanced Interface Converter for DH485 (1761-NET-AIC).

- 1747-L5xx

This object is hosted by **M1785KA5_GWY.**

To add the SLC500_DH485 object to your ABCIP hierarchy

1. Select and right-click on the **New_M1785KA5_GWY_000** branch.
2. Select **Add SLC500_DH485 Object** from the shortcut menu. The **New_SLC500_DH485_000** object is created.
3. Rename the newly created object as appropriate. The **SLC500_DH485** Parameters view is displayed.



Processor Type: The information is automatically provided (SLC500).

DH485 Node Address: The node address on the DH485 network.

- Valid range is 0 - 31.
- The default value is 1 (one).

Reply Timeout: The maximum amount of time (in seconds) that the DAServer will wait for a response from the controller.

- The valid range is 1 - 300 seconds.

- The default value is 15 seconds.

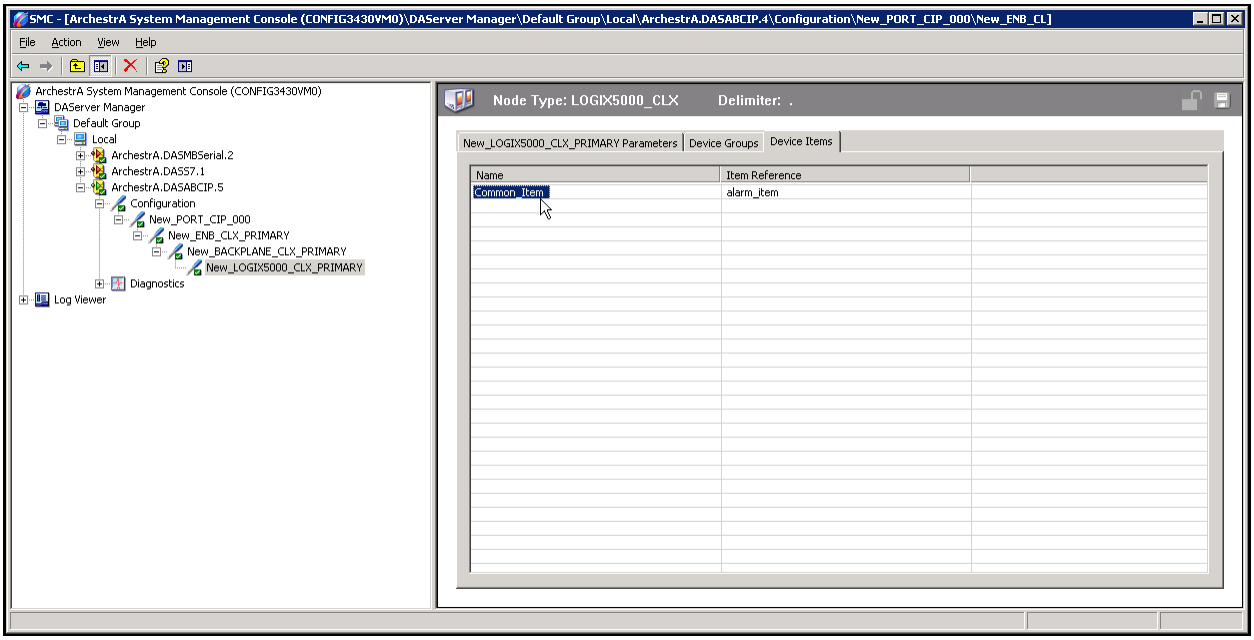
Configuring Device Redundancy

The DAServer Manager provides the ability to assign redundant devices for fail-over protection in the event of device failure. Two devices must be configured in the same DAServer having identical item syntax.

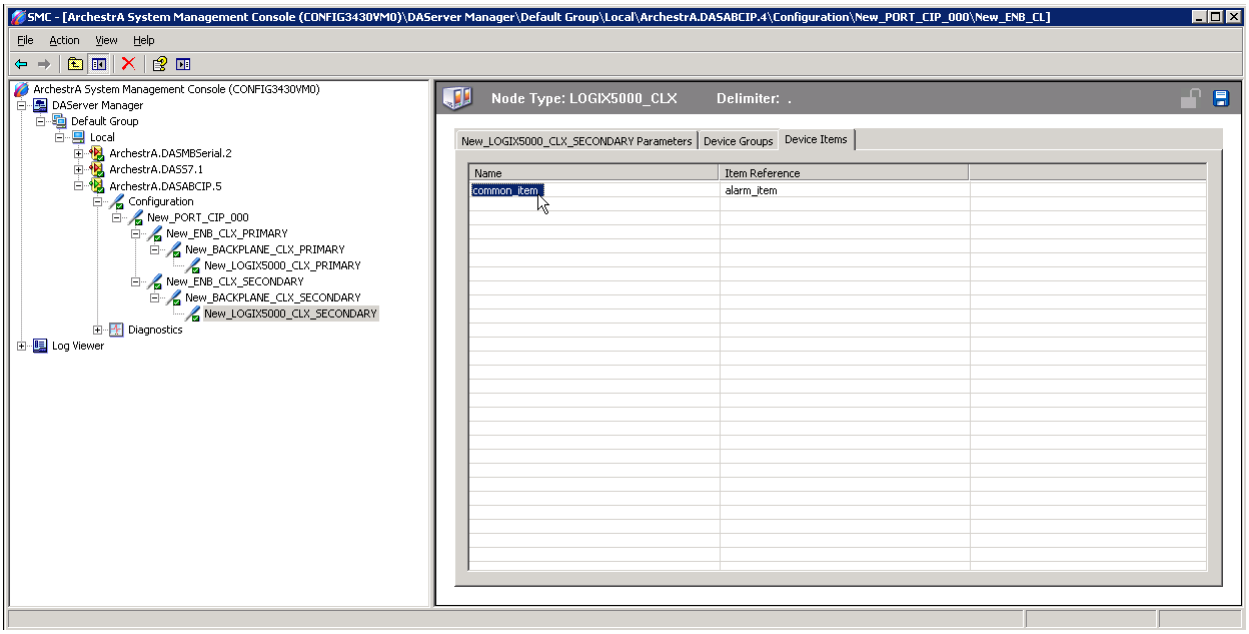
Primary and secondary devices will be setup in the REDUNDANT_DEVICE object in the SMC, along with a common item name (ping item) shared by each device to determine device status.

To setup up a REDUNDANT_DEVICE from the configuration branch:

1. Set-up a primary device and hierarchy in the DAServer Manager in the SMC.
2. Create at least one device item that can be shared between the primary and secondary devices to determine device status.

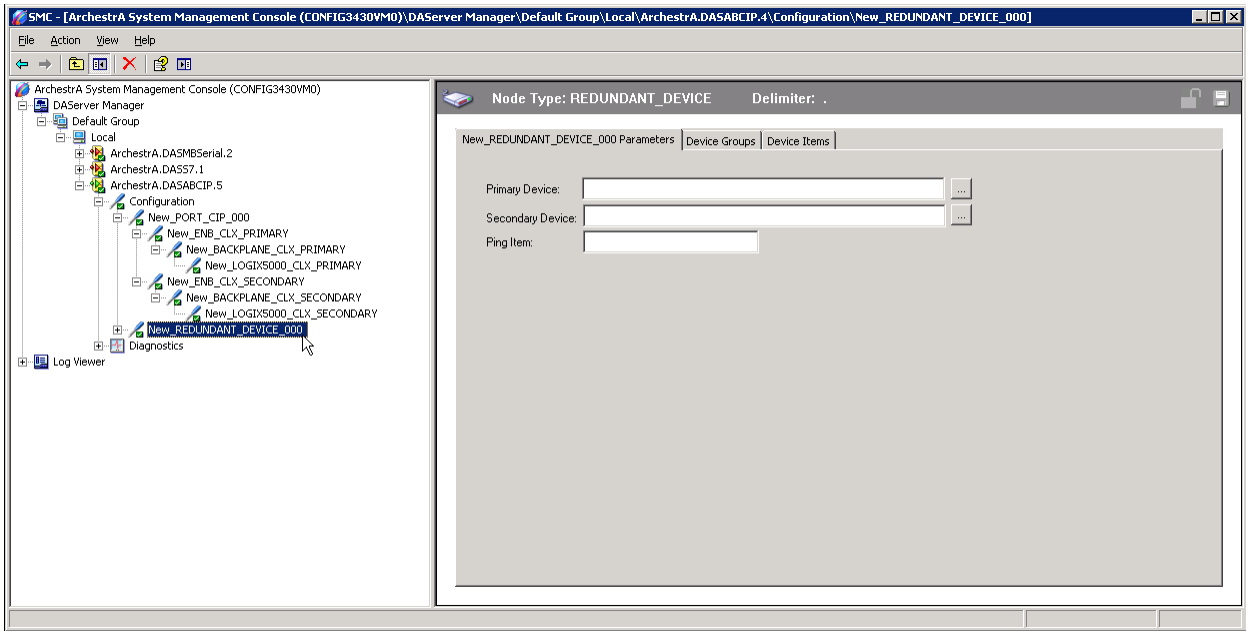


3 Set up an secondary device on the same DAServer. Once again, create an identical device item within the secondary device so that device status can be determined.



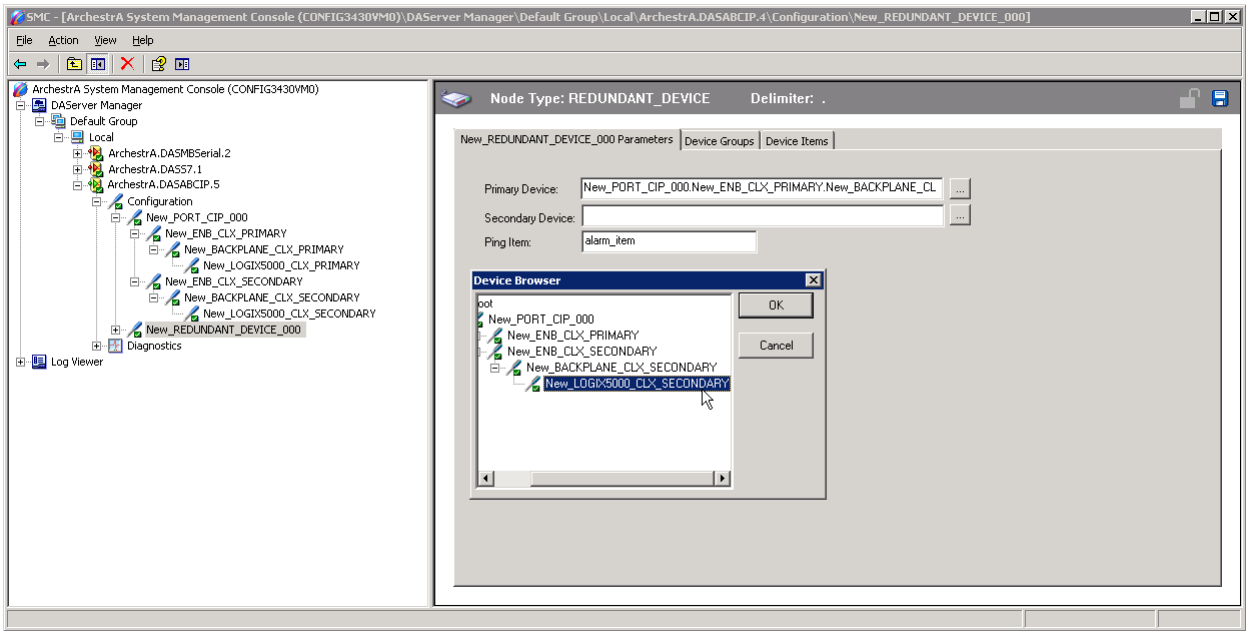
4 Select and right-click on the configuration node under the CIP Network Device object.

5 Select **Add REDUNDANT_DEVICE Object**. An object called **New_REDUNDANT_DEVICE_000** is created.



6 Rename the newly created object as appropriate. The **New_REDUNDANT_DEVICE_000** configuration view is displayed in the Configuration branch of the hierarchy.

7 Enter or use the device browser to select the primary and secondary devices. Save the hierarchy node configuration by clicking on the save icon.



Note: Unsolicited message configuration is not supported from the device redundant hierarchy.

Important: A Ping item must be specified and be a valid tag in both the primary and secondary controllers to determine the connection status for SYSStatus. The Ping item can be a static item in the device such as a firmware version or processor type. If the Ping item is invalid or does not exist in the controller, the failover operation may not work correctly as the value of SYSStatus may continue to stay as FALSE in the standby device.

Chapter 4

Device Groups and Device Items

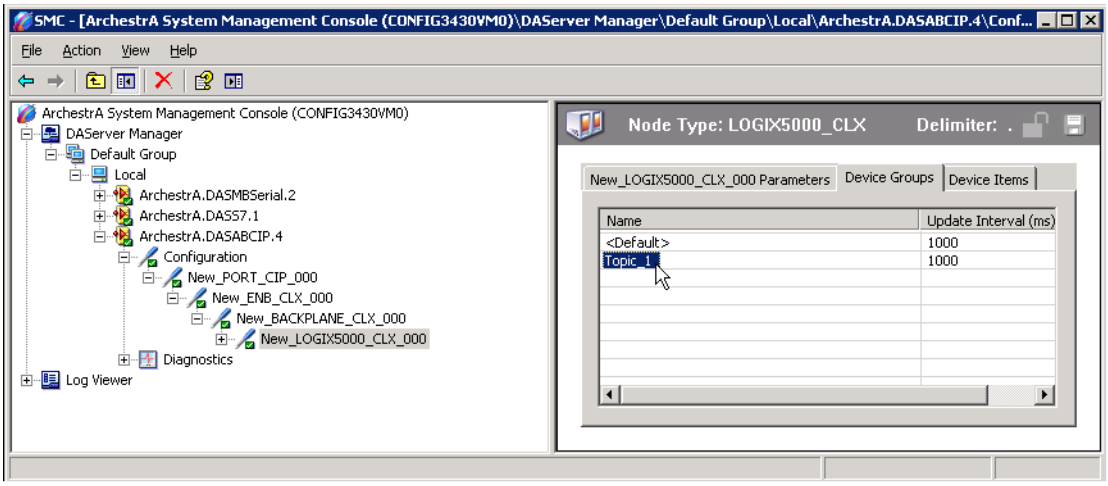
The **Device Group and Device Item** tabs in the DAServer Manager user interface are used to create new, modify, or delete device group and item definitions for an object.

For DDE/SuiteLink communications, one or more device group definitions must exist for each controller that the DAServer will communicate with. Each device group (topic) definition should contain a unique name for the controller associated with it.

Device Group Definitions

The **Device Groups** dialog box is displayed by clicking the **Device Groups** tab in the **CIP, LOGIX5000_CLX, LOGIX_FLX, ML_EN, LOGIX_CPLX, SLC500_EN, PLC5_CN, SLC500_CN, PLC5_DHP, SLC500_DHP, ML_DH485, SLC500_DH485** node configuration view. The **Device Groups** dialog box allows you to add, define, and delete device groups, in addition to configuring default update intervals and editing update intervals for the objects.

Note: When you add a new device group, enter a unique name. When you select another part of the DAServer tree hierarchy, you are prompted to save the modifications to the configuration set.



To create or add device groups

1. Right-click in the **Device Groups** box.
2. Select the **Add** command from the shortcut menu.
   - When adding a new device group, enter a unique name (up to 32 characters long).
3. Click the **Save** icon (the floppy disk icon in the upper right corner).

To make edits on device groups

1. In the **Name** column, double-click on the device group's name to be edited and make the change.
2. In the **Update Interval** column, double-click on the device group's value to be edited and make the change.
3. To enable unsolicited messages, right-click on the device group name and select "edit" to display Device Group Parameters. Select "Support Unsolicited Messages if desired and click OK.
4. Click the **Save** icon (the floppy disk icon in the upper right corner).

To delete device groups

1. Right-click on the device group to be deleted.
2. Select the **Delete** command from the shortcut menu.
   - The DAServer Manager confirmation box is displayed.
3. Click **Yes** to proceed with the deletion.
4. Click the **Save** icon (the floppy disk icon in the upper right corner).

To configure default update intervals

1. Right-click in the **Device Groups** box.
2. Select **Config Default Update Interval** from the shortcut menu.
3. Click the **Save** icon (the floppy disk icon in the upper right corner).

To edit update intervals

1. Double-click its value in the **Update Interval** column and make the edit.
   - Update Interval is the frequency (in milliseconds) that the DAServer acquires data from the topics associated with that device group.
   - Different topics can be polled at different rates in a controller by defining multiple device group names for the same controller and setting a different Update Interval for each device group.

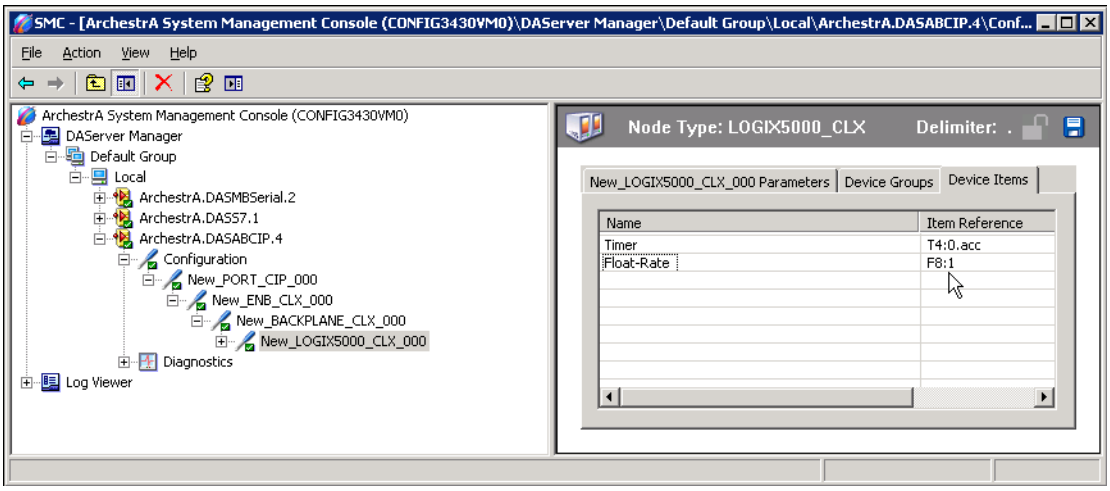
Device Item Definitions

To make it easier to remember lengthy or strictly structured item names, the DAServer enables you to create aliases for these item names. For example, it may be easier for you to remember the item syntax "T4:1.acc" as "Timer1."

The **Device Items** tab in the DAServer Manager user interface is used to create new, modify, delete, export, or import device item definitions for an object. The configuration is performed in the **Device Items** dialog box, which you can open by clicking the **Device Items** tab in the **LOGIX5000_CLX, LOGIX_FLX, ML_EN, LOGIX_CPLX, SLC500_EN, PLC5_CN, SLC500_CN, PLC5_DHP, SLC500_DHP, ML_DH485** or **SLC500_DH485** node configuration view.

Once the Device Items feature is used to configure item names, it provides the DAServer with the capability to perform OPC Item browsing. When the DAServer is running and an OPC client requests item information, the configured items will show up under the controller hierarchy node.

Note: Device items have the precedence in addressing items in the controller device at run time. Items request from the client would be searched from the Device Items Name list first before going out to the controller.



To create or add device items

1. Right-click in the **Device Items** box.
2. Select the **Add** command from the shortcut menu.
3. Type the item name (symbolic name) of your choice in the **Name** column.
   - The device item name must be unique and is limited to 32 characters long.
4. Double-click the line on the **Item Reference** column and enter the correlated item reference (the actual I/O item name in the device) for the device item name you have just selected.
   - For example, "n7:0."
5. Click the **Save** icon (the floppy disk icon in the upper right corner).

Note: System items are not valid item references, but DAServer-specific system items are valid.

To rename device items

1. Right-click on the device item to be renamed.
2. Select **Rename** from the shortcut menu, then make the change.
3. Click the **Save** icon (the floppy disk icon in the upper right corner).

To delete device items

1. Right-click on the device item to be deleted from the list.
2. Select the **Delete** command from the shortcut menu.
3. Click the **Save** icon (the floppy disk icon in the upper right corner).

To clear all device items

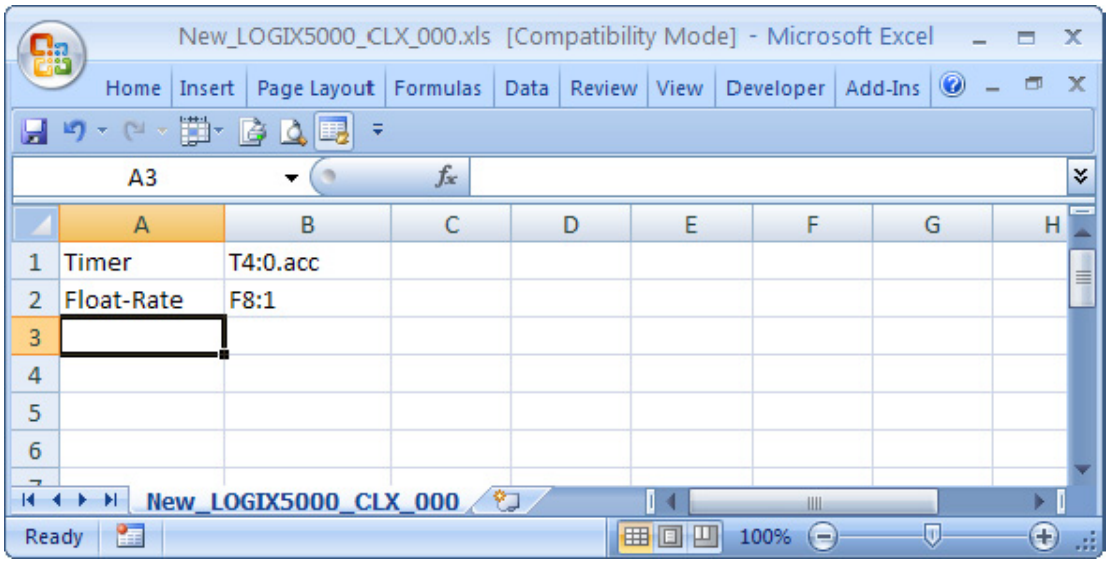
1. Right-click in the **Device Items** box.
2. Select the **Clear All** command from the shortcut menu.
   - The DAServer Manager confirmation box appears.
3. Click **Yes** to confirm the deletion.
   - All the device items listed will be cleared.

Exporting and Importing DAServer Item Data

The Export and Import commands on the shortcut menu enable you to export and import the DAServer item data to and from a CSV file, after the configuration of the Device Items has been completed. These commands will allow you to perform an off-line, large-scale edit on the item data configured for a controller, and import what has been edited back into the controller configuration.

To export DAServer item data to a CSV file

1. Right-click in the **Device Items** box.
2. Select the **Export** command from the shortcut menu.
   - The **Save As** dialog box appears.
   - The file name has defaulted into "PLCHierarchyNodeName.csv," within the current-system-configured default directory.
3. Accept the defaults to save the file or rename the file if appropriate.
   - The file is saved as New_PLC5_DHP_000.csv.
   - It is editable in Microsoft Excel.



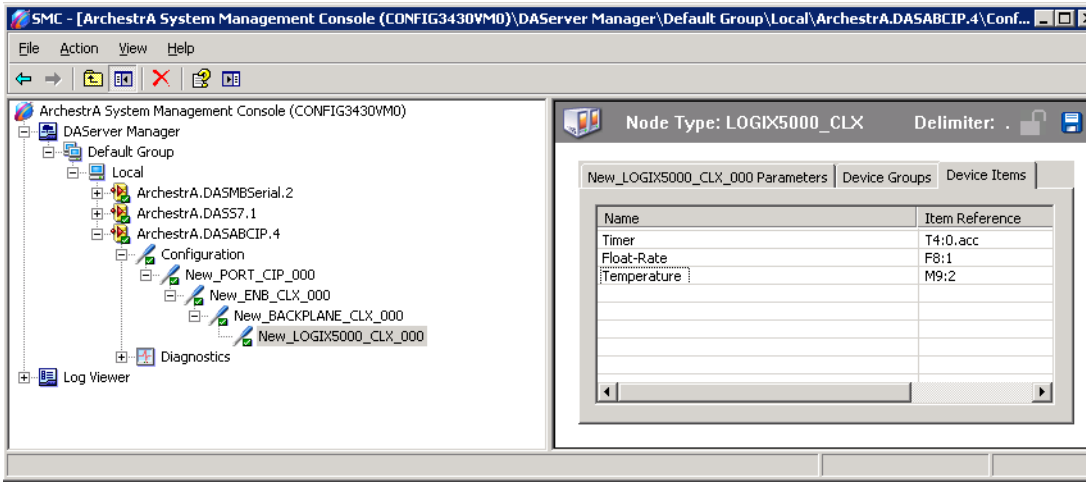
The file can now be edited off-line. It contains one row for each item configured with two columns, Name and Item Reference, respectively.

To import DAServer item data from a CSV file

1. Right-click in the **Device Items** box.
2. Clear all the item data you wish to replace with the edited.csv file by selecting the **Clear All** command.
   - The DAServer confirmation box is displayed.
3. Click **Yes** to confirm the deletion.
   - The data will be cleared.
4. Select the **Import** command from the shortcut menu.
   - The **Open** dialog box appears.
   - It defaults to the .csv file extension within the current-system-configured default directory.
5. Browse and select the specific CSV file you want to import, select it, then click **OK for confirmation**.

   The DAServer Manager will import the file and deposit it in the **Device Items** box.

   During the imported file processing:
   - New item references will be added based on unique names.
   - If there are duplicate names, you will be provided with the ability to replace the existing entry with the new entry, or ignore the new entry.



When the DAServer is running and an OPC client requests item information, the imported configured items will show up under the controller hierarchy node.

Note: When you select another part of the DAServer tree hierarchy, you are prompted to save the modifications to the configuration set.

Each configuration view associated with nodes in the DAServer hierarchy tree has a common feature, the **Save** button located on the upper-right corner of the configuration view.

When you modify any parameters in the **Device Groups** dialog box, click **Save** to implement the new modifications.

If you do not click **Save**, you will be prompted to save the new data to the configuration set.

Scan-Based Message Handling

Wonderware's DAServers are based on the concept of polling a hardware device for information. This polling is driven by a need which is expressed in the form of requests from one or more clients. After a particular piece of information has been requested by a client, the DAServer formulates its own request and sends that request to the hardware device. The DAServer then waits for a response to its request. After the information has been received, the DAServer passes that information back to the client, and repeats the process until all clients have ceased requesting information.

The rate at which the DAServer will poll a particular device for a specific piece of information is defined in the device group (topic definition) inside the DAServer, using a parameter called the Update Interval. When setting this parameter, there is always a trade-off between the update speed of the device group and the resulting decrease in system responsiveness.

Because you more than likely want very fast response, the temptation is to set the Update Interval to a value close to 0 seconds. However, if every point is polled at this rate, the entire system will suffer due to slow response time. Therefore, you should compromise, and set the Update Interval to a more reasonable value. You could also create multiple device groups for each device, setting the Update Interval to different values, then assigning different items to different device groups depending on how quickly the values change and how quickly you want to see an update of those changes.

Some items, like alarms, change very infrequently but because of their importance require very fast updates. For those kinds of items, you should set the Update Interval at a very small value. If you desire an immediate response, set the Update Interval at 1.

Unsolicited Message Handling

In the world of controllers and DAServers, it is obvious that a controller will know when a critical event has occurred before the DAServer will have a chance to poll for that data. Therefore, it would seem natural that if a critical event occurs, the controller should have the capability to inform the DAServer immediately, without having to wait for the DAServer to poll it.

This is the role of an unsolicited message. After a controller has determined that a critical condition exists, it can generate a message immediately sent to the DAServer without a prior request from the DAServer. The unsolicited message implementation requires both the messaging instructions properly programmed in the controller logic and the device group appropriately configured in the DAServer.

Note: The use of unsolicited messages requires configuration changes in the controller. Please refer to the related Rockwell Automation documentation for procedures to set up unsolicited messages from the supported controller processors.

The ABCIP DAServer supports unsolicited messages from the following processors:

- PLC-5 and SLC 5/04 processors on the Data Highway Plus network.

Note: The following non-Logix processor configuration does **not** support unsolicited message handling:
MicroLogix with 1785-KA5 via ControlLogix Gateway (1756-DHRIO and 1756-ENB).

- Logix5000 and MicroLogix 1100 processors on the Ethernet network.

Note: The following Logix5000 processor configuration does **not** support unsolicited message handling:
CompactLogix going through the EtherNet/IP interface module (1761-NET-ENI).

- Logix5000, PLC-5, and SLC500 processors on the ControlNet network.

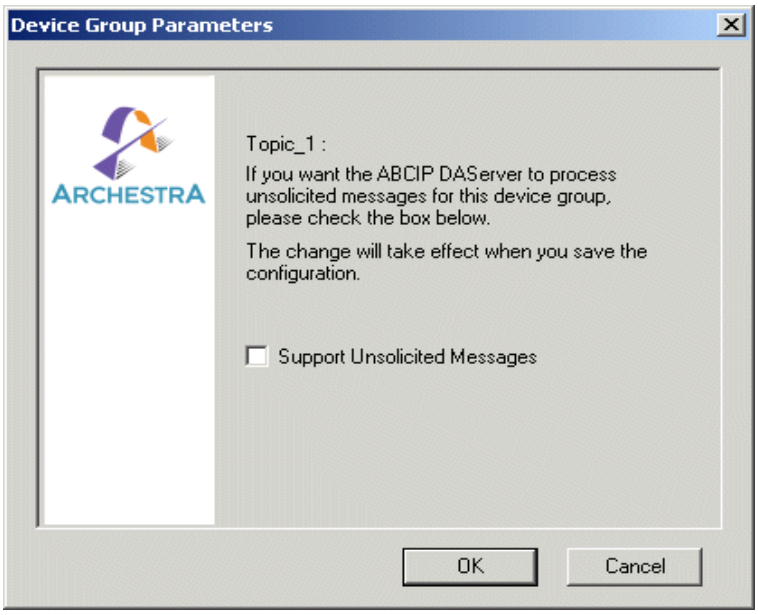
Note: The following SLC500 processor configuration does **not** support unsolicited message handling:
SLC500 using 1747-KFC15 interface on ControlNet via the ControlLogix Gateway (1756-CNB and 1756-ENB) to RSLinx on Ethernet.

To configure the ABCIP DAServer to receive unsolicited messages

This option is available only to the controllers listed above.

1. Click on the target controller node under the controller branch of the DAServer hierarchy.
2. Select the **Device Group** tab of the configuration view.
3. Add a new device group or select an existing device group.
4. Right-click on the device group name, then select **Edit** from the shortcut menu.

   The **Device Group Parameters** dialog box is displayed.



5. In the **Device Group Parameters** dialog box, select the **Support Unsolicited Messages** check box.
6. Click **OK**.

Note: Because the status of **Support Unsolicited Messages** check box cannot be readily viewed from the **Device Groups** tab, proper naming of device groups which support unsolicited messages is strongly recommended.

7 If appropriate, you can modify the Update Interval to "0".

8 Save the configuration change by clicking the **Save** button.

Note: To enhance performance in message handling, the default for the device group is to **not** provide unsolicited message data; therefore, the **Support Unsolicited Messages** check box is **not** checked. The setting of this check box is hot-configurable. Unsolicited message handling will take effect in the DAServer as soon as the change made in the configuration view is saved.

The DAServer supports target-specific unsolicited messages.

Target-specific: This method involves sending messages to its target as specified by a given static IP address.

- Configure the appropriate message instructions in the controller with the proper path (including the destination IP address) for sending the unsolicited messages.
- Configure the computer, in which the DAServer resides to receive unsolicited messages from the controller, with the corresponding IP address.
- Two instances of target-specific unsolicited messages are generated by the DAServer:
- If the value of "Update Interval" for a topic is 0 (zero), the server will poll this topic only one time at the start. After that, only an unsolicited message will update the data.
- If the value of "Update Interval" for a topic is >0 (greater than zero), the server will update the data for a particular item immediately upon receiving an unsolicited message for the item. The DAServer will also update the data at every Update Interval.

Note: For details in setting the Allen-Bradley hardware for unsolicited messaging, please refer to the related Rockwell Automation documentation.

Note: The DAServer requires unsolicited messages to be configured as "Connected" in Logix5000 processors.

To receive unsolicited messages

1. Activate the DAServer.
2. Add the items, defined in the controller for unsolicited messages, for updates under the device group set up for receiving unsolicited messages.

To access the settings for device groups

1. Click on the target controller node under the **Configuration** hierarchy of your DAServer.
2. Select the **Device Groups** tab of the configuration view pane at right.
3. Right-click on the device group name, then select **Edit** from the shortcut menu.

Note: Unsolicited PLC-5 Typed Write using CIP with Source ID method from the Logix5000 processor is not supported. Instead, configure the message instruction with CIP Data Table Write using CIP method from the Logix5000 processor.

Chapter 5

Managing Your DAServer

After you configure the DAServer, there are two steps to take before you can access data with your client application.

The first step is to determine what kind of client applications are to be used with this DAServer. If any of your client applications use DDE/SuiteLink, you must configure the DAServer as a service. If only OPC client applications will be used, you can configure the DAServer as a service or as not a service.

The last step is to activate the DAServer. Some client applications can programatically activate the DAServer. If you configure the DAServer as an automatic service, the DAServer is started and activated when the computer on which the DAServer is installed starts up. If you configure the DAServer as a manual service, the DAServer is not started when the computer starts up. Instead, it is started upon the first connection from an OPC client or when activated from the DAServer Manager.

After a DAServer is running as an auto or manual service, it stays running until explicitly stopped in the DAServer Manager or the computer shuts down.

Configuring the DAServer as Service

To support DDE/SuiteLink clients, the DAServer must be configured as a service.

To configure the DAServer as a service

1. In the DAServer Manager, navigate to the DAServer.
   - Expand **DAServer Manager**, expand the node group, and then expand **Local** or the remote computer name.
2. Right-click **ArchestrA.DASABCIP.5** and then click **Configure As Service**.
3. Click either **Auto Service** or **Manual Service**.
4. Read the warning message and click **Yes**.

Configuring the DAServer as Not a Service

The DAServer can only be set to run as not a service when the DAServer is in the deactivated state.

To configure the DAServer as not a service

1. In the DAServer Manager, navigate to the DAServer.
   - Expand **DAServer Manager**, expand the node group, and then expand **Local** or the remote computer name.
2. Right-click **ArchestrA.DASABCIP.5** and then click **Configure As Service**.
3. Click **Not a Service**.
4. Read the warning message and click **Yes**.

Archiving Configuration Sets

A configuration set includes the DAServer's global parameters; each channel and its parameters; and each device and its parameters, device groups, and device items. It lets you manage the settings of different DAServer configurations.

The DAServer contains a default configuration set named DASABCIP. You cannot delete the default configuration set.

Caution: Care should also be taken not to accidently delete the configuration set file outside of the DAServer Manager using Windows Explorer.

You can create multiple configuration sets and switch between them. Archiving, clearing, and switching configuration sets can only be done when the DAServer is deactivated.

Before you create a configuration set, verify that you have saved any changes you made to the global parameters. If you change a parameter and then immediately create a configuration set, the original parameter value is saved as part of the configuration set, not the changed value.

To save a global parameter, click the **Save** icon.

To archive a configuration set

1. In the DAServer Manager, navigate to the configuration node.
   a. Expand **DAServer Manager**, expand the node group, and then expand **Local** or the remote computer name.
   b. Expand the DAServer.
2. Click **Configuration**.
3. Right-click and click **Archive Configuration Set**.
4. In the dialog box, type the configuration set name, and click **Archive**. All the current configuration values are saved to the set.

After you archive at least one configuration set, you can select it for use.

To select a configuration set

1. In the DAServer Manager, navigate to the configuration node.
   a. Expand **DAServer Manager**, expand the node group, and then expand **Local** or the remote computer name.
   b. Expand the DAServer.
2. Click **Configuration**.
3. Right-click, point to **Use Another Configuration Set**, then click the desired name.

To change the parameter values saved in a configuration set, make sure the desired configuration set is shown, then follow this procedure.

To change the parameter values in a configuration set

1. In the DAServer Manager, navigate to the configuration node.
   a. Expand **DAServer Manager**, expand the node group, and then expand **Local** or the remote computer name.
   b. Expand the DAServer.
2. Click **Configuration**.
3. Change the parameters that you want to change.
4. Click the **Save** icon.

Clearing a configuration set returns the parameters to their default values.

To clear a configuration set

1. In the DAServer Manager, navigate to the configuration node.
   a. Expand **DAServer Manager**, expand the node group, and then expand **Local** or the remote computer name.
   b. Expand the DAServer.
2. Click **Configuration**.
3. Right-click, move the mouse over **Clear Configuration Set**, then left click.
4. Read the warning message, then click **Yes**. The parameters are set to the default values.

To delete a configuration set

1. In the DAServer Manager, navigate to the configuration node.
   a. Expand **DAServer Manager**, expand the node group, and then expand **Local** or the remote computer name.
   b. Expand the DAServer.
2. Click **Configuration**.
3. Right-click **Configuration,** point to **Delete Configuration Set** and select the configuration set to delete.
4. Read the warning message, then click **Yes**.

Activating/Deactivating the DAServer

When you activate the DAServer, it starts communicating and accepting requests from client applications. If a DAServer is configured as an automatic service, the DAServer is started and activated when the computer starts up. Also, a DAServer can be activated by the an OPC client connection request, but only out-of-proc..

To activate the DAServer

1. In the DAServer Manager, navigate to the DAServer.
   - Expand **DAServer Manager**, expand the node group, and then expand **Local** or the remote computer name.
2. Right-click **ArchestrA.DASABCIP.5** and then click **Activate Server**.

Deactivating your DAServer stops it from communicating with client applications.

A DAServer with active OPC clients does not stop until the last OPC client shuts down.

To deactivate the DAServer

1. In the DAServer Manager, navigate to the DAServer.
   - Expand **DAServer Manager**, expand the node group, and then expand **Local** or the remote computer name.
2. Right-click **ArchestrA.DASABCIP.5** and then click **Deactivate Server**.
3. Read the warning message and click **Yes**.

In-Proc/Out-of-Proc

The DAServer can run only as a stand-alone process (out-of-proc). If the CLXCTX_ALL option is chosen, out-of-proc activation for the DAServer is triggered. Explicitly starting as part of the client process (in-proc) is not supported. Activation using the CLSCTX_ACTIVATE _64_BIT_SERVER flag is also not supported.

When the DAServer is running out-of-proc, it supports requests from both DDE/SuiteLink and OPC client applications.

If the DAServer is running as a service, the icon on the DAServer node in the SMC is yellow. If the DAServer is not running as a service, the icon is white. For more information, see the *DAServer Manager User's Guide*.

Hot Configuration

If a parameter value change takes effect right away while the DAServer is running, the parameter is a hot-configurable parameter. Certain parameters in the ABCIP DAServer are hot-configurable. Incorporated in the DAServer are the following hot-configuration functionality:

- Modifying Global Configuration parameters.
- Adding, deleting, or modifying device nodes (NOT allowed on nodes having item subscription).
- Adding, deleting, or modifying device groups (including their parameters, such as Update Interval and Support Unsolicited Messages) and device items.

All other parameters are not hot-configurable.

Important: To have those non-hot-configurable changes take effect, you have to restart the DAServer or Reset the corresponding objects from within the SMC.

Note: If changes are made to server-specific parameters while the server is active, the DAServer will issue a warning message to the logger.

Archiving Configuration Sets

After you have configured your DAServer, you can archive that specific configuration. You can archive more than one configuration set, and subsequently choose different configurations for different purposes.

To archive configuration sets

1. In the DAServer Manager, right-click on the **Configuration** node in the hierarchy below your DAServer.
2. Select **Archive Configuration Set** from the shortcut menu.
3. In the **Archive Configuration Set** configuration view, provide a Configuration Set Name.
4. Click **Archive**.
   - All current configuration values are saved to the archived set.

After you have archived at least one configuration set, you can select it for use.

To use different configuration sets from the current one

1. In the DAServer Manager, right-click the **Configuration** node in the hierarchy below your DAServer.

2 Select **Use Another Configuration Set** from the shortcut menu and click on a configuration set in the sub-menu.

- All parameters in the DAServer configuration hierarchy change to the chosen configuration set.

Demo Mode

You can install the DAServer without a license. The DAServer runs without a license in Demo mode for 120 minutes. While in demo mode the DAServer checks for a license every 30 seconds. When the 120 minutes expires:

- The DAServer stops updating items.
- All non-system items have a Bad quality status.
- New items are rejected.

After the 120 minutes the DAServer checks for a license every thirty seconds. If a license is not found, the DAServer logs a warning.

You can use the SYSLicensed system item to check the status of your license. This item returns true if the proper license is found or the DAServer is in demo mode (the 120 minutes), otherwise, it returns false.

After the DAServer finds a valid license, it logs a message, stops looking for a license, and begins running normally. For more information, see the *License Utility User Guide*.

Chapter 6

Accessing the Data in Your DAServer

Client applications read and write to data items that are internal to the DAServer, as well as to the items located in the devices. Client application communication with the DAServer is done using either the OPC or DDE/SuiteLink protocols. The client application may or may not be on the same computer as the DAServer.

You do not need to create device items in the DAServer for your OPC client application.

Accessing Data Using OPC

To connect to the DAServer with an OPC client application, be aware of the following six parameters:

- **node name**: The computer name identifying the node where the DAServer is located. Only required for remote access.
- **program name**: ArchestrA.ABCIP.4
- **group name**: An OPC group defined and created by the client. The DAServer device group is used as the OPC access path.
- **device group**: A device group as defined on the DAServer. If omitted, the default device group is assumed.
- **link name**: The hierarchy of nodes names, from the channel node to the device node, separated by delimiters.

- **item name**: The specific data element. This can be the device item name or the item reference.

The combination of the link name and item name form the OPC data path for any OPC client to access DAServer data.

If the item specified is not valid for the device location, the DAServer does not accept the item. The DAServer returns bad quality and generates an error message in the logger.

Accessing Data Using DDE/SuiteLink

The DDE/SuiteLink address has four fields:

- **node name**: The computer name identifying the node where the DAServer is located. Only required for remote access.
- **application name**: DASABCIP
- **topic name**: A device group defined for the device.
- **item name**: The specific data element. This can be the device item name or the item reference.

The DDE/SuiteLink topic is the equivalent to the device group.

Chapter 7

ABCIP DAServer Features

The ABCIP DAServer supports item names for all supported controllers and also provides the following features:

- OPC Browsing
- Logix5000 Optimization Mode
- Logix5000 Write Optimization
- Data Type Determination
- Invalid Items Handling
- Logix5000 Online Tag Management
- Loading Tag Database from File
- Accessing Secured Logix5000-series Controllers
- Controller Time Stamping
- Device Redundancy

OPC Browsing

Two types of OPC browsing, namely off-line OPC browsing and on-line OPC browsing, are supported by the ABCIP DAServer.

Note: For tag items defined as array data types in an item addition request, the OPC_E_BADTYPE error is returned when an OPC client does not specify the array data type documented in the ABCIP User's Guide or the VT_EMPTY data type. The only exception is when an OPC client specifies VT_BSTR as the requested data type for an item that is defined as VT_ARRAY|VT_UI1. In this case, the DAServer accepts the item addition and returns the data as VT_BSTR.

Off-line OPC Item Browsing (Static Browsing)

The DAServer implements population of the namespace to enable OPC browsing of ControlLogix, CompactLogix, FlexLogix, GuardLogix, PLC-5, SLC500, MicroLogix, and SoftLogix processor items. Browsing can also be performed off-line using the .aacfg file for Device Items created and saved with the controller hierarchy node of the DAServer.

OPC browsing on item names is also provided to all controllers by means of importing a comma-separated-value (.csv) file, which provides symbolic names to tag names, into the .aacfg file.

On-line OPC Item Browsing (Dynamic Browsing)

The on-line OPC browsing for ControlLogix, CompactLogix, FlexLogix, GuardLogix, PLC-5, SLC500, MicroLogix, and SoftLogix processor items is implemented by the DAServer. Using the information retrieved from the processor's tag database, the ABCIP DAServer will dynamically create a configuration hierarchy that allows the DASEngine to browse into it.

When it detects that the processor's tag database has changed while browsing, the ABCIP DAServer will update the internal tag database but not the configuration hierarchy until the SYSBrowseTags system tag is poked with "1."

Note: The OPC item browsing capability is available on-line only when the ABCIP DAServer is connecting to the corresponding processor and its tag database is available for access. Otherwise, only off-line items (system items and saved device items) will be displayed.

Note: By default, dynamic OPC browsing of tags from the DAServer is disabled. In order to browse tags online from the DAServer, a "1" must be written to the SYSBrowseTags system tag associated with the chosen processor hierarchy node. Subsequent OPC item browsing operation on this particular processor should be enabled.

Logix5000 Optimization Mode

Operation of the ABCIP DAServer per device will be such that it can operate in the same multi-request service (non-optimized) mode or in optimized mode for any device. If the device and firmware support optimization, the default mode for the device will be with optimization. You will have the capability of disabling optimization even though the device and firmware may support it.

The optimization will require a tag database upload from the Logix5000-series controller (ControlLogix, CompactLogix, FlexLogix, GuardLogix, SoftLogix). The tag database contains the data types and unique references that can be used to reference the physical tags available in the controller. The tag database kept in the controller is versioned. ABCIP DAServer provides the option that can be used to periodically probe the controller for any version changes and to obtain the changes in the tag database.

- Starting with firmware version 21, DASABCIP will only support "No Optimization" and "Optimize for Read" mode.
- The "Optimize for Startup" mode will only be used to access controllers operating on firmware version 20 and lower versions.

- DASABCIP will switch to "Optimize for Read" mode from "Optimize for Startup" if firmware version 21 or higher is detected from the controller, even if the optimization setting is set to "Optimize for Startup". In this instance, DASABCIP will generate a warning line in the logger to alert that the "Optimize for Startup" mode is not supported for firmware version 21 and higher controllers.

Three selectable options are:

1 No optimization

All tags that communicate with the Logix processor will use the tag name. The tag database will be uploaded from the controller to validate the tag names. No optimization will have the fastest startup time, but will have the slowest read performance. It will create more messages for controller communication than the other two options. The length of the tag name will affect the number of messages created.

2 Optimize for read (Default)

All tags that communicate with the Logix processor will require a tag database to be available as a prerequisite. This operation also generates a memory buffer inside the controller and thus requires the longest startup time among the three options.

Despite the longer startup time, Optimize for Read provides the fastest read performance after the tag database upload operation has been completed. It will create fewer messages for controller communication.

3 Optimize for startup time

Available only for firmware versions 20 and lower, this option provides the best overall performance among all three optimization options. All tags that communicate with the Logix processor will require a tag database to be available as a prerequisite.

This option does not generate a memory buffer inside the controller and thus provides a faster startup time than the Optimize for read option. All tags communicating with the Logix processor will be using the physical tag address. It provides a faster read performance than the No Optimization option as multiple tags can be referenced in one request packet to the Logix processor. It will create a higher number of messages for controller communication than the Optimize for read option.

Note: If this option is checked, the 'Auto Synchronize Tag' option is checked automatically and cannot be unchecked.

UDT Optimization

A UDT (User-Defined Type) is a data type defined by the user in the Logix5000 processor. A UDT can group various data types, such as integers, floats, and so on, into a single structure. When this feature is enabled, the DAServer will attempt to group requests for a UDT's elements into a request for the whole structure. In fact, this feature also works for system predefined structure.

If the size of the structure exceeds 488 bytes, the DAServer will send separate requests for each structure's element. If the UDT involved is a nested structure (a UDT containing other UDTs), the DAServer will determine the optimal UDT to retrieve.

Note: Optimization and UDT Optimization features are selectable from all Logix5000-series controllers.

UDT Optimization with None Access Attribute

Starting with ControlLogix firmware version 18.x, using the Rockwell RSLogix 5000 Programming Software, UDT tags and their elements can be configured with an External Access property setting of Read/Write, ReadOnly or None. The None setting is specifically meant to define a private tag within the processor, which is not exposed to components outside of the controller, such as the ABCIP DAServer. This affects the UDT optimization capability in DASABCIP. This also affects Add On Instructions behavior, which makes extensive use of UDTs. For these reasons, UDTs with elements having the External Access property set to None is not supported. UDTs must not contain any elements with External Access property set to None when UDT optimization option is checked in DASABCIP.

Important: Changing a UDT containing an element with access rights of None to ReadOnly or Read/Write requires you to reset or deactivate and reactivate the DAServer.

Logix5000 Write Optimization

The Poke Mode parameter in the ABCIP DAServer Manager configuration screen (ArchestrA.DASABCIP.5) in the SMC controls how the DAServer treats pokes within a transaction with respect to optimization and folding.

You can select one of three modes:

- Control Mode
- Transition Mode

- Optimization Mode (ABCIP Default)

Control Mode - preserves the poke order without folding. Typically used by batch and control applications that depend on the order of the pokes and processing every item poked.

Transition Mode - preserves the poke order with minimum folding by keeping the first, second and last poke values of an item. Typically used by batch and control applications that depend on the order of pokes but not processing every item poked.

Optimization Mode (*ABCIP Default*) - does not preserve the poke order and has maximum folding by only poking the last value of an item.

When Poke Mode is set to Optimized, the DAServer will attempt to group consecutive tag writes (array elements) into a single request. Depending on the timing situation, there is no guarantee that consecutive tag writes will be grouped into a single request.

Note: For more information on all DAServer Global Parameters, see the *DAServer Manager User's Guide.*

Data Type Determination

When a client sends a read/write request to the ABCIP DAServer, the server needs to know if the tag is defined in the controller; it also needs to know the tag's data type and size. To determine this information, the ABCIP DAServer internally builds the item table (tag database) in the server-specific code before any item is created.

- This table includes information on the item's name, data type, and size.
- If an item is a structure, it also includes its members and their data types.

To build the table, the ABCIP DAServer sends a request to the controller for all the tag information defined in the controller. The controller then returns all the information needed. The table is built one time for each controller, unless a "refresh" request is received from the client. The ABCIP DAServer does not rely on the Allen-Bradley .csv files.

Important: The manual "refresh" tag database request for the Logix processor needs to be activated by your writing "true" (of type VT_BOOL) to the SysUpdateTagInfo; it is not activated by selecting the option (check box) as was implemented in the ABCIP DAServer 1.1.

Tag Database Status

To provide the status of the tag database for the Logix processor cached in the ABCIP DAServer, this version of the DAServer will implement a predefined, read-only system variable, SysTagDBStatus, of type VT_I2.

This system variable takes on any of the following values:

- 0 – No tag database
- 1 – Uploading tag database
- 2 – tag database uploaded
- 3 – tag database upload failed

The value of SysTagDBStatus can only be changed by poking to the system variable SysUpdateTagInfo. Poking a TRUE to SysUpdateTagInfo while SysTagDBStatus is 1 will not cause consecutive tag database uploads to the ABCIP DAServer.

Note: SysUpdateTagInfo and SysTagDBStatus are only available as item names associated with the Logix processor.

Regardless of the status of the tag database upload, the ABCIP DAServer periodically syncs the tag database from the controller. The Logix5000 controller has a journaling capability that keeps track of the changes in its tag database. Whenever the tag database in the Logix5000 controller is changed, a new journal and version are generated within the controller.

Tag Database Version

The ABCIP DAServer periodically checks for version changes and uploads the journal information from the controller, so that the tag database it maintains matches the corresponding database in the controller.

You can monitor changes in the Logix tag database version by subscribing to the tag database system item SYSTagDBVersion at the hierarchy of any CompactLogix, FlexLogix, or ControlLogix controller.

The ABCIP DAServer shows the new database major and minor versions presented as a number in addition to uploading the journal information from the controller.

Note: For tag items defined as array data types in an item addition request, the OPC_E_BADTYPE error is returned when an OPC client does not specify the array data type documented in the ABCIP User's Guide or the VT_EMPTY data type. The only exception is when an OPC client specifies VT_BSTR as the requested data type for an item that is defined as VT_ARRAY|VT_UI1. In this case, the DAServer accepts the item addition and returns the data as VT_BSTR.

Invalid Items Handling

Item syntax verification is based on the type of controllers associated with it. The PLC-5 and SLC500 controllers have predefined syntax on their item names. When an item is specified for these two types of controllers, its item syntax will be verified immediately. If the item syntax is incorrect, the item is rejected immediately and will not be added to the DAServer address space.

Note: An item can have a valid item syntax but invalid name depending on how the controller is programmed. In this case, the item will be added to the address space of the ABCIP DAServer with a BAD quality indication.

For the Logix5000 family of controllers, the ABCIP DAServer uses a different approach on item validation. When a Logix5000 controller item is requested to be subscribed or poked, the item is always added to the address space of the DAServer. If the item syntax does not match any of the items defined the Logix5000 controller, the item will maintain a BAD quality and is removed from any periodic scanning.

The ABCIP DAServer does periodically send messages to the Logix5000 controller for tag database update. The item that has a BAD syntax will be re-evaluated again when a new tag database has been downloaded to the Logix5000 controller. If the item is subsequently matched to an item in the new tag database, the item will automatically switch to a GOOD quality with the proper data value.

Logix5000 Online Tag Management

The ABCIP DAServer has the ability to detect online changes to the Logix5000 processor tag database and automatically update the status of these tags in your application.

Note: Tag change detection and updates are dependent on the Auto Load Tags on Activation, Auto Synchronize Tags, and Use Persisted Tags setting for these Logix5000-compatible controllers. For more information on On these tag database options, see "Loading Tag Database from File" on page 118

Adding or Removing Tags

When tags are added or removed from the Logix5000 processor, the DAServer can detect the change and update its internal tag database. If the newly added tags have already been accessed in your application, the quality of these tags will be changed to GOOD and their values updated. In the case when the tags are removed from the processor, the tags' quality will be changed to BAD.

Because the detection is done through periodic pollings of your Logix5000's status, there will be a delay between the time when tags are modified and the time when tags' information is updated in your application. The delay can be a few seconds to minutes, depending on how busy your DAServer is.

Making PLC Program Routine Changes While the Logix Controller is Online

If you import a routine containing new tags to an online PLC, the Auto Synchronize Tags option will not synchronize the new tags with the DAServer. The DAServer will reject the new tags as invalid even with the Auto Synchronize Tags option enabled until the tag database is re-reset by poking a 1 to the system item SYSUpdateTagInfo.

As a best practice, changes of PLC program routines should be done when the PLC is offline. Changing or importing the PLC program routines when the PLC is online is not supported.

Modifying Tags Through Downloaded Programs

Tag information can also be modified with an updated program. When a program is downloaded to the Logix5000 processor while data access is in progress, the DAServer can detect the change of state in your Logix5000 processor. A message will be displayed in the logger to inform you about the event and data access to the processor will be temporarily suspended.

As soon as the program downloading process has completed, the DAServer will re-upload all the tag database from the Logix5000 processor and resume your access to the processor. All tags in your application will be updated to reflect the change.

Loading Tag Database from File

The ControlLogix, GuardLogix, SoftLogix, CompactLogix and FlexLogix controller's have options to upload the Tag database from the file. Each option can improve the tag database upload time depending on your tag database management setup.

Auto Load Tags on Activation

When the DAServer is activated, it can perform an tag database upload.

If **Auto Load Tags on Activation** is selected, the DAServer will check the controller database version on startup. If it is different from the version stored in file, it will read the tags from the controller and synchronize the file. The "Tag Database from File Options Matrix" on page 120 explains the Tag database upload feature from the file.

If this option is **NOT** selected, the DAServer will not perform an upload upon activation, but will wait until an item has been advise by an client.

Auto Synchronize Tags

If **Auto Synchronize** is selected, the DAServer will periodically check the controller version number and perform an upload if a newer version is present.

Important: If optimization option **Optimize for Startup time** is selected, the **Auto Synchronize Tags** is automatically selected and unchangeable. In this situation, the DAServer needs to synchronize physical address of tags from device.

For information about importing new tags to an online PLC, see "Making PLC Program Routine Changes While the Logix Controller is Online" on page 117.

Persisted Tags

The ControlLogix, GuardLogix, SoftLogix, CompactLogix and FlexLogix controllers have an option to use Persisted Tags for uploading the tag database from the file. This feature will improve the tag database upload time.

When the DAServer is activated with the **Persisted Tags** option selected, it reads the tags from the controller and stores them into a file under the bin\CIPTagDB directory.

If the version of the tag database matches the tag database file persisted from the last run, the ABCIP DAServer will skip the tag database upload option and use the persisted file as the basis of the tag database.

If the DAServer detects the controller database version is different from the version stored in the file, it will read the tags from the controller and synchronize the file.

The subsequent restart of the DAServer will read the tag database from this file. This file will store the database major and minor version information.

Important: If secured controllers (password protected), are a part of your hierarchy, changes in the Persisted Tags functionality will occur. See the following Persisted Tag Functionality Matrix for a detailed description of each option.

Tag Database from File Options Matrix

Tag Database Options	Selected (checked)	Not Selected (Unchecked)
Auto Load Tags on Startup (Configurable parameter in the editor)	The tag database will be uploaded as soon as the DAServer is activated. The DAServer will attempt to connect to the device only one time. If the device is not connected, it will retry when the first item is subscribed.	The tag database will be uploaded as soon as the first device item is subscribed. The system item SYSUpdateTagInfo can not be used to trigger a tag database upload until the first device item is subscribed.
Auto Synchronize Tags (Configurable parameter in the editor)	The tag database in the DAServer will be synchronized periodically with the device. If the device is secured, the DAServer will not be able to automatically synchronize the tag database. Note: If the optimization option is set for **Optimize for startup time**, the value is always True. In this case the DAServer needs to synchronize the physical addresses of tags from the device.	The tag database in the DAServer will not be synchronized with the device. The system item SYSUpdateTagInfo can be used to synchronize the tag database manually.

Tag Database Options	Selected (checked)	Not Selected (Unchecked)
Use Persisted Tags	The DAServer will read the tags from the tag database file. If the file does not exist, it will then read the tags from the controller and store them into a file under bin\CIPTagDB directory. If the controller is unsecured and the database version is different from the controller version, then the DAServer will read the tags from the controller and store them into a file. The system item SYSUpdateTagInfo can be used to force the tag database upload from the device. Note: If the optimization option is **Optimize for startup time**, the physical address of the tags will also be stored in the file.	The DAServer will always upload the tag database from the device and store them in to a file.

Tag Database Options	Selected (checked)	Not Selected (Unchecked)
SYSUpdateTagInfo (System item can be accessed by any client application)	The tag database will be uploaded from the device if value **True** is poked to this item. This system item is provided for manual synchronization of the tag database. If the device is secured, use this item to synchronize tag database. Note: If **Use Persisted Tags** is enabled, the original file will be renamed to <####>_temp.aaTDB (where ### represent the serial number of the device). If the DAServer fails to upload tags from the device, it will use the renamed file to recover the database. The temporary file (<####>_temp.aaTDB) will be deleted, after the tag database is uploaded successfully.	Poking the value **False** will not affect the tag database.

Manual Tag Synchronization

This system item (SYSUpdateTagInfo) is provided for manual synchronization of tag database. If the device is secured (Password protected), use this item to synchronize tag database.

The system item SYSUpdateTagInfo can be accessed by any client application.

The tag database will be uploaded from the device if value **True** is poked to this item.

Note: If **Use Persisted Tags** is enabled, the original file will be renamed to <####>_temp.aaTDB (where ### represent the serial number of the device). If the DAServer fails to upload tags from device, it will use the renamed file to recover the database. The temporary file (<####>_temp.aaTDB) will be deleted after the tag database is uploaded successfully.

Poking the value **False** will not affect the tag database.

Accessing Secured Logix5000-series Controllers

When Logix5000 controllers are secured (Password protected), accessing the program version number will fail. When the controllers are secured, the tag database in the ABCIP DAServer may not be in-sync with the controller tag database.

If the run-time tag database synchronization has not been turned off and the controller is unsecured, the ABCIP DAServer will re-sync the tag databases at the next re-synchronization interval.

When the controller is secured an error is returned to the ABCIP DAServer indicating that the controller is secured and a message will be logged indicating that tag database re-synchronization failed because the controller is secured.

Because re-syncing is still running at the re-syncing interval, if the controller goes from secured to unsecured, the tag databases will be re-synced if necessary and a message will be logged indicating that the controller is unsecured.

The system variable SysDeviceSecurity will indicate if the controller security is On or Off.

You can turn on or off, through configuration, the tag database re-syncing, to minimize the traffic between ABCIP DAServer and the controller.

Note: The server can be started and in-sync with the controller, and the controller can be secured and un-secured with no changes, so that the server is still in-sync with the controller.

Note: Even though the controller is secured, the tag database can still be uploaded. The error returned from the controller when the controller is secured is only on the program version check.

The following Auto-synchronized and Persisted Tag Upload Functionality matrix shows when a tag database upload will occur or not occur based upon security.

Auto Synchronize Tag Functionality Matrix

Auto Synchronize Tag Configuration	Server Runtime behavior	
	For unsecured controller	For secured controller
Selected (Checked)	Tag database version in the controller will be checked periodically and the version changes will be uploaded to DASABCIP automatically.	Tag database version in the controller will be queried but no upload will be made automatically. Changes in the tag database in the controller will only be uploaded when the SYSUpdateTagInfo system tag in DASABCIP is written into.
Unselected (Unchecked)	Tag database version will not be checked. Changes in the tag database in the controller will only be uploaded if the SYSUpdateTagInfo system tag in DASABCIP is written into.	Same behavior as if the controller is unsecured.

Persisted Tag Functionality Matrix

Persisted Tags Configuration	Server Runtime behavior	
	For unsecured controller	For secured controller
Selected (Checked)	1. The DAServer will read the tags from the file. If the file does not exist then it will read the tags from controller and store them into a file under bin\CIPTagDB directory. 2. If the file database version is different from the controller version, then the DAServer will read the tags from the controller and store them into a file.	The DAServer will read the tags from the file. If the file does not exist, then it will read the tags from controller and store them into a file under the bin\CIPTagDB directory.
Unselected (Unchecked)	The DAServer will always upload the tags from the controller and store them into a file.	

Controller Time Stamping

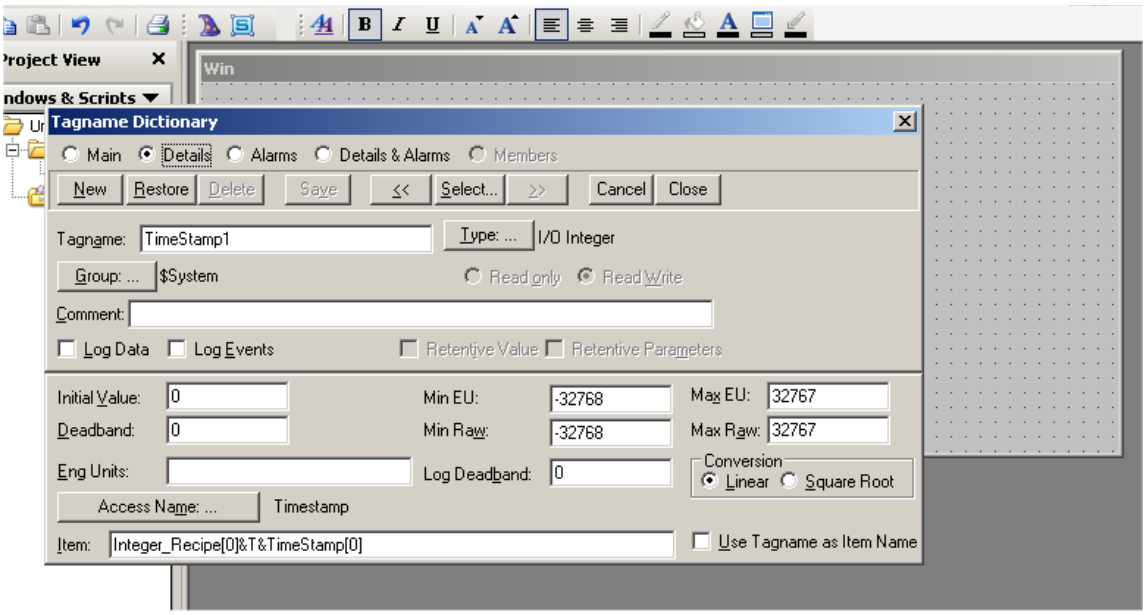
ABCIP DAServer has the capability to time stamp data changes with the controller's date and time as opposed to the PC's date and time. A new item syntax to time stamp data changes with the controller's date and time must be used.

Note: Controller Time Stamping is supported only in the Allen-Bradley Logix-family of controllers, version 16.x or later.

Important: The "TimeTag" in the controller must contain date and time as LINT type in UTC format. The logic behind the association between the specific DataTag & TimeTag pair is assumed to be user-defined in the controller program.

Specifying controller time stamping in native InTouch requires the time-stamping qualified DataTag plus both of its Date and Time Dotfield string tags.

The following sequence shows the Tagname Dictionary and the sample items.



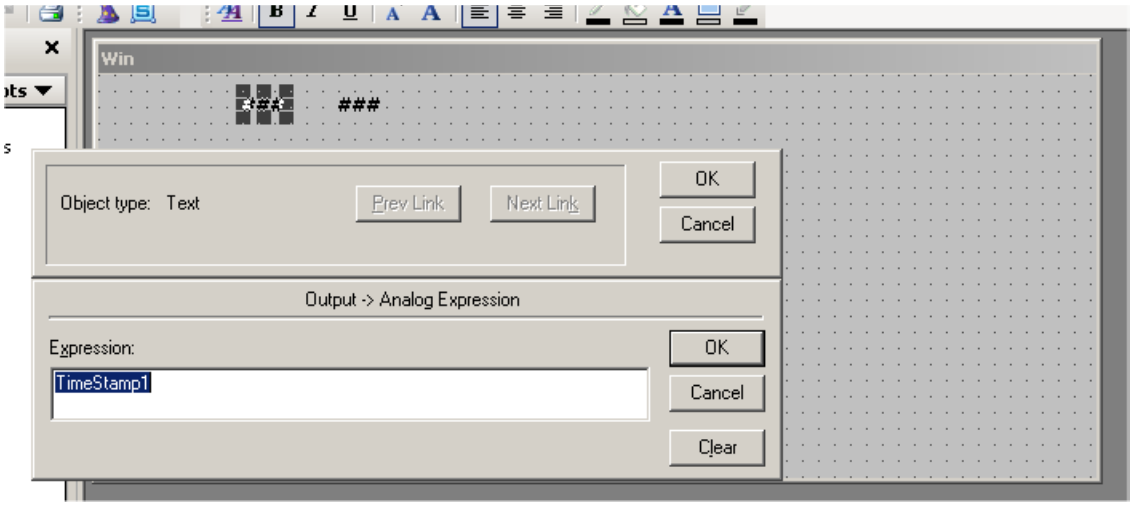
- DataTag: Can be almost any type including boolean, integer, string, and array. In the example, the Value tag is an integer for illustration purposes only.
- TimeTag: Must be string type.

When you enter an item name on the client side, you must enter an item name that is a Data tag and Time tag pair. You will use the "*&T&*" delimiter, to identify the time tag.

For example if you enter an item name such as *DataTag&T&TimeTag,* the DAServer will treat the item as two separate tags, "DataTag" and "TimeTag", and will validate each tag separately.

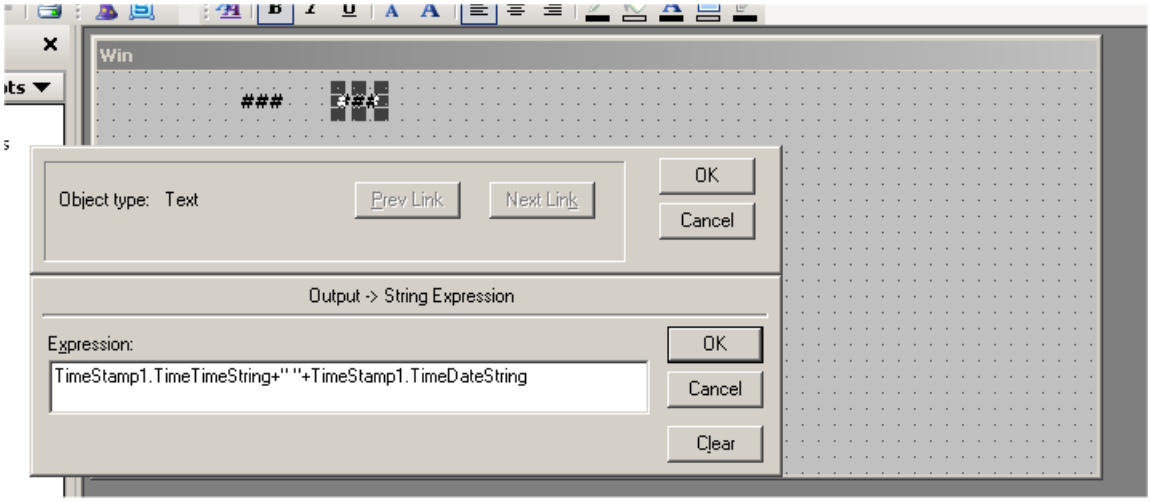
DataTag example "TimeStamp1":

```
Integer_Recipe(0)&T&TimeStamp(0)
```



TimeTag example, concatenated:

```
TimeStamp1.TimeTimeString+""+TimeStamp1.TimeDateString
```



The DAServer will read the data for the two tags from the controller.

When the pair of values is read by the server, the TimeTag value will be used to time stamp the DataTag value before sending the updates to the client.

If you enter an item name such as DataTag only, the value read from the controller is time stamped with the PC's date and time before sending the updates to the client.

Note: When advising an item using timestamping with "&T&", and communication is lost with the controller, the DAServer will timestamp the item and update its quality.

Device Redundancy

The DAServer Manager provides the ability to assign redundant devices for fail-over protection in the event of device failure. Two identical devices are expected to be configured in the DAServer Manager having identical item syntax, connected to the same DAServer.

Note: Items can not be added for updates through the Redundant Device Object if the items do not exist in both controllers.

If the Primary device fails, the DAServer will automatically switch over to the Secondary device. The Secondary device then becomes the active device with the failed Primary device reverting to the backup role. If the failed device recovers to good status, it will remain in the standby mode.

Runtime Behavior

The DAServer will start with the active device. The DAS Engine will switch to the standby device when the active device fails to communicate. The value of the SYSStatus will determine the communication failure.

Note: The value of the SYSStatus of the standby device must be TRUE in order to switch over to the standby device. Otherwise, there will not be any failover.

When SYSStatus shows a FALSE value at both active and standby devices, the DAS Engine will consider a complete communication failure and mark all the items subscribed to the redundancy device hierarchy with the current time and the appropriate OPC quality. The DAS Engine will activate the slow-poll mechanism to retry the communication to both devices until either one of the Ping Items returns to a good quality and update its SYSStatus item to TRUE.

When the DAS Engine switches to the standby device, the standby device becomes active and the originally active device becomes the standby.

When the active device becomes the standby device the Ping Item will not be deleted from that the standby device. This will ensure the standby will be able to recover the communication again.

Refer to "DAServer Redundant Device Specific System Items" on page 190 for system items specifically associated with a Redundant Device.

Note: The Ping Item must be a valid item from the controller that has not been rejected by the server for the failover to function properly.

The DAServer will log any failover activities. All other functionality such as diagnostics, enable/disable, and reset will be performed exactly same as it is performed for any other hierarchy node.

Note: Unsolicited message configuration is not supported in the Redundant Device Object itself. You can still receive unsolicited messages directly from device groups defined in the regular server hierarchy.

Chapter 8

Item Names/Reference Descriptions

The Wonderware ABCIP DAServer currently supports item names that follow the conventions described for the various Allen-Bradley ControlLogix, CompactLogix, FlexLogix, PLC-5, SLC500, and MicroLogix families of controllers.

- Logix5000 Item Naming
- PLC-5 Item Naming
- SLC500 Item Naming
- MicroLogix Item Naming
- DAServer Standard System Items
- DAServer Redundant Device Specific System Items
- Generic OPC Syntax

Logix5000 Item Naming

The Logix5000 controllers (ControlLogix, CompactLogix, FlexLogix, GuardLogix and SoftLogix) store data in tags, whose names you create. This is in contrast to the traditional Allen-Bradley PLC-5, SLC500 or MicroLogix controllers which store data in data/section files, whose names must follow the vendor-predefined naming convention.

The Logix5000 tags uses arrays instead of file numbers in addressing a set of multiple items. That is, "[]" would be accepted as a valid symbol but ":" would be rejected for the tag name. The Logix5000 item syntax is shown in the following table. The DAServer will adhere to this syntax for native mode.

Note: A tagname can be up to 40 characters in length and cannot include a file number. File numbers are not applicable to Control Logix. File numbers are valid for PLC5, SLC500 and MicroLogix only.

Reference	Syntax
Program tag	Program:<Program_Name>.<Tag_Name>
IO tag	<Location>:<slot_#>:<Data_ Type><Member_Name>.<SubMember_Name>.*[*<bit_#>*]*
Entire tag	<Tag_Name>
Member of structure tag	<Tag_Name>.<Member_Name>
Array element	<Tag_Name>[<element_X>]
Two-dimensional array element	<Tag_Name>[<element_X>,<element_Y>]
Three-dimensional array element	<Tag_Name>[<element_X>,<element_Y>,<element_Z>]
Block reads/writes of one-dimensional arrays (supported types: BOOLS, SINTS, INTS, DINTS, REALS, LONG)	<Tag_Name>[<element_X>],L<number_of_items_#>

Reference	Syntax
String tag	<String_Tag_Name>*[*.DATA*[*[<element_#>]*]]]*,SC<string_length_#>*]* <String_Tag_Name>*[*.DATA*[*[<element_#>]*]]]*,SP<string_length_#>*]* <String_Tag_Name>*[*.DATA*[*[<element_#>]*]]]*,SS<string_length_#>*]*
String tag array	<String_Tag_Name>[<element_X>]*[*.DATA*[*[<element_#>]*]]]*,SC<string_length_#>*]* <String_Tag_Name>[<element_X>]*[*.DATA*[*[<element_#>]*]]]*,SP<string_length_#>*]* <String_Tag_Name>[<element_X>]*[*.DATA*[*[<element_#>]*]]]*,SS<string_length_#>*]*
Bit within integer	<Tag_Name or Member_Name>.<bit_#>
Read-only item syntax to read controller time-stamped data	<Tag_Name>&T&<Time_Tag> <Hierarchy_Node_Path><Tag_Name>&T&[.]<TimeTag>
Note: When the data and timestamp are located in the same structure (e.g. UDT), the optional period following the &T& delimiter when entering the item name for structures reduces the need to retype the same structure name for the time tag.	**Example**: *A.B.C.D.DataTag&T&A.B.C.D.TimeTag* *A.B.C.D.DataTag&T&.TimeTag*
Note: [DT] qualifier is an option to subscribe the value (LINT) in date and time format.	**Example:** *A.B.C.D.TimeTag DT* **Note:** A space must be inserted between the <TimeTag> and DT qualifier. Using the [DT] qualifier causes the tag to become Read Only.

In the preceding table:

- *[]* italicized brackets designate element as optional.
- [] not italicized brackets denote array index.
- <> means user input (as defined in the controller program).
  - String placeholder (start with uppercase): Location, Program_Name, Tag_Name, Data_Type, Member_Name, SubMember_Name, and String_Tag_Name.
  - Numeric placeholder (all in lowercase): elemeny_#, element_X, element_Y, element_Z, string_length_#, slot_#, bit_#, and number_of_items_#.
  - <Location>identifies network location as:
    LOCAL = Local rail or chassis
    <Adapter_Name> = Name of the remote module
  - <Data_Type> is represented by a single letter as follows:
    I=input, O=output, C=configuration, and S=status.
  - All others are predefined keywords or symbols.

Examples:	
String tag array	BatchRecipe[4], BatchRecipe[4].DATA BatchRecipe[4].DATA[0],sc82 (all of them return the same data)
Two-dimensional array tag	Mixer_StepTimer_Preset[3,5]
User-defined structure tag	ProductionUnit.AssemblyLine[2].Counter[4]
Program tag	Program:MainProgram.Tank[1,2,4].Level Program:UserProgram.OperationMode
Module tag	Local:6:O.Data.31 Remote_IO:2:C.ProgValue

Note: A STRING type member is implicitly a structure in the form of StringTag.DATA and StringTag.LEN (where the DATA member is an array of 82 elements and the LEN member defines the actual length of the string). Therefore, a string member consumes two nesting levels by default.
As of ABCIP DAServer 3.0, the length field of a string will be used to determine the length of the string to be returned if the DATA member is not explicitly included in the string specification when the string is put into subscription.

Note: The "DT" qualifier returns a UTC date/time for OPC Clients requesting a "VT_DATE" binary value. For DDE and SuiteLink clients, requesting a "VT_BSTR", the date is converted to a UTC Date/Time string. The dates supported by the Date/Time string include values from 1/1/1970 12:00:00AM (GMT) to 8/30/2920 5:19:59AM (GMT).

Module-Defined Data Types

Module-defined data types are created automatically in the RSLogix5000 software after their corresponding I/O or DeviceNet modules are defined.

- Module-defined tags do not allow user modification.
- Formats are fixed by the Logix5000 controller.

User-Defined Data Types

The ABCIP DAServer supports read and write of user-defined data types. The Logix user-defined data type is a custom-made structure consisting of members that can be atomic, arrays (single dimension only), or structures themselves.

The user-defined data-type tags can be atomic or arrays up to three dimensions. The members of the structure can be any data types supported by this DAServer. If a structure contains another structure as its member, the maximum nesting supported is up to 20 levels.

Each level of members in a structure or each array dimension within a user-defined tag consumes one nesting level. The individual bits that make up a structure member do not constitute a nesting level.

The DAServer supports the optimization of user-defined data types. For information on UDT optimization, see Logix5000 Read Optimization.

Block Reads and Writes of Arrays

The ABCIP DAServer supports Block Reads and Writes of one-dimensional arrays from the supported ControlLogix, FlexLogix, and CompactLogix controllers.

The following features are **not** supported by the DAServer:

- Block Reads/Writes of strings.
- Block Reads/Writes of structures (either predefined or user-defined).

Note: The requested block size cannot exceed 486 bytes.

There are five different data types that are supported, each of which requires a different allowance on the qualifier due to the block size limitation.

There are three optimization modes supported, each with a different maximum qualifier allowance as shown in the following table: Optimize for Reads, Optimize for Startup, and No Optimization.

Note: The number in the "Ln" qualifier should not need an offset, because it is the total number counting from 1 (one).

	Qualifier Allowance (n)	
Data Type	**Optimize for Read**	**Optimize for Startup No Optimization**
Boolean (VT_BOOL)	3840	3831
SINT (VT_I1)	486	478
INT (VT_I2)	243	239
DINT (VT_I4)	114	114
Real (VT_R4)	121	119
LINT (VT_I8)	60	59

Note: Boolean array tags may allow up to 3872 items in a block if the specified range of array elements fits exactly into a contiguous block of DINT-based (4-byte) memory units. That is, Boolean array item block starting from array index zero or at every quadruple of byte (32-bits) margin.
For example, index 0, 32, 64, 96, … can exploit this feature to the maximum.

The Block Reads and Writes of Arrays feature works differently for a DDE/SuiteLink client and OPC client.

- In an OPC client, the array of data is displayed as an array of values (a series of data) separated by ";" according to their data types.
- In a DDESuiteLink client, the array of data is expressed as a string of Hex data block, of which each unit occupies the same byte size as defined by the data types.
  - The Hex value contained in each unit of the data block is equivalent to the decimal quantity stored in each individual item in the controller.
  - The data in the array block can be parsed according to the byte size of the data type.

- The Hex value can be converted to its equivalent decimal quantity for use in the application.

For example:

A DINT (double integer data type) item occupies 4 (four) bytes of data, which amounts to 8 (eight) Hex digits.
An array block of DINT items from the InTouch HMI using DDESuiteLink should be parsed into individual units of 8 (eight) Hex characters.
Then each unit of parsed data needs to be converted from Hex to its equivalent decimal value for usage.

PLC-5 Item Naming

The general format of item names for data from the PLC-5 controllers matches the naming convention used by the programming software. The following is the format:

[$] X [file] : element [.field] [/bit]

Note: The parts of the name shown in square brackets ([]) are optional.

Item Name	Description
$	Purely optional.
X	Identifies the file type. The following table summarizes the valid file types, the default file number for each type, and the fields allowed (if any).
file	File number (0 - 999 decimal). ● File 0 must be Output. ● File 1 must be Input. File 2 must be Status.
element	Element number within the file. ● For Input and Output files it is also called rack-and-group number and must be 0 - 777 octal. For all other file types, it must be 0 - 999 decimal.

Item Name	Description
.field	Valid only for Counter, Timer, ASCII String, PID, SFC Status, Block Transfer, and Control files. Refer to the following table.
/bit	Valid for all file types except ASCII String and Floating Point. ● For Input and Output files it must be 0 - 17 octal. ● For all other file types it must be 0 - 15 decimal.

Identifier	File Type	Default File #	.fields
O	Output	0	N/A
I	Input	1	N/A
S	Status	2	N/A
B	Binary	3	N/A
T	Timer	4	.PRE .ACC .EN .TT .DN
C	Counter	5	.PRE .ACC .CU .CD .DN .OV .UN
R	Control	6	.LEN .POS .EN .EU .DN .EM .ER .UL .IN .FD
N	Integer	7	N/A
F	Floating Point	8	N/A
A	ASCII	None	N/A
D	BCD	None	N/A
ST	ASCII String*	None	.LEN
PD	PID*	None	.ADRF .ADRE .BIAS .CA .CL .CT .DB .DO .DVDB .DVN .DVNA .DVP .DVPA .EN .ERR .EWD .INI .KD .KI .KP .MAXI .MAXO .MAXS .MINI .MINO .MINS .MO .OLH .OLL .OUT .PE .PV .PVDB .PVH .PVHA .PVL .PVLA .PVT .SO .SP .SPOR .SWM .TIE .UPD
SC	SFC Status*	None	.DN .ER .FS .LS .OV .PRE .SA .TIM
BT	Block Transfer* (Read-Only)	None	.EN .ST .DN .ER .CO .EW .NR .RW .TO .RLEN .DLEN .FILE .ELEM

Identifier	File Type	Default File #	.fields
MG	Message	None	.NR .TO .EN .ST .DN .ER .CO .EW .ERR .RLEN .DLEN .DATA[0] through .DATA[51]
CT	CNet Message	None	.TO .EW .CO .ER .DN .ST .EN .ERR .RLEN .DLEN .FILE .ELEM

Note: * Available only on certain PLC-5 models. Check the Processor Manual for the model being used.

Output File Items

O[n]:rg[/b]	**n** represents the file number and it is optional. If specified, it must be 0 (zero).
	r indicates the rack number (0 - 27 octal).
	g indicates the I/O group (0 - 7 octal).
	b specifies the bit (0 - 17 octal). **/b** may be omitted, if necessary, to treat the I/O group as a numeric value.

Examples:

O0:00/0

$O:177/17

O:3 4BCD (for 16-bit 7-segment display)

Input File Items

I[n]:rg[/b]	**n** represents the file number and is optional. If specified, it must be 1 (one).
	r indicates the rack number (0 - 27 octal).
	g indicates the I/O group (0 - 7 octal).
	b specifies the bit (0 - 17 octal). **/b** may be omitted, if necessary, to treat the I/O group as a numeric value.

Examples:

I1:0/0

I:177/17

I:3 4BCD (for 16-bit thumbwheel input)

Status File Items

S[n]:e[/b]	**n** represents the file number and is optional. If specified, it must be 2 (two).
	e indicates the element number in the file.
	b is optional. If specified, it indicates the bit (0 - 15 decimal).

Note: Refer to the 1785 PLC-5 Family Processor Manual (Allen-Bradley Publication 1785-6.8.2) for a complete description of the Status file information.

Examples:

$S:18	(year)
$S2:18	(year)
S2:19	(month)
S2:10/0	(battery low status bit)

Binary File Items

B[n]:e[/b] or B[n]/m	**n** represents the file number and is optional. If not specified, it is assumed to be 3 (three). If specified, the file number must be 3 - 999 decimal.
	e specifies the element (word) number within the Binary file. It must be 0 - 999 decimal.
	b specifies the bit number within the word and is optional. In the first form (where **:e** is present), the bit number must be 0 - 15 decimal.
	m specifies the bit number within the file. However, in the second form, no word numbers are specified and the bit number may be 0 - 15999.

Examples:

B:33	
B:6/4	(same bit as B/100)
B3/15999	(same bit as B:999/15)

Timer File Items

T[n]:e[.f][/b]	**n** represents the file number and is optional. If not specified, it is assumed to be 4 (four). If specified, the file number must be 3 - 999 decimal.
	e specifies the element number (three words per element) within the Timer file. It must be 0 - 999 decimal.
	f identifies one of the valid Timer fields. The valid fields for Timer Files are listed in the table. If **.f** is omitted, it is assumed to be the word containing the status bits.
	b is optional and is normally not used. All of the fields of a timer can be accessed by specifying the **.f** fields. However, it is possible to use **/b** to single out a bit in the .PRE or .ACC fields (which are words). If specified, the bit number must be 0 - 15 decimal.

Examples:

T4:0.ACC

T4:0.DN

T4:1.PRE

Counter File Items

C[n]:e[.f][/b]	**n** represents the file number and is optional. If not specified, it is assumed to be 5 (five). If specified, the file number must be 3 - 999 decimal.
	e specifies the element number (three words per element) within the Counter file. It must be 0 - 999 decimal.
	f identifies one of the valid Counter fields. The valid fields for the Counter files are listed in the table. If **.f** is omitted, it is assumed to be the word containing the status bits.
	b is optional and is normally not used. Specifying the **.f** fields can access all of the fields of a counter. However, it is possible to use **/b** to single out a bit in the .PRE or .ACC fields (which are words). If specified, the bit number must be 0 - 15 decimal.

Examples:

C5:0.ACC

C5:3.OV

C5:1.PRE

Control File Items

R[n]:e[.f][/b]	**n** represents the file number and is optional. If not specified, it is assumed to be 6 (six). If specified, the file number must be 3 - 999 decimal.
	e specifies the element number (three words per element) within the Control file. It must be 3 - 999 decimal.
	f identifies one of the valid Control fields. The valid fields for Control files are listed in the table. If **.f** is omitted, it is assumed to be the word containing the status bits.
	b is optional and is normally not used. Specifying the **.f** fields can access all of the fields of a Control file. However, it is possible to use **/b** to single out a bit in the .LEN or .POS fields (which are words). If specified, it indicates the bit (0 - 15 decimal).

Examples:

R6:0.LEN

R6:3.EM

R6:1.POS

Integer File Items

N[n]:e[/b]	**n** represents the file number and is optional. If not specified, it is assumed to be 7 (seven). If specified, the file number must be 3 - 999 decimal.
	e specifies the element number within the Integer file. It must be 0 - 999 decimal.
	b is optional. If specified, it indicates the bit (0 - 15 decimal).

Examples:

N7:0

N7:0/15

N7:3

Floating Point File Items

F[n]:e	**n** represents the file number and is optional. If not specified, it is assumed to be 8 (eight). If specified, the file number must be 3 - 999 decimal.
	e specifies the element number within the Floating Point file. It must be 0 - 999 decimal.

Examples:

F8:0

F8:3

ASCII File Items

An:e[/b] An:x-y	**n** represents the file number (NOT optional) and must be 3 - 999 decimal.
	e specifies the element number within the ASCII file. It must be 0 - 999 decimal. Each element in an ASCII file contains two ASCII characters.
	b is optional. If specified, it indicates the bit (0 - 15 decimal).
	x and **y** also specify element numbers. In this form, the item is an ASCII string occupying element **x** through element **y**. Each element contains two ASCII characters: the first character is the high-order byte and the second is the low-order, and so on.

Note: If reading only one word as a two-character string, the range must be "x-x." For example, A20:3-3.

Examples:

A20:3

A10:0/0

A9:0-19 (40-character ASCII string)

BCD File Items

Dn:e[/b]	**n** represents the file number (NOT optional) and must be 3 - 999 decimal.
	e specifies the element number within the BCD file. It must be 0 - 999 decimal. Each element in a BCD file contains a number between 0 - 9999.
	b is optional. If specified, it indicates the bit (0 - 15 decimal).

Examples:

D20:3

D10:0/3

ASCII String Section Items

STn:e[.f]	**n** represents the file number (NOT optional) and must be 3- 999 decimal.
	e specifies the element number within the String file. It must be 0 - 779 decimal. Each element in a String file contains an ASCII string with a maximum length of 82 characters.
	f identifies the following ASCII string field: .LEN. If **.f** is omitted, it is assumed to be the string.

Examples:

ST9:0

ST9:700

ST9:700.LEN

Block Transfer Section Items

BTn:e[.f][/b]	**n** represents the file number (NOT optional) and must be 3 - 999 decimal.
	e specifies the element number (three words per element) within the Block Transfer file (0 - 999 decimal).
	f identifies one of the valid Block Transfer fields. The valid fields for Block Transfer items are listed in the table. If **.f** is omitted, it is assumed to be the word containing the status bits.
	b is optional and is normally not used. Specifying the **.f** fields can access all of the fields of a Block Transfer. However, it is possible to use **/b** to single out a bit in the .FILE or .ELEM fields (which are words). If specified, the bit number must be 0 - 15 decimal.

Note: Block Transfer files are read-only.

Examples:

BT9:0.EN

BT9:3.RLEN

BT9:3.FILE

PID Section Items

PDn:e.f[/b]	**n** represents the file number (NOT optional) and must be 3 - 999 decimal.
	e specifies the element number within the PID file. It must be 0 - 398 decimal.
	f identifies one of the valid PID fields. The valid fields for PID files are listed in the table. If PID field .ADDR is needed, use .ADRE for element and .ADRF for file.
	b is optional and is normally not used. All of the fields of a PID can be accessed by specifying the **.f** fields. If specified, it indicates the bit (0 - 15 decimal).

WARNING! Access to PID files may degrade the DAServer's performance due to the extreme size of the PID element (82 words each). If accessing only a few PIDs at a time, performance will not be greatly affected. If accessing a few fields of many PIDs at the same time, it may be faster to move the needed fields to an intermediate file (Floating Point or Binary) and let the DAServer access the intermediate files.

Examples:

PD9:2.SP

PD9:3.OLH

PD9:0.INI

SFC Status Section Items

SCn:e[.f][/b]	**n** represents the file number (NOT optional) and must be 3 - 999 decimal.
	e specifies the element number within the SFC Status file. It must be 0 - 999 decimal.
	f identifies one of the valid SFC fields. The valid fields for SFC files are listed in the table.
	b is optional and is normally not used. Specifying the **.f** fields can access all of the fields of an SFC. If specified, the bit number must be 0 - 15 decimal.

Examples:

SC9:0

SC9:0.PRE

SC9:0.SA

Message Section Items

MGn:e[.f] [/b]	**n** represents the file number (NOT optional) and must be 3 - 999 decimal.
	e specifies the element number within the SFC Status file. It must be 0 - 999 decimal.
	.f identifies one of the valid MSG fields. The valid fields for MSG files are listed in the table.
	b is optional and is normally not used. Specifying the **.f** fields can access all of the fields of a .MG. However, it is possible to use **/b** to single out a bit in the word fields. If specified, the bit number must be 0 - 15 decimal.

Important: Access to MSG files may degrade the DAServer's performance, due to the extreme size of the MSG file element (56 words each). If accessing only a few MSG elements at one time, performance will not be affected greatly. However, if accessing a few fields of many MSG file elements at one time, it may be faster to move the needed fields to an intermediate file (Binary or Integer) and let the DAServer access the intermediate files.

Examples:

MG9:0.NR

MG255:1.DLEN

CNetMessage Control Block Items

CTn:e[.f][/b]	**n** represents the file number (NOT optional) and must be 3 - 999 decimal.
	e specifies the element number within the CT file. It must be 0 - 999 decimal.
	f identifies one of the valid CT fields. Valid CT fields are listed in the table.
	b is optional and normally not used. Specifying the **.f** fields can access all of the fields of a CT. If specified, the bit number must be 0 - 15 decimal.

Examples:

CT10:0

CT10:0.TO

CT10:0.ELEM

SLC500 Item Naming

The general format of item names for data from the SLC500 controllers matches the naming convention used by the programming software. The format is as follows:

[$] X [file] : element [.field] [/bit]

Note: The parts of the name shown in square brackets ([]) are optional.

Item Name	Description
$	Purely optional.
X	Identifies the file type. The following table summarizes the valid file types, the default file number for each type, and the fields allowed (if any).
file	Identifies the file number. ● File numbers must be 0 - 255 decimal. ● File 0 must be Output. ● File 1 must be Input. ● File 2 must be Status. ● All other file numbers, 9 - 255 decimal, are open to all file types.
element	Identifies the element number within a file. ● For Input and Output files it must be between 0 and 30 decimal. ● For all other file types, the element number must be 0 - 255 decimal.
.field	Valid only for Counter, Timer, and Control files. See the following table.
/bit	Valid for all file types except ASCII String and Floating Point. ● For Input and Output files it must be 0 - 17 octal ● For all other file types it must be 0 - 15 decimal.

Identifier	File Type	Default File #	.fields
O	Output*	0	N/A
I	Input*	1	N/A
S	Status	2	N/A
B	Binary	3	N/A
T	Timer	4	.PRE .ACC .EN .TT .DN
C	Counter	5	.PRE .ACC. CU .CD .DN .OV .UN .UA
R	Control	6	.LEN .POS .EN .DN .ER .UL .IN .FD
N	Integer	7	N/A
F	Floating Point*	8	N/A
A	ASCII*	None	N/A
ST	ASCII String*	None	.LEN

Note: *Available only on certain SLC500 models. Check the Processor Manual for the model being used.

Output File Items

O[n]:e[/b]	**n** represents the file number and is optional. If specified, it must be 0 (zero).
	e indicates the element number in the file (0 - 255).
	b specifies the bit (0 - 15 decimal). **/b** may be omitted, if necessary, to treat the I/O group as a numeric value.

Note: The elements in I/O modules are sequentially mapped into a memory table, and are different from the item names in the controller programming software. Refer to the Addressing SLC I/O Modules section.

Examples:

O0:0/0

$O:2/15

O:3 4BCD (for 16-bit 7-segment display)

Input File Items

I[n]:e[/b]	**n** represents the file number and is optional. If specified, it must be 1 (one).
	e indicates the element number in the file (0 - 255).
	b specifies the bit (0 - 15 decimal). **/b** may be omitted if necessary to treat the I/O group as a numeric value.

Note: The elements in I/O modules are sequentially mapped into a memory table and are different from the item names in the controller programming software. Refer to the Addressing SLC I/O Modules section.

Examples:

I1:0/0

I:2/15

I:3 4BCD (for 16-bit thumbwheel input)

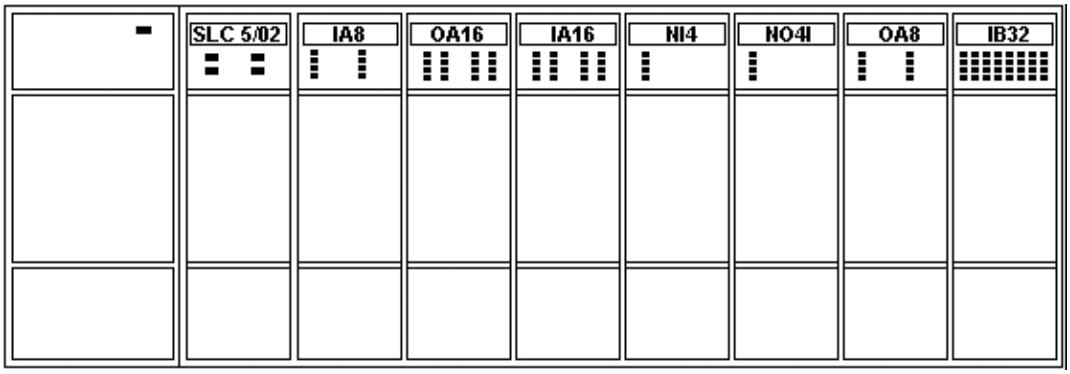
Addressing SLC I/O Modules

The elements (words) in I/O modules are mapped into a memory table. If the Analog I/O modules are being used, then the point naming will differ from the point naming in the programming software. The DAServer item name must be computed from the sum total of words used by the previous input or output blocks. The operator can use the programming software Data Monitor to look at the memory map of the I file or O file to verify your address. If the address is unsure, or if the controller configuration is likely to change, copy the points in question to the N table or B table, and access the data from there.

The naming conventions used in the Allen-Bradley programming software are not supported by the Allen-Bradley Ethernet Direct DAServer. The addressing convention is similar to that of the PLC-5 family processors. To derive the correct address for each I/O point, see the following Diagram System. Also see the following topics, Label I/O Modules with "Word Counts," Sequentially Number the Input Modules, and Sequentially Number the Output Modules, to complete addressing the SLC I/O modules.

Diagram System

Addressing of the I/O points begins by drawing a schematic of the system. The following figure is a diagram of the SLC-5/02 system.



The far left unit is the power supply.

From left to right, the modules are:

1747-L524	SLC-5/02 Module Processor
1746-IA8	8-point 120VAC input module
1746-OA16	16-point 120VAC output module
1746-IA16	16-point 120VAC input module
1746-NI4	4-point 20mA analog input module
1746-NO4I	4-point 20mA analog output module
1746-0A8	8-point 120VAC input module
1746-IB32	32-point DC input module

Label I/O Modules with "Word Counts"

The address of any point within the I/O data table space, in an SLC processor, is the sum of the words occupied by previous modules (to the left in the rack) of the same type. Therefore, to determine the correct address for any particular point in the I/O data table, the number of words each module will consume must be known. Refer to the following list:

Number of Words	Module	
0	1747-L524	SLC-5/02 Module Processor
1	1746-IA8	8-point 120VAC input module
1	1746-OA16	16-point 120VAC output module
1	1746-IA16	16-point 120VAC input module
4	1746-NI4	4-point 20mA analog input module
4	1746-NO4I	4-point 20mA analog output module
1	1746-0A8	8-point 120VAC input module
2	1746-IB32	32-point DC input module

Note: In the preceding table, the minimum number of words which can be consumed by a module is 1 (16-bits). This is due to the memory scheme of all Allen-Bradley processors.

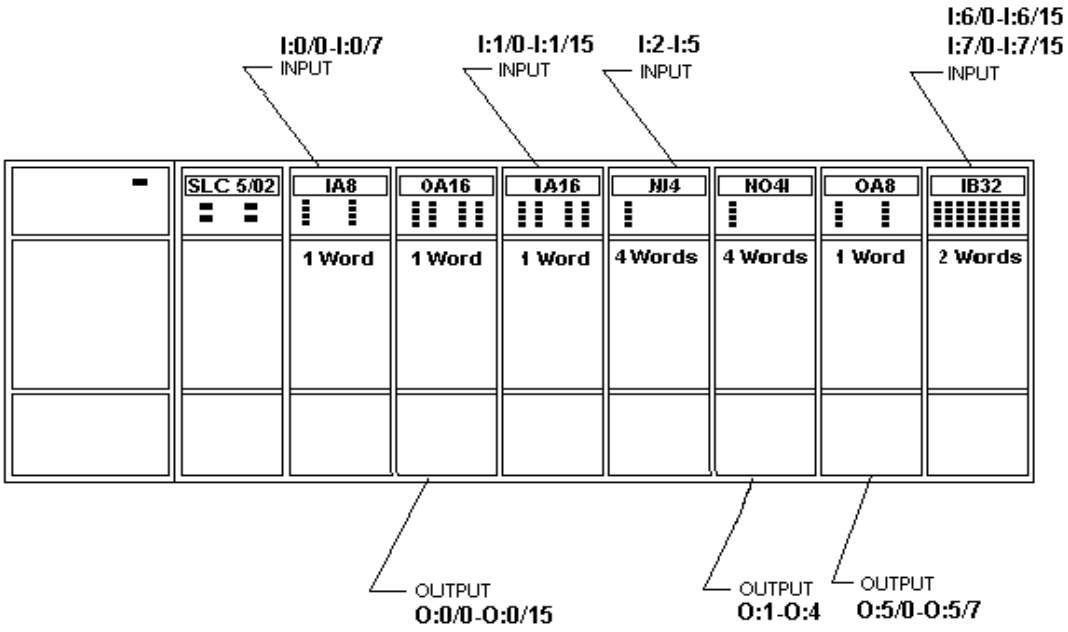
Sequentially Number the Input Modules

In the following I/O diagram, the first input module's addressing should start with "I:0." As previously noted, this module consumes one data table word. Therefore, the addressing of the next INPUT module encounter, moving from left to right, will begin with "I:1," regardless of the module's physical location.

Sequentially Number the Output Modules

In the following I/O diagram, the first output card encountered is the OA16. Although it is not in the first slot, its address will be "O:0" ("OHH, colon, ZERO"). This module consumes one data table word. Therefore, the addressing of the next OUTPUT module, moving from left to right, will begin with "O:1," regardless of the module's physical location.

I/O Diagram



Status File Items

S[n]:e[/b]	**n** represents the file number and is optional. If specified, it must be 2 (two).
	e indicates the element number in the file.
	b is optional. If specified, it indicates the bit (0 - 15 decimal).

Note: Refer to the SLC500 Family Processor Manual (Allen-Bradley Publication) for a complete description of the Status file information.

Examples:

S2:6	(major error fault)
S2:13	(math register)
S:1/5	(forces enabled)

Binary File Items

B[n]:e/b or B[n]/m	**n** represents the file number and is optional. If not specified, it is assumed to be 3 (three). If specified, the file number must be 3 or 9 - 255 decimal.
	e specifies the element (word) number within the Binary file. It must be 0 - 255 decimal.
	b specifies the bit number within the word. In the first form (where **:e** is present), the bit number must be 0 - 15 decimal.
	m also represents the bit number. However, in the second form, no word numbers are specified and the bit number may be 0 - 4095.

Examples:

B:33	
B:6/4	(same bit as B/100)
B3/4095	(same bit as B:255/15)

Timer File Items

T[n]:e[.f][/b]	**n** represents the file number and is optional. If not specified, it is assumed to be 4 (four). If specified, the file number must be 4 or 9 - 255 decimal.
	e specifies the element number (three words per element) within the Timer file. It must be 0 - 255 decimal.
	.f identifies one of the valid Timer fields. The valid fields for Timer Files are listed in the table. If **.f** is omitted, it is assumed to be the word containing the status bits.
	b is optional and is normally not used. All of the fields of a timer can be accessed by specifying the **.f** fields. However, it is possible to use **/b** to single out a bit in the .PRE or .ACC fields (which are words). The bit number must be 0 - 15 decimal.

Examples:

T4:0.ACC

T4:3.DN

T4:1.PRE

Counter File Items

C[n]:e[.f][/b]	**n** represents the file number and is optional. If not specified, it is assumed to be 5 (five). If specified, the file number must be 5 or 9 - 255 decimal.
	e specifies the element number (three words per element) within the Counter file. It must be 0 - 255 decimal.
	.f identifies one of the valid Counter fields. The valid fields for the Counter Files are listed in the table. If **.f** is omitted, it is assumed to be the word containing the status bits.
	b is optional and is normally not used. Specifying the **.f** fields can access all of the fields of a counter. However, it is possible to use **/b** to single out a bit in the .PRE or .ACC fields (which are words). The bit number must be 0 - 15 decimal.

Examples:

C5:0.ACC

C5:3.OV

C5:1.PRE

Control File Items

R[n]:e[.f][/b]	**n** represents the file number and is optional. If not specified, it is assumed to be 6 (six). If specified, the file number must be 6 or 9 - 255 decimal.
	e specifies the element number (three words per element) within the Control file. It must be 0 - 255 decimal.
	f identifies one of the valid Control fields. The valid fields for the Control files are listed in the table. If **.f** is omitted, it is assumed to be the word containing the status bits.
	b is optional and is normally not used. All of the fields of a Control file can be accessed by specifying the **.f** fields. However, it is possible to use **/b** to single out a bit in the .LEN or .POS fields (which are words). The bit number must be 0 - 15 decimal.

Examples:

R6:0.LEN

R6:3.EN

R6:1.POS

Integer File Items

N[n]:e[/b]	**n** represents the file number and is optional. If not specified, it is assumed to be 7 (seven). If specified, the file number must be 7 or 9 - 255 decimal.
	e specifies the element number within the Integer file. It must be 0 - 255 decimal.
	b is optional. If specified, it indicates the bit (0 - 15 decimal).

Examples:

N7:0

N7:0/15

N7:3

Floating Point File Items

F[n]:e	**n** represents the file number and is optional. If not specified, it is assumed to be 8 (eight). If specified, the file number must be 8 - 255 decimal.
	e specifies the element number within the Floating Point file. It must be 0 - 255 decimal.

Examples:

F8:0

F8:3

ASCII File Items

An:e[/b] An:x-y	**n** represents the file number (NOT optional) and must be 9 - 255 decimal.
	e specifies the element number within the ASCII file. It must be 0 - 255 decimal. Each element in an ASCII file contains two ASCII characters.
	b is optional. If specified, it indicates bit (0 - 15 decimal).
	x and **y** also specify element numbers. In this form, the item is an ASCII string occupying element **x** through element **y**. Each element contains two ASCII characters: the first character is the high-order byte and the second is the low-order, and so on.

Note: If reading only one word as a two-character string, the range must be "x-x." For example, A20:3-3.

Examples:

A20:3

A10:0/0

A9:0-19 (40-character ASCII string)

ASCII String Section Items

STn:e	**n** represents the file number (NOT optional) and must be 9 - 255 decimal.
	e specifies the element number within the String file. It must be 0 - 255 decimal. Each element in a String file contains an ASCII string with a maximum length of 82 characters.

Examples:

ST9:0

ST9:200

MicroLogix Item Naming

The general format of item names for data from the MicroLogix controllers matches the naming convention used by the programming software. The following is the format:

[$] X [file] : element [.field] [/bit]

Note: The parts of the name shown in square brackets ([]) are optional.

Item Name	Description
$	Purely optional.
X	Identifies the file type. The following table summarizes the valid file types, the default file number for each type, and the fields allowed (if any).
file	Identifies the file number. ● File numbers must be 0 - 999 decimal. ● File 0 (zero) must be Output. ● File 1 (one) must be Input. ● File 2 (two) must be Status. ● All other file numbers, 9 - 255 decimal, are open to all file types.
element	Identifies the element number within a file. ● For Input and Output files it must be between 0 and 777 octal. ● For all other file types it must be 0 - 999 decimal.
.field	Valid only for Counter, Timer, ASCII String, PID, SFC Status, Block Transfer, and Control files. Refer to the following table.
/bit	Valid for all file types except ASCII String and Floating Point. ● For Input and Output files it must be 0 - 17 octal. ● For all other file types it must be 0 - 15 decimal.

Identifier	File Type	Default File #	.fields
O	Output	0	N/A
I	Input	1	N/A
S	Status	2	N/A
B	Binary	3	N/A
T	Timer	4	.PRE .ACC .EN .TT .DN
C	Counter	5	.PRE .ACC .CU .CD .DN .OV .UN
R	Control	6	.LEN .POS .EN .EU .DN .EM .ER .UL .IN .FD
N	Integer	7	N/A
F	Floating Point	8	N/A
A	ASCII	None	N/A
L	Long	None	N/A
ST	ASCII String*	None	.LEN
PD	PID*	None	.TM .AM .CM .OL .RG .SC .TF .DA .DB .UL .LL .SP .PV .DN .EN .SPS .KC .TI .TD .MAXS .MINS .ZCD .CVH .CVL .LUT .SPV .CVP
MG	Message	None	.IA .RBL .LBN .RBN .CHN .NOD .MTO .NB .TFT .TFN .ELE .SEL .TO .CO .EN .RN .EW .DN .ER .ST

Note: * Available only on certain MicroLogix models. Check the Processor Manual for the model being used.

Output File Items

O[n]:e[/b]	**n** represents the file number and is optional. If specified, it must be 0 (zero).
	e indicates the element number in the file (0 - 255).
	b specifies the bit (0 - 15 decimal). **/b** may be omitted, if necessary, to treat the I/O group as a numeric value.

Note: The elements in I/O modules are sequentially mapped into a memory table, and are different from the item names in the controller programming software. MicroLogix and SLC500 adopt the same I/O addressing format. Refer to the Addressing SLC I/O Modules section for details.

Examples:

O0:0/0

$O:2/15

O:3 4BCD (for 16-bit 7-segment display)

Input File Items

I[n]:e[/b]	**n** represents the file number and is optional. If specified, it must be 1 (one).
	e indicates the element number in the file (0 - 255).
	b specifies the bit (0 - 15 decimal). **/b** may be omitted if necessary to treat the I/O group as a numeric value.

Note: The elements in I/O modules are sequentially mapped into a memory table and are different from the item names in the controller programming software. MicroLogix and SLC500 adopt the same I/O addressing format. Refer to the Addressing SLC I/O Modules section for details.

Examples:

I1:0/0

I:2/15

I:3 4BCD (for 16-bit thumbwheel input)

Status File Items

S[n]:e[/b]	**n** represents the file number and is optional. If specified, it must be 2 (two).
	e indicates the element number in the file.
	b is optional. If specified, it indicates the bit (0 - 15 decimal).

Note: Refer to the SLC500 Family Processor Manual (Allen-Bradley Publication) for a complete description of the Status file information.

Examples:

S2:6	(major error fault)
S2:13	(math register)
S:1/5	(forces enabled)

Binary File Items

B[n]:e/b or B[n]/m	**n** represents the file number and is optional. If not specified, it is assumed to be 3 (three). If specified, the file number must be 3 or 9 - 255 decimal.
	e specifies the element (word) number within the Binary file. It must be 0 - 255 decimal.
	b specifies the bit number within the word. In the first form (where **:e** is present), the bit number must be 0 - 15 decimal.
	m also represents the bit number. However, in the second form, no word numbers are specified and the bit number may be 0 - 4095.

Examples:

B:33	
B:6/4	(same bit as B/100)
B3/4095	(same bit as B:255/15)

Timer File Items

T[n]:e[.f][/b]	**n** represents the file number and is optional. If not specified, it is assumed to be 4 (four). If specified, the file number must be 4 (four) or 9 - 255 decimal.
	e specifies the element number (three words per element) within the Timer file. It must be 0 - 255 decimal.
	.f identifies one of the valid Timer fields. The valid fields for Timer Files are listed in the table. If **.f** is omitted, it is assumed to be the word containing the status bits.
	b is optional and is normally not used. All of the fields of a timer can be accessed by specifying the **.f** fields. However, it is possible to use **/b** to single out a bit in the .PRE or .ACC fields (which are words). The bit number must be 0 - 15 decimal.

Examples:

T4:0.ACC

T4:3.DN

T4:1.PRE

Counter File Items

C[n]:e[.f][/b]	**n** represents the file number and is optional. If not specified, it is assumed to be 5 (five). If specified, the file number must be 5 (five) or 9 - 255 decimal.
	e specifies the element number (three words per element) within the Counter file. It must be 0 - 255 decimal.
	.f identifies one of the valid Counter fields. The valid fields for the Counter Files are listed in the table. If **.f** is omitted, it is assumed to be the word containing the status bits.
	b is optional and is normally not used. Specifying the **.f** fields can access all of the fields of a counter. However, it is possible to use **/b** to single out a bit in the .PRE or .ACC fields (which are words). The bit number must be 0 - 15 decimal.

Examples:

C5:0.ACC

C5:3.OV

C5:1.PRE

Control File Items

R[n]:e[.f][/b]	**n** represents the file number and is optional. If not specified, it is assumed to be 6 (six). If specified, the file number must be 6 (six) or 9 - 255 decimal.
	e specifies the element number (three words per element) within the Control file. It must be 0 - 255 decimal.
	f identifies one of the valid Control fields. The valid fields for the Control files are listed in the table. If **.f** is omitted, it is assumed to be the word containing the status bits.
	b is optional and is normally not used. All of the fields of a Control file can be accessed by specifying the **.f** fields. However, it is possible to use **/b** to single out a bit in the .LEN or .POS fields (which are words). The bit number must be 0 - 15 decimal.

Examples:

R6:0.LEN

R6:3.EN

R6:1.POS

Integer File Items

N[n]:e[/b]	**n** represents the file number and is optional. If not specified, it is assumed to be 7 (seven). If specified, the file number must be 7 (seven) or 9 - 255 decimal.
	e specifies the element number within the Integer file. It must be 0 - 255 decimal.
	b is optional. If specified, it indicates the bit (0 - 15 decimal).

Examples:

N7:0

N7:0/15

N7:3

Floating Point File Items

F[n]:e	**n** represents the file number and is optional. If not specified, it is assumed to be 8 (eight). If specified, the file number must be 8 - 255 decimal.
	e specifies the element number within the Floating Point file. It must be 0 - 255 decimal.

Examples:

F8:0

F8:3

ASCII String Section Items

STn:e	**n** represents the file number (NOT optional) and must be 9 - 255 decimal.
	e specifies the element number within the String file. It must be 0 - 255 decimal. Each element in a String file contains an ASCII string with a maximum length of 82 characters.

Examples:

ST9:0

ST9:200

Long Integer Section Items

Ln:e[/b]	**n** represents the file number. If not specified, it is assumed to be 0 (zero). If specified, the file number must be 0 - 255 decimal.
	e specifies the element number within the Long Integer file. It must be 0 - 255 decimal.
	b is optional. If specified, it indicates the bit (0 - 31 decimal).

Examples:

L15:3

PID Section Items

PDn:e[.f][/b]	**n** represents the file number. If not specified, it is assumed to be 0 (zero). If specified, the file number must be 0 - 255 decimal.
	e specifies the element number within the PID file. It must be 0 - 255 decimal.
	.f identifies one of the valid PID fields. The valid fields for PID files are listed in the table.
	b is optional and is normally not used. Specifying the **.f** fields can access all of the fields of a PID. If specified, it indicates the bit (0 - 15 decimal).

WARNING! Access to PID files may degrade the DAServer's performance, due to the extreme size of the PID element (23 words each). If accessing only a few PIDs at one time, performance will not be affected greatly. However, if accessing a few fields of many PIDs at once, it may be faster to move the needed fields to an intermediate file (Floating Point or Binary) and let the DAServer access the intermediate files.

Examples:

PD:0.SP

PD9:3.LUT

PD1:0.CVP

Message Section Items

MGn:e[.f] [/b]	**n** represents the file number. If not specified, it is assumed to be 0 (zero). If specified, the file number must be 0 - 255 decimal.
	e specifies the element number within the String file. It must be 0 - 255 decimal.
	.f identifies one of the valid MSG fields. The valid fields for MSG files are listed in the table.
	b is optional and is normally not used. Specifying the **.f** fields can access all of the fields of a timer. However, it is possible to use **/b** to single out a bit in the .PRE or .ACC fields (which are words). For Timer files, the bit number must be 0 - 15 decimal.

Important: Access to MSG files may degrade the DAServer's performance, due to the extreme size of the MSG file element (56 words each). If accessing only a few MSG elements at one time, performance will not be affected greatly. However, if accessing a few fields of many MSG file elements at once, it may be faster to move the needed fields to an intermediate file (Binary or Integer) and let the DAServer access the intermediate files.

Examples:

MG9:0.NOD

MG255:1.ELE

DAServer Standard System Items

System items supply DAServer users with easy access to DAServer status and diagnostic information. They are treated just like ordinary items with respect to the client. However, in most cases these items are not directly acquired via the communications layer. System item values are usually generated through internal calculations, measurements, and the tracking of the DAS Engine.

System items, like ordinary items, are defined by the following properties:

- **Group** (client group/OPC group): The arbitrary collection of items, not correlated.
- **Hierarchical location** (link name/OPC path. The hierarchical node section of the fully qualified OPC Item ID.): The device the item is attached to.
- **Device group** (OPC access path/topic, or a Scan Group on a hierarchical branch.): A collection of items on the same physical location with the same protocol update rate.

Example:

To check the status of an external device, the reference might be:

AREA10.VESSEL1.TIC1.SYSStatus

Note: This syntax does not refer to the access path/device group. As long as the data requested is from the same external device, the value will always be the same.

Note: For DDE/SuiteLink clients, SYSStatus always comes from the leaf level of a DAServer hierarchy branch, which is the destination controller node. For OPC clients, SYSStatus can be accessed at all hierarchy levels. SYSStatus at the root level of the whole hierarchy tree is always GOOD, as it represents the quality status of the local computer itself. Hence, for practical application, OPC clients should reference SYSStatus at any hierarchy levels other than the root.

All system items follow the same naming convention:

- All system items start with SYS.
- System item name is not case-insensitive.

All system items can be accessed through subscriptions to a device group. However, while some system items return data for that device group, others are server-wide.

DAServer Global System Item

The following system item refers to specific information regarding a global condition of the DAServer.

System Item Name	Type/ Access Rights	Description	Values
SYSLicensed	Boolean/Read	Binary status indication of the existence of a valid license for the DAServer. If FALSE, this item causes the DAServer to stop updating existing tags, to refuse activation of new tags, and to reject write requests in addition to setting quality for all items to BAD. If TRUE, the DAServer functions as configured. All instances have the same value.	RANGE: TRUE, FALSE TRUE: Valid license exists. FALSE: No valid license exists.

DAServer Device-Specific System Items

The following system items refer to specific information regarding the device(s) the DAServer is connected to.

System Item Name (Type)	Type/ Access Rights	Description	Values
SYSStatus	Boolean/Read	Binary status indication of the connection state to the device (hierarchy level) the item is attached to. The device group (OPC access path/topic) does not affect the value. The status can be GOOD even if individual items have errors. For DDE/SuiteLink clients, SYSStatus always comes from the leaf level of a DAServer hierarchy branch, which is the destination controller node. For OPC clients, SYSStatus can be accessed at all hierarchy levels. SYSStatus at the root level of the whole hierarchy tree is always GOOD, as it represents the quality status of the local computer itself. Hence, for practical application, OPC clients should reference SYSStatus at any hierarchy levels other than the root.	RANGE: 0, 1 1: DAServer connection to the device is intact. 0: Error communicating with the device.

System Item Name (Type)	Type/ Access Rights	Description	Values
SYSErrorCode	LongInt/Read	Detailed error code of the communications state to the device. The device group (OPC access path/topic) does not affect the value.	>= 0: GOOD status (0 is the default state – connected. >0: is some device state like: connecting, initializing, etc. <0: Error status (value indicates the error).
SYSErrorText	String/Read	Detailed error string of the communications state of the device. The device group (OPC access path/topic) does not affect the value.	Descriptive text for the communications state corresponding to the error code.

System Item Name (Type)	Type/ Access Rights	Description	Values
SYSStoreSettings	Integer/ReadWrite	Used to make the temporary update interval changes via the SYSUpdateInterval item permanent. If the client pokes a value of 1 into this system item, the currently set update interval is written to the server's configuration file. The value of this system item clears to 0 after being set, if the configuration file write is successful. If the write fails, then the value is set to -1. If the update interval has been changed via the SYSUpdateInterval item and this item is not poked to 1, the DAServer uses the original update interval for that topic the next time it is started. Reading the item always provides 0. Read/Write values are persisted only if the user sets this system item. The values other than this persist only for the life of the DAServer.	RANGE: -1, 0, 1 -1: Error occurred during saving the configuration file. 0: Read value always if status is OK. 1: Persist settings (cleared immediately).

System Item Name (Type)	Type/ Access Rights	Description	Values
SYSEnableState	Integer/Read-only	Returns the current state of the hierarchy node.	RANGE: 0 to 3 0: Disabled 1: Enabled 2: Transitioning to Disabled 3: Transitioning to Enabled

DAServer Device-Group-Specific System Items

The following system items refer to specific information regarding device groups that have been configured in the DAServer.

System Item Name (Type)	Type/ Access Rights	Description	Values
SYSUpdateInterval	DWord/ReadWrite	Used to access the currently set update interval. It is the current update interval of the device group in milliseconds. A client can poke new values into this item. The value of zero indicates that no non-system items on that topic are updated (data for these items are not acquired from the device).	RANGE: 1…2147483647 0: Topic inactive, no items are updated. Data acquisition is stopped. >0: Expected updated interval for the set of all items in the device group.

System Item Name (Type)	Type/ Access Rights	Description	Values
SYSMaxInterval	DWord/Read	Used to access the currently measured maximum update interval in milliseconds of all items of the corresponding device group. This item is read-only. The value of the slowest item is displayed.	RANGE: 0…2147483647 0: If update interval is 0 or if the status is false. >0: Measured update interval

System Item Name (Type)	Type/ Access Rights	Description	Values
SYSWriteComplete	Integer/ReadWrite	Used to access the state of pending write activities on the corresponding device group. On device group creation (adding items to an OPC group), the value of this system item is initially 1, indicating all write activities are complete – no pokes are pending. If values are poked into any items of the device group, the value of this item changes to 0, indicating write activity is currently in progress. If the server has completed all write activities, the value of this item changes to 1 if all pokes were successful or to -1 if at least one poke has failed. If the value of this item is not zero, the client can poke 1 or -1 to it (poke a 1 to clear errors, or a -1 to test a client reaction on write errors). If the value of this item is zero, it cannot be poked.	RANGE: -1, 0, 1 1: Write complete (no writes are pending – initial state). 0: Writes are pending. -1: Writes completed with errors.

System Item Name (Type)	Type/ Access Rights	Description	Values
SYSReadComplete	Integer/ReadWrite	Used to access the state of initial reads on all items in the corresponding device group. The value is 1 if all active items in a device group have been read at least once. If at least one item in the device group is activated, this item changes to 0. It changes to 1 if all items have been read successfully, or to -1 if at least one item has a non-GOOD quality. Poking a 0 to this item resets the internal read states of all items in this device group. This resets this item to 0. If all items are read again after this poke, this item changes back to 1 or -1.	RANGE: -1, 0, 1 1: Read complete (all values have been read). 0: Not all values have been read. -1: All values have been read but some have a non-GOOD quality.
SYSItemCount	DWord/Read	Used to access the number of items in the corresponding device group. This item is read-only.	RANGE: 0…2147483647 >=0: Number of active items.
SYSActiveItemCount	DWord/Read	Used to access the number of active items in the corresponding device group. This item is read-only.	RANGE: 0…2147483647 >=0: Number of active items.

System Item Name (Type)	Type/ Access Rights	Description	Values
SYSErrorCount	DWord/Read	Used to access the number of active items that have errors (non-GOOD OPC quality) in the corresponding topic. If the communications status of a device group is BAD, all items have errors. This item is read-only.	RANGE: 0…2147483647 >=0: Number of all items (active and inactive) with errors.
SYSPollNow	Boolean/ReadWrite	Poking a 1 to this item forces all items in the corresponding device group to be read immediately (all messages in this device group become due). This is useful if you want to force to get the newest values from the device, regardless of its update interval. This also works on device groups with a zero update interval (manual protocol triggering).	RANGE: 0, 1

DAServer-Specific System Item

The following system items refers to specific information regarding the DAServer, DAServer Manager, and the controllers.

System Item Name	Type/ Access Rights	Description	Values
The following generic system items are supported for all Allen-Bradley controllers.			
SYSDeviceStatus	String/Read	Status of the processor.	RANGE: OK or faulted
SYSMode	String/Read	Current mode of the processor.	RANGE: Run, Program, Remote Run, or Remote Program
SYSPLCType	String/Read	Name of the process type.	Descriptive text for the process type.
SYSProcessorName	String/Read	Name of the program running in the processor.	Descriptive text for the corresponding processor name.
SYSRevision	String/Read	Firmware of the processor.	Descriptive text for the firmware revision.

System Item Name	Type/ Access Rights	Description	Values
SYSUpdateTagInfo	Boolean/ ReadWrite	Force update of the whole controller tag database. The DAServer returns WriteComplete for SYSUpdateTagInfo when poked. The transaction will be completed with no timeout. **Note:** The value in SYSUpdateTagInfo will return to "0" from "1" after the process is finished. **Note**: The DAServer will implement manual and automated updates of the ControlLogix tag database in the event that you add or delete items by direct controller programming. **Warning!** Updating a tag database online consumes resources. During the updating process, the DAServer may be held up from updating the client application.	RANGE: ON, OFF

System Item Name	Type/ Access Rights	Description	Values
The following tag-database-specific system items are supported for all Allen-Bradley controllers.			
SYSBrowseTags	Boolean/ ReadWrite	Indicates whether the controller tags are browsable from OPC client. If a TRUE value is written to this item, the controller tags will become browsable, provided that the tag database is ready at the time. If a FALSE value is written to this item, all controller tags will not be browsable.	RANGE: TRUE,FALSE

System Item Name	Type/ Access Rights	Description	Values
SYSTagDBStatus	String/ Read-Only	Indicates the status of the tag database as follows:	
		Uninitialized – The tag database is uninitialized, typically in a start state.	Uninitialized
		Uploading – The tag database is being uploaded from the controller.	Uploading
		Uploaded – The tag database has been completely uploaded from the controller.	Uploaded
		Error – The tag database is not uploaded because of errors encountered during the upload.	Error
SYSTagDBVersion	String/ Read-Only	Indicates the version of the Tag database. String format: MajorVersion. MinorVersion If version information cannot be acquired (for example, due to a bad PLC connection) the value initially displays "Uninitialized" as a string.	RANGE: Major version: 0–65535 (no padding) Minor version: 0–999 (no padding)

System Item Name	Type/ Access Rights	Description	Values
SYSUpdateTagInfo	Boolean/ ReadWrite	Forces update of the controller tag database upon adding the next item for advise or poking any value to an existing item (On or Off).	RANGE: TRUE,FALSE
The following Logix5000 system items are supported for ControlLogix, CompactLogix, and FlexLogix processors.			
SYSDeviceSecurity	Boolean/ Read-Only	Status of controller security	RANGE: True or False
SYSOptimization	Boolean/ Read-Only	Indicates the status of ControlLogix message optimization in handle mode as enabled with the ControlLogix Optimization check box under the Logix5000 node editor in DAServer Manager (True or False). If: FALSE - no message optimization will be used. TRUE - either the 'Optimized for read' option or the 'Optimized for startup' is being used.	RANGE: TRUE,FALSE

System Item Name	Type/ Access Rights	Description	Values
SYSUDTOptimization	Boolean/ Read-Only	Indicates the status of the ControlLogix user-defined data type optimization enabled with the **User Defined Data Type Optimization** check box under the Logix5000 node editor in the DAServer Manager (On or Off).	RANGE: TRUE,FALSE
SYSFreeMem	DWord/ Read-Only	Returns the current unused memory, in number of bytes, in the Logix processor (I/O + data table + general).	RANGE: 0…2147483647
SYSFreeMemDT	DWord/ Read-Only	Returns the unused data table memory in number of bytes (not applicable to 1756-L1).	RANGE: 0…2147483647
SYSFreeMemGM	DWordDWord/ Read-Only	Returns the total available general memory in number of bytes (applicable to 1756-L55M16 only).	RANGE: 0…2147483647
SYSFreeMemIO	DWord/ Read-Only	Returns the total available I/O memory in number of bytes.	RANGE: 0…2147483647
SYSTotalMem	DWord/ Read-Only	Returns the total memory, in number of bytes, in the Logix processor.	RANGE: 0…2147483647

System Item Name	Type/ Access Rights	Description	Values
SYSTotalMemDT	DWord/ Read-Only	Returns the total available data table memory in number of bytes (not applicable to 1756-L1).	RANGE: 0…2147483647
SYSTotalMemGM	DWord/ Read-Only	Returns the total available general memory in number of bytes (applicable to 1756-L55M16 only).	RANGE: 0…2147483647
SYSTotalMemIO	DWord/ Read-Only	Returns the total available I/O memory in n umber of bytes.	RANGE: 0…2147483647
The following system items are supported by each communications node in the ABCIP DAServer.			
SYSOpenConnections	DWord/ Read-Only	Returns the number of open CIP connections.	RANGE: 0…2147483647
SYSConnectionsInitiated	DWord/ Read-Only	Returns the number of CIP connections initiated by the server.	RANGE: 0…2147483647
SYSConnectionsRefused	DWord/ Read-Only	Returns the number of CIP connections refused by the communications module.	RANGE: 0…2147483647
SYSRequestSent	DWord/ Read-Only	Returns the number of message requests originating from the communications module.	RANGE: 0…2147483647

System Item Name	Type/ Access Rights	Description	Values
SYSReplyReceived	DWord/ Read-Only	Returns the number of reply packets received.	RANGE: 0…2147483647
SYSUnsolReceived	DWord/ Read-Only	Returns the number of unsolicited messages received by the communications module.	RANGE: 0…2147483647
SYSUnsolreplied	DWord/ Read-Only	Returns the number of replies sent in response to the unsolicited message.	RANGE: 0…2147483647
SYSRequestErrors	DWord/ Read-Only	Returns the number of errors for the requests sent.	RANGE: 0…2147483647
SYSRequestTimeout	DWord/ Read-Only	Returns the number of request timed out	RANGE: 0…2147483647
SYSResetStatistics	Boolean/Write-Only	The item SYSResetStatistics is available at root hierarchy PORT_CIP and will reset statistic counters of all child nodes.	RANGE: True,False
SYSTotalPacketSent	DWord/ Read-Only	Returns the number of data packets sent.	RANGE: 0…2147483647
SYSTotalPacketReceived	DWord/ Read-Only	Returns the number of replies received.	RANGE: 0…2147483647

System Item Name	Type/ Access Rights	Description	Values
SYSRateSent	DWord/ Read-Only	Returns the number of packets sent per second	RANGE: 0…2147483647
SYSRateReceived	DWord/ Read-Only	Returns the number of packets received per second.	RANGE: 0…2147483647
SYSItemUpdateRate	DWord/ Read-Only	Returns the number of read items received per second.	RANGE: 0…2147483647
SYSItemWriteRate	DWord/ Read-Only	Returns the number of write items sent out per second.	RANGE: 0…2147483647

DAServer Redundant Device Specific System Items

These system items are specific to the Redundant Device.

System Item Name	Type/Access Rights	Description	Values
SYSForceFailover	Boolean/ ReadWrite	This is required to achieve the failover condition to be forced by client. **Note:** By poking a value of "1" (True) into the Force Failover item, a client can conveniently switch to the secondary device.	TRUE, FALSE
SYSActiveDevice	String/Read-Only	This system item will show the current runtime active device.	Node Hierarchy Name

System Item Name	Type/Access Rights	Description	Values
SYSFailoverTime	Time/Read-Only	This system item will show the time at which the switch occurred.	Time at which the switch occurred
SYSStandbyDevice	String/Read-Only	This system item will show the current runtime standby device.	Node Hierarchy Name
SYSSecondaryDeviceStatus	Boolean/Read-Only	This system item will show the status of the secondary device. This is the status of the second device defined in the configuration and is not changed with any failover. RANGE: 0, 1	RANGE: 0, 1 (Contains the value of the system item SYSStatus)
SYSPrimaryDeviceStatus	Boolean/Read-Only	This system item will show the status of the primary device. This is the status of the first device defined in the configuration and is not changed with any failover. RANGE: 0, 1	RANGE: 0, 1 (Contains the value of the system item SYSStatus)
SYSFailoverReason	String/Read-Only	This system item will show the reason for the failover.	Descriptive text “ForceFailover” or the value of the system item SYSErrorText.

Important: The Redundant Hierarchy, including the Device Group, is not hot-configurable, and requires a Reset on the Redundant Hierarchy to effect a configuration change.

Generic OPC Syntax

A DAServer serves as a container for the OPC Groups, which provide the mechanism for containing and logically organizing OPC items. Within each OPC Group, an OPC-compliant client can register OPC items, which represent connections to data sources in the field device. In other words, all access to OPC items is maintained through the OPC Group.

The fully qualified name for an OPC item is called the Item ID (equivalent to the union of Link Name and Item Name). The syntax for specifying a unique Item ID is DAServer-dependent. In OPC data acquisition DAServers, the syntax can be as follows:

```
AREA10.VESSEL1.TIC1.PLC.N7:11
```

where each component (delimited by a hint, that is, a period in case of a DAServer) represents a branch or leaf of the field device's hierarchy.

In this example:

- AREA10.VESSEL1.TIC1 is the link name for a DAServer.
- PLC is the name of the target controller.
- N7:11 is the specific data point (Item) desired.
- An item is typically a single value such as an analog, digital, or string value.

Where Item ID describes the syntax for defining the desired data point, OPC provides for another parameter, called Access Path, that defines optional specifications for obtaining that data.

In DAServers, Access Paths are equivalent to Device Groups; it is this parameter that is used to define the update interval between the DAServer and the field device for accessing the values of data points in the controller.

Chapter 9

Troubleshooting

You can troubleshoot problems with the DAServer using the:

- Windows Task Manager
- Windows Performance and Alerts (PerfMon) application also called Performance Monitor
- DAServer Manager
- ArchestrA Log Flag Editor
- ArchestrA Log Viewer

Your client application may let you view error messages, monitor the status of requests, and allow you to request data on the status of the DAServer and connected devices. For more information, see your client application documentation.

Troubleshooting with Windows Tools

Windows has two tools that may be useful in troubleshooting performance problems.

You can find quick verification that the DAServer process is running by looking at the Windows Task Manager. It also provides information on the user, CPU, and memory usage of the processes.

If you need more information, or need to gather data while not logged in, you can use the Performance and Alerts application. For more information, see the Microsoft Management Console (MMC) help files on the Performance application. The Performance application is one of the administrative tools found in the Windows Control Panel.

Troubleshooting with the DAServer Manager

The DAServer Manager has information that may be useful in troubleshooting problems. When the DAServer is active, a diagnostic node is present below the configuration node in the console tree of the System Management Console.

Each diagnostic leaf contains information about DAServer activity. For more information, see the *DAServer Manager User Guide* or Help files.

Finding Version Information

If you contact Technical Support, you may need to supply version information.

To determine the DAServer Manager version

- In the DAServer Manager, right-click **DAServer Manager**, and then click **About DAServer Manager**. An **About** box appears showing the version and build date of the DAServer Manager.

To determine version information for DAServer components

- In the DAServer Manager, select the DAServer name in the console tree. The version information for each DAServer component is shown in the details pane.

Using the ArchestrA Log Viewer

Error messages are created by the DAServer and logged by the ArchestrA Logger. You can view these messages with the Log Viewer. The Log Viewer help files explain how to view messages and how to filter which messages are shown.

Log Flags are categories of messages. The *Log Flag Editor User Guide* contains an explanation of the categories. Using the Log Flag Editor, you can specify which log flags the DAServer creates.

Note: Generating large numbers of diagnostic messages can impact DAServer performance. You should not run in production with any more flags than those set when the DAServer is installed. To troubleshoot you can turn on more flags, but there is a performance impact. For more information, see the *Log Flag Editor User Guide*.

To open the Log Flag Editor

1. In the System Management Console, expand **Log Viewer** and then expand the log viewer group.
2. Select **Local**.
3. On the **Action** menu, click **Log Flags**.

In general, look at error and warning messages to determine if a problem exists. To determine whether the DAServer is communicating with a device, you can enable the DASSend and DASReceive log flags. From these you can determine whether or not the device is responding.

Basic Log Flags

The basic log flags for all ArchestrA components are:

- **Error**: A fatal error, the program cannot continue. By default set on by logger.
- **Warning**: The error is recoverable. A client called with a bad parameter, or the result of some operation was incorrect, but the program can continue. By default set on by logger.
- **Start-Stop**: Each main component logs a message to this category as it starts and stops.
- **Info**: General diagnostic messages.
- **Ctor-Dtor**: C++ classes of interest log messages to this category as they are constructed and destructed.
- **Entry-Exit**: Functions of interest log messages to this category as they are called and return.
- **Thread Start-Stop**: All threads should log messages to this category as they start and stop.

DAServer Log Flags

Messages created for these log flags are for DAServer common components and contain information about internal DAServer activities.

- **DACmnProtFail**: Some failure occurred in the common components while sending a message, updating an item, or otherwise moving data. Typically, this represents some unexpected behavior in the server-specific DLL.
- **DACmnProtWarn**: Some problem occurred that interfered with sending messages, updating items, or otherwise moving data. Common examples are slow poll, value limiting during type conversion, and transaction timeout messages.
- **DACmnTrace**: Normal processing of client program requests and data movement to and from the server-specific DLL are traced on this log flag. Use this in conjunction with DACmnVerbose to get the most information.
- **DACmnVerbose**: Many log flags used by the DAS common components are modified occasionally by DACmnVerbose. When DACmnVerbose is set, the logging of messages on other log flags includes more information.
- **DACmnSend**: Operations within the DAS Engine DLL that revolve around sending messages to the server-specific DLL.
- **DACmnReceive**: Events surrounding messages that are returned to the DAS Engine by the server-specific DLL, including the blocking and unblocking of hierarchies.

DAServer-Device Interface Log Flags

Messages created for the following log flags are specific to an individual DAServer and contain information about communications between the DAServer and device.

- **DASProtFail**: An error in the protocol occurred, for example, device disconnected. The program can continue, and, in fact, this category is expected during normal operation of the program. Must be set on by the generic DAS code when the DAServer starts.
- **DASProtWarn**: Something unexpected occurred in the protocol, for example, a requested item with an otherwise valid item name is not supported by this device. Must be set on by the generic DAS code when the DAServer starts.
- **DASTrace**: General diagnostic messages of a protocol-specific nature. For example, you can provide the number of items in a message for a specific protocol, then optimize based on the number.
- **DASVerbose**: Modifies all other DAS logging flags. When on, provides detailed messages.
- **DASSend**: Protocol messages sent to the device are logged to this category.
- **DASReceive**: Protocol messages received from the device are logged to this category.
- **DASStateCat1, DASStateCat2, DASStateCat3, DASStateCat4**: These are general categories for use by the server developer. As DeviceEngine-generated state machines are created by the DAServer, they can be told to log state machine messages to one of the following: DASStateCat1, DASStateCat2, DASStateCat3, or DASStateCat4. These messages indicate when a state is made the active state, when a state handler is run, when a state handler completes, and when a timeout occurs for a state machine.
- **DASStateMachine**: By default, DeviceEngine-generated state machines created by the DAServer log to this category unless specifically told to log to one of the DASStateCatN categories. In addition, general state machine messages are logged to this category. These messages indicate when a state machine is created and deleted.

ABCIP DAServer Error Messages

The following table lists the error messages produced by the DAServer that are logged to the Log Viewer.

- <Message ID> corresponds to the message ID displayed in the DAServer's Diagnostics root in the DAServer Manager.
- <Device> refers to the node name of the device.

Note: All of the error messages shown in the following table apply to the **DASProtFail** log flag.

Error Message	Explanation	Probable Cause	Solution
A PLC (IP:<IPAddress >) attempted to send us an unsolicited data packet. But the maximum number of simultaneous unsolicited data connections [MAX socket] has already been reached. Data packet ignored.	The maximum number of sockets used for unsolicited data communications has been reached. No more unsolicited data packages will be accepted.	The maximum number of sockets used for unsolicited data communications has been reached.	Decrease the number of unsolicited data to communicate to the socket.
ABCIPAccepted Socket: Initialize unable to associate an event with a handle	Unable to associate the event with a valid handle within the internal state computer.	Software internal error.	Restart the DAServer and try again.
Array index error found while formatting message for block <Block Number>	The Bit number specified in the item is out of range	Bit number is out of range	Verify and specify the correct bit number appropriate for the data type of the item tag.

Error Message	Explanation	Probable Cause	Solution
Attempt to resolve remote hostname <HostName> failed	Failed to resolve the HostName.	The HostName cannot be translated to a valid IP address internally.	Check if the HostName is configured correctly in the server.
Cannot create optimize structure for item <Item Name> message <MessageID>	The server failed to create internal structure.	It is an internal error.	
Connection to <Target Address> on port <Port Number> failed with error <Error Code>.	Error is returned from the OS while trying to establish the socket connection.	Indicated by OS returned <Error Code>.	
Connection to <Target Address> on port <Port Number> refused.	No connection can be made because the target device actively refused it.	The target address is not a ControlLogix Ethernet Module.	Check the device configured with the address.
Encountered the following error in reply message <Message ID> when reading from <Device>	Error codes are returned in the poll-message response from the device; further explanations follow.	Depends on the CIP errors returned (refer to the tables in Logix5000 Error Codes).	Check to see if there are other error messages in the logger. Check the DAServer diagnostics, if necessary.
Encountered the following error in reply message <Message ID> when writing to <Device>	Error codes are returned in the poke-message response from the device; further explanation follows.	Depends on the CIP errors returned (refer to the tables in Logix5000 Error Codes).	Check to see if there are other error messages in the logger. Check the DAServer diagnostics, if necessary.

Error Message	Explanation	Probable Cause	Solution
Encountered the following error when reading block <Block Item Name> in message <Message ID> from <Device>	Error codes are returned in the request block for Multi-Request messages. Further explanation of the errors will be listed.	Depends on the CIP errors returned (refer to the tables in Logix5000 Error Codes).	Check to see if there are other error messages in the logger. Check the DAServer diagnostics, if necessary.
Encountered the following error when reading optimized block <Internal Block Address> in message <Message ID> from <Node>	The error code is returned from the ControlLogix controller when the server tries to read the optimization structure in the controller.	It is an internal error. See the CIP Service error code for details.	
Encountered the following error when writing block <Block Item Name> in message <Message ID> from <Device>	Error codes are returned in the request block for Multi-Request messages. Further explanation of the errors will be listed.	Depends on the CIP errors returned (refer to the tables in Logix5000 Error Codes).	Check to see if there are other error messages in the logger. Check the DAServer diagnostics, if necessary.
Error encountered initializing Unsolicited Data Port. No direct (i.e.: peer-to-peer) unsolicited data will be accepted.	There is a failure to create a listening socket for the peer-to-peer unsolicited data used. As a result, no unsolicited data can be accepted.	Another application has already been listening at the same port. The network communications is having a problem. The controller is having a problem communicating.	Make sure that no other application is running and listening at the same port (such as RSLinx). Make sure the network is functioning. Make sure the controller is functioning.

Error Message	Explanation	Probable Cause	Solution
ExtSTS=<Extended Error Code>: <Description>	The error message shows the extended CIP error code and description, if there is one.	CIP-error dependent (refer to the tables in Logix5000 Error Codes).	
Failed to add block <block number> with base name <UDT base name> in <PLC address>, tag does not exist in the PLC or some of its UDT members are configured for External Access=None in the PLC	The error message shows when advising a UDT member with external access set to NONE or the PLC tag does not exist.	Either the tag does not exist in the PLC or one or more UDT members are configured for an External Access attribute of "None".	Set the UDT member external access from NONE to Read Only or Read/Write, or create the tag.
Failed to initialize Listen Socket (CIP port = <Port Number>)	Listening socket with the indicated port number is being used by another process. ABCIP DAServer cannot receive unsolicited message from controllers.	There may be third party products already listening on the same CIP port.	Shut down any third-party product (such as RSLinx) listening on the same CIP port and restart ABCIP DAServer.
Host EtherNet/IP <IP address> connect host failed, maximum number of socket <MAX socket> exceeded	The maximum number of sockets allowed was exceeded.	The maximum number of sockets allowed was exceeded.	System limit on TCP sockets on the machine hosting the DAServer is reached. Check if there are other programs on same the machine consuming a large number of sockets.

Error Message	Explanation	Probable Cause	Solution
Inconsistent message type encountered for <Device Name>	The DAServer encounters an internal error.	This is an internal error.	
Invalid item <Item Name> fields required for structure item	UDT member was not defined in the item syntax	Invalid Item syntax	Specify the member of the UDT structure in the item tag
Invalid item <Item Name> bit number not allowed or invalid	The Bit number specified in the item is out of range or invalid	The bit number is specified for a non-integer type.	Verify and specify the correct bit number appropriate for the data type of the item tag.
Invalid item <Item Name> structure not found	Specified UDT structure does not exist in the controller	Invalid item syntax	Verify if the UDT structure name is correct and exists in the controller.
Invalid item <Item Name>, invalid item format <format>	"," was used to specify the item's format. However, the format was found to be invalid for the item.	An incorrect format was used for the item.	Check the item's format.
Invalid item <Item Name>, invalid index	An invalid index is specified in the item.	An invalid index is found in the item.	Check the index specified. Only integer is accepted as an index.
Invalid item <Item Name>, member <Member Name> not found in structure	The member of the structure cannot be found.	The member is not defined in the structure.	Check the structure's definition in the device.
Invalid item <Item Name>, not defined in the processor	The item's definition cannot be found in the tag database.	The item is not defined in the Logix processor.	Check the item's definition in the device.

Error Message	Explanation	Probable Cause	Solution
Invalid item <Item Name>, offset dimension exceeded	The internal limit of the item's nesting level (20) is exceeded. In general, each "." and the index [x] increment the item's nesting level by 1 (one).	The item's nesting level exceeds the server's limit.	Reduce the item's nesting level.
Invalid item <Item Name>, retrieving whole structure not supported	The item points to a structure other than a string. The DAServer does not support this type of item.	A structure item is specified.	Retrieve an individual member of the structure instead of the whole structure.
Invalid item <Item Name>, structure not found	The structure definition for the item cannot be located.	Invalid item encountered.	Check the item's definition in the device.
Invalid item <Item Name>, syntax error	There is a syntax error in the item.	An invalid item is specified.	Check the item's syntax.
Invalid item <Item Name>, unknown suffix	The suffix specified after "," is not recognized by the DAServer.	The suffix is not supported.	Check the suffix specified for the item.
Invalid item <Item Name>, dimension mismatch	The dimension specified in the <Item Name> is different from what has been defined in the device.	The item's array dimension is different from the definition in the device.	Check the item's definition in the device.
Invalid item <Item Name>, index bracket mismatch	The bracket for the index is found mismatched. It is an item's syntax error.	An invalid item is entered.	Correct the item's syntax.

Error Message	Explanation	Probable Cause	Solution
Invalid item <Item Name>, index out of range	The index specified in the <Item Name> is outside the range defined in the device.	The item's index is too large.	Check the item's definition in the device.
Invalid item <Item Name>, invalid bit number	The bit number specified in the item is invalid.	An invalid item is specified.	The bit number specified cannot go beyond the range allowed for the item's data type. For example, 0-15 is allowed for an INT variable and 0-31 is allowed for a DINT item.
Item <ItemName> cannot be created, out of memory.	ABCIP DAServer failed to obtain memory during item creation.	The system ran out of memory.	Reduce the number of tags in the DAServer. Close other applications.
Message <Message ID> for <Device> timed out.	The DAServer does not get the message's response back from the device within the <Reply Timeout> specified.	The device is off line.	Check the device's network connection.
Mismatched bracket found while formatting message for block <Block Number>	Invalid item syntax	Missing bracket in the item tag	Specify the missing bracket in the item tag

Error Message	Explanation	Probable Cause	Solution
Received packet from [HostName] too big on port [PortNumber] ([#of bytes received] bytes)	The received packet from the controller exceeds the maximum packet size allowed for this type of protocol.	Incorrect data packet was read from the socket.	Check if the server is configured properly for the target ControlLogix controller.
Received incomplete response packet for message <Message ID> from <Device>	The response packet is incomplete or corrupted.	Bad connection or there is a DAServer problem.	Check to see if there are other error messages in the logger. Check the DAServer diagnostics, if necessary.
recv() for <HostName> on port <PortNumber> failed	Failed to read from the Window Socket specified.	Failed to read from the Winsock.	Repeat the operation by restarting the DAServer.
Register Session encountered the following error: recd packet from [HostName] too big on port [PortNumber] ([#of bytes received] bytes)	DASABCIP encountered an error while trying to establish an EtherNet/IP session with the controller. The received packet from the controller exceeds the maximum packet size allowed for this type of protocol.	Not communicating to a ControlLogix controller. Incorrect data packet was read from the socket.	Check the controller configuration. Check if the server is configured properly for the target ControlLogix controller.
Rejected %s ITEM = %s on plc %s	The item cannot be added or subscribed from the DAServer. The time tag portion in the &T& syntax is missing or invalid.	The time tag portion in the &T& syntax is missing or invalid.	If the &T& syntax is used, make sure that it consists of a valid data tag followed by the “&T&” and a valid time tag.

Error Message	Explanation	Probable Cause	Solution
Rejected <PLC Type Name> ITEM = <Item Name> on plc <PLC Node Name>. Time Tags not supported on Firmware Revision less than 16.0	The &T& time tag syntax cannot be used with controller using firmware version less 16.0	Controller firmware version is not compatible with &T& time tag requirement	Upgrade the controller firmware version to 16 or above if the &T& time tag syntax is to used.
Rejected <PLC Type Name> ITEM = <Item Name> on plc <PLC Node Name>. Invalid data type for time tag	The data type of the time tag is not correct	The data type of the time tag is not correct	The data type for time tag must be LINT
Response service code <ServiceCode> different from command <Service Code> for message <Message ID>	The service code in the message sent does not match the one in the reply.	Packet corrupted or DAServer problem encountered.	Check to see if there are other error messages in the logger. Check the DAServer diagnostics, if necessary.
Response service code <ServiceCode> not handled	Unexpected service code is encountered in the reply packet from the controller.	This is an internal error.	The service code received by the DAServer is not supported. Please verify the controller firmware version against supported version by the DAServer.
Session error <Error Code>, packet ignored	Communications errors encountered.	CIP-error dependent (refer to the tables in Logix5000 Error Codes).	

Error Message	Explanation	Probable Cause	Solution
Socket <SocketID> send() returned <ErrorNumber>, connection to be closed	This is an internal Winsock error.	This is an internal Winsock error.	
Socket <SocketID> sending packet with buffer size <BufferSize> larger than <MAX buffer size>	Failed to send on socket due to the data packet size.	This is an internal error.	
STS=<CIP Error Code>: <Description>	The error message shows the CIP error code and description.	CIP-error dependent (refer to the tables in Logix5000 Error Codes).	
Timeout waiting for an unknown event from PLC on an unsolicited data port connected to <PLC Host Name>	Timeout occurred while waiting for unsolicited data packet from a controller.	Failed to receive unsolicited data from a controller.	Make sure the controller is configured to send out unsolicited data correctly. Make sure the DAServer is functioning correctly.
Timeout waiting for data packet from PLC on an unsolicited data port connected to <PLC Host Name>	Timeout occurred while waiting for unsolicited data packet from a controller.	Failed to receive unsolicited data from a controller.	Make sure the controller is configured to send out unsolicited data correctly. Make sure the DAServer is functioning correctly.

Error Message	Explanation	Probable Cause	Solution
Timeout waiting for initialization packet from PLC on an unsolicited data port connected to <PLC Host Name>	Timeout occurred while waiting for unsolicited data header from a controller.	Failed to receive unsolicited data from a controller.	Make sure the controller is configured to send out unsolicited data correctly. Make sure the DAServer is functioning correctly.
Unsolicited socket not open for device <Device>	Unsolicited socket was not open when the DAServer tried to send a reply message to the device.	This is an internal error.	Reset the node hierarchy to restart the connection

ABCIP DAServer Error Codes

There are two server-specific error codes, shown in the following table, that augment those provided by the DAS Toolkit.

Error Code	Logger Message	Log Flag
-10001	PLC not connected	DASProtFail
-10002	PLC times out	DASProtFail

Logix5000 Error Codes

The Logix5000 processor generates error conditions. The following tables show these errors and the server-specific strings generated by the DAServer to the logger.

General Allen-Bradley Error Code (High byte = 00)	Logger Message	Log Flag
00	Success	
01	Connection failed	DASProtFail

General Allen-Bradley Error Code (High byte = 00)	Logger Message	Log Flag
02	Insufficient Connection Manager resources	DASProtFail
03	Invalid connection number	DASProtFail
04	IOI could not be deciphered. Either it was not formed correctly or the match tag does not exist	DASProtFail
05	The particular item referenced could not be found	DASProtFail
06	The amount of data requested would not fit into the response buffer. Partial data transfer has occurred.	DASProtFail
07	Connection has been lost	DASProtFail
08	Requested service not supported	DASProtFail
09	Error in data segment or invalid attribute number	DASProtFail
0A	An error has occurred trying to process one of the attributes	DASProtFail
0C	Service cannot be performed while object is in current state	DASProtFail
10	Service cannot be performed while device is in current state	DASProtFail
11	Response data too large	DASProtFail
13	Not enough command data/parameters were supplied in the command to execute the service requested	DASProtFail
14	Attribute not supported	DASProtFail
15	Too much data was received	DASProtFail
1C	An insufficient number of attributes were provided compared to the attribute count	DASProtFail
1E	Errors encountered with the items in the message	DASProtFail

General Allen-Bradley Error Code (High byte = 00)	Logger Message	Log Flag
26	The IOI word length did not match the amount of IOI which was processed	DASProtFail
None of the above codes	Unknown Status	DASProtFail

Extended Allen-Bradley Error Code (Hex)	Logger Message	Log Flag
2104	The beginning offset was beyond the end of the template.	DASProtFail
2105	You have tried to access beyond the end of the data object.	DASProtFail
2106	Data in use.	DASProtFail
2107	The abbreviated type does not match the data type of the data object.	DASProtFail
0100	Connection in Use or Duplicate Forward Open.	DASProtFail
0103	Transport Class and Trigger combination not supported.	DASProtFail
0106	Ownership Conflict.	DASProtFail
0107	Connection not found at target application.	DASProtFail
0108	Invalid Connection Type. Indicates a problem with either the Connection Type or Priority of the Connection.	DASProtFail
0109	Invalid Connection Size.	DASProtFail
0110	Device not configured	DASProtFail
0111	RPI not supported. May also indicate problem with connection time-out multiplier.	DASProtFail
0113	Connection Manager cannot support any more connections.	DASProtFail

Extended Allen-Bradley Error Code (Hex)	Logger Message	Log Flag
0114	Either the Vendor ID or the Product Code in the key segment did not match the device.	DASProtFail
0115	Product Type in the key segment did not match the device.	DASProtFail
0116	Major or Minor Revision information in the key segment did not match the device.	DASProtFail
0117	Invalid Connection Point.	DASProtFail
0118	Invalid Configuration Format.	DASProtFail
0119	Connection request fails since there is no controlling connection currently open.	DASProtFail
011A	Target Application cannot support any more connections.	DASProtFail
011B	RPI is smaller than the Production Inhibit Time.	DASProtFail
0203	Connection cannot be closed since the connection has timed out.	DASProtFail
0204	Unconnected Send timed out waiting for a response.	DASProtFail
0205	Parameter Error in Unconnected Send Service.	DASProtFail
0206	Message too large for Unconnected message service.	DASProtFail
0207	Unconnected acknowledge without reply.	DASProtFail
0301	No buffer memory available.	DASProtFail
0302	Network Bandwidth not available for data.	DASProtFail
0303	No Tag filters available.	DASProtFail
0304	Not Configured to send real-time data.	DASProtFail
0311	Port specified in Port Segment Not Available.	DASProtFail
0312	Link Address specified in Port Segment Not Available.	DASProtFail
0315	Invalid Segment Type or Segment Value in Path.	DASProtFail

Extended Allen-Bradley Error Code (Hex)	Logger Message	Log Flag
0316	Path and Connection not equal in close.	DASProtFail
0317	Either Segment not present or Encoded Value in Network Segment is invalid.	DASProtFail
0318	Link Address to Self Invalid.	DASProtFail
0319	Resources on Secondary Unavailable.	DASProtFail
031A	Connection already established.	DASProtFail
031B	Direct connection already established.	DASProtFail
031C	Miscellaneous.	DASProtFail
031D	Redundant connection mismatch.	DASProtFail
None of the above codes	Unknown Extended Status.	DASProtFail

Note: For more information about the general and extended Allen-Bradley error codes, please refer to the Allen-Bradley controller documentation.

Data Highway Plus Error Conditions

The Data Highway Plus generates error conditions. These error conditions and the server-specific strings are generated by the DAServer to the logger.

Note: All of the error messages shown in the following table apply to the **DASProtFail l**og flag.

Allen Bradley Error Code	Logger Message
DHPERR_DP_FNC (0x20000001)	Dual-port memory functionality test error
DHPERR_RAM (0x20000002)	Unknown random access memory test error
DHPERR_RAM (0x20000003)	Failure of Z80 RAM 0
DHPERR_RAM (0x20000004)	Failure of dual-port RAM
DHPERR_RAM (0x20000005)	Failure of Z80 RAM 1
DHPERR_RAM01 (0x20000006)	Failure of both Z80 RAM 0 and RAM 1
DHPERR_RAM1_DP (0x20000007)	Failure of both RAM 1 and Dual-Port RAM
DHPERR_CTC (0x20000008)	Unknown counter timer circuit test error
DHPERR_CTC_TMR (0x20000009)	Failure of CTC timer module
DHPERR_CTC_CNT (0x2000000A)	Failure of CTC counter module
DHPERR_CTC_TC (0x2000000B)	Failure of both CTC timer and counter modules
DHPERR_SIO (0x2000000C)	Unknown serial input output test error
DHPERR_SIO_INT (0x2000000D)	Failure of CIO channel: no interrupt
DHPERR_SIO_LOOP (0x2000000E)	Failure of SIO channel A: Loopback failure
DHPERR_PROT_LOAD (0x2000000F)	Protocol file download error
DHPERR_LOAD_BLK (0x20000010)	Block too large error
DHPERR_RAM_FULL (0x20000011)	Z80 RAM too full for next block
DHPERR_BD_WRITE (0x20000012)	Cannot write to adapter card memory
DHPERR_OPEN_LOADPCL (0x20000013)	Cannot open file LOADPCL.BIN

Allen Bradley Error Code	Logger Message
DHPERR_OPEN_KLPCL (0x20000014)	Cannot open file KLPCL.BIN
DHPERR_OPEN_KLST0 (0x20000015)	Cannot open file KLST0.BIN
DHPERR_OPEN_KLST1 (0x20000016)	Cannot open file KLST1.BIN
DHPERR_OPEN_KLST2 (0x20000017)	Cannot open file KLST2.BIN
DHPERR_OPEN_PROT (0x20000018)	Cannot open protocol file
TIMEOUT_ERR (0x01)	Timeout error
CANCELLED_ERR (0x02)	Cancelled error code

PLC-5 Error Messages

The error messages generated specifically for the PLC-5 family controllers are listed in the following table.

Error Message	Explanation	Possible Cause	Solution
BCD file number must be greater than 2	Incorrect format for the item. The BCD Item's File Number was smaller than 3 for PLC-5.	The BCD Item's File Number must be 3 or larger.	Only access the BCD Item with File Number equal to 3 or larger.
BINARY file number must be greater than 2	Incorrect format for this item. The Binary Item's File Number was smaller than 3 for PLC-5.	The Binary Item's File Number must be 3 or larger.	Only access the Binary Item with File Number equal to 3 or larger.
BT file number must be > 8	Incorrect format for the Item. The BT item's File Number was 8 or smaller for PLC-5.	The BT Item's File Number must be 9 or larger.	Only access the BT Item with File Number equal to 9 or larger.

Error Message	Explanation	Possible Cause	Solution
Cannot write to file BT[FileNumber]	Failed to write to a BT item for PLC-5.	For PLC-5, write operation to a BT item is not permitted.	Do not attempt to write to a BT item for PLC-5.
CONTROL file number must be greater than 2	Incorrect format for the item. The Control Item's File Number was smaller than 3 for PLC-5.	The Control Item's File Number must be 3 or larger.	Only access the Control Item with File Number equal to 3 or larger.
INTEGER file number must be greater than 2	Incorrect format for the item. The Integer Item's File Number was smaller than 3 for PLC-5.	The Integer Item's File Number must be 3 or larger.	Only access the Integer Item with File Number equal to 3 or larger.
item <ItemName> not valid, PLC does not have PID feature	PID feature is not supported for this PLC-5 configuration.	The PLC-5 configuration indicates that the PID feature is not supported.	Select the "Support PID" feature option for PLC-5, if the controller supports the feature.
PD file number must be > 8	Incorrect format for the item. The PID Item's File Number was 8 or smaller for PLC-5.	The PID Item's File Number must be 9 or larger.	Only access the PID Item with File Number equal to 9 or larger.
SC file number must be > 4	Incorrect format for the Item. The SC item's File Number was 4 or smaller for PLC-5.	The ST Item's File Number must be 5 or larger.	Only access the ST Item with File Number equal to 5 or larger.
ST file number must be > 8	Incorrect format for the Item. The ST item's File Number was 8 or smaller for PLC-5.	The ST Item's File Number must be 9 or larger.	Only access the ST Item with File Number equal to 9 or larger.
TIMER file number must be greater than 2	Incorrect format for the item. The Timer Item's File Number was smaller than 3 for PLC-5.	The Timer Item's File Number must be 3 or larger.	Only access the Timer Item with File Number equal to 3 or larger.

SLC500 and MicroLogix Error Messages

The following table lists all the SLC500 and MicroLogix controller-specific error messages.

Error Message	Explanation	Possible Cause	Solution
BINARY file number must be 3 or 9-255	Incorrect format for this item. The Binary Item's File Number was not 3 or 9-255 for SLC500 and MicroLogix.	Binary Item's File Number must be 3 or 9-255.	Only access Binary Item with valid File Number.
CONTROL file number must be greater than 6 or 9-255	Incorrect format for the item. The Control Item's File Number was not 6 or 9-255.	The Control Item's File Number must be 6 or 9-255.	Only access the Control Item with the valid File Number.
COUNTER file number must be greater than 5 or 9-255	Incorrect format for the item. The Counter Item's File Number was not 5 or 9-255.	Counter Item's File Number must be 5 or 9-255.	Only access Counter Item with valid File Number.
FLOAT file number must be 8 or 8-255	Incorrect format for the item. The Float Item's File Number was not 8 or 8-255.	The Float Item's File Number must be 8 or 8-255.	Only access the Float Item with the valid File Number.
INTEGER file number must be 7 or 9-255	Incorrect format for the item. The Integer Item's File Number was not 7 or 9-255.	The Integer Item's File Number must be 7 or 9-255.	Only access the Integer Item with the valid File Number.
TIMER file number must be 4 or 9-255	Incorrect format for the item. The Timer Item's File Number was not 4 or 9-255.	Timer Item's File Number must be 4 or 9-255.	Only access Timer Item with valid File Number.

PLC-5, SLC500, and MicroLogix Error Messages

The error messages listed in the following table pertain to the PLC-5, SLC500, and MicroLogix controllers.

Error Message	Explanation	Possible Cause	Solution
[Sub-Element] not valid for type [FileType] files.	Incorrect format for the item. The Sub-Element is not valid for this File Type.	Wrong Item format with a wrong Sub-Element type.	Only access the valid item format with the correct Sub-Element type.
ASCII file number must be greater than 2	Incorrect format for the item. The ASCII Item's File Number was smaller than 3.	The ASCII Item's File Number must be 3 or larger.	Only access the ASCII Item with a File Number equaling to 3 or larger.
BINARY file, bit>15 and element>0	Incorrect format for the item. The Binary Item contained an element number, but its bit number was larger than 15.	The valid format for a PLC-5 Binary Item is: B[FileNumber]:[Element]/[Bit], where Bit is from 0 to 15. In this case, the Bit field was larger than 15.	Only access the Binary Item with the valid range.
File numbers must be between 0 and 999	Incorrect format for the Item. The Item's File Number was out of range.	A bad item File Number was used.	Use a valid range for the Item's File Number.
FLOATING POINT file cannot have bit number	Incorrect format for the item. The Floating Point Item contained a bit number field.	The Floating Point Item must not contain a bit number field.	Only access the Floating Point Item without a bit number field.
FLOATING POINT file number must be greater than 2	Incorrect format for the item. The Floating Point Item's File Number was smaller than 3.	The Floating Point Item's File Number must be 3 or larger.	Only access the Floating Point Item with a File Number that equals to 3 or larger.

Error Message	Explanation	Possible Cause	Solution
INPUT file number must be 1	Incorrect format for the item. The Input Item's File Number was not 1.	The Input Item's File Number must be 1.	Only access the Input Item with a File Number that equals to 1.
OUTPUT file number must be 0	Incorrect format for the item. The Output Item's File Number was not 0.	The Output Item's File Number must be 0.	Only access the Output Item with a File Number equaling to 0.
STATUS file number must be 2	Incorrect format for the item. The Status Item's File Number was not 2.	The Status Item's File Number must be 2.	Only access the Status Item with a File Number equaling to 2.
Unsupported file type [File Type]	Incorrect format for the item. An invalid Item Type was used.	There was no such Item Type name.	Use the valid Item Type.
[Sub-Element not valid for type [FileType] section.	Incorrect format for the item. The Sub-Element is not valid for this section.	Wrong Item format.	Use only the valid item format.
Attempt to write read only item in file [FileNumber] element [Element#] subelement [Sub-Element#] ignored	Write operation failed due to an attempt to write to a read-only item.	An attempt to write to a read-only item caused the failure.	Do not attempt a write operation to a read-only item.

Chapter 10

Reference

DAServer Architecture

This DAServer is a collection of components that work in concert to provide communications access with hardware field devices. These components include:

- **DAServer Manager**: This is the Microsoft Management Console (MMC) snap-in, which is part of the ArchestrA System Management Console suite of utilities, supplied with the DAServer. It provides the necessary user-interface for configuration, activation, and diagnostics.
- **Client Plug-ins**: These are the components that are added to a DAServer to enable communications with clients. Examples are: OPC, DDE/Suitelink, and so on.
- **DAS Engine**: This is the library that contains all the common logic to drive data access.
- **Device Protocol**: This is the custom code provided by this DAServer to define the communications with a particular device.

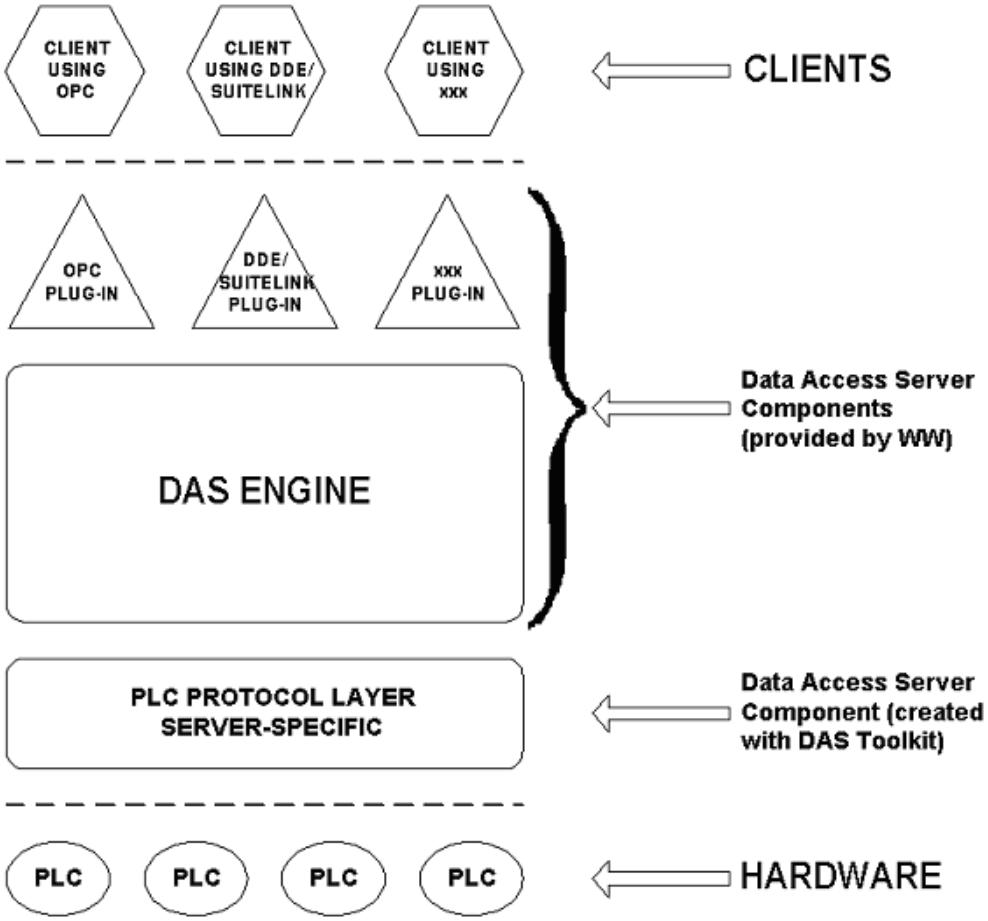
Note: NetDDE protocol is not supported by DAServers.

DAServers

A DAServer is comprised of three physical parts (see the following figure). They are the following:

- **Plug-in Component(s)**: Responsible for communicating with clients.

- **DAS Engine**: This common component is used by all DAServers.
- **PLC Protocol Layer**, DAServer-specific: This component is responsible for communicating with the hardware.



DAServer Architecture

Each physical part of a DAServer is comprised of a set of .exe and/or .dll modules. Wonderware provides the Plug-ins and the DAS Engine. The DAS Toolkit user creates the PLC Protocol Layer (DAServer-specific) modules. All three sets of modules are required for a fully functioning DAServer.

Plug-ins

Plug-ins provide a protocol-translation function for device integration clients. Typical Plug-ins communicate in DDE, SuiteLink, or OPC protocol, and serve as interfaces between their clients and the DAS Engine.

Note: Items of an array are not supported in the DDESuiteLink plug-in. These arrays are converted to HEXASCII strings, which provide legacy behavior for DAServers that support this in the DAServer-specific code.

Note: For tag items defined as array data types in an item addition request, the OPC_E_BADTYPE error is returned when an OPC client does not specify the array data type documented in the ABCIP User's Guide or the VT_EMPTY data type. The only exception is when an OPC client specifies VT_BSTR as the requested data type for an item that is defined as VT_ARRAY|VT_UI1. In this case, the DAServer accepts the item addition and returns the data as VT_BSTR.

DAS Engine

The DAS Engine is a middleware component that exposes two sets of unique interfaces, one for communicating with the Plug-ins and the other one for communicating with the PLC Protocol Layer components.

PLC Protocol Layer

The PLC Protocol Layer provides a protocol-translation function for specific hardware, such as ModBus; and it serves as an interface between the DAS Engine and the hardware.

Component Environments

Stand-alone DAServers have the following characteristics:

- The DAS Engine is dynamically linked to the other DAServer components. In other words, a new DAS Engine (feature enhancement or bug fix) would not require re-linking to the other components. When deployed to the system, the new DAS Engine would attach to all existing DAServer components.
- Newly deployed Plug-ins (feature enhancements or bug fixes) do not require re-linking. Even new Plug-ins (for example, OPC Alarm & Events) would not require any development changes to the other components, and therefore no re-linking in a customer-installed base. In fact, it is feasible to implement new functionality in a Plug-in to enhance the DAServer without any involvement of the code of the other components.
- DAServers can be configured in one stand-alone configuration utility (DAServer Manager), and is capable of displaying specific configuration views for all DAServers. This utility allows the browsing and editing of DAServers on different nodes.
- The DAServer Manager diagnostic tool displays generic diagnostic objects common to all DAServers, in addition to the DAServer-specific/DAServer-developer-defined diagnostic data.

The DAServer data configuration format is XML. Any XML-enabled program (for example, XML Editor) can read this format.

Chapter 11

Tested Logix5000 Firmware

The following table lists the Allen-Bradley Logix5000 firmware that have been tested with ABCIP DAServer Version 5.0.[1]

Allen-Bradley Module	Description	Firmware Revision
1756-L1	ControlLogix5550 Controller	v.13.034
1756-L55	ControlLogix5555 Controller	v.16.22
1756-L63	ControlLogix5563 Controller	v.20.013
1756-L64	ControlLogix5564 Controller	v.20.013
1756-L6xS LSP	GuardLogix Safety Controller	v.20.012
1756-L73	ControlLogix5573 Controller	v.21.011
1756-ENET/A	ControlLogix Ethernet Communication Interface Module (Series A)	v.1.018
1756-ENET/B	ControlLogix EtherNet Communication Interface Module (Series B)	v.2.007
1756-ENBT/A	ControlLogix EtherNet/IP 10/100Mbps Bridge Module (Twisted-pair Media)	v.6.006
1756-EWEB	ControlLogix EtherNet/IP Web Server Module	v.4.016
1756-EN2T/A/C	ControlLogix High Capacity EtherNet/IP Bridge Module	v.5.008
1756-EN2TR	ControlLogix 2-Port EtherNet/IP Bridge Module	v.5.008
1756-DHRIO/A/B	ControlLogix DH+/RIO Communication Interface Module (Series A or B)	v.2.021

Allen-Bradley Module	Description	Firmware Revision
1756-DHRIO/C	ControlLogix DH+/RIO Communication Interface Module (Series C)	v.5.004
1756-DHRIO/D	ControlLogix DH+/RIO Communication Interface Module (Series D)	v.6.003
1756-CNB(R)/A/B	ControlLogix ControlNet Communication Bridge Module (Series A or B)	v.2.030
1756-CNB(R)/D	ControlLogix ControlNet Communication Bridge Module (Series D)	v.7.016
1756-CNB(R)/E	ControlLogix ControlNet Communication Bridge Module (Series E)	v.11.005
1756-CN2(R)/B	ControlLogix ControlNet 2X Capacity Bridge Module	v.20.020
1756-DNB(R)/A/B	ControlLogix ControlNet Redundancy Bridge Module	v.7.003
1768-L43	CompactLogix 5343 Controller	v.20.013
1768-ENBT/A	CompactLogix EtherNet/IP Comm. Interface Module	v.4.004
1768-M04SE	CompactLogix SERCOS Interface Module	v.20.007
1769-L20	CompactLogix 5320 Controller	v.13.019
1769-L27ERM-QBFC1B	CompactLogix 5370 L2 Series Controller	21.011
1769-L30	CompactLogix5330 Controller	v.13.019
1769-L31	CompactLogix 5331 Controller	v.20.013
1769-L32C	CompactLogix 5332 ControlNet Controller	v.20.013
1769-L32E	CompactLogix 5332 Controller with built-in EtherNet/IP Port	v.20.013
1769-L35E	CompactLogix 5335 Controller with built-in EtherNet/IP Port	v.20.013
1769-L35CR	CompactLogix 5335 ControlNet Redundant Controller	v.20.013
1769-L36ERM	CompactLogix 5370 L3 Series Controller	21.011
1769-ADN/B	Compact I/O DeviceNet Adapter (Series B)	v.2.005
1769-SDN/B	DeviceNet Scanner for CompactLogix and MicroLogix	v.4.004

Allen-Bradley Module	Description	Firmware Revision
1783-ETAP	EtherNet/IP Tap Module (Embedded Switch Technology)	2.002
1794-L33	FlexLogix 5433 Controller	v.13.033
1794-L34	FlexLogix5434 Controller	v.16.022
1788-ENBT	FlexLogix Ethernet Communication Daughter Card	v.3.001
1788-CNC	FlexLogix ControlNet Communication Daughter Card	v.1.008
1788-DNBO	FlexLogix DeviceNet Communication Daughter Card	v.2.002
1789-L30	SoftLogix 5800 (5-Slot Virtual Chassis) Software	v.21.00
ControlLogix Standard Redundancy System SRM[2]		
1756-L55	ControlLogix 5555 Processor	v.16.022
1756-L6x	ControlLogix 556x Processor (L61/L62/L63)	v.20.013
1757-SRM	ControlLogix System Redundancy Module	v.5.004
1756-CNB(R)/D	ControlLogix ControlNet Bridge Module (Series D)	v.7.016
1756-CNB(R)/E	ControlLogix ControlNet Bridge Module (Series E)	v.11.005
1756-ENBT/A	ControlLogix EtherNet/IP Bridge Module (Series A)	v.6.006
ControlLogix Enhanced Redundancy System RM[2]		
1756-L6x	ControlLogix 556x Processor (L61/L62/L63)	v.20.054
1756-RM/B	ControlLogix Redundancy Module (Series B)	v.3.003
1756-CN2R/B	ControlLogix ControlNet Bridge Module (Series B)	v.20.020
1756-EN2T/C	ControlLogix EtherNet/IP Bridge Module (Series C)	v.5.028

[1] Upon the release of major revisions of the Allen-Bradley firmware from Rockwell Automation, tests will be performed on the ABCIP DAServer to ensure proper compatibility. Any updates to the ABCIP DAServer resulting from this testing will be posted for download at the Wonderware Technical Support Web site.

[2] ControlLogix Redundancy Bundle firmware version is v.20.54.

Index

O

P

R

S

T

U

V