Proficy HMI/SCADA - CIMPLICITY 2022

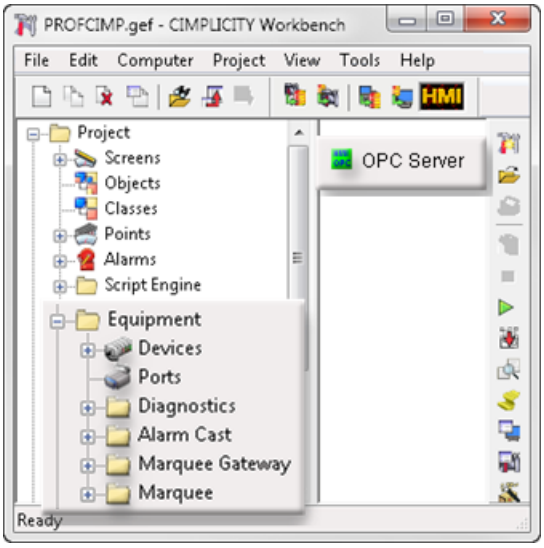
Device Communications

We want to hear from you. If you have any comments, questions, or suggestions about our documentation, send them to the following email address:

doc@ge.com

Chapter 1. About Communications Equipment

CIMPLICITY device communications tools and OPC Servers.



- CIMPLICITY native device communications.
- Quick device setup.
- Devices.
- Ports.
- Device QuickView.
- Device communications diagnostics.
- Alarm Cast. (Gateway).
- Marquee Gateway.
- OPC Servers.

Chapter 2. About Device Diagnostic Utilities

Device communications diagnostic utilities include:

- Allen-Bradley DF1 Setup utility.
- Mitsubishi A-Series serial communication diagnostics.
- OMRON Host Link communications diagnostics.
- OMRON TCP/IP communications.
- Series 90 Triplex installation verification procedures.

Chapter 3. Ports

About Ports

A port is a communication "socket" that connects one or more factory devices such as PLC's to the computer.

Use this function to configure ports and specify their characteristics. Configuration requirements for ports vary depending on the type of port and communications protocol being used.

Step 1 *(page 47)*	Enable a protocol in a CIMPLICITY project.
Step 2 *(page 48)*	Open a Port Properties dialog box.
Step 3 *(page 51)*	Configure port general properties.
Step 4 *(page 54)*	Configure port-specific properties.

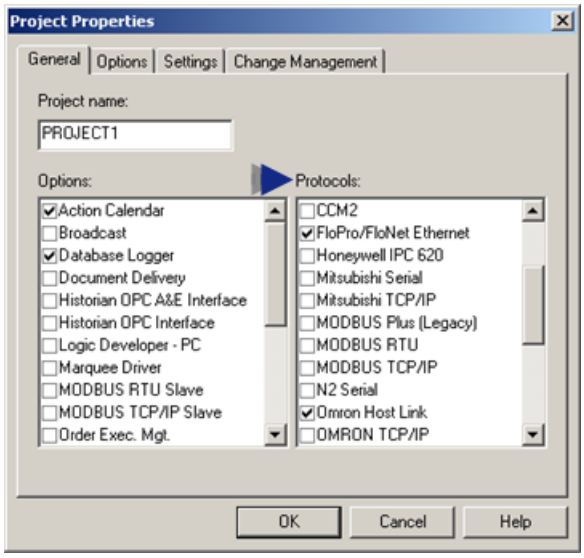
Note: You can create a maximum of 64 ports per project.

Step 1. Enable a Protocol in a CIMPLICITY Project

1. Click Project>Properties on the Workbench menu bar.

   The Project dialog box opens.

2. Select the General tab.



3. Scroll up and/or down in the **Protocols** box to find the protocol that will be used.
4. Check the protocol's check box if it is not already checked.

   The protocol is now enabled.

   Note: If the protocol does not appear in the **Protocol** box, use the CIMPLICITY installation CD to add the protocol.

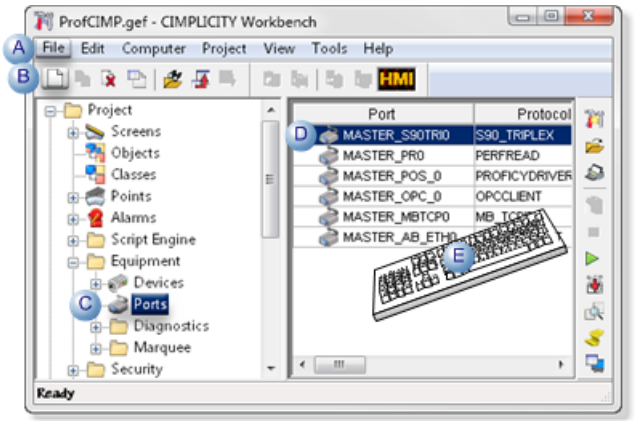
Step 2. Open a Port Properties Dialog Box

Step 2. Open a Port Properties Dialog Box

Option 2.1 *(page 48)*	Create a new port.
Option 2.2 *(page 50)*	Open an existing Port Properties dialog box.

Option 2.1. Create a new Port

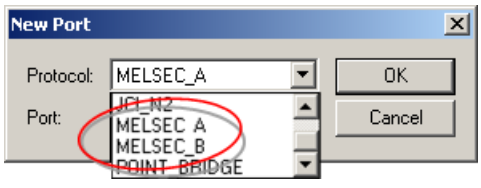
1. Select **Project>Equipment>Ports** in the Workbench left pane.
2. Do one of the following.



A	Click File>New>Object on the Workbench menu bar.	
B	Click the New Object button on the Workbench toolbar.	
C	In the Workbench left pane:	
	Either	Or
	Double click **Ports**.	a. Right-click **Ports**. b. Select New on the Popup menu.
D	a. In the Workbench right pane. a. Right-click any port. b. Select New on the Popup menu.	
E	Press Ctrl+N on the keyboard.	

The New Port dialog box opens when you use any method.

3. Right-click **Ports**.
4. Select New on the Popup menu.
5. Right-click any port.
6. Select New on the Popup menu.
7. Enter the following information to define the port.



Protocol	The protocols available to you will depend on the node you select, and the device communication protocols available on that node. To select a protocol, click the drop-down list button to the right of the input field to display the list of available protocols, then click the protocol you want to use.

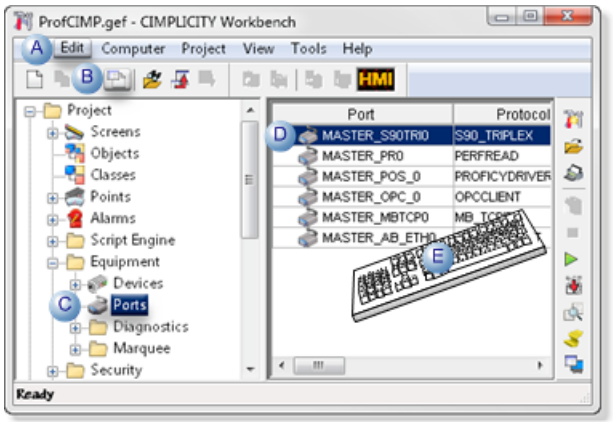
Port	The ports available to you will depend on the node and protocol you select. To select a port, click the drop-down list button to the right of the input field to display the list of available ports for the protocol, then click the port you want to use.

8. Click **OK** to continue configuring the new port.

The Port Properties dialog box for the new port opens. You will need to enter information for the General *(page 51)* and port-specific *(page 54)* properties.

Option 2.2. Open an Existing Port Properties Dialog Box

1. Select **Project>Equipment>Ports** in the Workbench left pane.
2. Select a port in the Workbench right pane.
3. Do one of the following.



A	Click Edit>Properties on the Workbench menu bar.	
B	Click the Properties button on the Workbench toolbar.	
C	In the Workbench left pane: a. Right-click **Ports**. b. Select Properties on the Popup menu.	
D	In the Workbench right pane:	
	Either	Or
	Double click a port.	a. Right-click a port. b. Select Properties on the Popup menu.
E	Press Alt+Enter on the keyboard.	

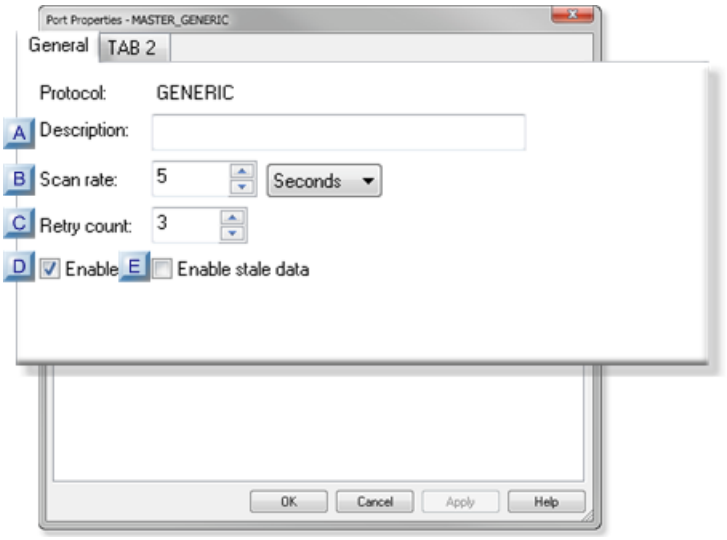
4. Right-click **Ports**.
5. Select Properties on the Popup menu.

6. Right-click a port.
7. Select Properties on the Popup menu.

Step 3. Configure Port General Properties

Step 3. Configure Port General Properties

General properties for a new port are as follows.



rect 0, 70, 20, 87 *(page 52)*
rect 0, 98, 21, 118 *(page 52)*
rect 1, 128, 19, 147 *(page 52)*
rect 0, 154, 20, 172 *(page 52)*
rect 73, 154, 91, 171 *(page 53)*

A *(page 52)*	Description
B *(page 52)*	Scan rate
C *(page 52)*	Retry count
D *(page 52)*	Enable

E *(page 53)*	Enable stale data

Note: Tab2 in the graphic above represents the port-specific tab, whose name depends on what port is being configured.

A	Description

A description can be up to 40 characters of explanatory text about the port.

B	Scan rate

The basic timer for points monitored from this port.

The rate at which points are polled is a multiple of this **Scan Rate**.

Configurable units can be set in any of the following units.

- Ticks (hundredths of seconds)
- Seconds
- Minutes
- Hours

C	Retry Count

Specifies the number of times to retry communications to devices on this port after a communications error is encountered

If communications cannot be established, devices on this port are considered to be down, and a $DEVICE_DOWN alarm is generated for each device.

Once a device is down, periodic attempts are made to resume communications to the device.

D	Enable

Do one of the following.

Check	Enable communications on this port.
Clear	Disable communications on this port.

guide: When a port is dynamically disabled, communication to all devices associated to that port will stop.

By default, when the port is dynamically disabled:

- The associated devices will be marked **Down** and their associated points will be marked **Unavailable**.
- Setpoints and processing of unsolicited data will not be allowed for the associated devices.

Tip: As an alternative to this default action, you may configure a Global Parameter, `ALLOW_UPDATE_WHEN_DISABLED`, to keep the associated devices alive.

If `ALLOW_UPDATE_WHEN_DISABLED` is configured, then when the port is dynamically disabled:

- The device is not marked **Down**.
- Polling stops.
- Setpoints and unsolicited messages will still be processed.

E	Enable Stale Data

Check Enable stale data to keep the point available in most circumstances that would have made it unavailable. However the point value will be stale, meaning it is the last known value and may or may not have changed in the device.

The following table displays a point's availability when events occur and **QUALITY.STALE_DATA** is configured as on or off.

	Point Availability when Stale Data is Configured	
Event	On	Off
Poll Failure	Available	Unavailable
Process Shutdown	Available	Unavailable
System Shutdown	Not Applicable	Unavailable
Ind. Point Unavailable	Available	Unavailable
Out of PTMRP Range Limits	Range Bit / Available	Unavailable
DC Sends Device State	Available	Unavailable
DynCfg Disable Point/Device		Unavailable

Tip: Use QUALITY.STALE_DATA (Attribute) to report if the point value is stale.

Point Availability with Enable Stale Data On or Off

	Point Availability when Stale Data is Configured	
Event	On	Off
Poll Failure	Available	Unavailable

Process Shutdown	Available	Unavailable
System Shutdown	Not Applicable	Unavailable
Ind. Point Unavailable	Available	Unavailable
Out of PTMRP Range Limits	Range Bit / Available	Unavailable
DC Sends Device State	Available	Unavailable
DynCfg Disable Point/Device		Unavailable

Step 4. Configure Port-Specific Properties

The port-specific properties depend on the type of port you are configuring.

- Allen-Bradley Communications.
- Allen Bradley DF-1 Communications
- Allen-Bradley Intelligent Antenna Communications
- CCM2
- (As-Is) DDE (Local only)
- FloPro/FloNet
- Honeywell IPC 620
- Johnson Control N2
- Marquee
- Mitsubishi A-Series Serial
- Mitsubishi TCP/IP
- Modbus RTU
- Modbus RTU Server
- Modbus TCP/IP
- Modbus TCP Server
- OMRON Host Link
- OMRON Ethernet
- OPC Client
- OPC UA Client
- Driver Server client
- Sharp TCP/IP
- Siemens TI Serial
- Smarteye Electronic Assembly
- SNP and SNPX
- Square D SyMax SyNet
- Toyopuc Ethernet Device
- Triplex

Note: The following port types require no port-specific properties.

- DDE
- FloPro/FloNet
- Mitsubishi TCP/IP(MELSEC_A or MELSEC_B)
- OMRON TCP/IP

Other Port Configuration

Other Port Configuration

- Saved settings startup action
- Saved settings deletion
- Change a port's protocol

Saved Settings Startup Action

Saved settings startup action is available for the following device communications.

- Allen-Bradley Communications
- FloPro/FloNet Communications
- Modbus RTU Communications
- Modbus TCP Communications
- SNPX Communications

Saved Settings Deletion

Saved settings deletion is available for the following device communications.

- Allen-Bradley Communications
- FloPro/FloNet Communications
- Modbus RTU Communications
- Modbus TCP Communications
- SNPX Communications

Change a Port's Protocol

In your CIMPLICITY project configuration, device and point data are protocol dependent. You cannot delete a port then re-add it using another protocol if device and point data for the original port still exist.

todo: To change a port's protocol:

1. Export the point data for the port using the Import/Export utility if you plan to re-use the same point data.
2. Delete the point data for the port.
3. Delete the device data for the port.
4. Delete the port.
5. Create the port with the new protocol.
6. Define the devices for the port.
7. If you exported the point data, then use the Import/Export utility to import the point data. Otherwise, define new points.

Chapter 4. Devices

About Devices

A device is anything that can communicate point data to CIMPLICITY software. CIMPLICITY software can read data from and write data to devices.

Use this function to configure devices and specify their characteristics. Configuration requirements for devices vary depending on the type of device and communications protocol being used. CIMPLICITY Device Communications documentation for detailed information about configuring a device for a particular protocol.

Step 1 *(page 57)*	Display devices in the Workbench.
Step 2 *(page 59)*	Open a Device dialog box.
Step 3 *(page 62)*	Configure device general properties.
Step 4 *(page 64)*	Configure device-specific properties.

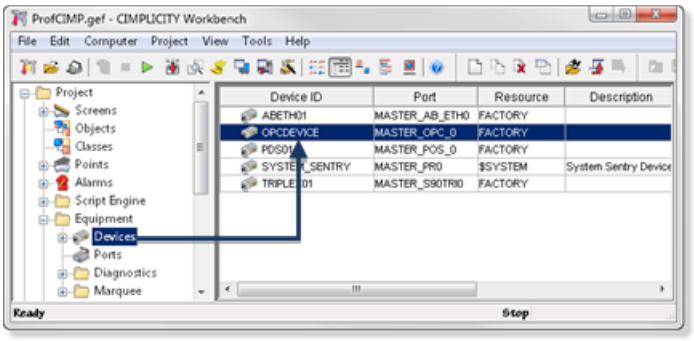
Step 1. Display Devices in the Workbench

1 *(page 57)*	Select devices.
2 *(page 58)*	Review device details.

Select Devices

Select **Project>Equipment>Devices** in the Workbench left pane.

The Workbench right pane displays the following fields for devices:



Device ID	The physical device identifier
Port	The port to which the device is connected
Resource	A configured resource Only users that are assigned this resource can view alarms for this device. **Note:** Alarms for points on the device may or may not use the same resource as the device.
Description	Up to 40 characters of text that describes the device

Note: Use the Workbench Field Chooser to remove or display any of the fields, except the Device ID. The Device ID is required.

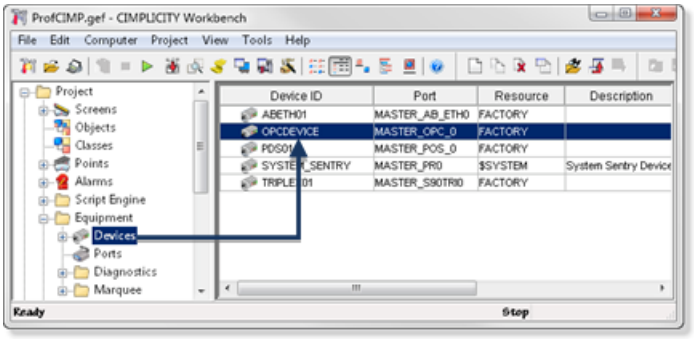
The Device list is initially sorted by **Device ID**. You can click on any of the other column titles at the top of the list to sort the list by that attribute.

Review Device Details

1. Select **Project>Equipment>Devices** in the Workbench left pane.
2. Expand **Devices**.

The existing devices display in a list under **Devices**.

1. Select a device.



The Workbench right pane displays fields that were selected in the Field Chooser for devices.

Example: Fields selected in the Field Chooser.

Point ID	All Point IDs that are associated with that device
Device ID	ID of the selected device (which is the source of the point data)
Resource	Resource associated with the device
Point Type	Type of point (e.g. **UINT**, **INT**)
Description	A description that may have been entered for the point in its Point Properties dialog box

Note: Information in the displayed fields comes from the associated point configuration. Use the Workbench Field Chooser to remove or re-display any of the fields, except the Point ID. The Point ID is required

The Device list is initially sorted by **Point ID**. You can click on any of the other column titles at the top of the list to sort the list by that attribute.

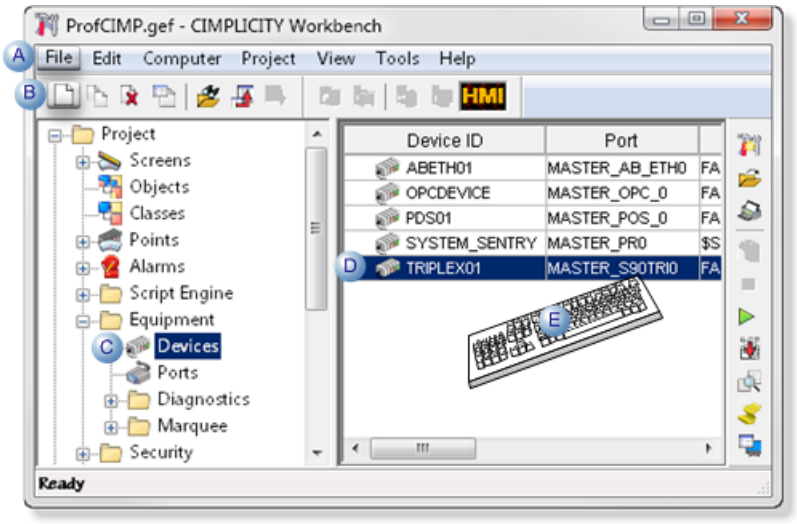
Step 2. Open a Device Dialog Box

Step 2. Open a Device Dialog Box

Option 2.1 *(page 59)*	Create a new device.
Option 2.2 *(page 61)*	Open an existing Device dialog box.

Option 2.1. Create a new Device

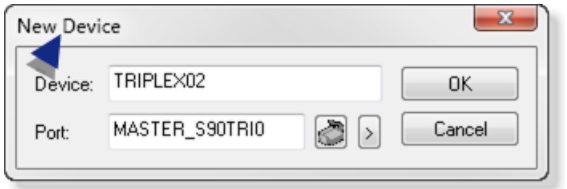
1. Select **Project>Equipment>Devices** in the Workbench left pane.
2. Do one of the following.



A	Click File>New>Object on the Workbench menu bar.	
B	Click the New Object button on the Workbench toolbar.	
C	In the Workbench left pane:	
	Either	Or
	Double click **Devices**.	a. Right-click **Devices**. b. Select New on the Popup menu.
D	a. In the Workbench right pane. a. Right-click any device. b. Select New on the Popup menu.	
E	Press Ctrl+N on the keyboard.	

The New Device dialog box opens when you use any method.

3. Right-click **Devices**.
4. Select New on the Popup menu.
5. Right-click any device.
6. Select New on the Popup menu.
7. Enter the following information to define the new device:



Field	Description

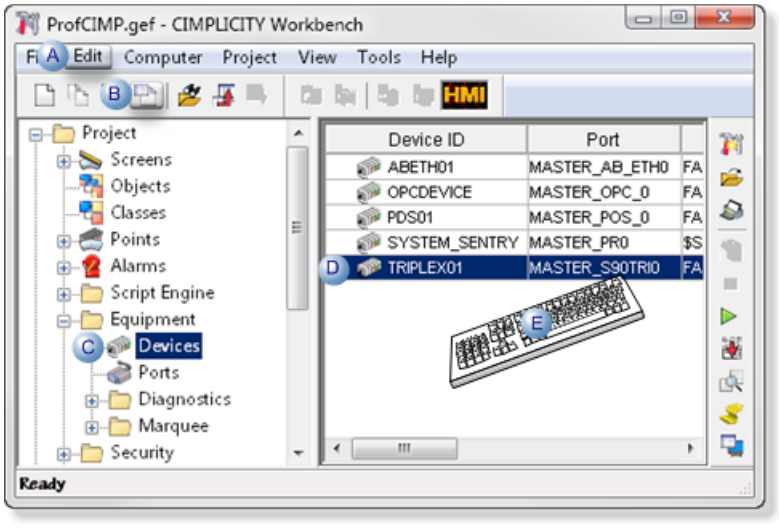
Device	Name of the new device. **Warning:** Do not name the device EGD. EGD is reserved for and will be assimilated by the Proficy Process Systems device, EGD.		
Port	Port that will be associated with the device. Buttons to the right of the Port field do the following.		
	Button		Description
		Broswer	Opens a Select a Port browser.
		Popup Menu	Opens a Popup menu to: • Create a new port. • Edit a current port. • Browse to find a port.

8. Click OK to continue configuring the new device.

The Device Properties dialog box for the new device opens. You will need to enter information for the General and device-specific properties.

Option 2.2. Open an Existing Device Dialog Box

1. Select **Project>Equipment>Devices** in the Workbench left pane.
2. Select a device in the Workbench right pane.
3. Do one of the following.



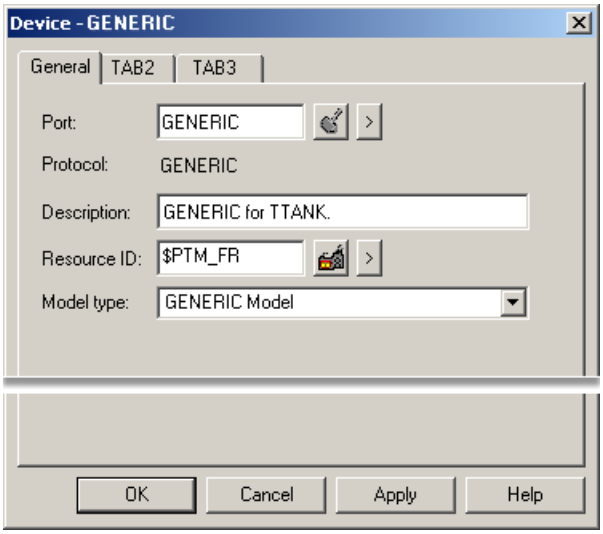
A	Click Edit>Properties on the Workbench menu bar.
B	Click the Properties button on the Workbench toolbar.

C	In the Workbench left pane: a. Right-click **Devices**. b. Select Properties on the Popup menu.	
D	In the Workbench right pane:	
	Either	Or
	Double click a device.	a. Right-click a device. b. Select Properties on the Popup menu.
E	Press Alt+Enter on the keyboard.	

4. Right-click **Devices**.
5. Select Properties on the Popup menu.
6. Right-click a device.
7. Select Properties on the Popup menu.

Step 3. Configure Device General Properties

Step 3. Configure Device General Properties



Properties to define on the General tab in a Device dialog box, are as follows.

Field	Description	
Port	The port selected in the New Device dialog box displays. Buttons to the right of the **Port** field enable you to do the following.	
	Button	Description

		Browser	Opens a Select a Port browser to select an existing port.
		Popup Menu	Opens a Popup menu to: • Create a new port. • Edit a current port. • Browse to find a port.
Description	Explanatory text about the device, up to 40 characters.		
Resource ID	Name of the device's resource. **Note:** Only the users that are assigned this resource will be able to see device alarms. Buttons to the right of the Resource ID field do the following.		
	Button		Description
		Browser	Opens a Select a Resource browser.
		Popup Menu	Opens a Popup menu to: • Create a new resource. • Edit a current resource. • Browse to find a resource.
Model Type	The type of device. The list of model types depends on the protocol.		

Ethernet Connections and Windows DHCP Media Sense

Media sense is a:

- Feature available on Windows-based computers that use TCP/IP.
- Mechanism for the network interface card to notify the TCP/IP protocol stack of media connect and disconnect events.

When, enabled DHCP can:

- Disable the network protocol stack when Windows detects that the network cable is disconnected from the network.
- Disable access to the local host (loopback address) when there are no other network connections.
- Re-enable the stack when the card is reconnected.

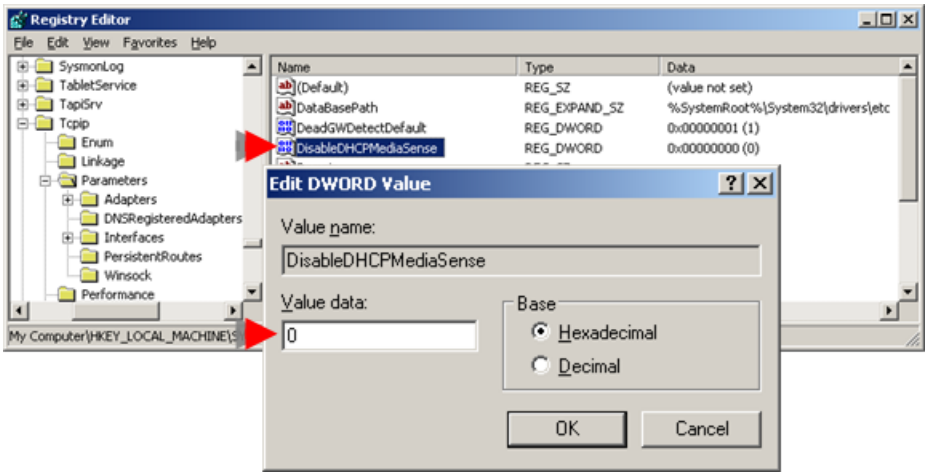
Support for Media Sense depends on the NIC and the media involved. For example, there is no support for Media Sense on 10Base2 coax cable because there is no line voltage supplied on the cable.

Important: Applications that require access to the loopback address even when the computer is not connected to a network should disable DHCP Media Sense by adding the following Registry key.

Path	\HKEY_LOCAL_MACHINE \System\CurrentControlSet \Services\Tcpip\Parameters	
Name	DisableDHCPMediaSense	
Type	REG_DWORDValue	
Data Range	1	True
Default	0	False (DHCP is enabled)

To re-enable Media Sense, do one of the following:

- Delete the registry entry or
- Change the value 0.



Step 4. Configure Device-Specific Properties

The device-specific properties depend on the type of device you are configuring.

- Allen Bradley
- Allen Bradley DF-1
- Allen-Bradley Intelligent Antenna
- CCM2
- (As-Is) DDE (Local only)
- FloPro/FloNet
- Genius

- Honeywell IPC 620
- Johnson Control N2
- Marquee Driver Operation
- Mitsubishi A-Series Serial
- Mitsubishi TCP/IP
- Modbus RTU
- Modbus RTU Server .
- Modbus TCP/IP
- Modbus TCP Server .
- OMRON Host Link
- OMRON TCP/IP
- OPC Client
- OPC UA Client
- Point_Bridge
- Driver Server client.
- Reflective Memory
- Sharp TCP/IP
- Siemens TI Serial
- Smarteye Electronic Assembly
- System Sentry
- SNP and SNPX
- Square D SyMax SyNet
- Toyopuc Ethernet
- Triplex

Chapter 5. Device QuickView

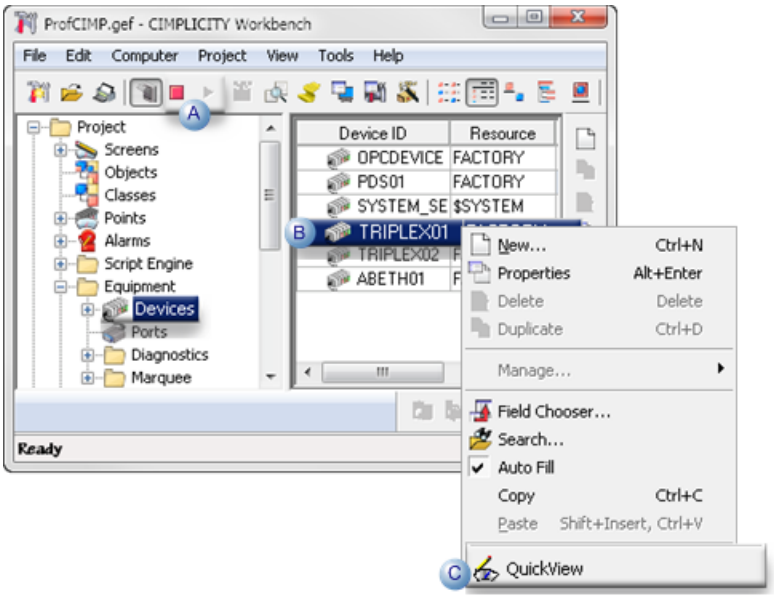
About Device QuickView

CIMPLICITY provides a tool to quickly see if a device is communicating successfully.

1 *(page 66)*	Open the Device QuickView tool.
2 *(page 67)*	Review device communication.

Open the Device QuickView Tool

Do the following.



A	Make sure the CIMPLICITY project is running.
B	Right-click the device that will be quick viewed.
C	Select QuickView on the Popup menu.

Result: A QuickView Device screen opens.

Review Device Communication

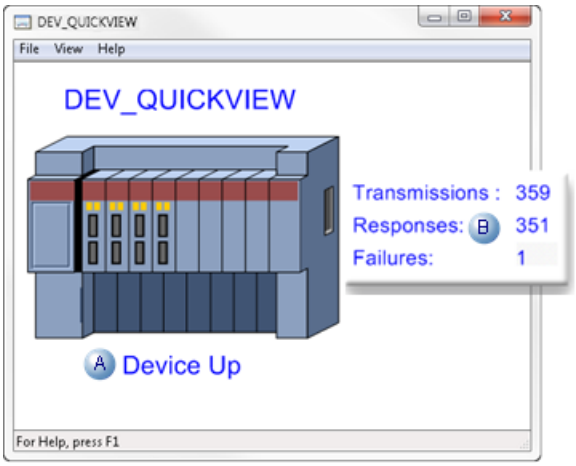
When the device's QuickView screen opens it instantly displays the device's diagnostic points; the points are updated (start counting) when the device communication starts.

The device is either up or down.

- Device Up
- Device Down

Device Up

The QuickView screen displays the following for a device that is currently communicating.

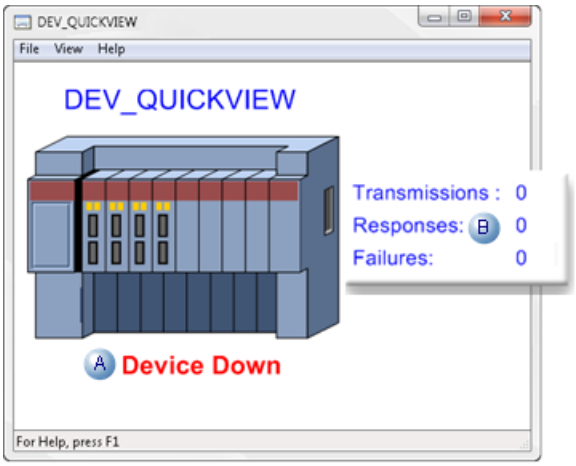


A	Device Up displays.	
B	The following diagnostics display counting from when the communication started.	
	Transmissions	Number of successful transmissions.
	Responses	Number of successful responses.
	Failures	Number of failures that might have occurred.

Note: If communication was uninterrupted, QuickView would display transmissions equal to responses and 0 failures.

Device Down

The QuickView screen displays the following for a device that is currently not communicating.



A	Device Down displays.	
B	The following diagnostics display 0.	
	Transmissions	Number of successful transmissions.
	Responses	Number of successful responses.
	Failures	Number of failures that might have occurred.

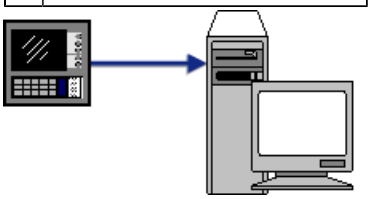
Chapter 6. Quick Device Setup

Quick Device Setup

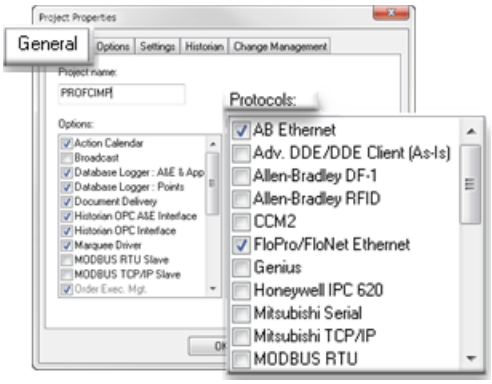
CIMPLICITY provides you with the tools to quickly include devices in your project and then create basic device points that immediately begin to monitor your systems.

To do this quick setup, you:

1	Connect a device to a server.

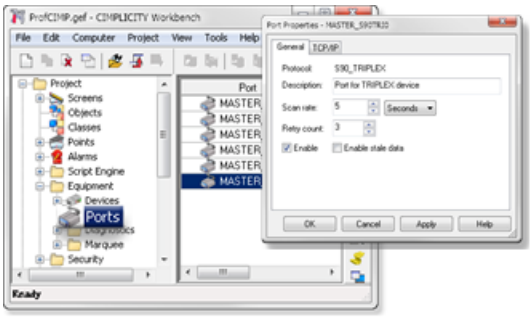


2 *(page 47)*	Enable a protocol in a CIMPLICITY project.



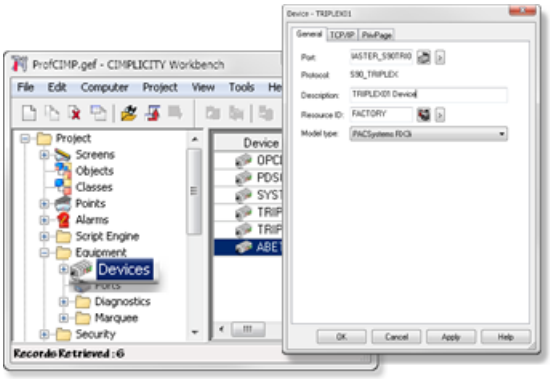
rect 130, 49, 306, 228 Step 1. Enable a Protocol in a CIMPLICITY Project *(page 47)*
rect -3, 12, 57, 42 Step 1. Enable a Protocol in a CIMPLICITY Project *(page 47)*

3 *(page 47)*	Configure a port for the device.



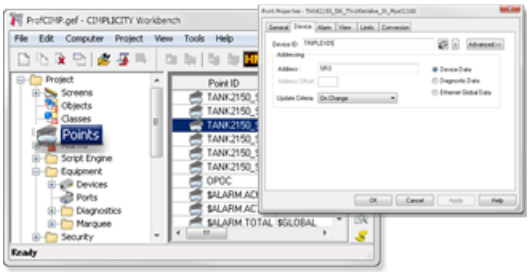
rect 161, 4, 325, 152 About Ports *(page 47)*
rect 35, 122, 83, 146 About Ports *(page 47)*

4 *(page 57)*	Configure a CIMPLICITY device.



rect 172, 0, 333, 219 About Devices *(page 57)*
rect 37, 154, 104, 178 About Devices *(page 57)*

5	Configure device points.



Chapter 7. Native Device Communications

About Native Device Communications

CIMPLICITY provides device communications to enable reading and writing point data to and from the following protocols.

	Allen Bradley
	Allen Bradley DF-1
	Allen-Bradley Intelligent Antenna
	CCM2
	(As-Is) DDE Client (Local only)
	FloPro/FloNet
	Honeywell IPC 620
	Johnson N2
	Marquee Driver Operation
	Mitsubishi A-Series Serial
	Mitsubishi TCP/IP
	Modbus RTU
	Modbus RTU Server .
	Modbus TCP/IP
	Modbus TCP Server .
	OMRON Host Link
	OMRON TCP/IP
	OPC Client
	Sharp TCP/IP
	Siemens TI Serial
	Smarteye Electronic Assembly
	SNP and SNPX
	Square D SyMax SyNet
	Toyopuc Ethernet
	S90 Triplex

Logical Names used for Native Device Communications

Logical Names used for Native Device Communications

Logical names are used to override default values in the `log_names.cfg` file for the CIMPLICITY Base System and options.

Note: Do not confuse logical names with environment variables. Logical names are found in the `log_names.cfg` file, while environment variables are accessed through the Control Panel.

The following device communications have logical names that can be modified:

- Native device communications (except OPC) supporting unsolicited data.
- Allen-Bradley Communications
- Mitsubishi TCP/IP Communications.
- SNP Communications.

Native Device Communications Supporting Unsolicited Data

The following native device communications support unsolicited data.

- Allen-Bradley Communications
- DDE Communications
- Modbus RTU Server Communications
- Modbus TCP Server Communications
- Smarteye Electronic Assembly Communications
- Square D SY/MAX Communications
- Mitsubishi TCP/IP Communications
- Modbus TCP/IP Communications
- Sharp TCP/IP Communications

Multiple Point Updates via a Logical Name (ALL_UNSO)

Since there are some applications for which you may not wish to enable these multiple point updates, this facility is controlled via a logical name, `ALL_UNSO`, which is defined in the CIMPLICITY logical names file.

`ALL_UNSO` applies to all device interfaces that support unsolicited data except OPC.

Enable a multi-point update function for all device interfaces configured for your system:

1. Define the logical `ALL_UNSO` to True by
2. Add a record similar to the following to log_names.cfg, a text file located in the Program Files \Proficy\Proficy CIMPLICITY\data directory.

   ```
   ALL_UNSO|P|default|10|true
   ```

3. Restart the device communications process to make the change active.

   Important: In a multi-node system, you need to make this change on the node where the device communications interface is configured to run.

   Enable a multi-point update function for a specific device communications interface:

   Add a logical

   `<PORT>_ALL_UNSO|P|default|10|true`

   Where

   <PORT> corresponds to the CIMPLICITY port for the specific interface.

   For the Device Communications Interface, the valid ports depend on the device communications that you are configuring.

   If a logical is not defined in the log names file, the default mode, single point update, will apply.

   The following native device communications (that are not OPC) support unsolicited data.

Allen-Bradley Communications Logical Names

Logical names that can be modified in the log_names.cfg file for Allen-Bradley Communications are:

PLC2_PROT_WRITE_ALL	Enable protected writes to all PLC-2 devices
PLC2_PROT_WRITE_< port **>**	Enable protected writes to all PLC-2 devices on the specified port
ABI_MAXDEF	Maximum number of devices and unsolicited data messages

Mitsubishi TCP/IP Communications Logical Names

Logical names that can be modified in the log_names.cfg file for Mitsubishi TCP/IP Communications are:

< prcnam **>_UNSO_ALL_TYPES**	Unsolicited point addresses are matched to both their Hexadecimal and Decimal format addresses.
< prcnam **>_UNSO_DEC**	Unsolicited point addresses are matched to their Decimal format addresses.
< prcnam **>_UNSO_EITHER**	Unsolicited point addresses are matched to either their Hexadecimal or Decimal format addresses or both.
< prcnam **>_UNSO_HEX**	Unsolicited point addresses are matched to their Hexadecimal format addresses.
DC_CONNECT_MS	Change the number of milliseconds to wait for a connection to be established between poll checks.
DC_CONNECT_URETRY_CNT	Change the number of retries to re-establish communications between poll checks.
DC_TCP_POLLS_MS	Change the number of milliseconds in the polling interval.
DCQ_DEAD_TIME	Change the number of seconds to wait before declaring a device dead.
MMAX_SYNC_TICKS	Changes how long (in ticks) to wait for a synchronous response from a device, once the request has been made of the device.
MSYNC_TICKS	Changes how long to wait for a synchronous response from a device, once the request has been made of the device.

SNP Communications Logical Names

Logical names that can be modified in the log_names.cfg file for SNP Communications are:

BSM$SNP_T1_TIME	Controls the time between SNP messages.
BSM$SNP_ATTACH_DELAY	Controls the time delay before SNP attach messages.

Chapter 8. Allen-Bradley Communications

About Allen-Bradley Communications

The Allen-Bradley Communications enabler lets you exchange data between CIMPLICITY software and AllenBradley PLC processors.

This enabler requires Rockwell's RSLINX™ Classic software in order to communicate with Allen-Bradley PLCs. You must have an RSLINX™ OEM license available to the application in order for the software to operate.

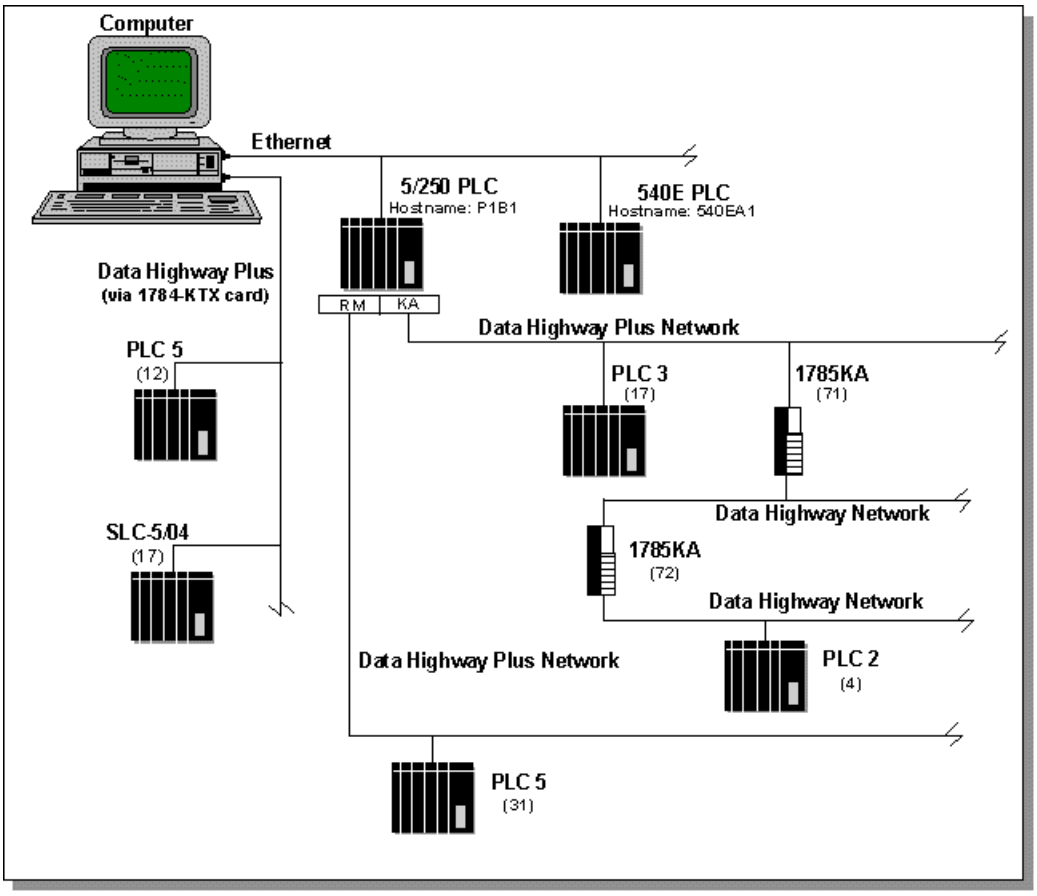
The physical communications link between a CIMPLICITY computer and the PLC processors is an Ethernet network or a Data Highway Plus network via an Allen-Bradley 1784 KTX card.

- The RSLINX™ software provides an interface to this layer; it is through this layer that the CIMPLICITY device communications enabler interfaces.
- The RSLINX™ OEM software also handles all of the validation and management of the Rockwell RSLINX licensing.

For the most current information on the version of RSLINX™ qualified with this version of CIMPLICITY, please refer to the Important Product Information (readme.chm) that accompanies this release.

Warning: Because of a registry conflict between the Allen-Bradley Data Highway Plus Communications enabler and RSLinx software, you cannot use the Allen-Bradley Data High Plus Communications enabler and the Allen-Bradley Communications enabler on the same computer.

Sample Allen Bradley Network



Allen-Bradley Communications Features

The Allen-Bradley Communications enabler:

- Provides the following features:
- Configuration functions for defining the logical communication ports used by CIMPLICITY software for the device interface. A device communications process is attached to each port, and up to four communication processes can be configured for a CIMPLICITY computer.
- Configuration functions for defining the devices accessed via each communication port and the data or "points" to be exchanged.
- Asynchronous exchange of data between CIMPLICITY software and a variety of devices.
- Support for collecting data from the AllenBradley processors via unsolicited data messages.
- Alarm generation when a communication failure occurs.

A utility program for Ethernet communications, **abiping.exe**, which is used for validating the network addresses that must be specified for each device to which communications are to be established.

- Supports the following CIMPLICITY features:
- Polled reads at user defined rates
- Poll after setpoint
- Triggered reads
- Analog deadband
- Alarm on communications failure
- Server Redundancy configurations
- Supports the following data types:
- Signed 8, 16, and 32 bit integer
- Unsigned 8, 16, and 32 bit integers
- Text
- Arrays
- Floats (for most supported device types).

Required Hardware/Software

You must have the following:

- Rockwell RSLINX™ OEM Software
- ControlLogix Gateway with the appropriate modules and ControlLogix Gateway Configuration software (see Allen-Bradley Publication 1756-10.2 ControlLogix Gateway Configuration Software and Communication through ControlLogix Gateway *(page 114)* in this document).

Supported Allen-Bradley Devices

The following Allen-Bradley devices are supported locally via an Ethernet link:

- AllenBradley PLC5 processors with a built-in Ethernet interface such as the PLC5/20E™, PLC-5/40E™, or PLC5/80E™
- AllenBradley PLC5/250 Pyramid Integrator Systems with an Ethernet Interface Module (5820-EI)

The following Allen-Bradley devices can be accessed through a 1784-KTX card:

- PLC5 Family PLCs

PLC5/12 (1785LT3), PLC5/25 (1785LT2)

- SLC-5/04

- SLC-5/05

The following AllenBradley devices can be accessed remotely through a Pyramid Integrator via a Resource Manager Module (5130RM1, RM2), or a KA Data Highway / Data Highway Plus Interface Module (5130KA).

- PLC5 Family PLCs

PLC5/12 (1785LT3), PLC5/25 (1785LT2)

- PLC2 Family PLCs

PLC2, 2/20 (LP1), MINI PLC2, 2/20 (LP2), PLC2/15, 2/16, 2/17, 2/30, 2/02, 2/05

DH/DH+ interfaces: 1171KC, 1171KD, 1771KA2, 1771KA, 1771KA4, 1771KE, 1771KF, 1770-KF2, 1771KG, 1771KGM, 1785KA3

- PLC3 with one of the following DH/DH+ interfaces: 1771KA, 1775SA, 1775SR5, 1775KA232, 1775S5, and 1775SR5 (new revisions)
- PLC5/250
- Supported PLCs via a 1785KA Bridge using offlink addressing
- SLC-5/04
- SLC-5/05

Supported File Types

For PLC3, PLC5/250, and PLC5 devices, data can be read and written to the following file types:

File Type	Description	Comments
N	Integer	
I	Input	Not supported for PLC-5/250, SLC-5/04 or SLC-5/05
O	Output	Not supported for PLC-5/250, SLC-5/04 or SLC-5/05
F	Floating Point	Not supported for SLC-5/04 or SLC-5/05
B	Bit	
T	Timer	PLC-5/250 CTL field is read-only
C	Counter	PLC-5/250 CTL field is read-only
S	Status	
A	ASCII	Not supported for PLC-5/250
D	BCD	Not supported for PLC-5/250, SLC-5/04 or SLC-5/05

H	High Order Integer	PLC-3 only
L	Long Integer	PLC-5/250 only
R	Control	Not supported for PCL-3 PCL-5/250 CTL field is read-only

For points in the Floating Point (F) file type, use the FLOATING point type.

For points in the BCD (D) file type, use the 3D_BCD or 4D_BCD point type.

When using Recipe Management to read and write in Floating Point files, only floating point types are supported.

For the PLC2, data can be read from and written to Register Memory only.

Required Allen-Bradley Software

You must install RSLINX OEM and have an available software license from Rockwell to use this device communication interface. The version of the software qualified with this release of CIMPLICITY is documented in the IPI.

The part number to be used when ordering RSLINX OEM software from Rockwell is 9355WABOEM.

If you need the `bootpd` service, you must call Rockwell Software for the latest copy of the bootptab file. The file is self-documenting.

Important: A sample bootptab file is included in the Installation Procedure section. If your file differs from the sample, follow the instructions in your bootptab file for adding PLC/5-xxE Ethernet processors and 5820-EI Ethernet Interface modules.

Allen-Bradley Related Documents

You should have one or more of the following documents available when configuring your communications network:

RSLinx OEM User's Guide

RSLinx Installation Procedures

RSLinx Installation Procedures

The RSLinx installation procedure you use depends on whether you are installing Ethernet communications or 1784-KTX communications.

Set up Contrologix Gateway to Communicate to CIMPLICITY

Set up ControLogix Gateway to Communicate to CIMPLICITY

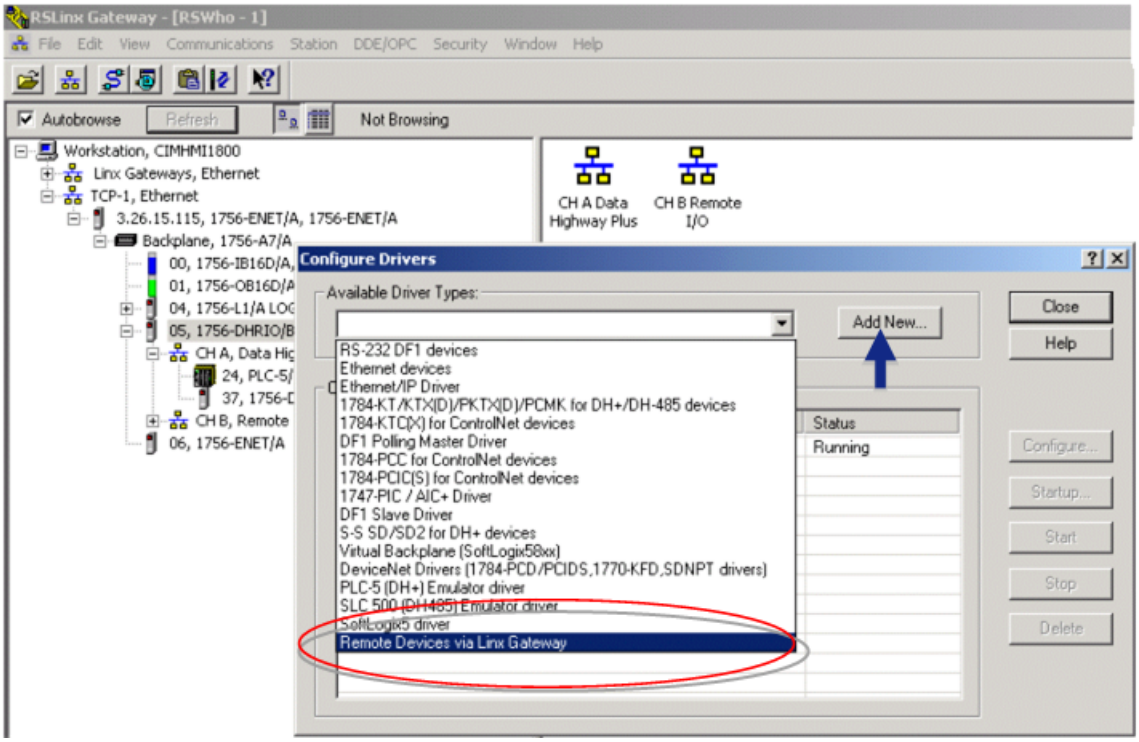
The following steps provide an overview on how to set up ControLogix Gateway to communicate to CIMPLICITY software.

Step 1 *(page 80)*	Add the Allen Bradley Driver in RS Linx.
Step 2 *(page 82)*	Review the Allen Bradley Driver in RS Linx configuration.
Step 3 *(page 84)*	Create a new port and device using the AB Ethernet driver.

Step 1. Add the Allen Bradley Driver in RS Linx

In RS Linx add the Allen Bradley driver Remote Devices via Gateway and configure it to point to the gateway device that you are using.

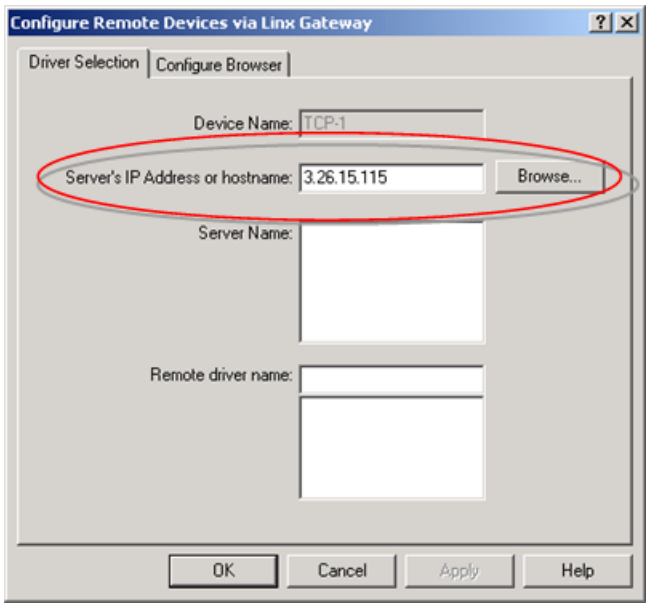
1. Select Remote Devices via Linx Gateway in the Configure Drivers dialog box.



2. Click Add New.

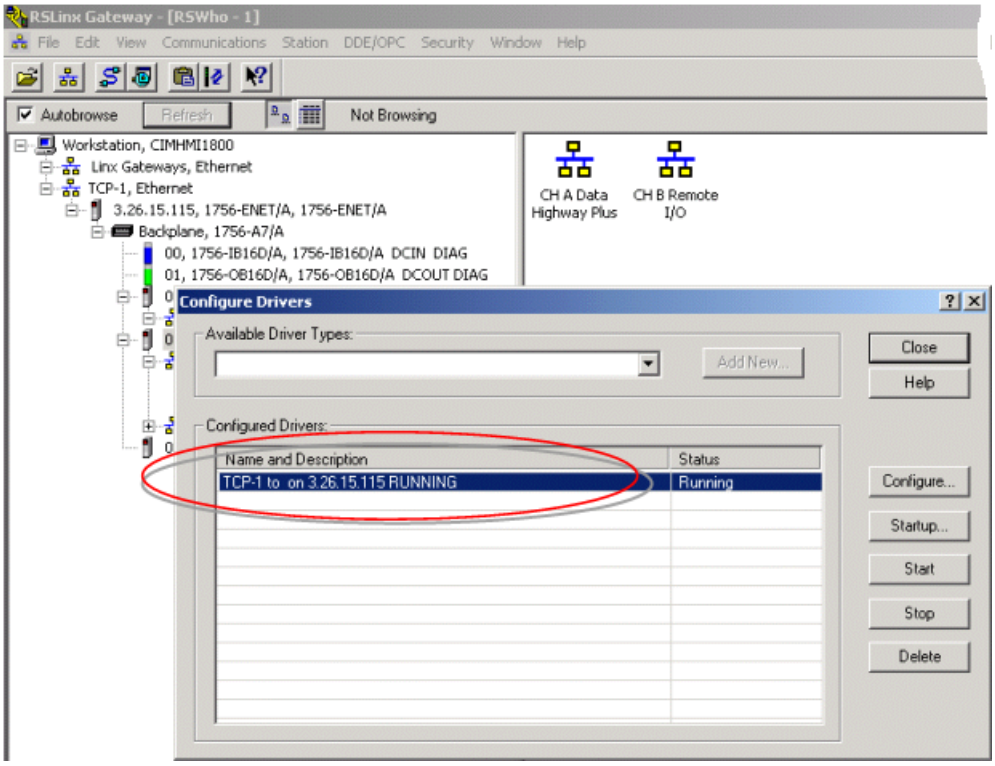
   The Configure Remote Devices via Linx Gateway dialog box opens.

3. Enter the Server's IP Address or host name, e.g. `3.26.15.115`.



4. Click OK.

The name and description display in the list of Configured Drivers.



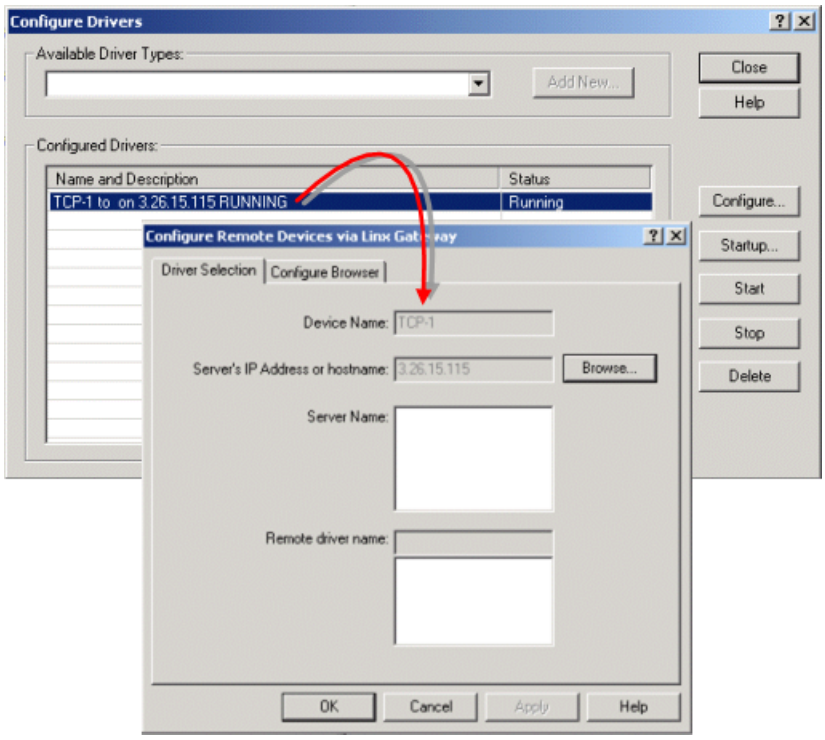
When the driver is configured and the PLC is running you can set up CIMPLICITY to communicate to the PLC via the AB Ethernet driver.

Step 2. Review the Allen Bradley Driver in RS Linx Configuration

1. Double-click the device and address in the Name and Description list, in the Configure Drivers window.

   The Configure Remote Devices via Linx Gateway dialog box opens.

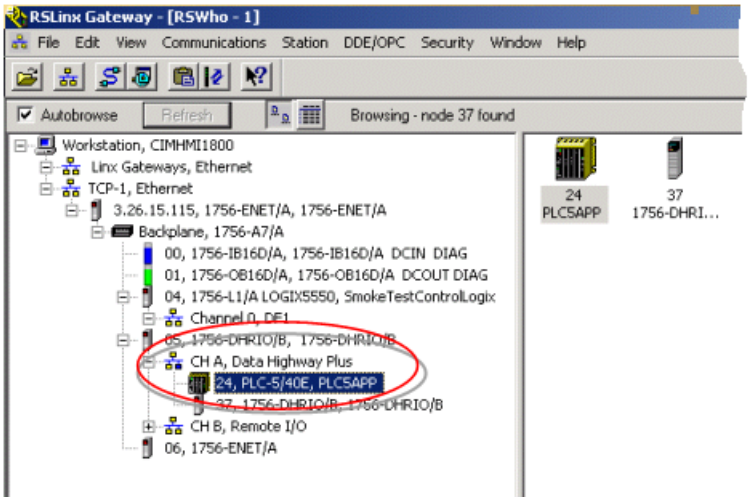
   The configured device Name and server address is now read-only.



2. Click OK to close the Configure Remote Device via Linx Gateway dialog box.

3. Click Close to close the Configure Drivers dialog box.

   Your RS Linx should look something like the configuration in this graphic; the DH+ network module is on slot 5, Channel A.



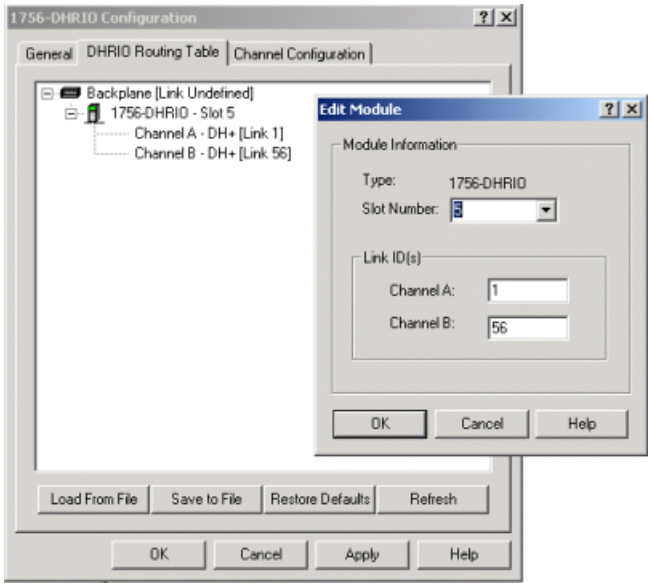
4. Double-click the 24PLC5APP.

   The DHRIO Configuration browser opens

5. Right-click the DHRIO module.

   A Popup menu opens.

6. Choose Module Configuration.



   You can change the Link ID for the module (or see what Link ID is currently being used).

Step 3. Create a new Port and Device using the AB Ethernet Driver

1. Create a new port and device using the Allen Bradley Ethernet driver.
2. For the address on the device, use an address as follows:

   ```
   Communication through ControlLogix Gateway
   ```

3. Specify the Network Address in the CIMPLICITY Device Configuration as follows

   For the AB Communications Driver, the network address for CIMPLICITY device configuration has following format:

   ```
   Driver_name, Port_id, Station_number
   ```

   Where

Driver_name	Is the name of the driver user configured in the Rockwell Software RSLinx OEM driver configuration. For communicating to a remote device via a ControlLogix Gateway, use TCP-n where n is the driver number.
Port_id	Has the following format: AB:KEYWORD/L:l Where

	`KEYWORD`	Is the port type. The port types for the ControlLogix Gateway is ASA.
	`/L:l`	Is the Destination Link ID of the Data Highway Plus Network to which you are bridging to (in the range from 0–65535 (decimal)).

Note: The Link ID used for the address must be represented in OCTAL. In RS Linx the Link ID will show up in decimal – in plant edition you must put this number in OCTAL.

4. See your Allen-Bradley network administrator if you need help defining the Port ID.

   `Station_number` specifies the station number of a remote processor on an Allen-Bradley network. For a remote device via the ControlLogix Gateway, the format is

   `Port.Slot.Channel.Station`

   Where

Port	Is always equal to 1 for Ethernet.
Slot	Is the location of the DHRIO module in the backplane.
Channel	Is the DHRIO module channel. Enter 2 for channel A, or 3 for channel B.
Station	Is the destination node number in decimal.

Note: The destination node number is represented in RS Linx as OCTAL and must be represented in in DECIMAL. For example the address that is shown in the screenshots is 24 for the device (This is in octal) and must be converted to 20 (decimal) to be used inCIMPLICITY .

Example

A device that is connected to a Contrologix Gateway has the following RS Linx attributes.

Port	1
Slot	5 for the DHRIO module
Channel	A
Node ID (in RS Linx)	24
Link ID (in RS Linx)	10

Then the following would be the connection string in Plant Edition for the device address:

`TCP-1,AB:ASA/L:12,1.5.2.20`

Important: Make sure you note that the Link ID has been converted TO Octal and the station number has been converted FROM octal.

Ethernet Communications RSLinx Installation Procedure...

Ethernet Communications RSLinx Installation Procedure

1. Install the RSLinx software.
2. Configure at least one PC to serve as Ethernet Boot server for the EI modules and Ethernet PLC-5 processor.
3. Create a LMHOSTS file in a directory of your choice, and use Notepad to enter the PLC IP address, node name and alias for each EI module and Ethernet PLC-5 processor. The format is:

   <IP_Address> <node_name> <alias>

4. Import the LMHOSTS file. See below for detailed instructions if you do not know how to do this.
5. Use the Rockwell Software RSLinx OEM driver configuration to configure the RSLinx Ethernet to PLC-5 or 5820-EI driver to communicate with the PLC-5 or EI module processors. Follow the instructions in your RSLinx documentation to do this.

Sample bootptab File

The following is an example of a **bootptab** file:

```
#
# Legend:
#      bf -- bootfile
#      gw -- gateways
#      ha -- hardware address
#      ht -- hardware type
#      ip -- host IP address
#      sm -- subnet mask
#vm -- BOOTP vendor extensions format
defaults5E: ht=1:vm=rfc1048
defaultsPI: ht=1:vm=rfc1048:hd="c:/rsi/bootpd":bf=sgmpccc.bin
# Copy and modify this entry for each PLC/5-xxE Ethernet processor:
# 1.   Change plc5name to the host name of the processor.
# 2.   Change aa.bb.cc.dd to the processor's IP address.
# 3.   Change xxyy to the last 4 digits of the processor's Ethernet
#      hardware address. This is found on a printed label on the top
#      edge of the circuit board, in the form 00-00-BC-1C-xx-yy.
#plc5name: tc=defaults5E:ip=aa.bb.cc.dd:ha=0000BC1Cxxyy
ALPLCE:tc=defaults5E:sm=255.255.255.0:ip=205.133.104.112:ha=0000BC1C05bd
# Copy and modify this entry for each 5820-EI Ethernet Interface module
# 1.   Change einame to the host name of the module.
# 2.   Change aa.bb.cc.dd to the module's IP address.
```

```
# 3.   Change xxyy to the last 4 digits of the module's Ethernet
#      hardware address. This is found on a printed label on the top
#      edge of the circuit board, in the form 00-00-BC-01-xx-yy.
#einame: tc=defaultsPI:ip=aa.bb.cc.dd:ha=0000BC01xxyy
```

Sample LMHOSTS File

The following is an example of a LMHOSTS file:

```
205.133.104.111    abplcd    ALPLCD
205.133.104.112    abplce    ALPLCE
```

Importing an LMHOSTS File

1. From the Start menu, select Control Panel.
2. In the Control Panel group, click **Network**.
3. In the Network Settings dialog box, select the Protocols property tab.
4. On the Protocols property tab, double-click on **TCP/IP Protocol** in the **Network Protocols** list.
5. In the Microsoft TCP/IP Properties dialog box, select the WINS Address property tab.
6. In the WINS Address property tab, select **Import LMHOSTS...**.
7. In the Open dialog box, specify the pathname for your LMHOSTS file, then select **Open**.
8. When the import is complete, select **Cancel** in the Microsoft TCP/IP Properties dialog box and the Network Settings dialog box.

1784-KTX Data Highway Plus Communications RSLinx Installation Procedure

You must configure the 1784-KTX card using the RSLinx Driver Configuration software. You cannot use the configuration driver supplied with the 1784-KTX card.

Follow the instructions found in the Rockwell RSLinx software to install the RSLinx software.

Use the Rockwell Software RSLinx OEM driver configuration to configure the 1784-KTX card. Follow the instructions in your RSLinx documentation to do this. See the following sections in the on-line Help for RSLinx software:

- How to Configure Communications Hardware
- How to Configure Communications Drivers

Allen-Bradley Communications Ethernet Installation Verification

Allen-Bradley Communications Ethernet Installation Verification

You can do the following to verify your installation:

- In RSLinx OEM, run the **SuperWho** utility to verify that the RSLinx driver has detected the existence of the configured PLC-5 or EI module processors on the Ethernet.
- Run the **abiping.exe** program.

Using abiping.exe

The **abiping.exe** program is provided to assist you in validating the network address to be configured for a device and verifying the communications link. This program is invoked by using the following command:

```
\CIMPLICITY\exe\abiping.exe PLCX Netname Port_id station_num
```

Where:

PLCX can be PLC2, PLC3, PLC5, or PLC5250

Netname, Port_id and station_num are as described in the Device Configuration section.

This is especially helpful when your communications network contains gateways and bridges.

If you enter a valid network address and your communications hardware is properly installed, the following message displays:

```
Communications successfully detected the device
```

If you enter an invalid address or your communication link is not operating properly, a message similar to the following displays:

```
Unable to successfully DTL_C_CONNECT to Device
Status = 158 address >PLC2< hi_xd = 0
```

Important: Enter the station ID in RSLINX software in Decimal format. In ABIPING, enter the address in Octal.

Examples of Valid Ethernet Network Addresses

The sample networkdepicts a PLC network utilizing a CIMPLICITY computer using Allen-Bradley Communications to:

- Communicate to various PLCs directly over Ethernet, through a PLC5/250 and other bridge devices, and through a 1784-KTX card.
- Communicate to various PLCs via a 1784-KTX card using Data Highway Plus protocol.

The network address for CIMPLICITY device configuration has following format :

Driver_name, Port_id, Station_number

The network address for each device in the sample Ethernet network is:

Device	Address
PLC5/250	AB_ETH-1,AB:LOCAL,1
PLC540E	AB_ETH-1,AB:LOCAL,2
PLC5	AB_ETH-1, AB:PIGATEWAY/P:0/M:RM/C:2/E:1,31
	Communication is via the Resource Manager in the 5/250 on channel 2
PLC3	AB_ETH-1,AB:PIGATEWAY/P:1/M:KA/C:2/E:1,6
	Communication is via the KA module in the 5/250 on channel 2
	This example assumes that the pushwheel setting on the KA Module is 1.
PLC2	AB_ETH-1,AB:PIGATEWAY/B:71/P:1/M:KA/C:2/E:1/KA,4
	Communication is via the KA module in the 5/250 on channel 2, a 1785KA bridge, and a 1785KA module

The network address for each device in the sample Data Highway Plus network is.

Device	Address
PLC-5	AB_KT-1,AB:LOCAL,12
SLC-5/04	AB_KT-1,AB:LOCAL,17
SLC-5/05	AB_KT-1,AB:LOCAL,17

Note: When directly connected to the Data Highway Plus network via the 1784-KTX adapter, the driver name is **AB_KT-1**, the Port ID is **AB:LOCAL**, and the Station Number is the number assigned to the PLC.

Refer to the Device Configuration section of this document for information on the format of the network address.

Refer to the RSLinx Reference Manual for additional detailed information on offlink addressing.

CIMPLICITY Configuration for Allen-Bradley

CIMPLICITY Configuration for Allen-Bradley

When you are configuring ports, devices, and device points, you will be asked for some information that is specific to the Allen-Bradley Communications enabler.

Allen-Bradley Port Configuration

Allen-Bradley Port Configuration

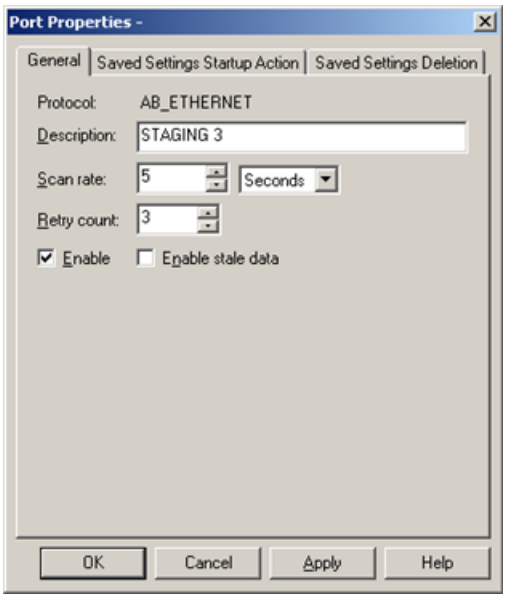
1. Select the following in the New Port dialog box.

Field	Description
Protocol	Select AB_ETHERNET from the list of available protocols.
Port	Select the communication port that will be used for Allen-Bradley Communications.

2. Click OK.

The Port Properties dialog box for Allen-Bradley Communications opens.

General Port Properties



Use the General properties to enter general information for the port. You can define the following:

Description	Enter an optional description to help you identify the port.
Scan Rate	Enter the base scan rate for the port. Point scan rates will be multiples of the base rate. You can specify a scan rate in **Ticks** (100 ticks = 1 second), **Seconds**, **Minutes**, or **Hours**.
Retry Count	If communications cannot be established to a device on the port, the device is considered to be down and a $DEVICE_DOWN alarm is generated. Once the device is down, periodic attempts are made to resume connections to it.
	Enter the number of scans to wait before attempting to reconnect to a device on this port after a communications error is detected.
Enable	Set this check box if you want the port to be enabled when the project starts. If you clear this check box, the port will not be enabled when the project starts, and points will not be available for devices on the port.
Enable Stale Data	Set this check box to keep the point available in most circumstances that would have made it unavailable. However, the point value will be stale, meaning it is the last known value and may or may not have changed in the device.

Saved Settings Startup Action

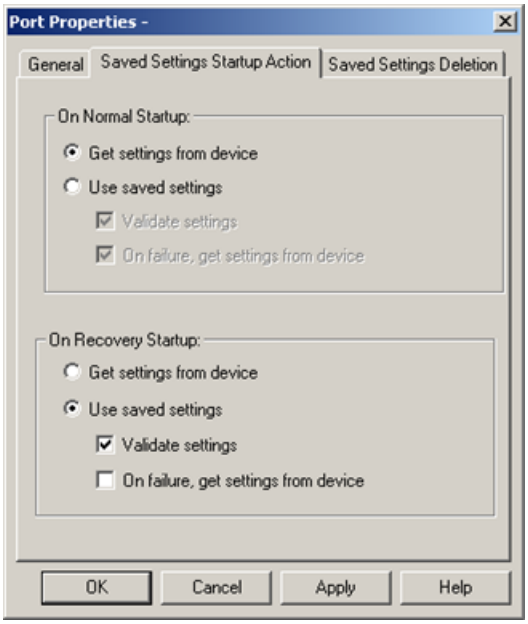
Allen-Bradley Communications provides you with the option to reduce normal and/or recovery start up time by saving device characteristics for subsequent re-use.

Note: Saved Settings Startup Action is intended for devices whose characteristics, such as the memory size, do not change. If the settings change, make sure they are deleted .

Select the Saved Settings Startup Action tab in the Allen-Bradley Communications Port Properties dialog box.

Selections can be made for:

- Normal startup.
- On Recovery Startup.



rect 30, 61, 123, 81 *(page 92)*
rect 33, 85, 161, 105 *(page 92)*
rect 31, 221, 159, 241 *(page 92)*
rect 32, 107, 232, 173 *(page 92)*
rect 29, 244, 229, 310 *(page 92)*
rect 27, 198, 155, 218 *(page 93)*

Normal Startup

- Normal Startup occurs when CIMPLICITY starts.
- CIMPLICITY is started either when:
- The computer is booted up or
- A CIMPLICITY user starts the project.

Options for each of the startups are as follows.

Get settings from device

Defines the actions that the device communication interface takes to determine the supported memory types and ranges for a specific device.

The methods vary by device communication interface.

Use saved settings

Checked: The device communication interface will use the stored settings to define the device-specific memory types and ranges.

- The device configuration data is recorded and stored for later use.
- Options if Use saved settings is checked are:

Option	Description	
Validate settings	Checked	The device communication interface will query the high range of each memory range as a quick check to confirm that the memory type and range represented is valid.
	Clear	The device communication interface will not do a quick check.
On failure, get settings from device	When using saved settings, failures may occur. Two typical failures are: • An integrity error is detected in the saved information. • There is a failure to verify a memory range when Validate settings is selected.	
	Checked	When a failure occurs the device communication interface will immediately attempt to determine the valid device information using the standard method for obtaining the information.
	Clear	When the failure occurs, the: • Device will immediately be marked down. • Valid device information will not be collected during initial startup. It will be determined later during retry processing.

On Recovery Startup

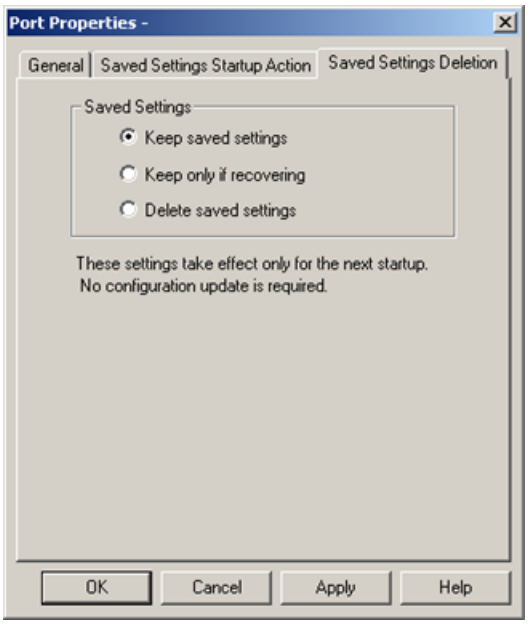
Recovery startup occurs after:

- A cluster fails.
- Process health kills a project as a result of a process failure.

Options *(page 92)* for On Recovery Startup are the same as they are for On Normal Startup.

Saved Settings Deletion

Check one of the following to specify what Allen-Bradley Ethernet Communications should do with saved settings.



Option	Description
Keep saved settings	Do not automatically delete the saved settings on the next startup
Keep only if recovering	Delete the current saved settings unless the next startup is in recovery mode.
Delete saved settings	Deletes: • The current saved settings at the next startup. • Settings for all the devices that are configured for the port. **Note:** If the configuration of the device is changed, make sure to check and apply Delete saved settings.

Note: These settings affect the next startup only. The Saved Settings Deletion tab will always default to the Keep saved settings option for subsequent startups.

The memory sizes for files in the Allen-Bradley are sized based on the highest referenced point configured for the file in the project. When adding a higher ordered element in a file statically (i.e., not through dynamic configuration), or adding a point that references a file previously unused in the configuration, you must use the Delete saved settings option just prior to the execution of the Configuration Update. If this action is not performed, the newly added point(s) at the higher addresses in the file may be declared invalid. Should this occur, rather than shutting down the project, one option is to dynamically edit the points and they will be re-validated.

Allen-Bradley Device Configuration

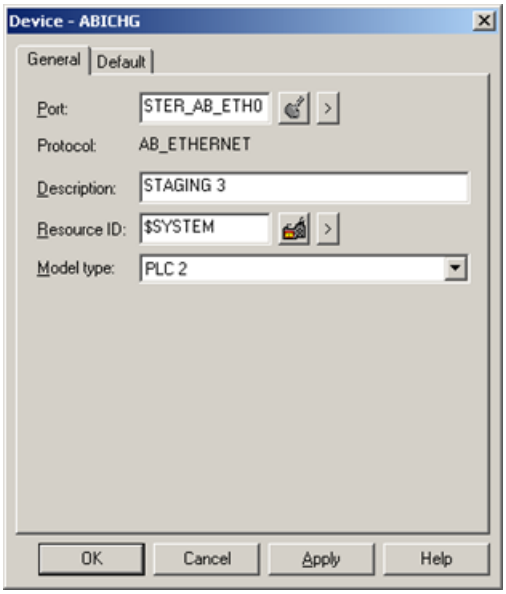
Allen-Bradley Device Configuration

1. In the **Device** field, enter the name of the device you are configuring.
2. In the **Port** field, select the Allen-Bradley Communications port to be used for the device.

   When you click **OK** to create the device, the Device Properties dialog box for devices using this protocol opens.

   Note: The following special characters cannot be used for an Allen-Bradley Communications device name: \, /, :, *, ?, ", <. >, |, [,]. If the device name contains any of these characters, the device settings will not be preserved and saved startup will not be possible.

General Device Properties

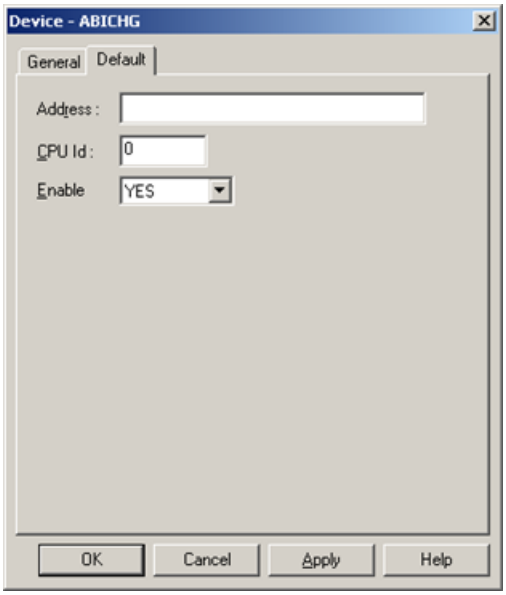


Use the General tab to enter general information for the device. You can define the following:

Port	Select the port for this device. Click buttons to the Port field to select the port as follows.		
	Browser		Display the list of ports and select one.
	Popup Menu		Create a new port, edit the current port, or select a port from the list of ports.
Description	Enter an optional description to help you identify the device.		

Resource	Enter a resource that can be associated with this device for alarm generation. Click buttons to the right of the **Resource** field to select the resource, as follows.		
	Browser		Display the list of resources and select one.
	Popup Menu		Create a new resource, edit the current resource, or select a resource from the list of resources.
Model Type	Select the type of device. Click the drop-down button to the right of the input field to display your choices, then make a selection. For this protocol, the choices are: • PLC 2 • PLC 3 • PLC 5 • PLC 5/250 • SLC-5/04 • SLC-5/05		

Default Device Properties



Use the Default tab to enter information about AllenBradley Communications for the device. You can define the following:

Address	Enter the Network Address for the device.
	The network address for CIMPLICITY device configuration has following format :
	Driver_name, Port_id, Station_number
	Where:
	Driver_name is the name of the driver user configured in the Rockwell Software RSLinx OEM driver configuration. For the Ethernet driver, use **AB_ETH-1**. For the 1784-KTX Data Highway Plus driver, use **AB_KT-1**.

	Port_id has the following format :	
	AB :**KEYWORD/B:b/L:l/G:g/P:p/M:m/C:c/E:e/KA**	
	Where:	
	KEYWORD	is the port type. Valid port types are: LOCAL OFFLINK PIGATEWAY DF1MASTER
	/B:b	is the bridge address in the range from 1–376 (octal).
	/L:l	is the Destination Link ID in the range from 0–65535 (decimal).
	/G:g	is the Gateway to final DH485 Link in the range from 1–367 (octal).
	/P:p	is the Pushwheel Number in the range from 0–4 (Ethernet).
	/M:m	is the Module Type. Valid types are: RM KA
	/C:c	is the Channel Number. Valid numbers are 0, 2 or 3.
	/E:e	is the Station number used in the Rockwell Software RSLinx OEM driver configuration. The number is an octal number in the range of 0-77 (RSLinx software used a decimal number here). If you are using the CIMPLICITY Host Redundancy option and the KT or KTX card, do not use the **/E:e** option.
	/KA	is used if the Bridge requires 1785-KA addressing mode. **/KA** is required to communicate through a 1785-KA from Data Highway Plus to Data Highway.
	If you need help defining the **Port ID**, see your Allen-Bradley network administrator.	
	Station_number specifies the station number of a remote processor on an Allen-Bradley network. Valid station numbers are : For Ethernet, use the station number used in the RSLinx Ethernet driver configuration. The number is an octal number in the range of 0-77 (RSLinx software uses a decimal number). For Data Highway channels, 1-376 (octal). For Data Highway Plus channels, 0-77 (octal). For Data Highway-485 channels, 0-37 (octal). For RS-232 Full Duplex DF1, 0-77 (octal).	
CPU ID	Not used.	
Enable	Select **Yes** in this box to enable the device when the project starts. Select **No** in this box to disable the device. Points associated with the device will be unavailable.	

Important: Devices that use the Port_ID keyword `LOCAL` must specify the RSLinx station number in the **station_number** field.

Allen-Bradley Point Configuration

Allen-Bradley Point Configuration

When you define a point, the following fields have values that are unique to or have special meanings for the Allen-Bradley Communications enabler.

General Tab

On the General tab in the Point Properties dialog box, you may configure points for Read or Read/ Write access.

When you are reading / writing data between CIMPLICITY software and a Floating Point file (F), define CIMPLICITY points using the REAL data structure type.

When you are configuring array points, make sure that the size of the array is within the limits of the file. CIMPLICITY software does not declare an array configured beyond the end of a file as invalid.

Device Tab

On the Device tab in the Point Properties dialog box.

Field	Description
Address	Point address requirements are device dependent. Valid devices are: PLC-2 PLC-3 PLC-5 PLC-5/250 SLC-5/04

Update Criteria	The update criteria determine how the data will be requested.	
	Enter	For
	On Poll or **On Scan**	Points whose values should be polled by the Allen-Bradley Communications enabler at regular intervals.
	On Demand On Poll or **On Demand On Scan**	Points whose values should be polled by the Allen-Bradley Communications enabler at regular intervals while users are displaying them.
	Unsolicited	Points that will accept unsolicited criteria.
	Poll Once	Points that are to be polled once when the Allen-Bradley Communications enabler starts.

When you are configuring Boolean points, you must also enter data in the following field:

Bit Offset: Enter the bit offset that corresponds to the bit position within the word. The valid range for bit offset is 0–15 where 0 is the least significant bit. The bit offset must be specified as a decimal value.

General Point Configuration

On the General tab, you may configure points for **Read** or **Read/Write** access.

When you are reading / writing data between CIMPLICITY software and a Floating Point file (F), define CIMPLICITY points using the REAL data structure type.

When you are configuring array points, make sure that the size of the array is within the limits of the file. CIMPLICITY software does not declare an array configured beyond the end of a file as invalid.

Device Point Configuration

On the Device tab:

Address	Point address requirements are device dependent. Valid devices are: PLC-2 PLC-3 PLC-5 PLC-5/250 SLC-5/04 SLC-5/05
Update Criteria	The update criteria determine how the data will be requested. Enter **On Poll** or **On Scan** for points whose values should be polled by the Allen-Bradley Communications enabler at regular intervals. Enter **On Demand On Poll** or **On Demand OnScan** for points whose values should be polled by the Allen-Bradley Communications enabler at regular intervals while users are displaying them. Enter **Unsolicited** for points that will accept unsolicited data. Enter **Poll Once** for points that are to be polled once when the Allen-Bradley Communications enabler starts.

When you are configuring Boolean points, you must also enter data in the following field:

Bit Offset	Enter the bit offset that corresponds to the bit position within the word. The valid range for bit offset is 0–15 where 0 is the least significant bit. The bit offset must be specified as a decimal value.

Point Address Formats

Point Address Formats

- PLC-2 address formats.
- PLC-3 address formats.
- PLC-5 address formats.
- PLC-5/250 address formats.
- SLC-5/04 and SLC 5/05 address formats.

PLC-2 Address Formats

PLC-2 devices support the following address formats:

File Type	Address Format
R	**<ELEMENT>**
	where **<ELEMENT>** is an octal offset

PLC-3 Address Formats

PLC-3 devices support the following address formats:

File Type	Address Format
D, A, H, S, N, F, B	**<FILE><FILE NUMBER>:<ELEMENT>**
	The use of leading zeros when specifying the **<FILE NUMBER>** and **<ELEMENT>** is not supported.
I, O	**<FILE><FILE NUMBER>:<OFFSET>** where **<FILE>** is **I** for inputs or **O** for outputs.
T, C	**<FILE><FILE NUMBER>.<TYPE>**

	where **<TYPE>** is one of the following: **ACC** - Accumulator **PRE** - Preset **CTL** - Control Word
	Bit offsets for the Control Word (CTL) in Counters and Timers are:

	Counters		**Timers**	
	Control Bit	**Offset**	**Control Bit**	**Offset**
	CU	15	EN	15
	CD	14	TT	14
	DN	13	DN	13
	OV	12		
	UF	11		
	EN	10		
	ZR	9		
	FN	8		

PLC-5 Address Formats

PLC-5 devices support the following address formats:

File Type	Address Format
D. A. N, F, B	**<FILE><FILE NUMBER>:<ELEMENT>**
	The use of leading zeros when specifying the **<FILE NUMBER>** and **<ELEMENT>** is not supported.
S, I, O	**<FILE>:<OFFSET>**
	where **<FILE>** is **I** for inputs, **O** for outputs, and **S** for status.
T, C, R	**<FILE><FILE NUMBER>:<ELEMENT>.<TYPE>**
	where **<TYPE>** is one of the following: **ACC** - Accumulator **PRE** - Preset **CTL** - Control Word **LEN** - Length (R only) **POS** - Position (R only) Accumulators, Presets, and Control Words are 16 bits.
	Bit offsets for the Control Word (CTL) in Controls are:

Control Bit	**Offset**	**Control Bit**	**Offset**
EN	15	ER	11
EU	14	UL	10
DN	13	IN	9
EM	12	FD	8

Bit offsets for the Control Word (CTL) in Counters and Timers are:

Counters			
Control Bit		**Offset**	
CU	15	EN	15
CD	14	TT	14
DN	13	DN	13
OV	12		
UN	11		

PD	**<FILE NUMBER>:<ELEMENT>**
	For more information about using this file type, see PLC-5 PD File Support.

PLC-5/250 Address Formats

PLC-5/250 devices support the following address formats:

File Type	Address Format
N, F, B, L	**<MODULE><FILE> <FILE NUMBER>:<ELEMENT>**
	The use of leading zeros when specifying the **<FILE NUMBER>** and **<ELEMENT>** is not supported.
S	**<MODULE><FILE>:<OFFSET>**
I, O	**<FILE>:<OFFSET>**
	where **<FILE>** is **I** for inputs or **O** for outputs.
T, C, R	**<MODULE><FILE> <FILE NUMBER>:<ELEMENT> .<TYPE>**
	where **<TYPE>** is one of the following: **ACC** - Accumulator **PRE** - Preset **CTL** - Control Word **LEN** - Length (R only) **POS** - Position (R only) Accumulators and Presets are 32 bits, and Control Words are 16 bits. Accumulators, Presets, and Control words for Counters are 16 bits. **Note For PLC-5/250 devices, Control Words are read-only .**
	Bit offsets for the Control Word (CTL) in Controls are:

	Control Bit	**Offset**	**Control Bit**	**Offset**
	EN	15	ER	11
	EU	14	UL	10
	DN	13	IN	9
	EM	12	FD	8

	Bit offsets for the Control Word (CTL) in Counters and Timers are:

	Counters		Timers	
	Control Bit	**Offset**	**Control Bit**	**Offset**
	CU	15	EN	15
	CD	14	TT	14
	DN	13	DN	13
	OV	12		
	UN	11		

SLC-5/04 and SLC-5/05 Address Formats

SLC-5/04 and SLC-5/05 devices support the following address formats:

File Type	Address Format
A, N, B	**<FILE><FILE NUMBER>:<ELEMENT>**
	The use of leading zeros when specifying the **<FILE NUMBER>** and the **<ELEMENT>** is not supported.
S	**<FILE>:<OFFSET>**
T, C, R	**<FILE><FILE NUMBER>:<ELEMENT>.<TYPE>**
	where **<FILE>** is **T** for timers or **C** for counters, and **<TYPE>** is one of the following: **ACC** - Accumulator **PRE** - Preset **CTL** - Control Word **LEN** - Length (R only) **POS** - Position (R only) Accumulators, Presets and Control Words are 16 bits.
	Bit offsets for the Control Word (CTL) in Controls are:

	Control Bit	**Offset**	**Control Bit**	**Offset**
	EN	15	ER	11
	EU	14	UL	10
	DN	13	IN	9
	EM	12	FD	8

	Counters		Timers	
	Control Bit	**Offset**	**Control Bit**	**Offset**
	CU	15	EN	15
	CD	14	TT	14
	DN	13	DN	13

	OV	12		
	UN	11		

PLC-5 PD File Support

PD File structures are extremely large (41 real values) and can be addressed using two different methods.

- The first method addresses the complete 41word real array (164 bytes) so that the entire PD structure can be read/written.

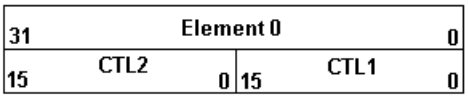
For example: PD30:5 returns 164 bytes.

- You can access multiple PD structures by creating larger arrays, which would be in increments of 41 real words or 164 bytes.
- The second method of addressing is to access each PID Mnemonic individually by using the .XXX format.

For example: **PD30:5.SP** returns the Setpoint value.

- These points may also be accessed as arrays. For example, to read the 8 setpoint values for **PD30:0.SP** through **PD30:7.SP**, you can create a standard real array of 8 elements with a starting at address at **PD30:0.SP**.

The first two mnemonics in the array are actually short integers called CTL1 and CTL2. These integers are mapped into the first real array value like this:



When you request CTL1 or CTL2 the appropriate 16 bit unsigned integer is returned.

The table below identifies each item within the array so that they can be paired up with their mnemonic. Information for this table can be found in the **Rockwell Software Hardware Interface Configuration User's Guide** and the **Instruction Set Reference** for PLC-5 Programmable Controllers. Bit access mnemonics are not implemented. They are supported using the same method as Timers and Counters. The particular bits are also listed in the table below.

Mnemonic		Bit	Index as Real	Description
CTL1			0	Control Word 0
	EN	15		Enable

	CT	9		Cascaded Type
	CL	8		Cascaded Loop
	PVT	7		Process variable tracking
	DO	6		Derivative action
	SWM	4		Set Output
	CA	2		Control Action
	MO	1		Mode
	PE	0		PID Equation
CTL2			0	Control Word 1
	INI	12		PID Initialized. This bit is cleared during pre-scan
	SPOR	11		Set Point out of range
	OLL	10		Output alarm, lower limit
	OLH	9		Output alarm, upper limit

Define Points for Unsolicited Data

Define Points for Unsolicited Data

The AllenBradley Ethernet Device Communications Interface can process unsolicited data messages in PLC2 style format. Unsolicited data can be generated by PLCs, which are directly connected to the Ethernet, such as PLC5 E series and 5/250 systems. PLCs that are connected to Data Highway or Data Highway Plus networks can indirectly send messages through a bridge such as a 5/250 Resource Manager.

You can configure unsolicited data points via the Device Point Configuration transaction by specifying an address in the following format:

```
point address>application address
```

Where

point address is the memory location in the PLC as described in the previous section. This address must be specified in Uppercase.

application address is a unique tag that identifies the unsolicited data item. This is an octal value and must be in the range of 0 – 77777.

Example: **N7:10>020**

The application address is a logical identifier for CIMPLICITY software, and the data will be mapped to the point address.

Rules For Defining Unsolicited Points

Rules For Defining Unsolicited Points

The following rules also apply when defining unsolicited data points:

- Application addresses.
- Data type and size.
- Update criteria.
- Array points.
- Unique point registration.

Application Addresses Rule

Rules for application addresses include:

- An application address must be unique for each point on your computer.
- An application address does not have to be sequentially assigned.
- The same application address can be used for multiple devices communicating to the same CIMPLICITY device communications process. When multiple AllenBradley Ethernet device communications processes are configured for a CIMPLICITY host computer, the same application address cannot be sent through the same network interface to two different CIMPLICITY device communications processes.
- You cannot dynamically update the application address for an unsolicited data point.

The application address is used to identify the CIMPLICITY point to the RSLINX interface software. As a result, there are several restrictions that must be applied to the use of application addresses with unsolicited point data:

- For each unsolicited message that a device sends to your CIMPLICITY project you must configure a device point with an application address. Configure the device point to have the exact same size and data type as the message being sent. Also the application address should match the CIF offset of that message.

For example, if you are sending a message containing integers from N7:20 to N7:40 (inclusive) at CIF offset 3, you should configure a point of type INT or UINT with 21 elements at the address N7:20>3.

- If you use the **ALL_UNSO** parameter and configure additional overlapping points for an unsolicited message, the additional points should not have an application address in their point address and do not need to be the exact same size as the unsolicited message.
- To insure that unsolicited communications is correctly initiated for every message, points that have been configured with an application address should never be configured as Delay Load point or as Just-in-time attributes of a point object.

Data Type and Size Rule

The data type and size of the data transmitted via an unsolicited message must match exactly the type and size configured for the CIMPLICITY point to be used for this data. For example, if ten 16-bit unsigned analog values are being transmitted in a message, a CIMPLICITY data structure **Type** of UINT with an **Element** attribute of 10 (ten) will be required. Messages that do not conform will cause an error code to be generated, and the data point will not be updated.

When multiple devices route unsolicited messages through the same PLC 5/250 to the same CIMPLICITY device communications process, and use the same application address, the data type and size of the messages must match exactly the type and size configured for the CIMPLICITY point.

Note: If possible, you should use unique application addresses for all messages being routed through a specific PLC 5/250.

Update Criteria Rule

You must specify the **Update Criteria** attribute as UNSOLICITED when configuring an unsolicited data point.

Array Points Rule

When configuring CIMPLICITY data points for an unsolicited data message that contains a block of data from a PLC, a CIMPLICITY array point must be configured. The **Element** attribute specifies the number of data points contained in the message, and the **Address** specifies the starting address where the data is to be mapped.

For example, if the unsolicited data consists of one register, a single point with an address of **N7:10** is updated with the unsolicited data. If the unsolicited data message contains 10 registers of data, then: element [0] of the array point contains the data for the address **N7:10**, element [1] contains the data for **N7:11**,... , and element [9] contains the information for data originating at **N7:19**.

Certain restrictions apply to CIMPLICITY array points. For example, alarms cannot be configured for array points, and setpoint actions are not allowed.

- To circumvent these restrictions, you can configure individual points with their corresponding PLC memory addresses. These points can be configured as polled points with a very slow scan rate. They will be updated each time the scan interval occurs. You can also set a global parameter to have these individual points updated by unsolicited data messages that map to the corresponding memory addresses.
- For example, if an unsolicited data point is configured as a ten element array point with a point address of **N7:100**, you can also configure ten points addressed at **N7:100** through **N7:109**. If the multiple point update global parameter is set to "true", then each time an unsolicited data message is received for the array points, the individual points addressed at **N7:100** through **N7:109** are also updated.

Unique Point Registration Rule

If you are implementing Allen-Bradley RSLinx applications directly, be aware that the same unsolicited points cannot be registered for both CIMPLICITY software and your application. Conflicts in these Allen-Bradley RSLinx resources may cause either your application or CIMPLICITY software to be unable to receive certain unsolicited messages.

Control Multiple Point Updates via a Logical Definition

Since there are some applications for which you may not wish to enable these multiple point updates, this facility is controlled via a logical defined in the CIMPLICITY logical names file. This file is located in your project's **\data** directory and is named **log_names.cfg.**

To enable this multi-point update function for all device interfaces configured for your system, define the logical ALL_UNSO to True by adding a record similar to the following to this file:

ALL_UNSO|P|default|10|true

Since this file is a text file, you can add this record with any text editor. After the edit has been completed, you need to restart the device communications process to make the change active. In a multi-node system, you need to make this change on the node where the device communications interface is configured to run.

Alternatively, you can enable this function for a specific device communications interface by adding a logical

<PORT>_ALL_UNSO|P|default|10|true

where **<PORT>** corresponds to the CIMPLICITY port for the specific interface. For the Allen-Bradley Device Communications Interface, the valid ports are ABINT0, ABINT1, ABINT2, and ABINT3.

If a logical is not defined in the log names file, the default mode of single point update will apply.

This logical applies to all device interfaces, which are based on the CIMPLICITY Device Communications Toolkit. This applies to those interfaces provided by GE Intelligent Platforms, Inc. as well as those developed by third parties.

Important: If you use OPC Client device communications, do not configure the ALL_UNSO logical. Instead, use the <PORT>_ALL_UNSO parameter.

Enable PLC2 Protected Writes

The AllenBradley Device Communications Interface utilizes unprotected writes when communicating to the PLC2.

For applications where "Protected Writes" are required, you can configure CIMPLICITY software to use protected writes by adding a logical to the CIMPLICITY software logical names files. This file is located in your project's **\data** directory, and is named **log_names.cfg.**

To enable protected writes for all AllenBradley Ethernet Communications Interfaces on a specific CIMPLICITY node, define the logical PLC2_PROT_WRITE_ALL to "true" by adding a record to this file similar to the following:

PLC2_PROT_WRITE_ALL|P|default|10|true

Alternatively, you can enable this function for a specific device communications interface by adding a logical

PLC2_PROT_WRITE_<PORT>|P|default|10|true

where **<PORT>** corresponds to the CIMPLICITY port for the specific interface. For the Allen-Bradley Device Communications Interface, the valid ports are ABINT0, ABINT1, ABINT2, and ABINT3.

For example, the record that must be added to the Logical Names file to enable this feature for an interface configured for port ABINT0 is:

PLC2_PROT_WRITE_ABINT0|P|default|10|true

If a parameter is not added to the logical names file, the default mode of unprotected write will apply. You can also disable protected writes on a node or port basis by specifying the logical value as "false".

Configure PLCs for Unsolicited Data

Configure PLCs for Unsolicited Data

The following sections describe how to set up unsolicited data on Allen-Bradley PLCs. It is assumed that Rockwell RSLOGIX programming software is used to set up the ladder logic on the PLC.

Ethernet Direct

Ethernet Direct

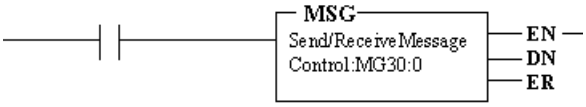
This is used for PLCs, which support direct Ethernet communication such as PLC-5/40E. It is assumed that the RSLinx Driver AB_ETH-1 is configured to include these PLCs as LOCAL.

Control Address:	Use an MG file address (ex. MG30:0)
Communication Command:	PLC-2 Unprotected Write
PLC-5 Data Table Address:	Starting address in the PLC of unsolicited data
Size In Elements:	Number of Elements of Data
*Host / Internet IP address:	IP address of CIMPLICITY host
Destination DT address:	PLC-2 data table address configured for CIMPLICITY point
Port Number:	Channel to be used for communication (ex. 2A)

You can also use the word **CLIENT** instead of specifying the IP address. In that case, you need to configure **piunsol.ini** file on the host computer.

PLC-5 Ethernet Direct Example

The following sample message block is an example of sending an unsolicited message from a PLC-5/40E to CIMPLICITY project running RSLinx software.



Control Address:	MG30:0
Communication Command:	PLC-2 Unprotected Write
PLC-5 Data Table Address:	N55:0

Size In Elements:	2
*Host / Internet IP address:	3.26.5.205 (or CLIENT)
Destination DT address:	30
Port Number:	2A

In CIMPLICITY software, if you define an INT type array point of 2 elements with address N55:0>30, and scan type UNSOLICITED for the device sending the unsolicited data, values are updated each time the data is received from the device.

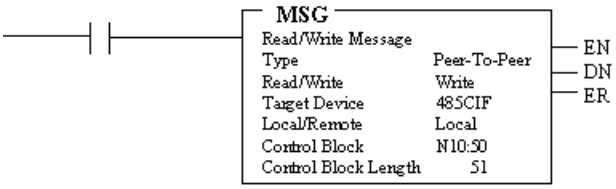
If you specify **CLIENT** instead of a host IP address, the following section must be included in the **piunsol.ini** file on the host computer.

[plc IP address]

30=2

SLC-505 Ethernet Direct Example

The following sample message block is an example of sending an unsolicited message from a SLC-505 to a CIMPLICITY project running the Allen Bradley Communications Enabler.



Control Block:	N10:50
Control Address Length	51
Read/Write	Write
Target Device	485CIF
Local / Remote:	LOCAL
Channel:	1
IP Address.	IP Address of the Machine running CIMPLICITY.
Our Source File Address:	The address of the unsolicited point in HMI.
Targets CIF offset:.	Offset in bytes, the >value in HMI will be ½ this value.

In CIMPLICITY software, if you define an INT type point with 1 element with address N20:10>2,.the scan type is Unsolicited for the point. The device will be sending the data unsolicited so the values are updated each time the data is received from the device.

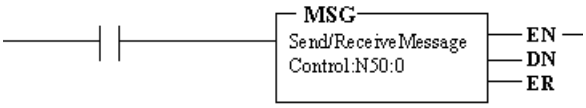
Ethernet through PI Gateway

Ethernet through PI Gateway

This is used for PLCs that are connected to the CIMPLICITY host computer through a PI Gateway. The connection between the PI Gateway and CIMPLICITY software is through Ethernet, and PLCs are connected to the Gateway through a Data Highway Plus network. The unsolicited messages are sent locally on DH+ to the EI module of the PI Gateway. The **piunsol.ini** file must be configured on the host computer to receive unsolicited messages from the PI Gateway.

PLC-5 Example

The following sample message block is an example of sending unsolicited message from a PLC-5 to a CIMPLICITY host running RSLinx software through a PI Gateway.



Control Address:	N50:0
Communication Command:	PLC-2 Unprotected Write
PLC-5 Data Table Address:	N55:0
Size In Elements:	2
Local / Remote:	LOCAL
Remote Station:	N/A
Link ID:	N/A
Remote Link type:	N/A
Local Node Address:	15 (This is the DH+ node address of the EI module on the PI Gateway. It is also called the virtual port no. on PI Gateway)
Destination DT address:	30

In CIMPLICITY software, if you define an INT type array point of 2 elements with address N55:0>30, and scan type UNSOLICITED for the device sending the unsolicited data, values are updated each time the data is received from the device.

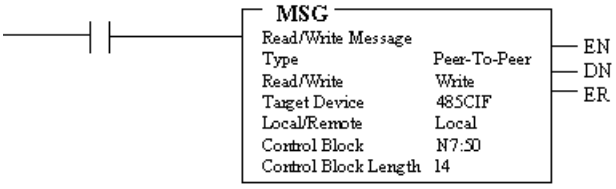
The following section must be included in **piunsol.ini** file on host computer.

```
[IP address of PI Gateway]
```

```
30=2
```

SLC-5/04 or SLC-5/05 Example

The following sample message block is an example of sending unsolicited message from an SLC-5/04 or SLC-5/05 to a CIMPLICITY host running RSLinx software through a PI Gateway.



Control Block:	N7:50
Read/Write:	Write
Target Device:	485-CIF
Local / Remote:	Local
Channel:	1
Target Node:	13 (015) (Octal 15 is DH+ node address of EI module on PI Gateway)
Local File address:	N55:0
Target file address/ offset:	48 (The target offset is in byte count. Enter PLC-2 application address*2 expressed in decimal)

In CIMPLICITY software, if you define an INT type array point of 2 elements with address N55:0>30, and scan type UNSOLICITED for the device sending the unsolicited data, values are updated each time the data is received from the device.

The following section must be included in **piunsol.ini** file on host computer.

```
[IP address of PI Gateway]
30=2
```

Data Highway Plus

Data Highway Plus

Use this when the host computer has an KT (or KTx) card and the RSLinx driver (AB_KT) has been configured for this card. The unsolicited messages are sent locally on DH+ to the CIMPLICITY host computer. The configuration of unsolicited messages in this case is the same as described in Ethernet through PI Gateway. The only differences are:

- The **piunsol.ini** configuration is not required on host.

- The DH+ node address of the CIMPLICITY host is entered under **local node address**.

Configuring PIUNSOL.INI

This is a configuration file for RSLinx.

There are two cases when you need to configure the **piunsol.ini file**:

1. When the PLC sends unsolicited messages through PI Gateway.

   You need to configure the PI Gateway's IP address in the **piunsol.ini** file

2. When the PLC sends unsolicited messages directly on Ethernet by specifying **CLIENT** as destination IP address.

   You need to configure the PLC's IP address in the **piunsol.ini** file.

   This file needs to be in RSLinx root directory (generally **\RSI\RSLINX**) of the host computer running the CIMPLICITY application. The format of this file is :

```
[<IP address 1>]
<application data table address> = <Number of elements>
<application data table address> = <Number of elements>
     .
     .
[<IP address 2>]
<application data table address> = <Number of elements>
<application data table address> = <Number of elements>
     .
     .
[<IP address n>]
<application data table address> = <Number of elements>
<application data table address> = <Number of elements>
     .
     .
```

   Where

   <application data table address> is a unique octal number associated with the unsolicited point.

   For example, **30** is the application data table address in the CIMPLICITY point address **N55:0>30**. Assuming the point is configured with a size of 2 elements, then the entry is:

```
30=2
```

   in the **piunsol.ini** file.

Communication through ControlLogix Gateway

Specifying the Network Address in the CIMPLICITY Device Configuration

For the AB Communications Driver, the network address for CIMPLICITY device configuration has following format:

Driver_name, Port_id, Station_number

Where:

Driver_name is the name of the driver user configured in the Rockwell Software RSLinx OEM driver configuration. For communicating to a remote device via a ControlLogix Gateway, use **TCP-n** where **n** is the driver number.

Port_id has the following format:

AB:KEYWORD/L:l

Where:

KEYWORD is the port type. The port types for the ControlLogix Gateway is ASA.

/L:l is the Destination Link ID of the Data Highway Plus Network to which you are bridging to (in the range from 0–65535 (decimal)).

If you need help defining the **Port ID**, see your Allen-Bradley network administrator.

Station_number specifies the station number of a remote processor on an Allen-Bradley network. For a remote device via the ControlLogix Gateway, the format is:

Port.Slot.Channel.Station

Where:

- Port is always equal to 1 for Ethernet.
- Slot is the location of the DHRIO module in the backplane.
- Channel is the DHRIO module channel. Enter 2 for channel A, or 3 for channel B.
- Station is the destination node number in decimal.

Network/Cable Redundancy Support

Network/Cable Redundancy Support

CIMPLICITY network/cabling redundancy for Allen Bradley Communications requires each Allen-Bradley PLC to be connected to two different physical networks.

Following are some examples of redundant networks as regards Allen-Bradley PLCs and CIMPLICITY software.

- The CIMPLICITY computer has two KT (or KTX) cards. (A combination of one KT and other KTx is also allowed). An Allen-Bradley PLC is connected to computer using two Data Highway plus networks (cables).

In order to have these networks redundant, user has to take following steps.

1. Configure a driver in RSLinx for each card (AB_KT-1 and AB_KT-2). The Data Highway Plus node address for the two drivers have to be different.
2. In CIMPLICITY device configuration, provide two network addresses for the device. You can specify these addresses by separating them by a ";" (semicolon).

   For example, if the two network addresses for a device are AB_KT-1,AB:LOCAL,22 and AB_KT-2,AB:LOCAL,22, the Network Address field in device configuration is:

   AB_KT-1,AB:LOCAL,22;AB_KT-2,AB:LOCAL,22

   - The CIMPLICITY computer has one Ethernet card and one KT (or KTx) card. An Allen-Bradley PLC is connected to computer using the Ethernet and Data Highway Plus networks.

   In order to have these networks redundant, user has to take following steps.

3. Configure a driver in RSLinx for each card (AB_ETH-1 and AB_KT-1).
4. In CIMPLICITY device configuration, provide two network addresses for the device. You can specify these addresses by separating them by a ";" (semicolon).

   For example, if the two network addresses for a device are AB_ETH-1,AB:LOCAL,22 and AB_KT-1,AB:LOCAL,22 , network address field in device configuration is:

   ```
   AB_ETH-1,AB:LOCAL,22;AB_KT-1,AB:LOCAL,22
   ```

Important: CIMPLICITY software currently does not support network redundancy for Allen-Bradley Communications when the computer has two Ethernet cards.

Features of network redundancy

A PLC, which is connected on two physical networks, is considered to be two logical devices having two different network addresses. The two devices are termed as Primary and Secondary. Only one network at a time is used to access the PLC.

CIMPLICITY network redundancy for Allen-Bradley Communications works in the following manner:

- On start-up, the Allen-Bradley Communications software uses the primary device to access the PLC.
- When the communication software determines that the primary device has failed, it switches communications to the secondary device. When this switchover occurs, **REDUND_DEVICE_DOWN** and **the communication software generates DEVICE_FAILOVER alarms**.
- If subsequently , the primary device comes up, the communication software clears the **REDUND_DEVICE_DOWN** alarm. However device communications continues to use secondary device for communication with the PLC.
- If secondary device fails while the communication software is using it, the communication software will switch back to primary device after generating proper alarms.
- As long as the communication software can talk to the PLC over a network, the device will always remain on-line (Alive).
- The device will be declared dead only when both the networks are down.

The Resource ID for all secondary devices is **$SYSTEM**. Because the Resource ID is used when **REDUND_DEVICE_DOWN** and **DEVICE_FAILOVER** alarms are generated and cleared, it is recommended that you not enter **$SYSTEM** in the **Resource** field when you configure redundant devices.

With this scheme, there is a single alarm listed for all **REDUND_DEVICE_DOWN** alarms on the secondary network. You can use the **View Stack** option in the Alarm Viewer to look at individual alarms.

All **DEVICE_FAILOVER** alarms are generated using Resource ID you configure for the device.

Unsolicited data

To support unsolicited data over redundant networks, you need to configure the PLC ladder logic to send the unsolicited data over both the networks. To do this, you need to use separate Application Data Table addresses for the point on the primary device and the secondary device.

To calculate the point's address on the secondary device, add 10,000 (octal) to its address on the primary device.

Example

You have configured an unsolicited point with address **N7:0>06** in your CIMPLICITY project. The Application Data Table address for the unsolicited point is 06. You must configure PLC ladder logic to send the unsolicited data to this Application Data Table address. If network redundancy is supported for this PLC, you also must configure the ladder to send the same unsolicited data at same time to the address 10006 (10000 + 06) over the PLC's secondary network.

Allen-Bradley Ethernet Global Parameters

Allen-Bradley Ethernet Global Parameters

- AB_WS_UNSO_PLC5_FLOA
- ABI_MAXDEF
- ABETH_PLC_POLL_TIMEOUT
- ABETH_PLC_REQUEST_TIMEOUT
- ABETH_PLC_RESPONSE_TIMEOUT
- ABETH_UNSO_QUEUE_SIZE

AB_WS_UNSO_PLC5_FLOAT

`AB_WS_UNSO_PLC5_FLOAT` deals with floating point numbers received via unsolicited using a PLC 2 unprotected write that could be word-swapped for data that is mapped to a floating point Point Type.

AB_WS_UNSO_PLC5_FLOAT

For	Project
Purpose	To control how incoming data is mapped to the point by swapping/not swapping word order.
Value	Enter one of the following: Y, N, or Default. deflaut

ABI_MAXDEF

By default, the AllenBradley Communications enabler supports a combination of 100 devices and unsolicited data messages. If the total number of devices and unsolicited data messages exceeds this number, you must define an additional logical **ABI_MAXDEF** in the **log_names.cfg** file in order to specify the number of definitions required for your installation.

The following entry would increase the maximum definitions to a value of 300.

```
ABI_MAXDEF|P|default|6|300
```

ABETH_PLC_POLL_TIMEOUT

The Allen-Bradley Communications enabler uses asynchronous requests to poll point data. By default, time-out for an asynchronous request is 10 seconds. You can use this global parameter to increase or decrease the time-out value.

To change the default time-out, add the following global parameter:

ABETH_PLC_POLL_TIMEOUT

For	Project
Purpose	To change the default time-out for an asynchronous poll request.
Value	n (where n is a value greater than 0)
Default Value	10

ABETH_PLC_REQUEST_TIMEOUT

The Allen-Bradley Communications enabler uses synchronous requests during device initialization and during setpoint operations. By default, time-out for a synchronous request is 5 seconds. You can use this global parameter to increase or decrease the time-out value.

To change the default time-out, add the following global parameter:

ABETH_PLC_REQUEST_TIMEOUT

For	Project
Purpose	To change the default time-out for a synchronous request.
Value	n (where n is a value greater than 0)

Default Value	5

ABETH_PLC_RESPONSE_TIMEOUT

When the queue for point polls is nearly full the Allen-Bradley Communications enabler waits for responses to the poll requests. The default wait time is 1 second. You can use this global parameter to increase the wait time.

To change the default wait time, add the following global parameter:

ABETH_PLC_RESPONSE_TIMEOUT

For	Project
Purpose	To change the default wait time for poll request responses.
Value	n (where n is a value greater than 0)
Default Value	1

ABETH_UNSO_QUEUE_SIZE

When the Allen-Bradley Ethernet Communications driver is used, the unsolicited data received from the PLC is held in a queue.

To change the default queue size, add the following global parameter.

ABETH_UNSO_QUEUE_SIZE

For	Project
Purpose	To change the size of the unsolicited data queue.
Value	n (where n is a value greater than or equal to 500)
Default Value	500

Note: When you change the queue size, a message stating "PLC Queue Size set to n" is logged to the cor_recstat file.

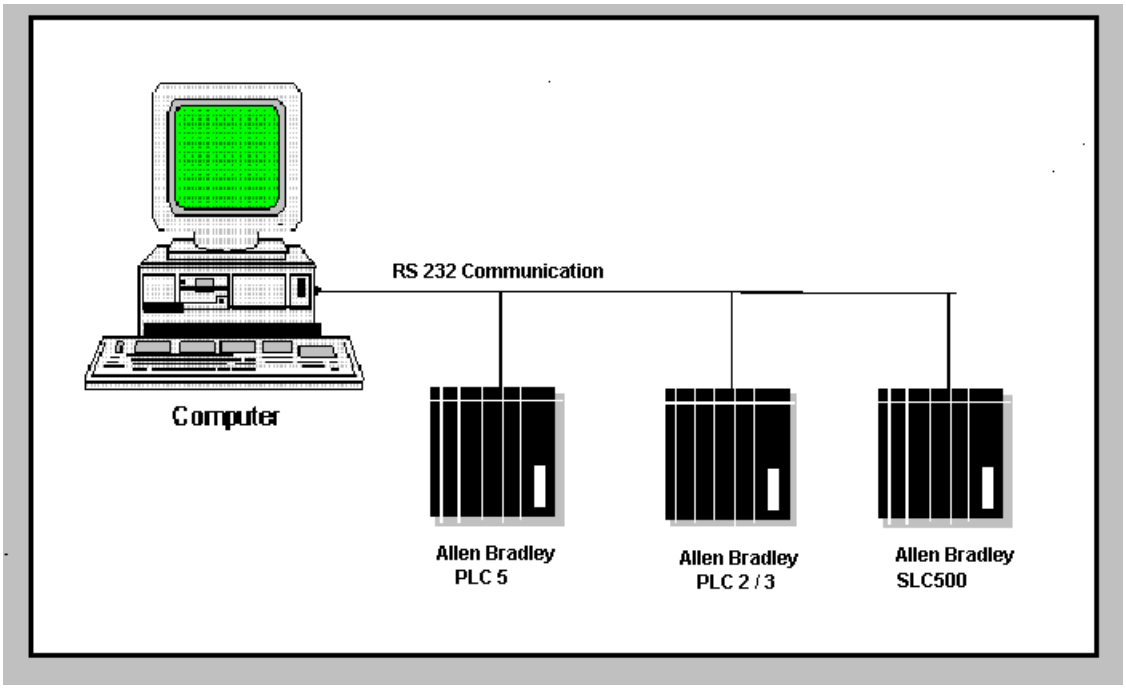
Chapter 9. Allen-Bradley DF-1 Communications

About Allen-Bradley DF-1 Communications

The CIMPLICITY Allen-Bradley DF-1 Communication Enabler can be used to communicate with Allen-Bradley Programmable Controllers through the DF-1 serial communication protocol.

Using this option, CIMPLICITY software can:

- Poll points at a user-defined scan rate or on demand.
- Read and write single points or arrays.
- Perform Engineering Unit conversion for meaningful point value displays
- Issue alarms to user when a device goes down.



Set up Allen-Bradley DF-1 Communications

Set up Allen-Bradley DF-1 Communications

After installing CIMPLICITY software's Allen-Bradley DF-1 communication enabler, you are ready to set up Allen-Bradley DF-1 Communications:

Step 1 *(page 121)*	Run the DF-1 Setup Utility.
Step 2 *(page 124)*	Configure the Allen-Bradley DF-1 Application.

Step 1. Run the DF-1 Setup Utility

Step 1. Run the DF-1 Setup Utility

1. Click Start on the Windows task bar.
2. Select All Programs>HMI SCADA - CIMPLICITY<version>Allen Bradley DF-1 Setup Utility.

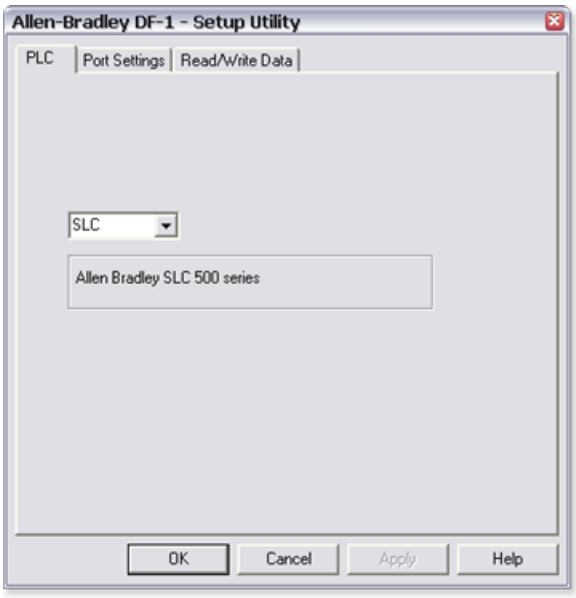
You can perform the following functions:

Step 1.1 *(page 121)*	Set up PLC Properties
Step 1.2 *(page 122)*	Set up Serial Port Properties.
Step 1.3 *(page 124)*	Test Communications (Read/Write).

For this program to function, CIMPLICITY software's Allen-Bradley DF-1 communication enabler must be successfully installed. To start the installation verification program, select the DF-1 Setup icon in the CIMPLICITY menu.

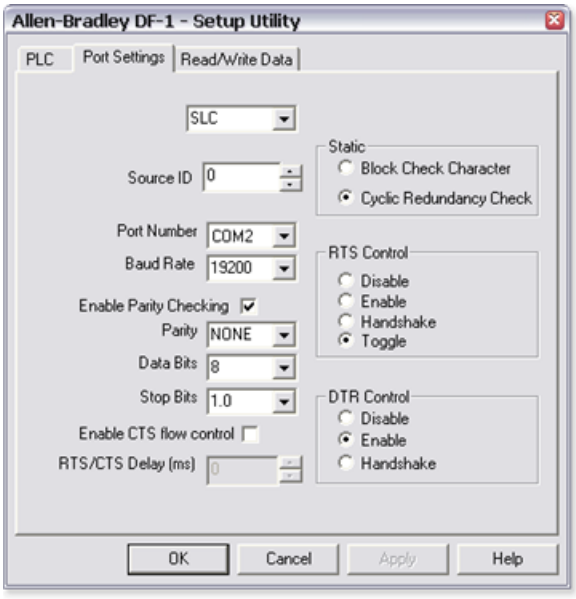
Step 1.1. Set up PLC Properties

When the program starts, you must select the Model type of PLC from the drop-down list box. The list is on the PLC tab of the Setup Utility dialog box.



Step 1.2. Setup Serial Port Properties

Use the field on the Port Settings tab of the Setup Utility dialog box, to configure the serial port characteristics:

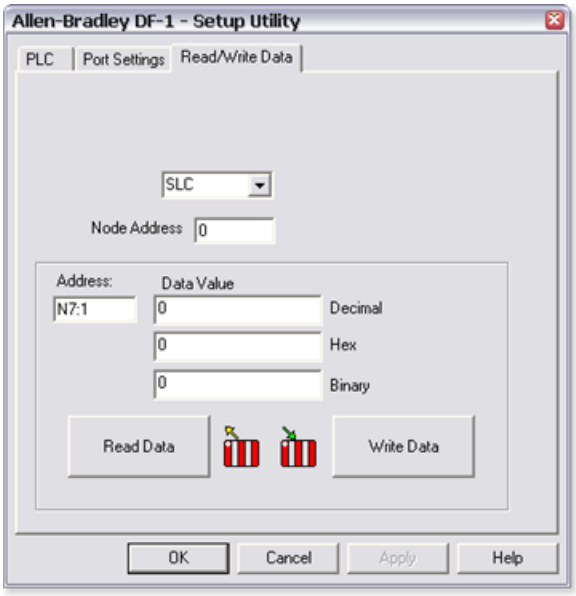


rect 194, 234, 307, 306 *(page 123)*
rect 191, 147, 321, 230 *(page 123)*
rect 191, 81, 339, 141 *(page 123)*
rect 25, 258, 189, 305 *(page 123)*
rect 22, 178, 187, 258 *(page 123)*
rect 50, 153, 181, 174 *(page 123)*
rect 51, 133, 185, 155 *(page 123)*
rect 63, 97, 189, 119 *(page 123)*

Source ID	Enter the station number of the DF-1 device.
Port Number	Select the COM port attached to the device.
Baud rate	Enter the baud rate for communications from the drop-down list.
Enable Parity Checking	Check to enable parity checking and activate **Parity** fields below.
Parity	Select the parity to be used for communications.
Data Bits	Select the number of data bits in the data from the list.
Stop Bits	Number of stop bits.
Enable CTS flow control	Check to enable Clear to Send flow control and activate the **RTS/CTS Delay** field below.
RTS/CTS Delay (ms)	Enter the delay desired between RTS and CTS.
Static	Static error test. Can be a Block Check character (BCC) type or Cyclic Redundancy Check (CRC) type.
RTS Control	Request to Send flow control. Default is Toggle.
DTR Control	Data Terminal Ready signal. Default is Enable.

Step 1.3. Test Communications (Read/Write Data)

After configuring the port parameters the communication can be tested with the Read/Write Data tab of the Setup Utility dialog box.



rect 201, 254, 299, 298 *(page 124)*
rect 29, 251, 136, 298 *(page 124)*
rect 87, 165, 238, 251 *(page 124)*
rect 19, 165, 84, 210 *(page 124)*
rect 91, 102, 175, 124 *(page 124)*

Node Address	Enter the same as the **Source ID** on the Port Settings tab.
Address	Enter the address to be tested.
Data Value	Enter data value in decimal to be written on the PLC.
Read Data	Reads data from the address specified in the **Address** field and updates the **Data Value** field.
Write Data	Writes the data from the **Data Value** field to the address in the **Address** field.
OK / Cancel	Closes the application.
Help	Provides the online help about the Setup utility.

Step 2. Configure CIMPLICITY for Allen-Bradley DF-1

Step 2. Configure CIMPLICITY for Allen-Bradley DF-1

When configuring ports, devices, and points that use the DF-1 enabler, some fields must contain unique values for the communications to work successfully.

Step 2.1 *(page 125)*	Configure Allen-Bradley DF-1 Ports.
Step 2.2 *(page 128)*	Configure Allen-Bradley DF-1 Devices.
Step 2.3 *(page 130)*	Configure Allen-Bradley DF-1 Points.

Step 2.1. Configure Allen-Bradley Df-1 Ports

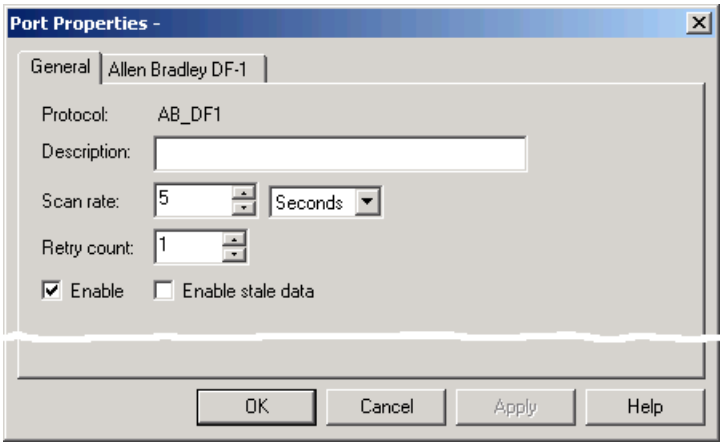
Step 2.1. Configure Allen-Bradley Df-1 Ports

When you configure a port for DF-1, enter the following in the New Port dialog box:

1. In the **Protocol** field, select AB-DF-1 from the list of available protocols.
2. In the **Port** field, select the communication port that will be used for the Allen-Bradley DF-1 enabler.
3. When you click OK, the Port Properties dialog for the protocol opens.

Step 2.1.1 *(page 125)*	Set general port properties.
Step 2.1.2 *(page 126)*	Set Allen-Bradley DF-1 port properties.

Step 2.1.1. Set General Port Properties

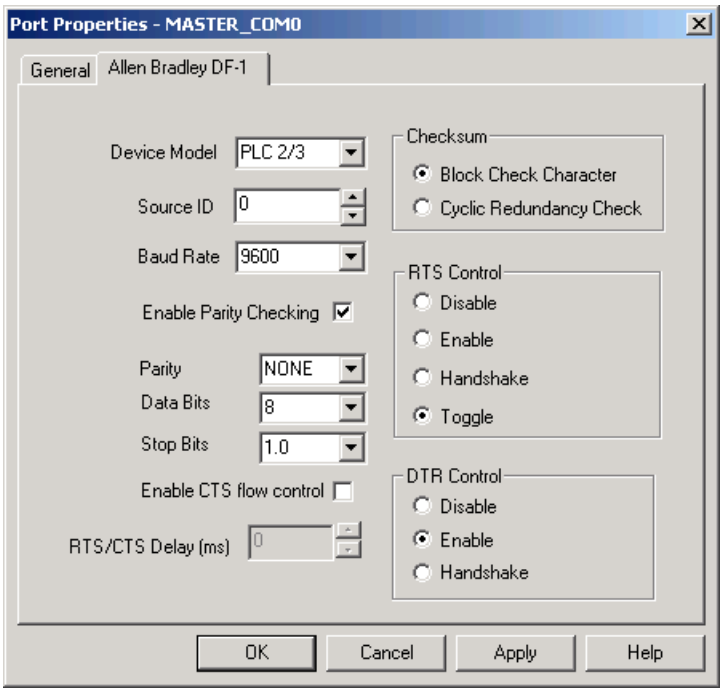


rect 22, 83, 329, 104 *(page 126)*
rect 23, 112, 237, 133 *(page 126)*
rect 23, 141, 153, 161 *(page 126)*
rect 21, 168, 80, 187 *(page 126)*
rect 92, 170, 197, 187 *(page 126)*

Use the General tab of the Port Properties dialog box to enter general information for the port. You can define the following:

Description	Enter an optional description to help identify the port.
Scan Rate	Enter the base scan rate for the port. Point scan rates will be multiples of the base rate. You can specify the scan rate in Ticks (100 ticks = 1 second), Seconds, Minutes, or Hours.
Retry Count	If communications cannot be established for a device on the port, the device is considered to be down, and a $DEVICE_DOWN alarm is generated. Once a device is down, periodic attempts are made to resume connections to it. Enter the number of scans to wait before attempting to reconnect to a device after a communications error has been detected.
Enable	Check if the port is to be enabled when the project starts. If you clear this check box, the port will not be enabled when the project starts, and points will not be available for devices on the port.
Enable stale data	Check to keep the point available in most circumstances that would have made it unavailable. However the point value will be stale, meaning it is the last known value and may or may not have changed in the device.

Step 2.1.2. Set Allen-Bradley DF-1 Port Properties



rect 64, 82, 226, 102 *(page 127)*
rect 80, 114, 227, 137 *(page 127)*
rect 81, 146, 227, 168 *(page 127)*
rect 85, 182, 225, 289 *(page 127)*
rect 38, 292, 221, 348 *(page 128)*
rect 242, 76, 412, 142 *(page 128)*
rect 243, 160, 412, 270 *(page 128)*
rect 239, 285, 408, 372 *(page 128)*

Use the Allen Bradley DF-1 tab of the Port Properties dialog box to enter Allen Bradley DF-1 protocol information for the port. You can also define these settings using the independent test utility.

Device Model	Selected model fills in default values for the Allen Bradley DF-1. Models include: • PLC 2/3 • PLC 5 • SLC
Source ID	Enter the station number of the DF-1 device.
Port Number	Select the COM port connected to the device.
Baud rate	Enter the baud rate for communications. Click the drop-down list to the right of the input field and select a rate from the list.
Enable Parity Checking	Select to enable parity checking and activate the **Parity** fields below.
Parity	Select the parity to be used for communications.
Data Bits	Select the number of data bits.

Stop Bits	Select the number of stop bits.
Enable CTS flow control	Select to enable Clear to Send flow control and activate the **RTS/CTS Delay** field.
RTS/CTS Delay (ms)	Enter the delay desired between RTS and CTS.
Checksum	Select the Checksum method to use. Your selection must match the PLC.
RTS Control	Select the RTS Control option to use. The default is Toggle.
DTR Control	Select the DTR Control option to use. The default is Enable

These settings will override those from the independent test utility *(page 121)* .

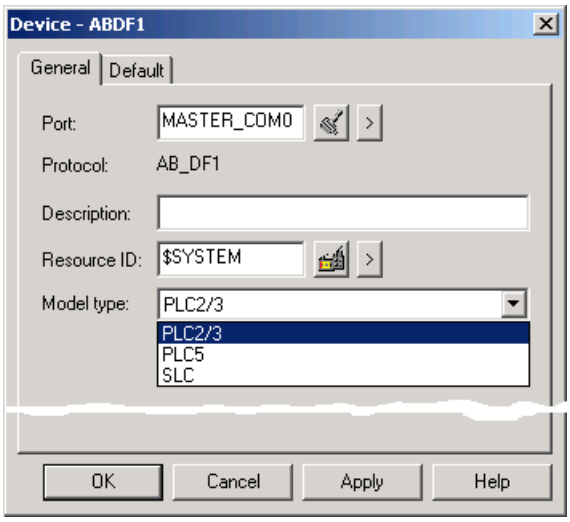
Step 2.2. Configure Allen-Bradley DF-1 Devices

Step 2.2. Configure Allen-Bradley DF-1 Devices

1. In the **Device** field, enter the name of the device you are configuring.
2. In the **Port** field, select the Serial Communications port to be used for the device.
3. When you click OK, the Device Properties dialog box for devices using this protocol opens.

Step 2.2.1 *(page 128)*	Set general device properties.
Step 2.2.2 *(page 129)*	Set default device properties.

Step 2.2.1. Set General Device Properties

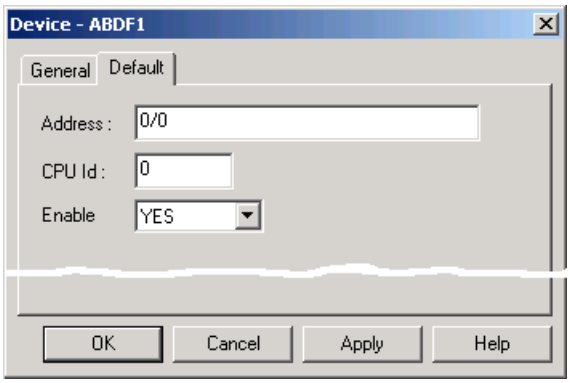


rect 20, 62, 236, 83 *(page 129)*
rect 21, 116, 330, 140 *(page 129)*
rect 23, 143, 237, 170 *(page 129)*
rect 23, 176, 331, 240 *(page 129)*

Use the General tab of the Device dialog box to enter general information for the device. You can define the following:

Port	Select the port for this device. You can click the Port button to the right of the field to display the list of ports and select one. You can click the Pop-up Menu button to create a new port, edit the current port, or select a port from the list of ports.
Description	Enter an optional description to help identify the device
Resource ID	Enter a resource that can be associated with this device. You can click the Resource button to the right of the field to display the list of resources and select one. You can click the Pop-up Menu button to create a new resource, edit the current resource, or select a resource from the list of resources.
Model Type	Select the type of device. Click the drop-down button to the right of the input field to display your choices, then make a selection. For Allen-Bradley DF-1, the choices are: • PLC2/3 • PLC5 • SLC

Step 2.2.2. Set Default Device Properties



rect 21, 62, 297, 84 *(page 130)*
rect 24, 93, 144, 115 *(page 130)*
rect 23, 119, 162, 141 *(page 130)*

Use the Default tab of the Device dialog box to enter information about DF-1 communications for the device.

Address	Enter the node ID for the device. • For PLC 2 or PLC 3 this is an optional Station Number Address used only when writing to a PLC 2 via a KA or KG module. If this field is not used, it should be set to zero. • For PLC 5 this is an address consisting of two fields as follows: <Octal digit station address>/< Octal digit station address of the KF2 Module> • For SLC this is an address consisting of one field as follow: <Source ID(00-31)> **Note:** The Source ID is also knows as the PLC Node address. For special network configurations a KF3 Node address can be used.
CPU ID	Not used.
Enable	Select Yes to enable the device when the project starts. Select No to disable the device. If you select No, the device will not be enabled at startup and points associated with the device will not be available.

Step 2.3 Configure Allen-Bradley DF-1 Points

Step 2.3 Configure Allen-Bradley DF-1 Points

When defining a point, set the following properties. These properties have values that are unique, to or have special meaning for, DF-1 communications.

Step 2.3.1 *(page 131)*	Set general point properties.
Step 2.3.2 *(page 131)*	Set device point properties.

Step 2.3.1. Set General Point Properties

On the General tab in the Point Properties dialog box, the type of access you choose depends on the point address:

- Check the Read Only checkbox for points from Discrete Inputs or Input Registers.
- Clear this checkbox (i.e. Read/Write access) for points from Coils or Holding Registers.

Step 2.3.2. Set Device Point Properties

On the Device tab of the Point Properties dialog box, you may configure:

Address	Point address requirements are device-dependent. Valid device types are PLC2/3, PLC5, and SLC. See the sections below for detailed information on the point address requirements for each device type. **Important:** The CIMPLICITY point addressing format does not include the I/O card rack slot position as part of the address.
Update Criteria	The update criteria determines how the data will be requested.
	Select On Change or On Scan for points whose values should be polled by the Allen-Bradley DF-1 Communications enabler at regular intervals.

When configuring Boolean points, you must also enter data in the following field:

Bit Offset	Enter the bit offset that corresponds to the bit position within the word.
	The valid range for bit offset is 0 - 15, where 0 is the least significant bit. The bit offset must be specified as a decimal value.

When configuring array points, make sure that the size of the array is within the limits of the file. CIMPLICITY software does not declare an array configured beyond the end of a file as invalid. If an array point is configured that extends beyond the end of a file, a $DEVICE_DOWN alarm will be generated each time an attempt is made to read data from or write data to the point.

Allen-Bradley DF-1 Technical Notes

Allen-Bradley DF-1 Technical Notes

Refer to the technical notes below for reference information regarding Allen-Bradley DF-1:

- Allen-Bradley DF-1 supported devices.
- Allen-Bradley DF-1 supported memory types.

- Allen-Bradley DF-1 global parameters.

Allen-Bradley DF-1 Supported Devices

Allen-Bradley DF-1 Supported Devices

This communication enabler supports the following Allen-Bradley Programmable Controllers:

- PLC 2 / PLC 3.
- PLC 5
- SLC 500 Series.

PLC2 and PLC3 Address Format

PLC 2 and PLC 3 devices support the following address format:

File Type	Address Format
BN, B3, B4 BT	<HIGHWAY NUMBER><FILE TYPE><ELEMENT>
	Where <HIGHWAY NUMBER> is an octal number.
	Where <FILE TYPE> is:
	The <ELEMENT> is the internal address.

PLC 5 Address Format

PLC-5 devices support the following address formats:

File Type	Address Format
N, F, B	<FILE><FILE NUMBER>:<ELEMENT>
	The use of leading zeros when specifying the <FILE NUMBER> and the <ELEMENT> is not supported.
I, O, S	<FILE>:<ELEMENT >
	Where <FILE> is:
T, C	<FILE><FILE NUMBER>:<ELEMENT>.<TYPE>

File Type	Address Format
	Where <FILE> is:
	and <TYPE> is:

Bit offsets for the Control Word (CTL) in Counters and Timers are:

Counters Control Bit	Offset
CU	15
CD	14
DN	13
OV	12
UN	11

SLC Address Format

SLC 5/04 devices support the following address formats:

File Type	Address Format
`N, F, B`	<FILE><FILE NUMBER>`:`<ELEMENT> **Note:** The use of leading zeros when specifying the <FILE NUMBER> and the <ELEMENT> is not supported.
`I, O, S`	<FILE>`:`<ELEMENT>
	Where <**FILE**> is:
`T, C`	<FILE><FILE NUMBER>`:`<ELEMENT>`.`<TYPE>
	Where <**FILE**> is:
	and <TYPE> is:

Bit offsets for the Control Word (CTL) in Counters and Timers are:

Counters	
Control Bit	**Offset**
`CU`	`15`

Counters	
Control Bit	**Offset**
CD	14
DN	13
OV	12
UN	11

Allen-Bradley DF-1 Supported Memory Types

Data may be read and written to the following memory types.

- Analog Inputs
- Analog Outputs
- Integer Files
- Timers
- Counters
- Bits
- Status Registers
- Files

Allen-Bradley DF1 Global Parameters

This global parameter allows you to configure the use of single precision floating point values when accessing an Allen-Bradley device. A value of Y will configure ABDF-1 to use single precision floating point values on the specified port name, which is required by some devices (e.g. SLC devices). By default, the value is N, which configures the ABDF-1 to use double precision floating points, which is required by some devices (e.g. PLC-5 devices).

Note: if this global is configured to Y, a message will be written to the Project Status Logs indicating that single precision floating point values have been selected for the channel listed in the message.

ABDF1_<portname>_USESPFP

For	Project
Purpose	To configure single precision floating points.
Value	Y
Default Value	N

Chapter 10. Allen-Bradley DF1 Setup Utility

About the Allen Bradley DF-1 Setup Utility

The DF-1 Setup Utility Program is a program provided with the Allen-Bradley DF-1 communication enabler that you can use to check the basic configuration and operation of the communication path without starting CIMPLICITY software This test utility is normally installed in C:\Program Files \Proficy\Proficy CIMPLICITY\EXE directory.

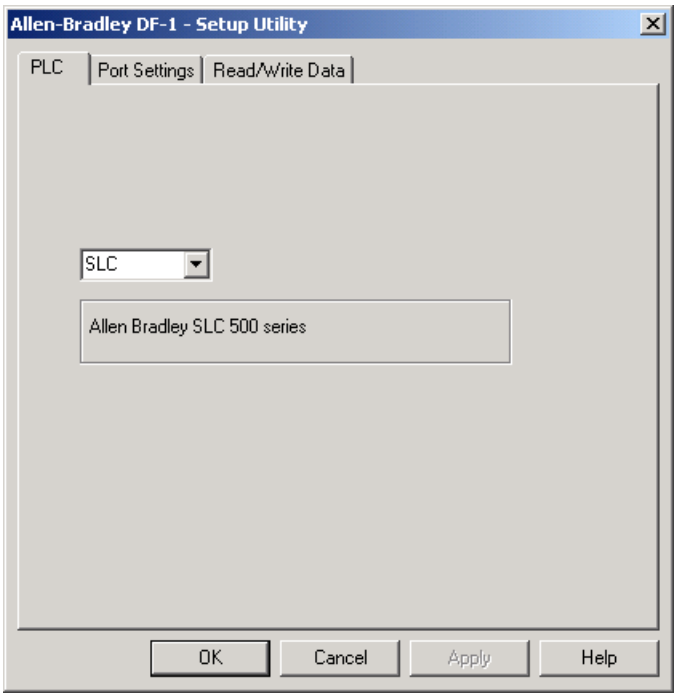
You can perform the following functions:

- Configure Advanced Port Settings
- Read test
- Write test

For this program to function, CIMPLICITY software's Allen-Bradley DF-1 communication enabler must be successfully installed. To start the installation verification program, select the DF-1 Setup icon in the CIMPLICITY menu.

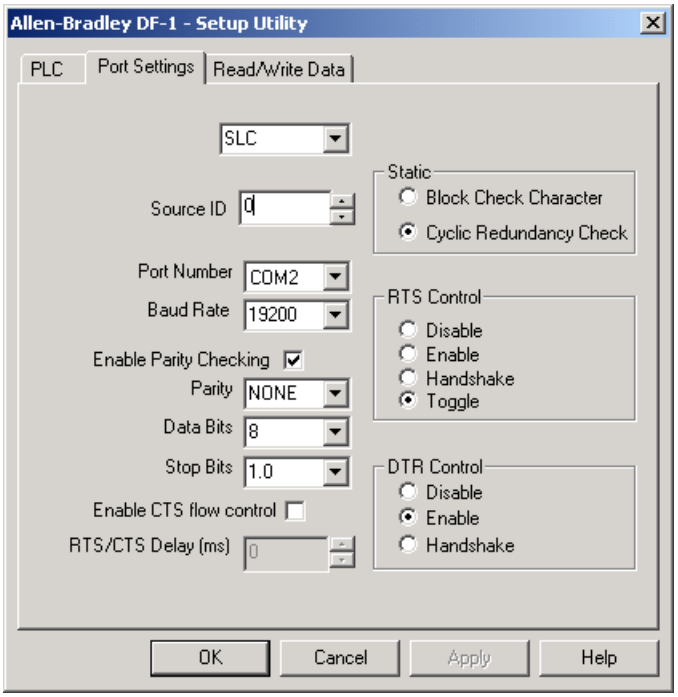
PLC Setup Utility Properties

When the program starts, you must select the Model type of PLC from the drop-down list box. The list is on the PLC tab of the Setup Utility dialog box.



Port Setting Setup Utility Properties

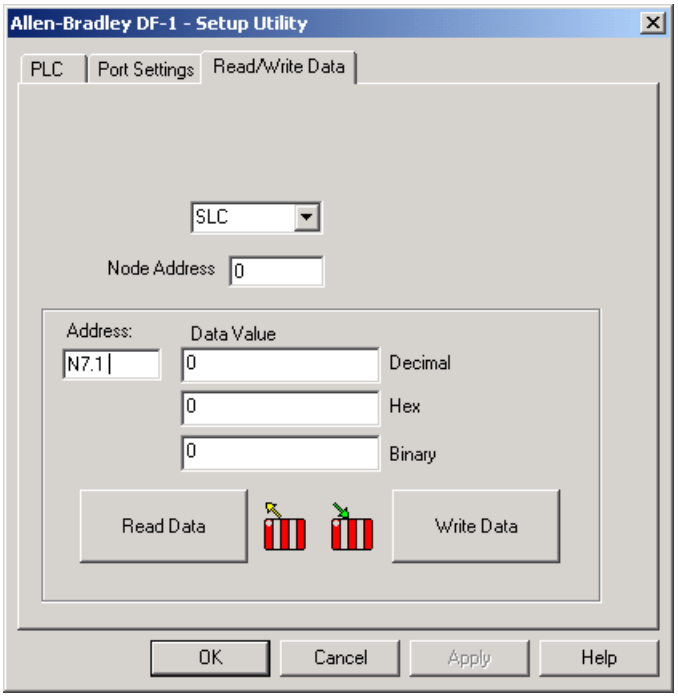
Use the field on the Port Settings tab of the Setup Utility dialog box, to configure the serial port characteristics:



Source ID	Enter the station number of the DF-1 device.
Port Number	Select the COM port attached to the device.
Baud rate	Enter the baud rate for communications from the drop-down list.
Parity	Select the parity to be used for communications.
Data Bits	Select the number of data bits in the data from the list.
Stop Bits	Number of stop bits.
Error Type	Can be a Block Check character (BCC) type or Cyclic Redundancy Check (CRC) type.
RTS Control	Default is Toggle.
DTR Control	Default is Enable.

Read/Write Data Setup Utility Properties

After configuring the port parameters the communication can be tested with the Read/Write Data tab of the Setup Utility dialog box.



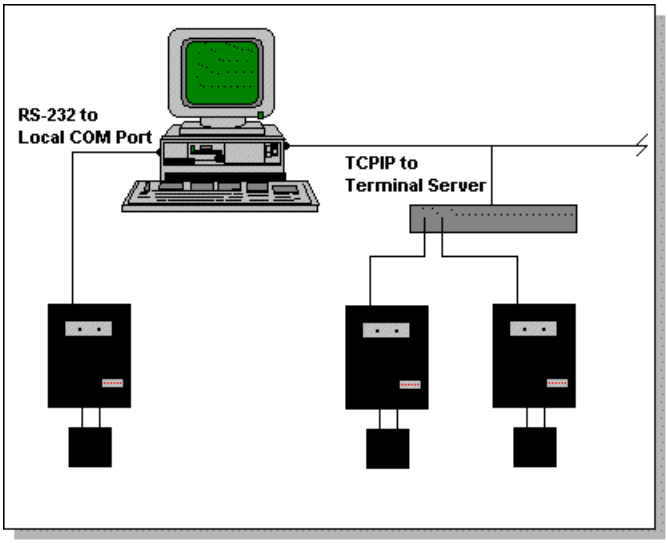
Node Address	Enter the same as the **Source ID** on the Port Settings tab.
Address	Enter the address to be tested.
Data Value	Enter data value in decimal to be written on the PLC.
Read Data	Reads data from the address specified in the **Address** field and updates the **Data Value** field.

Write Data	Writes the data from the **Data Value** field to the address in the **Address** field.
OK / Cancel	Closes the application.
Help	Provides the online help about the Setup utility.

Chapter 11. Allen-Bradley Intelligent Antenna Communications

About Allen-Bradley Intelligent Antenna Communications

The Allen-Bradley Intelligent Antenna Communications enabler uses the computer's local serial ports, or a remote terminal server to communicate with Allen-Bradley Intelligent Antenna devices.



The Allen-Bradley 2750-AS series antenna is one component of a Radio Frequency Identification (RFID) system. Antennas collect information from RF tags attached to objects. There are three basic types of RF tags:

- Read Only Tags
- Read/Write Tags
- Programmable Tags.

The Allen-Bradley Intelligent Antenna Communications enabler supports read access to the byte addressable memory of the all three RF Tag types (2-8K bytes of data depending on model). It can also write to the Read/Write Tags and the Programmable Tags. The Program Tag transaction is not supported.

The Allen-Bradley Intelligent Antenna Communications enabler also supports the automatic download of antenna setup parameters during startup and re-establishment of communications. The setup parameters are configurable for each individual device (antenna).

Communications Enabler Features

Using this option, CIMPLICITY software can:

Detect the presence of a Tag within the antenna's range

- Read Tag data at a user-defined scan rate
- Write data to the Tag
- Issue alarms to users when a read/write error occurs
- Monitor antenna configuration through diagnostic points

This enabler can:

- Support an infinite number of Intelligent Antennas in any combination of local serial ports and/or remote terminal servers.

This enabler supports the following CIMPLICITY features:

- Polled read at user defined rates
- Poll after setpoint
- Triggered reads
- Alarm on communications failure

This enabler supports the following data types:

- Signed 8, 16, and 32 bit integers
- Unsigned 8, 16, and 32 bit integers
- Floating point
- Text
- Arrays

Allen-Bradley Intelligent Antenna Supported Data Types

The Allen-Bradley Intelligent Antenna communicates supports the following tag types:

Tag Type	Access type
0	Read
16	Read/Write
17	Read/Write

32	Read/Write
33	Read/Write
34	Read/Write

Allen-Bradley Intelligent Antenna Related Documents

You should have the following documentation available when configuring this interface option:

Allen-Bradley Intelligent Antenna User Manual (Cat. No. 2750-ND002)

This document is shipped with each Intelligent Antenna. It discusses the issues involved in configuring the communication and operation parameters of the Intelligent Antenna.

Allen-Bradley Intelligent Antenna Configuration

Configure the Allen-Bradley Intelligent Antenna with the following communication and operational parameters.

Physical Protocol	RS-232
Baud Rate	9600
Parity	None
Communication Mode	Byte swapping mode Enabled

See the Allen-Bradley Intelligent Antenna User Manual for details on setting these parameters

Allen-Bradley RFID Diagnostic Program

You can use the AB RFID Diagnostics program, **abrfid_diag.exe**, to verify hardware installation and communication operation.

To run the AB RFID diagnostics, execute the following command in a Command Prompt window opened from the CIMPLICITY project's Configuration cabinet:

```
abrfid_diag.exe <port> <baud_rate> [<init_flag> <counter>]
```

where:

< **port** > is the name of the port connected to the connection cable.

< **baud_rate** > has a default value of 9600.

< **init_flag** > is optional to let the program initialize antenna setting only.

< **counter** > is optional to indicate number of read operations that should be invoked. Default value of counter is 20.

For example, executing the following command initializes the antenna setting only:

```
abrfid_diag.exe COM2 9600 i
```

The following command reads data for three points:

```
abrfid_diag.exe COM2 9600 10
```

After the command is issued, the program will ask:

```
Do you like to test write tag command?(y/n)
```

Answer **Y** to allow the program to write data to the tag.

The program then will ask:

```
Please enter the number of read tag operations invoked before each write
```

Enter the number of operations you want.

When the program is ready to write data to tag, user will be asked for the data to be written.

The following is sample output for the program:

```
abrfid_diag parameters: Port = COM1, baud_rate = 9600, repeat count = 10
Do you like to test write tag command? (y/n)
Y
Please enter number of read tag operations invoked before each write:
1
!!!!!use echo command to test the communication link to the antenna.
The antenna echoed: Hello!
!!!!!echo command finished.
Counter = 1
Read data for R0, element = 1.
Operation failed. Possible cause – no tag present.
   Data from antenna:
Read data from R0, element = 6.
   Data from antenna: 016405
Read data from R1, element = 2.
   Data from antenna: 16
Please input 2 bytes of data to write to tag memory:
```

```
11
Write data to R1, element = 2.
Write tag command: W 1 002 11
Tag detected, operation failed. Return code: 9
Possible cause – RF power low, wrong tag type.
Write data for point R1 failed.
…
abrfid_diag program finished successfully.
```

CIMPLICITY Configuration for Allen-Bradley Intelligent Antenna

CIMPLICITY Configuration for Allen-Bradley Intelligent Antenna

When you configure ports, devices, and points that use Allen-Bradley Intelligent Antenna Communications enabler, some fields must contain unique values for the communications to work successfully.

Allen-Bradley Intelligent Antenna Port Configuration

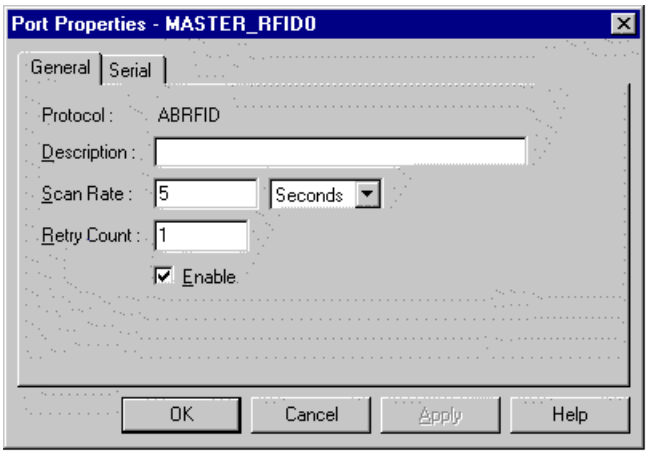
Allen-Bradley Intelligent Antenna Port Configuration

When you configure a new port for Allen-Bradley Intelligent Antenna communications, enter the following information in the New Port dialog box:

1. In the **Protocol** field, select **ABRFID** from the list of protocols.
2. In the **Port** field, select the communication port that will be used for Allen-Bradley Intelligent Antenna communications.

   When you click **OK** to create the port, the Port Properties dialog box opens.

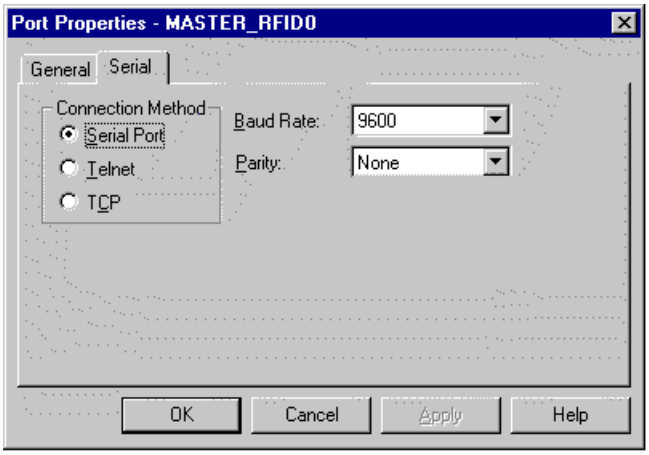
General Port Properties



Use the General tab in the Port Properties dialog box to enter general information for the port. You can define the following:

Description	Enter an optional description to help you identify the port.
Scan Rate	Enter the base scan rate for the port. Point scan rates are multiples of the base rate.
	You can specify a scan rate in Ticks (100 ticks = 1 second), Seconds, Minutes, or Hours.
Retry Count	If communications cannot be established, the device is considered to be down and a $DEVICE_DOWN alarm is generated. Once a device is down, periodic attempts are made to resume connections to it.
	Enter the number of scans to wait before attempting to a device on this port after a communications error is detected.
Enable	Set this check box if you want the port to be enabled when the project starts. If you clear this check box, the port will not be enabled when the project starts, and points will not be available for devices on the port.

Serial Port Properties - Serial Connection

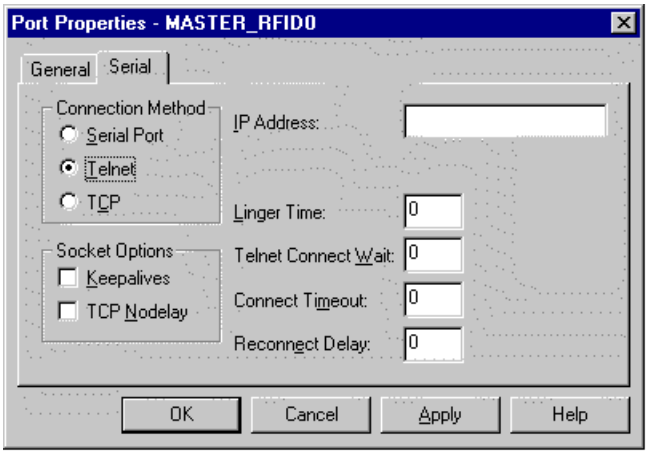


If you check the **Serial Port** connection method on the Serial tab in the Port Properties dialog box, you need to define the following:

Baud Rate	Enter the baud rate for communications. Click the drop-down list to the right of the input field and select a rate from the list.
Parity	Select the parity to be used for communications.

Remember that you must configure the same baud rate, data bits, parity, stop bits and flow control for all PLCs using the serial port.

Serial Port Properties - Telnet or TCP Connection



If you select the **Telnet** or **TCP** connection method on the Serial tab in the Port Properties dialog box, you need to define the following:

Socket Options	Check the **Keepalives** check box to use keepalives to detect the loss of the terminal server. Clear the check box if you do not want to use keepalives to detect the loss of the terminal server.
	Check the **TCP Nodelay** check box if you want to set the Nodelay flag on the socket. Clear the check box if you do not want to set the Nodelay flag.
IP Address	Enter the IP address of the terminal server.
Linger Time	Enter the time in seconds to wait after closing the socket before aborting the socket.
Telnet Connect Wait	Enter the time in seconds to wait for the Telnet protocol to initialize.
Connect Timeout	Enter the time in seconds to wait for the TCP connection to form.
Reconnect Delay	Enter the time in seconds to wait before attempting to reconnect to a device.
	If you set this value to zero and the terminal server is not available, no attempts will be made to reconnect to the terminal server.

Allen-Bradley Intelligent Antenna Device Configuration

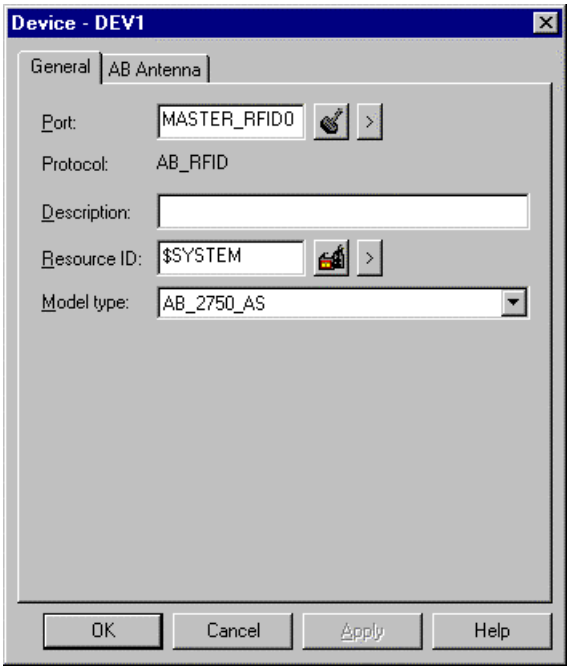
Allen-Bradley Intelligent Antenna Device Configuration

When you create a new device for Allen-Bradley Intelligent Antenna Communications, enter the following information in the New Device dialog box:

1. In the **Device** field, enter the name of the device you are configuring.
2. In the **Port** field, select the ABRFID port to be used by the device.

   When you click **OK** to create the port, the Device Properties dialog box opens.

General Device Properties

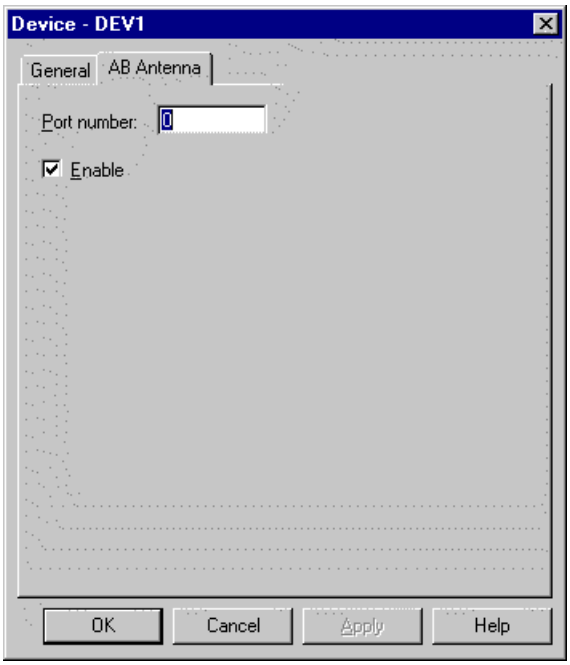


Use the General tab in the Device dialog box to enter general information for the device. You can define the following:

Port	Select the port for this device.
	You can click the Port button to the right of the field to display the list of ports and select one.
	You can click the **Popup Menu** button to create a new port, edit the current port, or select a port from the list of ports.

Description	Enter an optional description to help you identify the device.
Resource	Enter a resource that can be associated with this device for alarm generation.
	You can click the Resource button to the right of the field to display the list of resources and select one.
	You can click the **Popup Menu** button to create a new resource, edit the current resource, or select a resource from the list of resources.
Model Type	There is only one model, **AB_2750-AS**.

Default Device Properties



Use the Default tab in the Device dialog box to enter information about Allen-Bradley Intelligent Antenna communications for the device. You can define the following:

Port number	For serial port connections, enter the local port number
	For TCP or Telnet connections, enter the TCP port number on the terminal server where this antenna resides.
Enable	Select this check box to enable the device when the project starts. Clear this check box to disable the device. Points associated with the device will be unavailable.

Allen-Bradley Intelligent Antenna Point Configuration

Allen-Bradley Intelligent Antenna Point Configuration

When you define a point, the following fields have values that are unique to or have special meaning for Allen-Bradley Intelligent Antenna communications:

General Point Properties

On the General tab in the Point Properties dialog box, you may configure points for **Read** or **Read/ Write** access as follows:

Tag Type	Range	Access type
0	6-digit	Read
	20 or 40 char	
16	0 - 2047	Read/Write
17	0 - 8191	Read/Write
32	0 - 6	Read/Write
33	0 – 20	Read/Write
34	0 - 40	Read/Write

Note: For tag type 0, there are 3 possible ranges. Points must be configured within the memory range of the tag, otherwise read operations might fail and the communication with the tag would die.

Intelligent Antenna

Intelligent Antenna

The Allen-Bradley Intelligent Antenna contains four configuration settings. These four settings are: Configuration

- Tag Type
- Object Detect Mode
- Object Detect Timeout
- RF Field Strength

CIMPLICITY software stores these four settings for every device (antenna) in your project in a standard Windows INI file named **ABRFID.INI**. This file resides in the **%SITE_ROOT%** (project root) directory.

Every time you start your CIMPLICITY project, the Allen-Bradley Intelligent Antenna communications enabler queries each antenna for its current configuration settings. It then compares these values against those in the **ABRFID.INI** file.

If the configuration settings for a particular antenna do not exist in the **ABRFID.INI** file (or the **ABRFID.INI** file does not exist), the file is updated (or created) with the antenna's current configuration settings.

If the configuration settings for a particular antenna exist in the **ABRFID.INI** file, and the four values match the current antenna configuration, then no further action is taken.

If the configuration settings for a particular antenna exist in the **ABRFID.INI** file, and any of the four values do not match the antenna's current configuration, then the values from the **ABRFID.INI** file are written to the antenna, updating the antenna's configuration.

Updating an Antenna's Configuration File

You can override the antenna's default configuration by maintaining the values in the **ABRFID.INI** file. As with any Windows INI file, modifications are made with a text editor (such as Notepad). Use the following procedure to make changes to the **ABRFID.INI** file:

1. From the Configuration cabinet for your CIMPLICITY project, select **Command Prompt** from the **Tools** menu.

   The Command Prompt window opens.

2. In the Command Prompt window, type NOTEPAD ABRFID.INI

   The contents of the file will look similar to the lines below, depending upon the names you have chosen for the Allen-Bradley Intelligent Antenna devices in your project:

```
[ANTENNA1]
TagType=16
ObjectDetectMode=1
ObjectDetectTimeout=1000
RFFieldStrength=3
[ANTENNA2]
TagType=16
ObjectDetectMode=1
ObjectDetectTimeout=1000
RFFieldStrength=3
[ANTENNA3]
```

```
TagType=16
ObjectDetectMode=1
ObjectDetectTimeout=1000
RFFieldStrength=3
```

The text inside the square brackets identifies which device (antenna) the settings are for. This is the name you gave the device during Device Configuration. The values correspond to those from table 6.B in the Allen-Bradley Intelligent Antenna User's Manual. Please refer to the user manual for the definitions of these values.

After you complete the modifications you need to make, and save the **ABRFID.INI** file, close Notepad and then close the Command Prompt window.

Updating the Antenna Configuration Settings

There are two methods of getting the Allen-Bradley Intelligent Antenna communications enabler to update the antenna's configuration settings based on the modifications to the **ABRFID.INI** file.

- You can stop and restart the CIMPLICITY project. This is usually not a feasible thing to do.
- You can update the configuration settings dynamically for a particular antenna, without affecting the flow of data from the rest of the antennas in your project:

To updated the configuration settings dynamically, use the following procedure:

1. In the project's Configuration cabinet, select **Dynamic** from the Tools menu. You can now change the device properties without having to stop and restart the project.
2. In the project's Configuration cabinet, double-click on the Devices icon. The Devices window opens and shows you a list of all of the devices (antennas) in your project.
3. From the list of devices in the Devices window, double click on the antenna you wish to update. This opens the Device Properties dialog box.
4. In the Device Properties dialog box for the antenna:
5. Clear the **Enabled** check box.
6. Click **Apply**.
7. Check the **Enabled** check box.
8. Click **OK** to close the dialog box.
9. Close the Devices window.

Viewing an Antenna's Current Configuration

Viewing an Antenna's Current Configuration

You can configure Diagnostic Points for each antenna in your CIMPLICITY project to display an antenna's current configuration values.

Diagnostic Points are configured the same as any other device point, except that a special address value is used, and the **Diagnostic Data** checkbox is selected. The four special addresses you will use to create these Diagnostic Points are:

Configuration Value	Point Address	Point Type
Tag Type	**$TAG_TYPE**	**USINT**
Object Detect Mode	**$OBJECT_DETECT_MODE**	**USINT**
Object Detect Timeout	**$OBJECT_DETECT_TIMEOUT**	**UINT**
RF Field Strength	**$RF_FIELD_STRENGTH**	**USINT**

Configuring a Diagnostic Point

1. Open the New Point dialog box.

2. In the New Point dialog box:
   - Enter a name for the point in the **Point ID** field.
   - Select **Device Point** for **Point Origin**.
   - Select **Analog** for **Point Class**.
   - Click **OK**.

3. In the General tab of the Point Properties dialog box for the new point, make sure you:
   - Select a **Resource ID** for the point.
   - Enter the correct Point Type in the **Type** field.

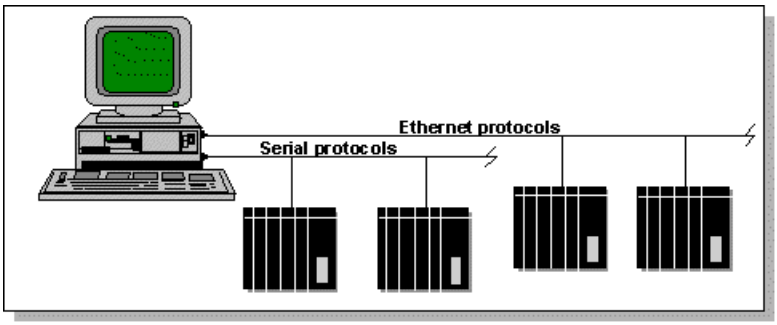
4. In the Devices tab of the Point Properties dialog box for the new point, make sure you:
   - Enter the correct Point Address for the Diagnostic Point in the **Address** field.
   - Check the **Diagnostic Data** check box.
   - For **Update Criteria**, select **On Change**.
   - Set the scan rate as you see fit.

5. Click **OK** to add the Diagnostic Point to the list of points.

Chapter 12. APPLICOM Communications

About APPLICOM Communications

The APPLICOM Communications enabler uses APPLICOM Communication Server hardware and software to communicate with a variety of devices from different vendors using appropriate protocols.



Using this enabler, CIMPLICITY software can:

Poll points at a user-defined scan rate, on demand, or once at startup

- Read and write single points
- Read arrays
- Perform Engineering Unit conversion for meaningful point value displays
- Issue alarms to users when a device goes down
- Access the APPLICOM interface database

This enabler can support up to 8 APPLICOM interface cards in one computer.

This enabler supports the following CIMPLICITY features:

- Polled read at user defined rates
- Poll after setpoint
- Triggered reads
- Analog deadband
- Alarm on communications failure
- Server redundancy configuration

This enabler supports the following data types:

- Signed 8, 16, and 32 bit integers

- Unsigned 8, 16, and 32 bit integers
- Floating point
- Text
- Arrays

APPLICOM Communications Supported Protocols

Through the APPLICOM Communications enabler, the APPLICOM device communications enabler supports the following protocols:

- April JBUS
- Klöckner-Moeller SUCOMA
- Modbus RTU
- Omron SYSMAC-WAY
- Otic Fischer & Porter Datalink
- Profibus DP
- SAIA SBUS
- Siemens 3964 or 3964R
- Siemens PPI
- Siemens Profibus FMS
- Siemens SINEC H1 HTB and H1-TF
- Siemens SINEC L2/S5
- Télémécanique ETHWAY
- Télémécanique FIPWAY
- Télémécanique UNI-TELWAY

The APPLICOM device communications enabler also supports access to the database on each APPLICOM interface card.

For more information on these protocol and their hardware and software requirements, see the APPLICOM Communication Server documentation.

APPLICOM Communications Supported Data Types

APPLICOM Communications Supported Data Types

APPLICOM Communications supported data types include:

- Generic device data types
- Interface database data types

- Seimens device data types

APPLICOM Generic Device Data Types

The following data types may be read from and written to generic devices accessed through the APPLICOM Communications enabler. Not all data types are available for all devices.

Tag	Memory domain
B	Bits
BI	Input bits
BO	Output bits
O	Bytes
OI	Input bytes
OO	Output bytes
W	Words
WI	Input words
WO	Output words
D	Double words
F	Floating point values

APPLICOM Interface Database Data Types

APPLICOM Interface Database Data Types

The following data types may be read from and written to the APPLICOM interface internal database:

Type	Description
B	Bit data
O	Byte (octet) data
W	Word data (16 bits)
F	Floating point data
D	Double word data (32 bits)

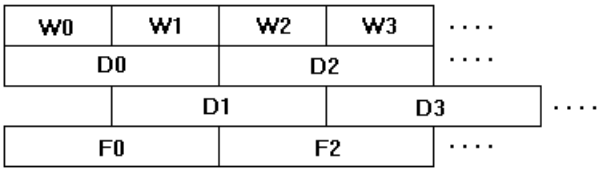
Notes on Database Data Types

Byte, Word, Double Word and Floating Point data are all stored in the same area of the database. This area is organized in 16 bit words addressed from 0 to 32,767 (decimal).

There are 32,766 double words. Each double word starts on a word boundary thus a DINT point configured at address D0 overlaps one configured at D1 by one word. The most significant word of the point at D0 is the least significant word of the point at D1. In addition, a point configured at D0 is identical to one configured at W0. The last double word is at D32766 and includes words 32,766 and 32,767.

There are 16,383 floating-point values addressed with even numbers from 0 to 32,766. Each value takes up two words starting on an even word boundary. Thus, a REAL point at address F0 coincides with a DINT point at address D0, and a REAL point at F2 coincides with a UDINT point at D2.

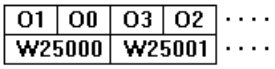
The following illustrates the relationship between Word (W), Double Word (D), and Floating Point (F) data in the internal database:



For example:

- If R is a 3-element REAL array point at address F4, R[0] contains F4, R[1] contains F6, and R[2] contains F8.
- If S is a 3-element DINT array point at address D3, S[0] contains D3, S[1] contains D5 and S[2] contains D7.
- If T is a 3-element DINT array point at address D4, T[0] contains D4, T[1] contains D6, and T[2] contains D8.

There are 14,000 bytes stored in words 25,000 through 31,999. In this Byte addressable area, the relationship between Word (W) and Byte (O) data is:



A point of type SINT with address O0 (oh zero) retrieves the least significant byte of word 25,000. Address O1 (oh one) is the most significant byte of word 25,000. A SINT or USINT point configured at O0 (oh zero) is identical to one configured at W25000.

Siemens Device Data Types

The following data types may be read from and written to Siemens devices accessed through the APPLICOM Communications enabler:

Type	Description
M	Bits
E	Input bits
A	Output bits
MB	Bytes
EB	Input bytes
AB	Output bytes
MW	Words
EW	Input words
AW	Output words
MD	Double words

The following data types may be read from and written to Siemens data blocks accessed through the APPLICOM Communications enabler:

Type	Description
DP	Bits
DW	Words
DD	Double words

APPLICOM Required Hardware and Software

The APPLICOM Communications enabler is supported on personal computers based on Intel 80486 and compatible processors. It requires the following hardware and software:

- One or more APPLICOM interface cards
- APPLICOM Communication Server version 2.8 or later

These items are available from your APPLICOM distributor.

APPLICOM Related Documents

When configuring this interface option you should have the APPLICOM Communication Server manual, as well as documentation specific to the communication protocol and devices you are using.

APPLICOM Card Configuration

The APPLICOM interface cards are configured with PCCONF. This APPLICOM Server Configuration tool comes with each card.

To start the card configuration program, double-click the **PCCONF** icon 

in the APPLICOM SERVER Common project.

Refer to the APPLICOM documentation for details on using this utility.

You may have up to eight cards (numbered 1 through 8) installed in your computer. Each card supports up to 4 channels (numbered 0 through 3). The channels may be serial, Ethernet, or protocol-specific.

Directory Path for APPLICOM Communications

After you install the APPLICOM Communication Server drivers, you must include their directory path in the `PATH` environment variable on your CIMPLICITY computer.

To do this:

1. Shut down all CIMPLICITY projects on the computer.
2. From the Start menu, select Settings.
3. In the Settings menu, select Control Panel.
4. In the Control Panel window, click **System**.
5. In the System Properties dialog box, select the Environment tab.
6. Highlight the **Path** environment variable in the **System Variables** list.
7. In the **Value** field, add the directory location of the APPLICOM Communication Server drivers to the end of the list.

   For example, if the drivers are located in c:\applicom, add ;C:\APPLICOM to the end of the value.

8. Select **Set** to make the addition part of the **Path** environment variable.
9. Click OK to close the System Properties dialog box.
10. Exit the Control Panel window.
11. Reboot the computer to make the environment variable change active.
12. Follow the directions for starting the APPLICOM Communication Server drivers.
13. Start your CIMPLICITY projects.

APPLICOM Installation Verification

The APPLICOM Communication Server comes with diagnostic and utility programs that you can use to verify configuration and communication.

Refer to the APPLICOM device communications documentation for details.

APPLICOM Application Configuration

Port Configuration

Port Configuration

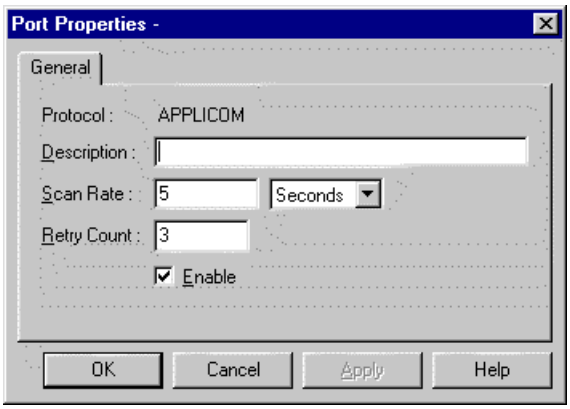
When you configure a new port for APPLICOM communications, enter the following information in the New Port dialog box:

1. In the **Protocol** field, select **APPLICOM** from the list of protocols.

2. In the **Port** field, select the communication port that will be used for APPLICOM communications.

When you click **OK** to create the port, the Port Properties dialog box for the protocol opens.

General Port Properties



Use the General properties tab in the Port Properties dialog box to enter general information for the port. You can define the following:

Description	Enter an optional description to help you identify the port.
Scan Rate	Enter the base scan rate for the port. Point scan rates will be multiples of the base rate. You can specify a scan rate in **Ticks** (100 ticks = 1 second), **Seconds**, **Minutes**, or **Hours**.
Retry Count	If communications cannot be established for a device on the port, the device is considered to be down, and a $DEVICE_DOWN alarm is generated. Once a device is down, periodic attempts are made to resume connections to it.
	Enter the number of scans to wait before attempting to reconnect to a device after a communications error has been detected.
Enable	Set this check box if you want the port to be enabled when the project starts. If you clear this check box, the port will not be enabled when the project starts, and points will not be available for devices on the port.

APPLICOM Device Configuration

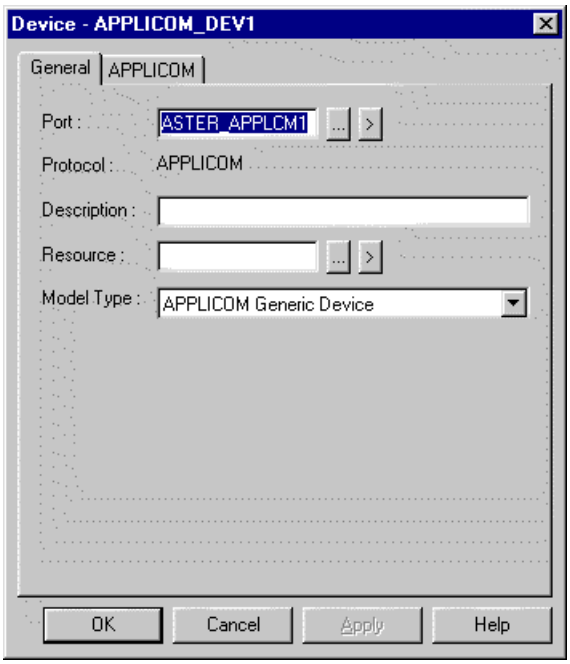
APPLICOM Device Configuration

When you create a new device for APPLICOM Communications, enter the following information in the New Device dialog box:

1. In the **Device** field, enter the name of the device you are configuring.
2. In the **Port** field, select the APPLICOM port to be used by the device.

   When you click **OK** to create the port, the Device Properties dialog box opens.

General Device Properties



Use the General device properties to enter general information for the device. You can define the following:

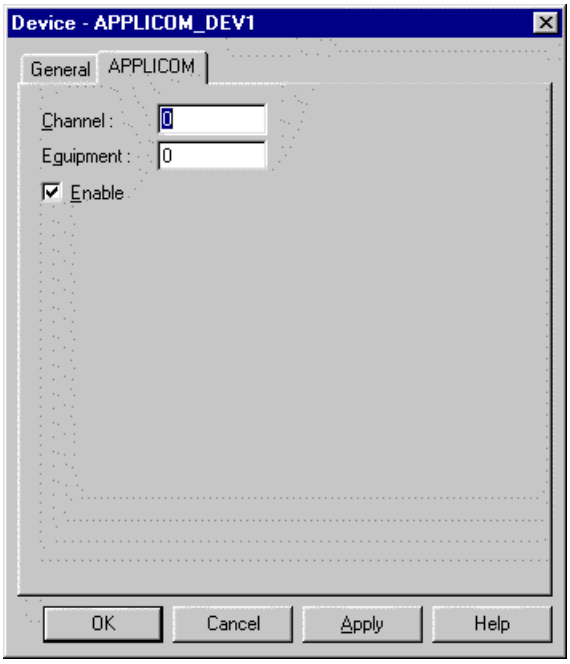
Port	Select the port for this device. • You can click the Browser button ... to the right of the field to display the list of ports and select one. • You can click the Pop-up Menu button > to create a new port, edit the current port, or select a port from the list of ports.
Description	Enter an optional description to help you identify the device.
Resource	Enter a resource that can be associated with this device for alarm generation. • You can click the Browser button ... to the right of the field to display the list of resources and select one. • You can click the Pop-up Menu button > to create a new resource, edit the current resource, or select a resource from the list of resources.
Model Type	Select the type of device. Click the drop-down button to the right of the input field to display your choices, then make a selection. The options are: APPLICOM Generic Device APPLICOM Interface Database Siemens Device
	For Siemens devices, select **Siemens Device**. This will let you configure points with Siemens addressing rather than generic addressing.
	For all other devices supported by the APPLICOM Communication Server, select **APPLICOM GenericDevice**.
	For access to the interface database, select **APPLICOM Interface Database**.

Depending on the Model Type you choose, one of two APPLICOM dialogs will be available. For the type **APPLICOM Interface Database**, you may specify the card number of the database. For other models types, you may specify the channel and equipment numbers for the device.

APPLICOM Device/Database Properties

- APPLICOM device properties
- APPLICOM database properties

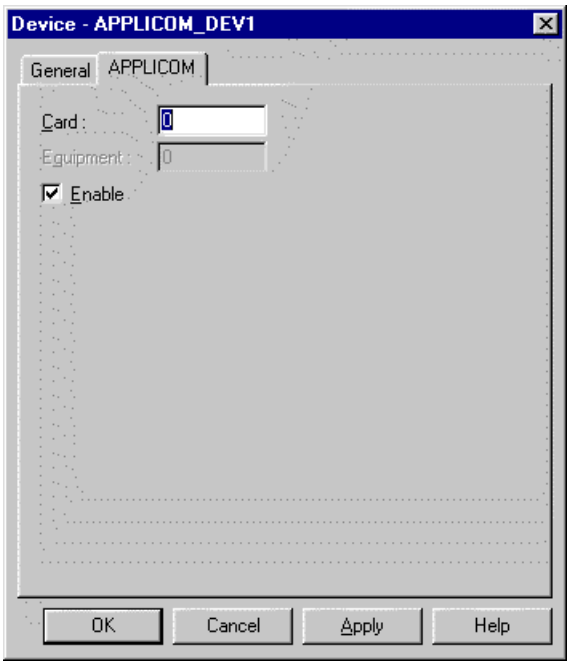
APPLICOM Device Properties



When you configure a physical device on the APPLICOM protocol, use the APPLICOM properties to enter information about APPLICOM communications for the device. You can define the following:

Channel	Enter the APPLICOM channel ID for the device. The channel ID is an integer from 0 to 31. Channels 0 to 3 are on card 1, channels 4 to 7 are on card 2, etc.
Equipment	Enter the APPLICOM equipment ID for the device as configured in PCCONF.
Enable	Set this check box if you want the device to be enabled when the project starts. If you clear this check box, the device will not be enabled when the project starts, and points will not be available for this device.

APPLICOM Database Properties



When you configure an **APPLICOM Database**, use the APPLICOM properties to enter information about the interface card. You can define the following:

Card	Enter the APPLICOM card ID for the interface card. The card ID is an integer from 1 to 8.
Equipment	Not used.
Enable	Set this check box if you want the device to be enabled when the project starts. If you clear this check box, the device will not be enabled when the project starts, and points will not be available for this device.

Point Configuration

Point Configuration

You can configure points that correspond to values in a physical device, or to values in each APPLICOM interface's internal database.

When you define a point, the several fields in the Point Properties dialog box have values that are unique to or have special meaning for APPLICOM communications:

General Point Properties

On the General tab in the Point Properties dialog box, you may configure points for **Read** or **Read/Write** access.

Device Point Properties

On the Device tab in the Point Properties Dialog Box: Properties

Update Criteria	The update criteria determines how the data will be requested. Click the drop-down button to the right of the input field to display your choices, then make a selection.
	Select **On Change** or **On Scan** for points whose values should be polled by the APPLICOM driver at regular intervals.
	Select **On Demand On Change** or **On Demand** On Scan for points whose values should only be polled when needed.
	Select **Poll Once** for points whose values should be polled by the APPLICOM driver only at startup or when the point is dynamically configured
Address	Use the point address format for the device's model type. The address formats are: APPLICOM Generic Device *(page 163)* APPLICOM Interface Database Siemens Device *(page 165)*

When you are configuring array points, make sure that the size of the array is within the limits of the domain. CIMPLICITY software does not declare an array configured beyond the end of a domain as invalid. If an array point is configured that extends beyond the end of a domain, a $DEVICE_DOWN alarm will be generated each time an attempt is made to read data from or write data to the point.

APPLICOM Point Address Formats

APPLICOM Generic Point Addresses

The general point address format is:

```
<TAG><OFFSET>
```

Where:

< **TAG** > is a model-dependent prefix, which specifies the memory domain. Valid prefixes are:

Tag	Memory domain
B	Bits
BI	Input bits
BO	Output bits
O	Bytes
OI	Input bytes
OO	Output bytes
W	Words

WI	Input words
WO	Output words
D	Double words
F	Floating point values

Be aware that not all devices support all memory types.

<OFFSET> is the offset of the item within the domain. The generic APPLICOM device type supports the following offsets:

- Decimal addressing (radix 10)
- Address of the first element (minimum address) is 0.

APPLICOM Interface Database Addresses

The general point address format is:

```
<TAG><OFFSET>
```

Where:

< **TAG** > is a model-dependent prefix, which specifies the memory domain. Valid prefixes are:

Tag	Memory Domain
B	Bits
O	Bytes
W	Words
D	Double words
F	Floating point values

<OFFSET> is the offset of the item within the domain. The APPLICOM Interface Database device type supports the following offsets:

- Decimal addressing (radix 10)
- Address of the first element (minimum address) is 0.

For example, when configuring a point in the APPLICOM interface database, **W23** is the 24th word value in the database (because database domains are numbered starting at 0). The **W** specifies the word domain and the **23** specifies which word.

Siemens Point Addresses

Data in Siemens devices can be read from device memory or data blocks.

When you select Siemens Device for the device model, you enter point addresses in Siemens nomenclature. These addresses are then translated by the APPLICOM Communications enabler into the corresponding APPLICOM references according to the formulas shown in the APPLICOM Communication Server manual.

For example, each data block is offset 256 items from the previous, so **DB150DW23** converts to an APPLICOM generic address of 38423 (or [150 * 256] + 23).

Device Memory Addresses

When you specify addresses for points in device memory, use the following general format:

```
<TAG><INDEX>
```

Where:

< TAG> is memory type. Valid memory types are:

Type	Description
M	Bits
E	Input bits
A	Output bits
MB	Bytes
EB	Input bytes
AB	Output bytes
MW	Words
EW	Input words
AW	Output words
MD	Double words

< INDEX> is the index of the item within the device memory. The index may be in octal, decimal, or hexadecimal depending on the device model.

Data Block Addresses

When you specify addresses for points in data blocks, use the following general format:

```
        DB<BLOCK><TAG><INDEX>
```

Where:

< BLOCK> is the data block number.

< TAG> is data type. Valid data types are:

Type	Description
DP	Bits
DW	Words
DD	Double words

< INDEX> is the index of the item within the data block. The index may be in octal, decimal, or hexadecimal depending on the device model.

For example, **DB150DW23** is data word 23 in data block 150.

APPLICOM Domain Configuration

APPLICOM Domain Configuration

Some aspects of APPLICOM point configuration are controlled by the configuration file **%SITE_ROOT%\data\applicom.cfg**. Generally, you should not have to modify this file. However, you may modify it to:

- Add explicit support for new devices
- Change point address tags to match local conventions
- Limit address ranges

APPLICOM Application Configuration

Up to 8 APPLICOM interface cards may be installed in a single computer. Each interface card may have up to four channels, and each channel may have up to 256 devices. A single APPLICOM Communication Server provides access to all these devices for up to eight client programs.

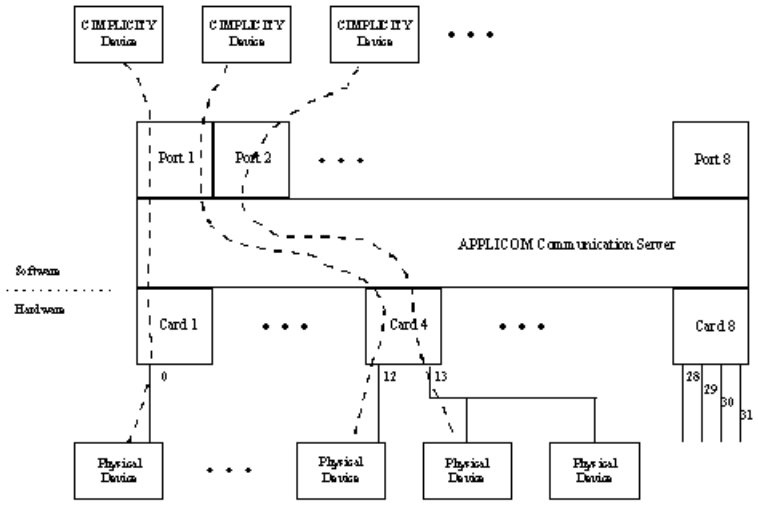
Each CIMPLICITY port using the APPLICOM Communications enabler is a client of the APPLICOM Communication Server. If no other client programs are accessing the server, you can configure up to eight ports.

Because all device access in through a single APPLICOM server, all devices on all channels may be accessed via any CIMPLICITY port. You may:

- Use a single port for all your devices
- Use ports to logically group devices by APPLICOM interface, by device type, or by process area, or by some other criteria.

In the latter case, you can control a logical group of devices by enabling and disabling a single CIMPLICITY port. Devices on other ports accessed via the APPLICOM Communications enabler would continue to be available.

The following figure illustrates the relationship between CIMPLICITY ports, the APPLICOM Communication Server, APPLICOM interfaces, and communication channels.



When you configure ports, devices, and points that use the APPLICOM Communication enabler, some fields must contain unique values for the communications to work successfully.

Field Definitions

Field Definitions

The memory domains for devices using the APPLICOM Communications enabler are configured in **%SITE_ROOT%\data\applicom.cfg**. This file is in IDT format with the following fields:

```
model|applicom type|name|tag|radix|min_addr|max_addr
```

- Model represents the model number of the device. This corresponds to an entry in the .MODEL file for the protocol. The valid model numbers for the APPLICOM Communications enabler are: 60 - APPLICOM Interface Database, 61 - APPLICOM Generic Device, and 62 Siemens Device.
- Valid types for APPLICOM type domain include: 0 - Bits, 1 - Input bits, 2 - Output bits, 3 - Bytes, 4 - Input bytes, 5 - Output bytes, 6 - 16 bit words, 7 - Input 16 bit words, 8 - Output 16 bit words, 9 - BCD words, 10 - Double (32 bit) words, and 11 - Floating point values (CIMPLICITY Real values).
- The maximum length of the domain name field is 16 characters.
- The tag field represents the point address in the domain. For a model, each domain must have a unique tag. This is used in point configuration. The tag may contain any character except |, -, *, ?, and \. The combined length of the tag and index for a point cannot exceed 32 characters.
- The radix field is the base for converting point address offsets. Defaults to 10 if blank. Suggested values are 8 (octal), 10 (decimal), and 16 (hexadecimal); but any value from 2 to 36 is acceptable.
- The minimum address (min_addr) field is the address of the first element in the domain. Typically, 0 or 1 but it may be anything. This number is in the base specified in the radix field.
- The maximum address (max_addr) field is the address of the last element in the domain. Must be greater than or equal to min_addr. This number is in the base specified in the radix field.

model

The model number of the device. This corresponds to an entry in the .MODEL file for the protocol.

The valid model numbers for the APPLICOM Communications enabler are:

60	APPLICOM Interface Database
61	APPLICOM Generic Device
62	Siemens Device

applicomtype

A valid APPLICOM domain type. Valid types are:

0	Bits
1	Input bits
2	Output bits
3	Bytes
4	Input bytes
5	Output bytes
6	16 bit words

7	Input 16 bit words
8	Output 16 bit words
9	BCD words
10	Double (32 bit) words
11	Floating point values (CIMPLICITY **Real** values)

name

The name of the domain. The maximum length is 16 characters.

tag

The tag for point addresses in the domain. For a model, each domain must have a unique tag. This is used in point configuration.

The tag may contain any character except **|**, **-**, *****, **?**, and \.

The combined length of the tag and index for a point cannot exceed 32 characters.

radix

Base for converting point address offsets. Defaults to 10 if blank. Suggested values are 8 (octal), 10 (decimal), and 16 (hexadecimal); but any value from 2 to 36 is acceptable.

min_addr

Minimum address. The address of the first element in the domain. Typically, 0 or 1 but it may be anything.

This number is in the base specified in the **radix** field.

max_addr

Maximum address. The address of the last element in the domain. Must be greater than or equal to **min_addr** .

This number is in the base specified in the **radix** field.

Sample File

For example, the following file shows the default domain configuration for the database on the APPLICOM interface card:

```
|-*
* model|applicom type|name|tag|radix|min_addr|max_addr
* Addresses are 32-bit unsigned integers
60|0 |Bits|B|10|0|32767
60|3 |Bytes|O|10|0|13999
60|6 |Words|W|10|0|32767
60|10|Double words|D|10|0|32767
60|11|Floating|F|10|0|32767
```

For another example, a domain of 65,535 16-bit input words which is naturally addressed **%I[x]** (where **x** is a hexadecimal number), can be specified like:

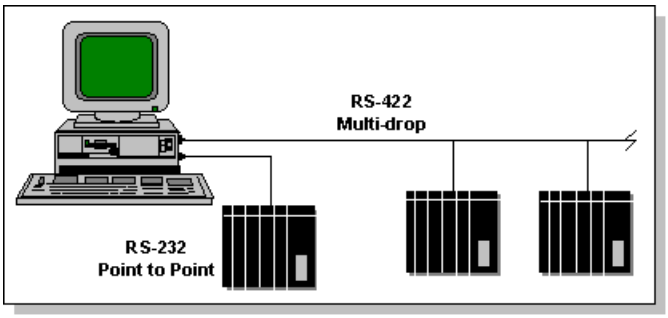
```
61|7|Word Inputs|%I[|16|0|FFFF
```

A point address like "**%I[6f]**" is parsed into a tag of "**%I[**" and an offset of "**6f**". The "**]**" suffix is ignored.

Chapter 13. CCM2 Communications

About CCM2 Communications

The CCM2 Communications enabler uses the CCM2 protocol to communicate with Series Six™, Series Five™, and Series 90™ programmable controllers. Using CCM2 active-standby protocol, this enabler acts as a server PLC and supports point to point and multi-drop configurations.



This enabler supports the following CIMPLICITY features:

- Polled reads at user defined rates
- Poll after setpoint
- Triggered reads
- Analog deadband
- Alarm on communications failure
- Server Redundancy configurations

This enabler supports the following data types:

- Signed 8, 16, and 32 bit integer
- Unsigned 8, 16, and 32 bit integer
- Floating point
- Text
- Arrays

Supported CCM2 Devices and Memory Types

Supported CCM2 Devices and Memory Types

The following devices and memory types are supported by the CCM2 Communications option:

- Series 90-30
- Series 90-70
- Series six
- Series Five

Series 90-30

The following memory types are support on a Series 90-30:

Memory Type	Description
R	Registers
I	Inputs
O	Outputs

Series 90-70

The following memory types are support on a Series 90-70:

Memory Type	Description
R	Registers
I	Inputs
O	Outputs

Series Six

The following memory types are supported on a Series Six:

Memory Type	Description
R	Registers
I	Inputs
O	Outputs
IO	Input Overrides
OO	Output Overrides
AI	Auxiliary Inputs

AO	Auxiliary Outputs
AI-	Internal Auxiliary Inputs
AO-	Internal Auxiliary Outputs
I<chan>+	Real Expanded (Channelized) Inputs
O<chan>+	Real Expanded (Channelized) Outputs
I<chan>-	Internal Expanded (Channelized) Inputs
O<chan>-	Internal Expanded (Channelized) Outputs

Where

< **chan** > is the hexadecimal channel number from 1 to F.

Series Five

The following memory types are supported on a Series Five:

Memory Type	Description
R	Registers
LI	Local Input
LIO	Local Input Overrides
I1+I	I1+ Inputs
I2+I	I2+ Inputs
I1+IO	I1+ Input Overrides
I2+IO	I2+ Input Overrides
SI	Special Inputs
LO	Local Output
LOO	Local Output Override
O1+O	O1+ Outputs
O2+O	O2+ Outputs
O1+OO	O1+ Output Overrides
O2+OO	O1+ Output Overrides
O1IC	O1- Internal Coils
O2IC	O2- Internal Coils
O1ICO	O1- Internal Coil Overrides
O2ICO	O2- Internal Coil Overrides

Related CCM2 Documents

You should have the appropriate documentation for your programmable controllers available when configuring this interface option. The following documents are available from GE Intelligent Platforms, Inc.:

Series Six Programmable Controllers Data Communications Manual (GEK-25364)

Series Five Data Communications (GFK-0244)

Series 90 PLC Serial Communications User's Manual (GFK-0582)

CCM2 Hardware Installation External to the Computer

Configure the following parameters for the CCM2 communications port on each programmable controller that is to communicate with CIMPLICITY software:

Item	Description
CPU ID	Each device on the same port must have a unique CPU ID.
Baud Rate (9600 or less)	All devices on the same port must be set to the same baud rate.
Parity (usually odd)	All devices on the same port must be set to the same parity.
Protocol	All devices must be configured as server devices. Depending on your network configuration, select the RS232 or RS422 protocol.

Note: If you are using the J1 port on a Series Six CCM2 card, the parity is always odd.

Since this information is required when configuring the CIMPLICITY communications port and the CIMPLICITY device, you should record these values at this time.

Install the correct cables for CCM2 communications. See the appropriate Data Communications manual for cabling instructions. In addition, if you are using Series 90 programmable controllers, the cabling is the same as for the SNP protocol.

CIMPLICITY Configuration for CCM2

CIMPLICITY Configuration for CCM2

CCM2 requires some CIMPLICITY port, device and point configuration.

CCM2 Port Configuration

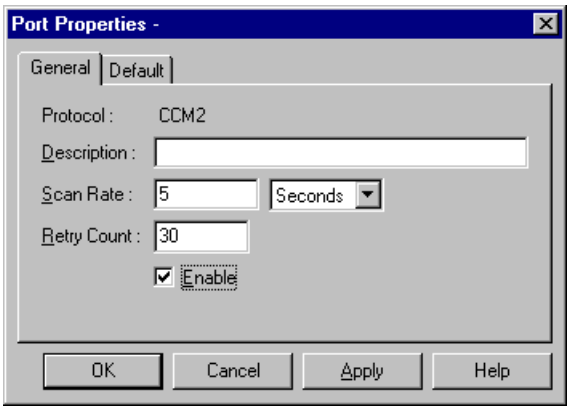
CCM2 Port Configuration

When you create a new port for CCM2 communications, enter the following information in the New Port dialog box:

1. In the **Protocol** field, select **CCM2** from the list of available protocols.
2. In the **Port** field, select the communication port that will be used for CCM2 communications.

   When you click **OK** to create the port, the Port Properties dialog box for the protocol opens.

General Port Properties

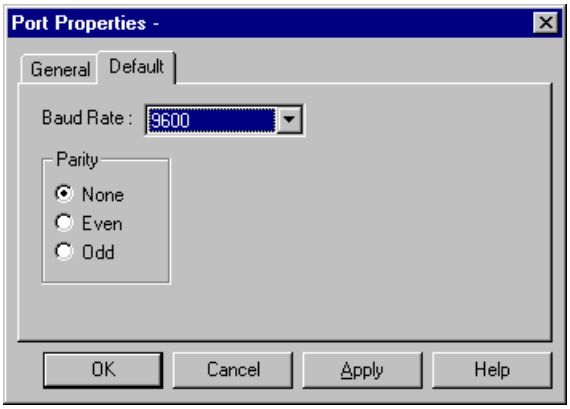


Use the General tab to enter general information for the port. You can define the following:

Description	Enter an optional description to help you identify the port.
Scan Rate	Enter the base scan rate for the port. Point scan rates will be in multiples of the base rate. You can specify a scan rate in Ticks (100 ticks = 1 second), Seconds, Minutes, or Hours.
Retry Count	If communications cannot be established, the device is considered to be down and a $DEVICE_DOWN alarm is generated. Once a device is down, periodic attempts are made to resume connections to it.

	Enter the number of scans to wait before attempting to reconnect to a device after a communications error has been detected.
Enable	Set this check box if you want the port to be enabled when the project starts. If you clear this check box, the port will not be enabled when the project starts, and points will not be available for any devices on the port.

Default Port Properties



Use the Default properties to enter information about the CCM2 communications for the port. You can define the following:

Baud Rate	Enter the baud rate for communications. Click the drop-down list to the right of the input field and select a rate from the list.
Parity	Select the parity for communications.

Note: Make sure the baud rate and parity you select match those on the programmable controller being used on this port.

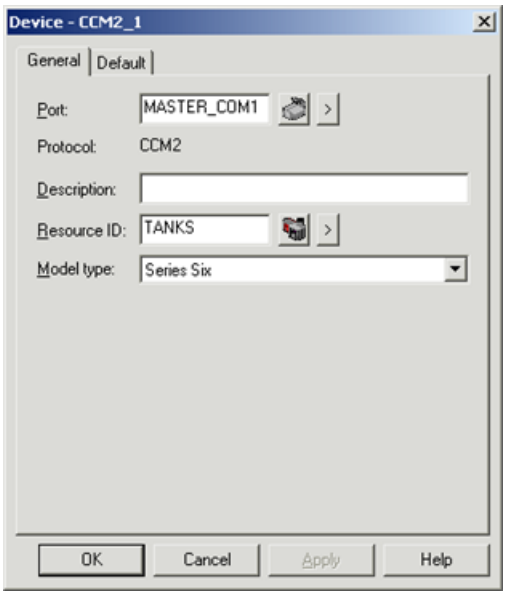
CCM2 Device Configuration

CCM2 Device Configuration

1. In the **Device** field, enter the name of the device you are configuring.
2. In the **Port** field, select the CCM2 port to be used by the device.

   When you click **OK** to create the device, the Device Properties dialog box opens.

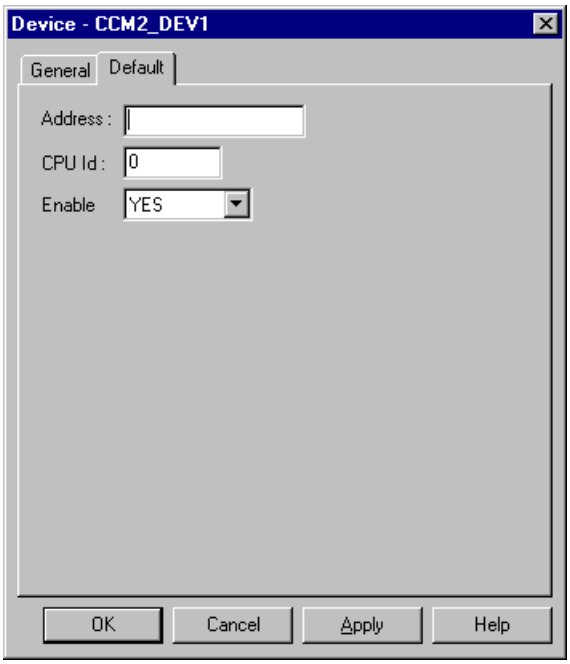
General Device Properties



Use the General Device properties to enter general information for the device. You can define the following:

Port	Select the port for this device. • You can click the Port button to the right of the field to display the list of ports and select one. • You can click the Popup Menu button to create a new port, edit the current port, or select a port from the list of ports.
Description	Enter an optional description to help you identify the device.
Resource	Enter a resource that can be associated with this device for alarm generation. • You can click the Resource button to the right of the field to display the list of resources and select one. • You can click the Popup Menu button to create a new resource, edit the current resource, or select a resource from the list of resources.
Model Type	Select the type of device. Click the drop-down button to the right of the input field to display your choices, then make a selection. For this protocol, the choices are:
	Series Six
	Series Five
	Series 90-30
	Series 90-70

Default Device Properties



Use the Default tab in the Device dialog box to enter information about CCM2 device communications for the device. You can define the following:

Address	Not used.
CPU ID	This field must match the CPU ID that was set in the programmable controller.
Enable	Select Yes in this box to enable the device when the project starts. Select No to disable the device. The points associated with the device will be unavailable when the project starts.

CCM2 Point Configuration

CCM2 Point Configuration

When you define a point, fields in the Point Properties dialog box have values that are unique to or have special meanings for CCM2 communications.

General Point Properties

On the General tab in the Point Properties dialog box, you may configure points for **Read** or **Read/ Write** access.

Device Point Properties

On the Device tab in the Point Properties dialog box:

Address	Point address requirements are device-dependent. For CCM2 communications, the address format is: **<memory_type><offset>** The memory types you can use are listed in the Supported CCM2 Devices and Memory Types *(page 171)* section.
Update Criteria	The update criteria determines how the data will be requested.
	Select **On Change** or **On Scan** for points whose values should be polled by CCM2 communications at regular intervals.
	Select **On Demand On Scan** or **On Demand OnChange** for points whose values should be polled by CCM2 communications at regular intervals when users are viewing them

When you are configuring Boolean points for memory types that are not bit addressable, you must also enter data in the following field:

Address Offset	Enter the bit offset that corresponds to the bit position within the word.

CCM2 Point Configuration Notes

CCM2 Point Configuration Notes

GE Intelligent Platforms, Inc. programmable controllers follow the same basic addressing scheme. The address of a point is specified by selecting a memory type, a location in the memory type, and for Boolean points, a bit offset.

The range of addresses for each memory type varies from controller to controller and depends on the model and its configuration (for example, the amount of memory installed in the controller). Ranges are not checked during Point configuration. If you enter a Point address that is out of range (such as R2000 on a controller that has Registers 1-1024), run-time errors will occur. Generally, a point with a bad address displays the unavailable value. For CimView screens, the unavailable value is the default text for the point. For Point Control Panel, the unavailable value is ******.

Register Memory Addressing

Registers are 16 bit words. You can configure Analog or Boolean point types to read registers.

- For Analog point types, enter the address in the **Address** field. For example, R10 specifies Register 10.
- For Boolean point types, enter the register number in the **Address** field and the bit offset (from 0 to 15) in the **Address Offset** field. For example, an address of R25 with an address offset of 0 specifies the first bit in Register 25, while an address of R25 and an address offset of 4 specifies the fifth bit in Register 25.

Input/Output Memory Addressing

Input/Output Memory Addressing

These memory types include:

- For the Series Six:

Inputs	Outputs
Input Overrides	Output Overrides
Auxiliary Inputs	Auxiliary Outputs
Internal Auxiliary Inputs	Internal Auxiliary Outputs
Real Expanded Inputs	Real Expanded Outputs
Internal Expanded Inputs	Internal Expanded Outputs

- For the Series Five

Local Inputs	Local Outputs
Local Input Overrides	Local Output Overrides
I1+ Inputs	O1+ Outputs
I2+ Inputs	O2+ Outputs
I1+ Input Overrides	O1+ Output Overrides
I2+ Input Overrides	O2+ Output Overrides
Special Input	O1- Internal Coils

O2- Internal Coils

O1- Internal Coils Overrides

O2- Internal Coils Overrides

- For the Series 90

Input	Output

You can configure Analog *(page 181)* and Boolean *(page 181)* points in these memory types.

Analog Addressing

For Analog point types in Input/Output memory types, only addresses that fall on byte boundaries are valid for analog points. To verify that the address is valid, use the following formula:

```
check = (address - 1) mod 8
```

where **address** is the number of one of the starting bit in the memory type that you want to use for the Analog point address. If **check** is zero (0), the address is valid. Otherwise, the address is invalid.

For example, an address of I18 is not valid because:

```
check = (18 - 1) mod 8 = 17 mod 8 = 1
```

An address of I25 is valid because:

```
check = (25 - 1) mod 8 = 24 mod 8 = 0
```

Boolean Addressing

For Boolean point types in Input/Output memory types, enter the bit address in the **Address** field. For example, an address of I32 specifies Input 32.

Any values you enter in the **Address Offset** field are ignored for these memory types. For example, an address of O14 with an address offset of 2 specifies Output 14 (not Output 16).

CCM2 Global Parameters

<PORT>_TURN_AROUND_DELAY

For	System	
Purpose	To set the amount of time the device communication enabler waits before sending a control character, start of header, or start of a data block.	
Value	0 ms	for any CCM to CCM direct wire connections
	10 ms	for situations causing slow response times
	500 ms	for radio transmissions
	500 ms	with time outs disabled for testing.
Default Value	0 ms	

Comments	• Pin 11 on the J1 port provides a keying signal for radio transmission. The keying signal allows the radio transmitter to warm up for the length of the turnaround delay before data begins flowing. • When the 500 ms turnaround delay with timeouts disabled is selected, timeout error conditions are ignored.

Note: The configuration in software must match the configuration on the CCM card for this to work correctly.

Chapter 14. CIMPLICITY OPC Products

About CIMPLICITY OPC Products

HMI/SCADA CIMPLICITY v10.0 provides a family of products that offer a wide variety of OPC functionality.

- CIMPLICITY OPC Servers
- CIMPLICITY OPC Clients
- OPC Simulator
- OPC general information

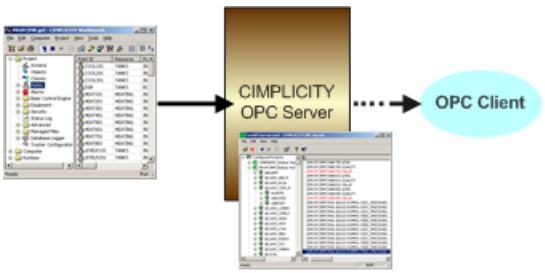
CIMPLICITY OPC Servers

CIMPLICITY offers the following OPC servers.

	CIMPLICITY OPC Server

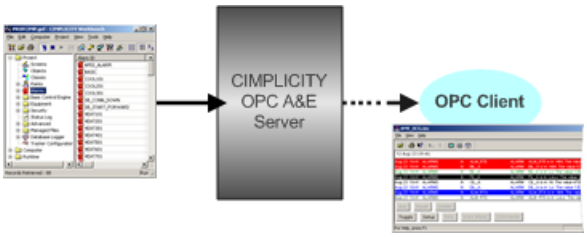
The CIMPLICITY OPC Server allows OPC clients to read, write, and subscribe to changes in CIMPLICITY runtime database points.

Note: The CIMPLICITY OPC Server also provides its own window for viewing point details.



	CIMPLICITY OPC Alarm and Event Server

The CIMPLICITY OPC Alarm and Event Server sends requested data from Alarm Managers in all connected source projects to OPC A&E clients.

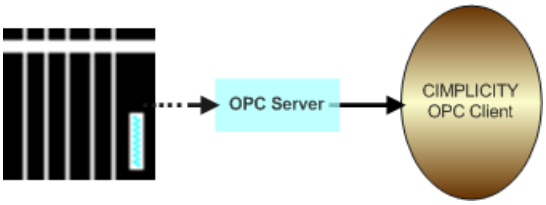


CIMPLICITY OPC Clients

CIMPLICITY offers the following OPC clients.

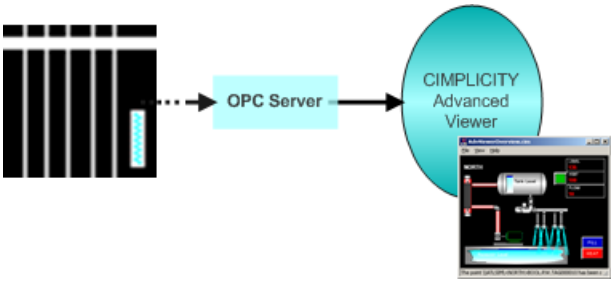
	OPC client

The CIMPLICITY OPC Client provides CIMPLICITY users with access to process data from OPC servers.



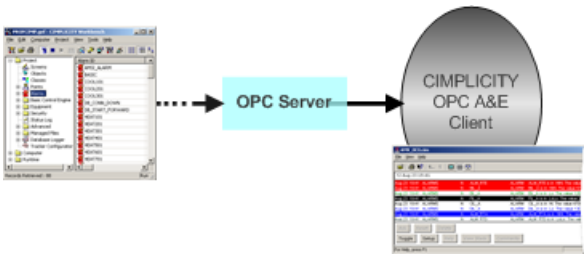
	Advanced Viewer

Displays point values collects from an OPC Server through CimView screens and the Point Control Panel without requiring a running CIMPLICITY project.



	OPC A&E client

Displays alarm and event data that was received from an OPC server in the Alarm Viewer



OPC Simulator

An OPC Server simulator is included in the CIMPLICITY distribution.

The filename is ATLSimServer.exe and is located in the following folders.

Machine	Path
32-bit	..\Program Files\Proficy\Proficy CIMPLICITY\exe
64-bit	..\Program Files (x86)\Proficy\Proficy CIMPLICITY\exe.

OPC General Information

An OPC server provides a standards-based interface to some form of runtime data. The data may come from a specific physical device (e.g. a PLC) or from a Distributed Control System.

CIMPLICITY OPC servers:

- Conform to Data Access 3.0 specification.
- Are backward compatible with Data Access 2.0 specifications.
- Do not support Data Access 1 specification; version 1 is obsolete.

OPC is a technology standard initially developed by a group of automation industry companies and now managed by the OPC Foundation, a not-for-profit organization. The standard was developed to provide a common de-coupling mechanism for automation system software components.

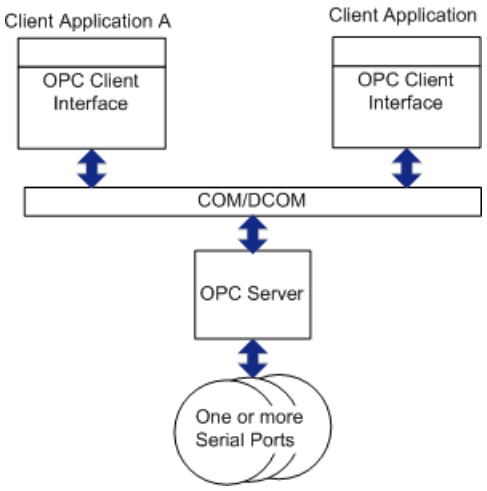
In depth information about OPC compliance is available at the OPC Foundation Web site http://www.opcfoundation.org.

- The OPC Foundation states that "Data Access 3 specification leverages earlier versions while improving the browsing capabilities and incorporating XML-DA Schema."
- As of the CIMPLICITY 7.0 release the OPC book OPC - Fundamentals, Implementation and Application, Revision 3, New Revised and Extended Edition 2006, Soft Cover with DVD is available at the OPC Foundation Web site.

OPC leverages Microsoft's COM/DCOM technology. The OPC specification defines the COM interfaces and object behaviors common to automation software applications. Since the OPC standard is COM compliant, DCOM can be leveraged for distributed deployments. For example, an OPC client application can run on a computer node different from that of an OPC server. Neither application (the client or the server) is aware of this distributed architecture.

OPC Client/Server Architecture

The following diagram illustrates the client / server architecture defined by the OPC specification.



- Multiple OPC compliant client applications can communicate with an OPC server simultaneously.
- DCOM, client and server software programs can be configured to run on the same computer node or be distributed across a network of computers.
- OPC servers provide a common view of automation information managed by the system for which the server was written.
- OPC clients use this common view of automation information in a variety of ways.

This includes providing:

- Human machine interfaces.
- Historical data logging.
- Data mirroring services.

Users can write their own custom programs in languages such as Visual Basic or Visual C++.

Desktop programs can reference OPC server information. For example, users can write VBA scripts in Microsoft Excel.

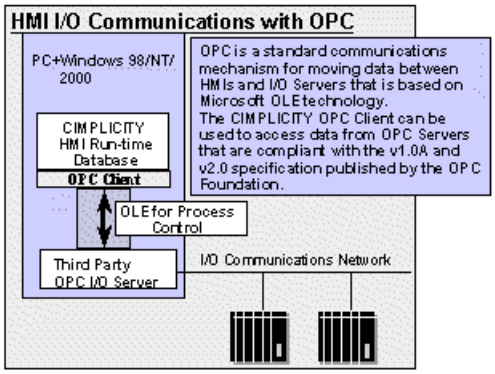
Chapter 15. CIMPLICITY OPC Client

About CIMPLICITY OPC Client

CIMPLICITY OPC Client software provides CIMPLICITY users with access to process data from OPC servers.

OPC (OLE for Process Control) is an emerging standard that is rapidly gaining acceptance among vendors of process automation products in response to the complexities of inter-operability among industrial control vendors. Developed by the OPC Foundation and endorsed by Microsoft, OPC can provide lower costs and increased productivity for end-users, systems integrators, and process control vendors alike by focusing communications issues on a single technology and strategy. Like CIMPLICITY®, OPC builds on the success and strength of Microsoft® Windows operating systems and OLE (Object Linking and Embedding) technology.

OPC defines a single common interface for both C++ and Visual Basic programs and is based on Microsoft's Component Object Model (COM) and the Distributed Component Object Model (DCOM). Through this interface, software as diverse as Microsoft BackOffice™,CIMPLICITY, and custom Visual Basic applications are able to access process data using the same methods.



OPC Client Required Documentation

All hardware requirements are OPC Server specific.

You should have the following documentation available when configuring this interface:

- OPC Server Manual

- OPC - Fundamentals, Implementation and Application, Revision 3 (optional)

CIMPLICITY Configuration for the OPC Client

CIMPLICITY Configuration for the OPC Client

You will be asked for some information that is specific to OPC Clients when you:

Step 1 *(page 188)*	Configure the OPC Client Port.
Step 2 *(page 190)*	Configure the OPC Client Device.
Step 3 *(page 195)*	Configure the OPC Client Points.

OPC Client Configuration Checklist

Prior to using this communication interface, you must do the following:

1. Install the OPC Server (and proxy if on a different machine) as detailed in its OPC Server manual.
2. Configure the OPC Server as detailed in its OPC Server manual.

Step 1. Configure the OPC Client Port

Step 1. Configure the OPC Client Port

To create a new port for OPC Clients, you enter the information in

Step 1.1 *(page 189)*	Create a new OPC port.
Step 1.2 *(page 189)*	Set Port General Properties.
Step 1.3 *(page 190)*	Set OPC Port Settings Properties.

Step 1.1. Create a New OPC Port

1. To create a new port, do one of the following:

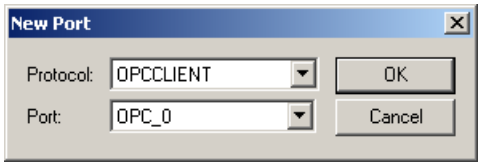
   Method 1

   Right-click the **Ports** icon Ports in the Workbench left pane and choose **New**.

   Method 2

   a. Select the **Ports** icon in the Workbench left pane.
   b. Click **File** on the menu bar, point to **New**, and choose **Object**.

   The New Port dialog box opens when you use either method.

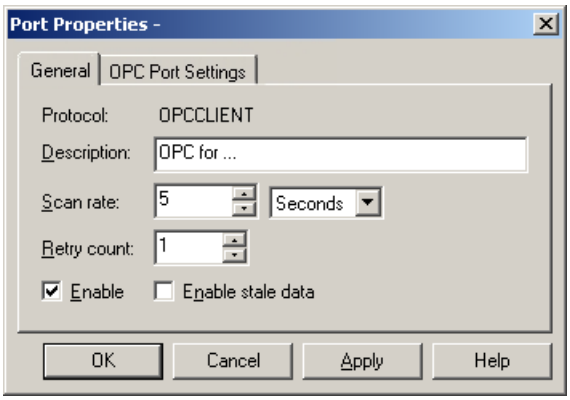


2. Select OPCCLIENT in the **Protocol** field.
3. Select the port for the protocol in the **Port** field.
4. Click OK to create the port.

The Port Properties dialog box for the protocol opens.

Step 1.2. Set Port Properties

The OPC protocol you selected in the New Port dialog box appears in the General tab of the Port Properties dialog box. Enter information that pertains to that protocol in this tab.



Enter the following information in the General tab of the Port Properties dialog box.

Description	(Optional) Add a description of the port.	
Scan Rate	Enter the base scan rate for the port. Point scan rates will be multiples of the base rate. You can specify a scan rate in Ticks (100 ticks = 1 second), Seconds, Minutes, or Hours. **Notes:** • This scan rate applies only to points that are not configured with an unsolicited or unsolicited on change update criteria. • This value does not define how often the OPC Server polls its hardware. It defines only how often CIMPLICITY polls the OPC Server. See the Scan Rate *(page 195)* variable on the OPC Group Settings tab of the Device dialog box for more information.	
Retry Count	If communications cannot be established with a device on this port, the device is considered to be down. For polled communication only, this is the number of scans until polled communications is attempted in the absence of the receipt of unsolicited data.	
Enable	Check	The port is enabled when the project starts. Enabling and disabling the port only affects polled communication. If you clear this check box.
	Clear	The port will not be enabled when the project starts, and points will not be available for devices on the port
Enable stale data	Check	Point values will remain available under most circumstances that would have made them unavailable. Under these conditions, the values displayed will be the last known value for the point. Since the current values are unknown, the values may or may not represent the current values on the device.
	Clear	Point values are no longer available under circumstances that make them unavailable.

Step 1.3. Set OPC Port Settings Properties

200 (2 seconds).Value in ticks (1/100 of a second). T`BatchTimeToLive` determines how long the OPC client waits for a completion notification that a dynamic batch addition has finished.When `BatchDynamicAdditions` is True,The OPC client

1. Waits until it receives a completion notification that the dynamic additions have completed.
2. Upon receiving this notification, the OPC client adds the batch of items to the OPC server.
   In heavily loaded systems, it is possible for the completion notification to be delayed or missed. In these cases, a timer (i.e. this property) in the OPC client causes the batched points to be added without waiting any longer for the completion notification.

Step 2. Configure an OPC Client Device

Step 2. Configure an OPC Client Device

To create a new Client Device configuration for OPC Clients, you enter the following information:

Step 2.1 *(page 191)*	Create a new OPC device.

Step 2.2 *(page 191)*	Set device general properties.
Step 2.3 *(page 192)*	Set OPC device settings properties.
Step 2.4 *(page 193)*	Select OPC server.
Step 2.5 *(page 194)*	Set OPC Group settings properties.

Step 2.1. Create a new OPC Device

1. To create a new device, do one of the following:

   Method 1

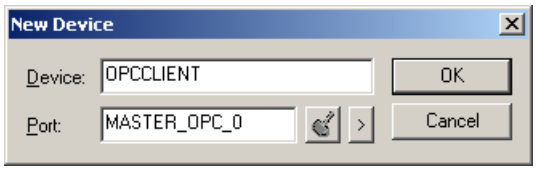
   Right-click the **Devices** icon 

in the Workbench left pane and choose **New**.

   Method 2

   a. Select the **Devices** icon in the Workbench left pane.
   b. Click **File** on the Workbench menu bar. Point to **New,** and choose **Object**.

   The New Device dialog box opens when you use either method.

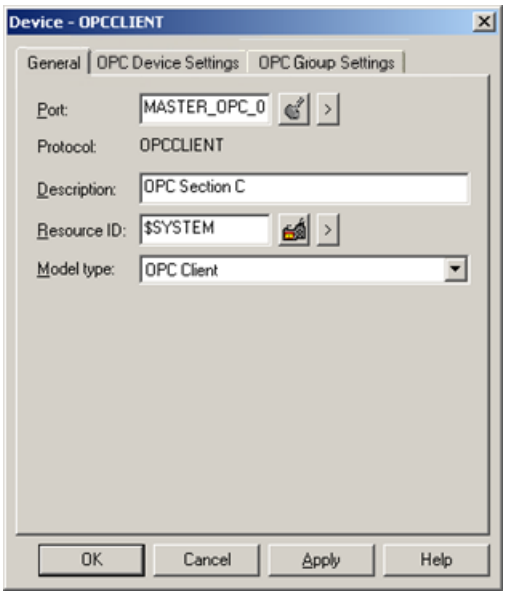


2. Enter a name for the device in the **Device** field.
3. Select the OPC Client port to be used by the device in the **Port** field.
4. Click OK.

The Device dialog box opens.

Step 2.2. Set Device General Properties

When you create a new device, the Device dialog box opens displaying the OPC protocol associated with the port you selected in the New Device dialog box.



Enter information on the General tab of the Device dialog box as follows:

Port	Select the port for this device. Click the buttons to the right of the Port field to select the port, as follows.	
		Display the list of ports and select one.
		Create a new port, edit the current port, or select a port from the list of ports.
Description	Enter an optional description to help you identify the device.	
Resource	Enter a resource that can be associated with this device for alarm generation. Click the buttons to the right of the **Resource** field to select the resource, as follows.	
		Display the list of resources and select one.
		Create a new resource, edit the current resource, or select a resource from the list of resources.
Model Type	Always select OPC Client.	

Step 2.3. Set OPC Device Settings Properties

Supercedes `DetectPingHang` and the default logic so the software is not shutdown as a result of a `GetStatus()` call not returning within the configured `PingTimeout` interval.

Note: When `IgnorePingHangTimeout` is set to True:

1. The software that periodically confirms that the OPC Server is running and responsive is disabled.

2. If the OPC Server becomes unresponsive or terminates, the condition may not detected.

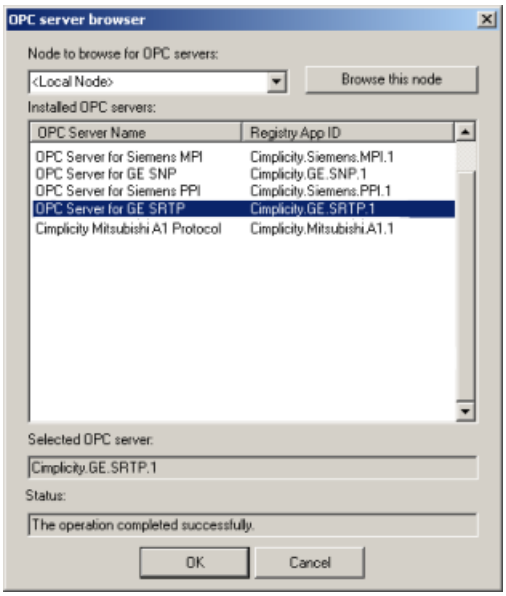
In this state, the:

a. Device state for the OPC Server will remain Up.

b. Point values will reflect the last reported value from the OPC Server.

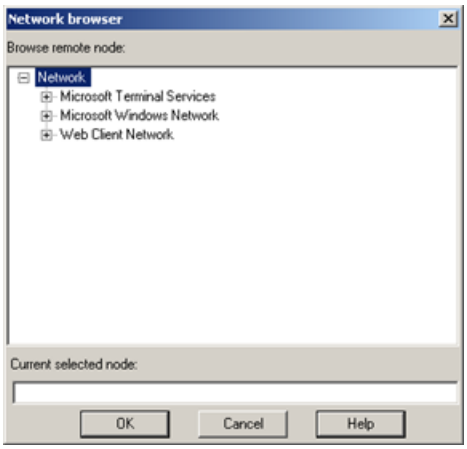
Step 2.4. Select OPC Server

OPC server address information is required for the OPC Client configuration.



Select or enter information on the OPC Server Select dialog box as follows:

Node to browse for OPC servers	Select the node you want to browse for OPC servers. When you select <Local Node>, all OPC servers installed on the local computer are listed. When you select Network..., the Network Browsing dialog box *(page 193)* opens, where you can look for and choose a remote node. At any time, you can manually enter a remote node name in this field to display the OPC servers installed on that node. If you want to choose a different OPC server from the same remote node, the remote node name appears in this list and can be selected.
Browse this node Button	Click this button to display a list of OPC servers installed on the selected node.
Installed OPC Servers	Lists the OPC servers installed on the node selected in the Node to browse for OPC servers list.
Selected OPC server	Displays the OPC server you selected from the Installed OPC servers list.
Status	Displays the status of the last operation.

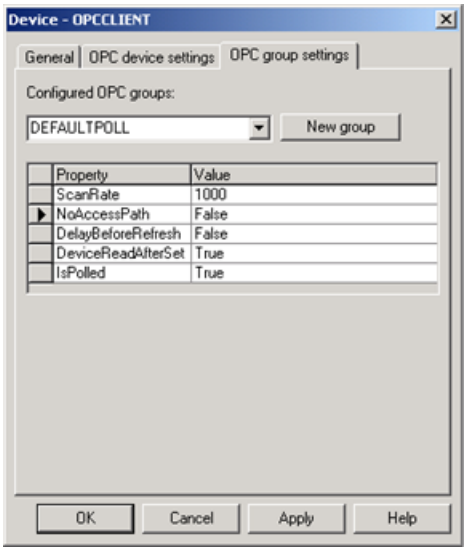


Browse for and select a remote machine to search for an OPC Server:

Browse remote node	Expand the network tree and browse to locate the remote machine with the OPC server you want to access.
Current selected node	Displays the remote machine you selected from the Browse remote node list, or you can enter the name of the remote machine if you know it.

Step 2.5. Set OPC Group Settings Properties

OPC group settings information is required for the OPC configuration.



Enter information on the OPC Group Settings tab of the Device dialog box as follows:

Configured OPC groups	Select a group from the **OPC groups** field. DEFAULTPOLL and DEFAULTUNSO are groups created by the OPC client and always exist. You can enter a group name to create a new group, then use the New Group button to add the group to the list.
New group Button	Click this button to add a new group to the Configured OPC groups list.

Property	Value	Description
Scan Rate	Default	1000 ms. Determines how often the OPC server updates the items in this group with fresh device data. It also determines how often to send DataChange events to the OPC client. Enter the scan rate in milliseconds (ms). If you require faster value updates, this value can be set as low as zero (0). If set to 0, the OPC server updates values "as fast as possible". However, caution is urged when setting this property to zero because it is possible for the OPC server to send the data faster than the hardware or operating system can support on a sustained basis. As the scan rate decreases, CPU usage may rise significantly and a corresponding increase in network traffic may be observed in a DCOM environment.
NoAccessPath	True	Enables OPC servers to have the Item IDs include the access path, separated by a period (.). Set to False if your OPC server (e.g. OMNISERVER), does not use the Access path when adding points.
	False	Default
DelayBeforeRefresh	True	Enables the OPC server to delay before refreshing after adding the items.
	False	Default
DeviceReadAfterSet	True	Requests that the OPC server read the data from the device. Default
	False	Forces the OPC server to read from its cache.
UseAsyncWrite	True	Write support will be asynchronous.
	False	Write support will be synchronous. Default
IsPolled	True	Enables polling. Default

Note: For existing devices that were configured in earlier versions of the OPC Client using the .INI file, all groups listed in the .INI file are displayed in the OPC groups list.

Step 3. Configure OPC Client Points

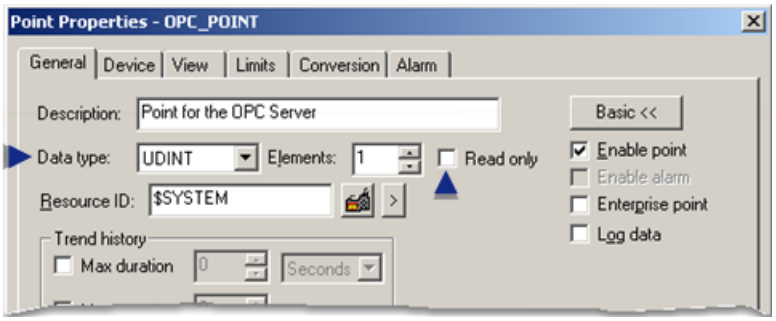
Step 3. Configure OPC Client Points

Once your devices are configured, you may configure points for them. You do this through the Point Configuration window.

Configure properties specific for the OPC Client in the Point Properties dialog box:

Step 3.1 *(page 196)*	Set general point properties.
Step 3.2 *(page 196)*	Set device point properties.
Step 3.3 *(page 197)*	Select OPC item.

Step 3.1. Set General Point Properties



When you configure a new point in the Point Properties dialog box, entries on the General tab that apply particularly to the OPC configuration include the following.

- You may configure points for:
- Read only (check Read only) or
- Read/Write (clear Read only) access.
- When you select the **Data Type** for the point, the OPC Server, with which the OPC Client communicates, determines the validity of that type for the selected point.

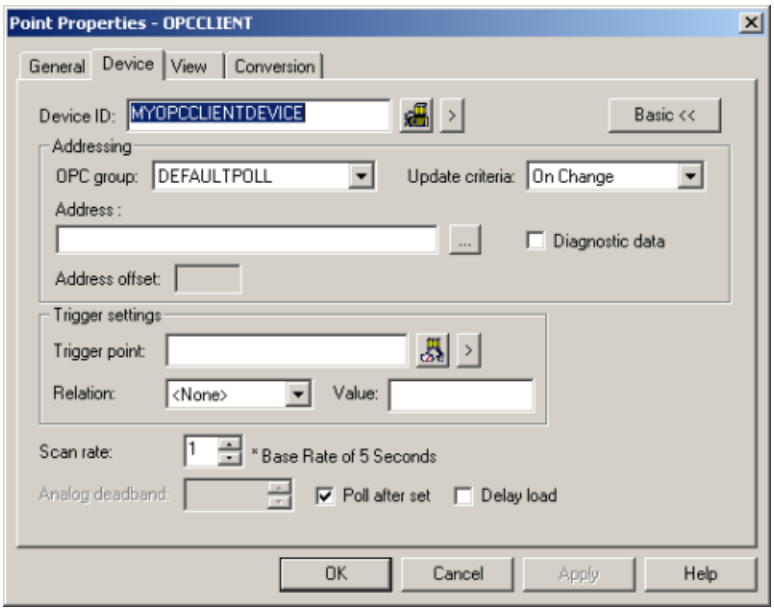
If the point is rejected by the OPC Server because the requested type is not valid:

- CIMPLICITY will declare the point as invalid.
- The OPC Client will log a message to the Status Log indicating that the point was rejected because the point type was invalid.

Note: In some OPC Servers, points that are configured within the OPC Server may be any of several allowed point types.

Check your OPC Server documentation! The more familiar you are with your particular OPC Server, the easier it will be to configure points successfully.

Step 3.2. Set Device Point Properties



Fill in the following information on the Device Tab of the Point Properties dialog box.

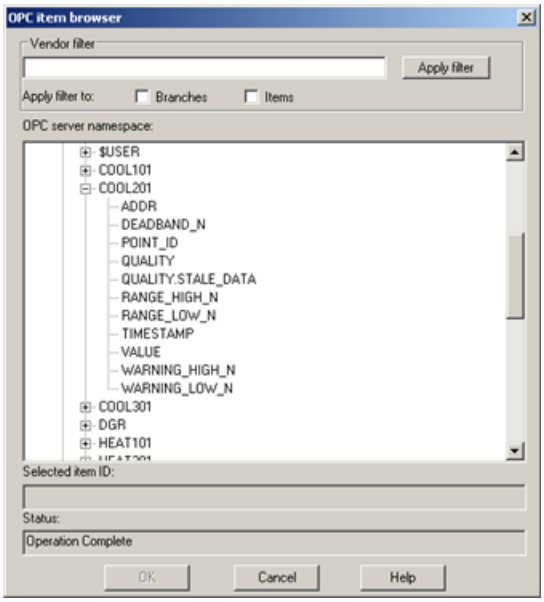
OPC Group	Select the group from the **OPC group** field. DEFAULTPOLL and DEFAULTUNSO are groups created by the OPC client and always exist.
Update Criteria	The update criteria determine how the data will be requested. • With OPC communication, Unsolicited or Unsolicited On Change are the preferred update criteria. The OPC Server will perform the polling operation and notify the OPC Client of the configured values that have changed. Additional characteristics that affect the data collection are configured within the group to which the point is assigned. **Important:** You cannot use an unsolicited point as a trigger point.
Address	The **Address** field may: • Contain up to 256 characters. • Include the OPC Group name or Item ID, and the Item name. The address for the point is OPC Server specific. Consult the OPC Server's manual for addressing information.
Browse Button	Click this button to open the OPC Item Browser where you can browse for and locate the item ID you want to use.
Diagnostic Data	Check this box if you want the point to be a diagnostic data point. If you check this box the address must match one of the addresses described in the topic OPC Client Diagnostic Points *(page 208)*.

Note: If you use Poll after Set with the OPC client device communications, you may decrease performance. Using **Poll after set** with OPC device communications is not recommended.

Step 3.3. Select OPC Item

OPC item information is required for the OPC Client configuration.

Note: Click the Browse button to the right of the **Address** field in the Device Point Properties dialog box to open the OPC Item browser.



Vendor Filter	Enter the criteria to use to filter the item ID's displayed in the OPC server name space. Similar to a **Find** procedure, items that contain the entered criteria will pass the filter. **Note:** : Filtering is OPC Server vendor-specific, e.g. if the OPC Server does not support filters, nothing will happen.
Apply Filter Button	Click this button to filter the OPC items specified in the Vendor Filter field.
Apply filter to:	Check either of the following.
Branches	Only branches that contain the filtered items will be listed.
Items	All of the branches are listed. However, only branches that contain the filtered items can be opened.
OPC Server Namespace	Lists the OPC item information in a tree format. The generated tree is the result of calls into the OPC server. Expand each level to find the item ID you want.
Selected Item ID	Displays the item ID you selected from the OPC server name space list.
Status	Displays the status of the last operation.

Important: : By default, the `ItemAccessPathDelimiter` uses a semi-colon (;) as the access path delimiter if the `ItemAccesspathEnable` property on the OPC Device dialog box>OPC device settings tab is set to True.

As a result, if a node includes a semi-colon, the OPC Item browser discards all characters that come after the semi-colon. If you attempt to expand the node a The parameter is incorrect message will display.

Guidelines for Configuring Group Addresses

guide: Guidelines for configuring group addresses:

1. You can specify a group from the **OPC group** list.

   If you do not specify a group, CIMPLICITY puts the point into one of two default groups, based on the point's **Update Criteria**.

Default Group	For points with Update Criteria that is:
DEFAULTPOLL	Polled
DEFAULTUNSO	Unsolicited

   NOTE: If all of the points in a group have polled update criteria, CIMPLICITY treats the group the same way it treats DEFAULTPOLL. As soon as one point has unsolicited update criteria, CIMPLICITY treats the group the same as it treats DEFAULTUNSO.

2. A group name can be up to 16 characters.

3. An OPC Client can have up to 510 groups. Two groups are defined by default, leaving 508 possible user defined groups.

   In the OPC Client, the maximum number of points in a group depends on the size of the points (in bytes). The total size of all points in a group cannot exceed 32K bytes. For example, a DINT point 4-Bytes in length; an INT point is 2 Bytes in length; string points are whatever length they are configured to be in Bytes.

   Please note that configuration restrictions may exceed practical limits for project implementation.

DCOM Overview

DCOM Overview

The OPC device communications enabler for CIMPLICITY is based on Microsoft's Object Linking & Embedding (OLE) model. OLE, in turn, was based on a specification for application architecture called the Component Object Model (COM).

COM is a method of writing a software application–a design model. Its most basic principle is that instead of writing an application as one large mass of code, the application should be broken down into independently running parts that each performs a specific function within an application.

In addition, COM specifies exactly how these components will interact with one another to form the whole application. This is known as the component's interface. The interface is actually a set of functions within the component, which other components can call from outside the component. These functions are exposed to the outside world. Any other component can call these functions to tell the component to do something, or ask it for some result. When a component exposes functions that anyone can call, it is called an OLE Server. Any other component calling an interface function is that server's OLE Client.

Each component is actually a program, running on the computer that may have exposed interfaces. The component program might do a variety of functions including things like displaying a chart, data collection from a PLC via a serial port, etc. Another program could then access the data through the exposed set of interface functions. This is the basic relationship between an OPC Server, and the CIMPLICITY OPC Client communications enabler.

Distributed COM is exactly the same principle as COM, with one important difference. The OLE Server component, instead of just serving up functions to other components running on the same computer, can serve up functions to other components running on ANY computer on the network. Using DCOM, the CIMPLICITY OPC Client can talk to OPC Servers running anywhere on the network.

There are some basic requirements and some configuration necessary to make this happen successfully.

Requirements for Distributed COM

For reliability, security and ease of configuration, both nodes (i.e. the node where CIMPLICITY is installed and the node where the OPC Server is installed) should be running Windows 2003 (or higher).

The OPC Server software must be properly installed and registered, and the CIMPLICITY OPC Client device communications enabler must be properly installed within CIMPLICITY.

CIMPLICITY and OPC Server Nodes' DCOM Setup

⚠ **Important:** DCOM settings for the OPC Server must be set up properly for the system to work as required. This includes setting up the OPC Server to allow access by the OPC Client, as well as impose restrictions consistent with your site's security guidelines and practices.

When the OPC Server is located on a different computer from CIMPLICITY, it is essential that the network be operating within its designed capacity and that it be free of anomalous behavior in order

for the OPC Communication to be reliable with good performance. This recommendation is not specific to CIMPLICITY. This suggestion is true for any application that uses DCOM.

You will need to be familiar with Microsoft's DCOMCNFG utility in order to configure the DCOM settings.

Optional OPC Client Debug Tracing

Optional OPC Client Debug Tracing

The OPC Client includes the ability to enable tracing in order to help diagnose communication problems between the OPC Client and OPC Server. Tracing is managed by a number of parameters and device diagnostic points. By default, if no trace parameters are specified and no diagnostic points are configured, only the default parameter values will be written to the trace file. The trace log is located in the LOG directory of your project.

Important: Enabling tracing beyond the defaults will reduce the performance of your OPC Communication and your PC. Tracing should only be enabled for the purpose of diagnosing an issue, should one arise.

Port Parameters that manage Trace output

Parameter	Description
CircularLog	Trace output can be written to either a circular log file (1) or to a flat file (0). If CircularLog=1, trace output goes to the circular log. The circular trace log is named according to the CIMPLICITY Port it represents. MASTER_OPC_0.LOG for the first port, MASTER_OPC_1.LOG for the next port, etc. If CircularLog=0 (flat file), the file is also named according to the CIMPLICITY Port, but the convention is slightly different: OPC_0.OUT for the first port, OPC_1.OUT for the next port, etc. The only advantage of the flat file is that, since it never wraps around, no trace data will be overwritten (nothing is lost). This is only an issue when a large amount of trace data is written over an extended period of time. As you will see, output to a flat file is seldom necessary to diagnose an issue. Flat files can grow very large, very quickly. You must monitor their size closely. If the flat file is used, the LogFileSize parameter is ignored.
LogFileSize	This property determines the maximum size the circular log can grow to before it wraps around and begins writing to the top of the file again. The default size is one megabyte (1000000). If you find that the circular log wraps too soon and begins overwriting old trace records you feel may be important to the diagnosis of a problem, increase this property to prolong the wrap.

Enable Tracing

The OPC Client performs various activities while it is running. It may be polling the OPC server or writing a value to a point in the OPC Server. It pings the OPC server. At startup, it creates OPC Groups and Items in the server. It handles point update events being fired by the OPC server.

You decide which of these activities you want to see trace data for, and you decide that on a Device by Device basis. You can set any of the trace properties for every CIMPLICITY device you configure. These are TRUE or FALSE (1 or 0) properties.

These properties can be manipulated at run time by defining device diagnostic points of type BOOL using the point addresses specified in the following table. Remember to check the Diagnostic option next to the **Point Address** when configuring these points in Workbench. This allows you to modify the activities that are traced, while the project is running.

With some creative engineering, these device diagnostic points can be very powerful. You could start and stop tracing based on a CIMPLICITY event manager script, or with a button on a CimView screen.

Trace Activity properties and corresponding diagnostic point addresses:

Device Parameter	Diagnostic Address	Type	Description
TraceAll	**!OPC_TRACE_ALL**	BOOLEAN	Trace all activities. This is effectively the same as setting the rest of these properties to 1.
TraceConnection	**!OPC_TRACE_CONNECTION**	BOOLEAN	Trace activity related to 'connecting to' and 'disconnecting from' the OPC Server.
TraceGroupActivity	**! OPC_TRACE_GROUP_ACTIVITY**	BOOLEAN	Trace activity related to OPC Groups; adding them to the server, calling Refresh, etc.
TraceItemActivity	**!OPC_TRACE_ITEM_ACTIVITY**	BOOLEAN	Trace activity related to OPC Items; adding them to the group, removing them from the group, adding them to the OPC Server, etc.
TracePinging	**!OPC_TRACE_PINGING**	BOOLEAN	Trace activity related to Pinging the OPC server. Did it succeed, did it fail, how long did it take, etc.
TracePolling	**!OPC_TRACE_POLLING**	BOOLEAN	Trace activity related to polling the OPC server for point values (calling the SyncIO.Read method). Successfully retrieved values are added to the point update queue.
TraceEvent	**!OPC_TRACE_EVENTS**	BOOLEAN	Trace activity related to point update 'events' 'fired' by the OPC server. When these events occur, the OPC client puts the new value into the point update queue, which is then emptied as the new values are passed into CIMPLICITY.
TraceWriting	**!OPC_TRACE_WRITING**	BOOLEAN	Trace activity related to writing a new value to a point in the OPC server from CIMPLICITY.
TraceDequeue	**!OPC_TRACE_DEQUEUE**	BOOLEAN	Trace activity related to processing the point update queue.

Every CIMPLICITY Device you configure can be set to output its trace records to it's own log file with the TraceSeparate property.

Trace Separate Property

Value	Description
TraceSeparate	TraceSeparate is TRUE or FALSE (1 or 0). The default value is FALSE. If one of your devices contains TraceSeparate=0, this device will output its trace records to the CIMPLICITY Port's circular log file (MASTER_OPC_x.LOG). Otherwise, this device will output its trace records to a file of its own, named with the device ID as follows: MASTER_OPC_x_<YourDeviceID>.LOG In this way, trace data for each configured CIMPLICITY Device is logged separately. This will aid in isolating an issue. Turning TraceSeparate off for one or more devices is useful when trying to diagnose a timing related issue, or some interaction between devices.

Every CIMPLICITY Device you configure contains one more, very important, trace property: TraceLevel.

TraceLevel can be an integer value between one (1) and seven (7). Think of the TraceLevel as the 'volume knob' for tracing. It determines the granularity, or how detailed the trace records will be. These seven values are defined as follows:

Trace Level granularity settings:

Value	Name	Description
1	**BIGERRORS**	This is the default value for TraceLevel. At this level, only big 'showstopper' type errors that occur during the activities you have enabled will be traced.
2	**ALLERRORS**	At this level, all errors, major and minor, are sent to the Trace file. This is perhaps the best setting to use for the diagnosis of an issue. Only error conditions will be traced, so you will not have to sift through a growing list of successful activities when searching for the cause of an issue.
3	**BIGSUCCESSES**	At this level, in addition to all error conditions, big successes like 'poll successful', 'write successful', and 'success connecting to OPC server' are also included. This level can be very useful when you need to see an error condition within the context of what else was going on at the time. Such as, what was the last successful operation before the error, or did the next attempt succeed.
4	**ALLSUCCESSES**	At this level, all errors and all successes are written to the trace log. Trace Levels of four or higher will cause the log to fill up quickly. In addition, the log will fill with successful operations, making it harder to diagnose problems. This setting is useful when trying to isolate a timing issue or an abnormal termination.
5	**ITEMDETAILS**	This Trace Level is like `ALLSUCCESSES`. In addition, details about every point are included. You can use this level to see point updates, complete with value, as they arrive from the OPC server. **Note:** There are other ways to get detailed point information without setting the Trace Level this high. See the section below on Other Trace related diagnostic addresses *(page 204)*.
6	**DEVELOPER Level 1**	Trace statements at this level are intended for the CIMPLICITY development team. They are not likely to be useful unless you have an in-depth knowledge of the OPC client source code.
7	**DEVELOPER Level 2**	**Important:** Do not use either of these 'Development' Trace Levels unless instructed to do so by CIMPLICITY support. These two Trace Levels provide an extreme amount of detail that will fill the log file very quickly.

Note:

- You should never need a Trace Level higher than three (BIGSUCCESSES) to diagnose an issue. It may be feasible to leave minimal tracing enabled, even in production, without severely affecting performance. With TraceAll=1 and TraceLevel=1 or 2, error conditions will be traced as they happen, and only error conditions will be traced. If an issue arises only after the project has been running for some time, this is the way to trap it. While everything is running fine, the log will remain empty. As errors occur, they will be written to the log.
- A device diagnostic point can be configured so that the Trace Level can be adjusted while the project is running. The address for this point is: !OPC_TRACE_LEVEL, and the point must be configured as an analog point of type SINT or USINT.
- Three more device diagnostic point addresses provide useful trace functionality. When configuring CIMPLICITY points with these addresses, the point type must be of type BOOL. Writing a value of one (1) will cause the action to occur. A status log message will indicate that the command was executed.

Other Trace related diagnostic point addresses

Point Address	Description
$OPC_FLUSH_TRACE_NOW	When trace records are written to the log, they are not expressly flushed to the file on disk. Instead, the OS is in charge of this. If you want to view the trace log while the project is still running, you can write a one (1) to this point. This will cause all pending trace records to be written to disk.
$OPC_DUMP_ALL_POINTS	Writing a one (1) to this point will cause the OPC client to immediately dump the status of every point in the device to the trace log. Each trace record contains detailed information about the point. This option will allow you to immediately know the answer to the two most popular support questions we get concerning the OPC client: • "Why is that point unavailable?", and • "Why hasn't that point updated recently?" This option makes it generally unnecessary to set the Trace Level above two (2). The "ITEMDETAILS" level is just not needed since you can dump point information on demand (as you need).
$OPC_DUMP_BAD_POINTS	This diagnostic point performs the same operation as the one above, but it only dumps points marked as unavailable. It will allow you to answer the first of the two big questions above without having to sift through points that are updating fine.

Viewing Trace Information

Whether using the circular log or the flat file, the format of the trace record is the same. Each trace record contains several fields of data, separated by the vertical bar character (|). This design was adopted because the trace log can be easily imported into most database and spreadsheet programs. This allows you to employ the powerful features of these programs to help view the trace data, and diagnose any issue more effectively.

Depending upon the context of the trace record, some fields may be empty. If the message does not pertain to a certain point, then Point ID, value, address, etc., will be empty. In general, the first four fields, and the last field will always contain data. The **Result Code** field will often contain zero (0x00000000), which means either success, or there was no **Result Code** to report.

Trace Record Fields:

The fields contained in every trace record are as follows (in order, left to right):

Field name	Description
Time Stamp	The time stamp when the record was logged to the file in the following format: yyyymmdd.hhmmssttt **Note:** This format represents times to a precision of one ten thousandth of a second.
Device ID	If the trace record was written by the Port, this will contain PortLevel, otherwise it will contain the CIMPLICITY Device ID of the device that generated the trace record.
Thread ID	This contains the name of the thread that generated the trace record. The OPC Client has four threads of execution. • TOOLKIT – CIMPLICITY devcom toolkit thread. • PING – Ping thread owned by device object. • WATCHER – Watcher thread owned by device object. • SERVER – OPC Server's thread (when events are fired).
Message	The message to you. Many of these have additional data embedded in them. Some are very verbose. This message can be up to 512 bytes in length.
Result (error) Code	If the trace record represents an error, and an error code was returned to the OPC Client, this field will hold that error code. If the OS has a text description of the error code, that will display as well. If there is no error code, this field will be zero (0x00000000), and the description will indicate success.
Group ID	Each CIMPLICITY Device contains at least two groups (DEFAULTPOLL and DEFAULTUNSO). There will be more if they were defined in the project. If the trace record is associated with a group, this field will contain the group's name.
Point ID	If the trace record is associated with a certain point, this field will contain the point's ID (name).
Point Address	If the trace record is associated with a certain point, this field will contain the point's address (OPC Item ID).
Point Value	If the trace record is associated with a certain point, this field will contain the point's value and variant type.
Point Quality	If the trace record is associated with a certain point, this field will contain the point's quality.
Trace Message ID	Each trace record in the OPC Client has a unique ID. This will be used for future enhancements.

OPC Client Technical Notes

OPC Client Technical Notes

- Supported CIMPLICITY features.
- Supported OPC Servers.
- Supported OPC Client types.
- OPC Client unsolicited data.
- OPC Client diagnostic points.

Supported CIMPLICITY Features

The OPC Client supports the following CIMPLICITY features:

- Collection of unsolicited data from an OPC Server.
- Poll after setpoint.
- Triggered reads.
- Analog Deadband via CIMPLICITY filtering

This enabler supports the following data types:

- Boolean
- Signed 8, 16, and 32 bit integers.
- Unsigned 8, 16, and 32 bit integers.
- Floating point
- Text.

Supported OPC Servers

OPC Servers provide real-time data by firing events whenever the value (or quality) of an item added by the OPC client changes. This OnDataChange event is defined by the OPC specification. An OPC Client can subscribe to this event and provide a handler function that gathers the new item values whenever the OPC Server fires the OnDataChange event.

The event handlers for gathering incoming unsolicited point updates in the CIMPLICITY OPC Client are compliant with the Data Access 3 and Data Access 2 specifications. The OPC Client tests to see if it has connected to a 3 or 2 compliant OPC Server and sets up the proper event handler for optimum performance.

Review data access specification documentation for the most current details *(page 183)* about valid interfaces and methods.

Note: CIMPLICITY does not support Data Access 1 specification.

Supported OPC Client Types

The following types are supported by this communications interface:

Analog

3D_BCD	Positive BCD values ranging from 0 to 999.
4D_BCD	Positive BCD values ranging from 0 to 9999.
DINT	Integers ranging from -2,147,483,648 to + 2,147,483,647.
INT	Integers ranging from -32,768 to +32,767.
REAL	Floating-point numbers.
SINT	Integers ranging from -128 to +127
UDINT	Unsigned integers ranging from 0 to 4,294,967,295.
UINT	Unsigned integers ranging from 0 to 65,535.
USINT	Unsigned integers ranging from 0 to 255.

Boolean

BOOL	A one digit Boolean point with a value of 0 or 1.
BYTE	An 8-bit array of Boolean points.
WORD	A 16-bit array of Boolean points.
DWORD	A 32-bit array of Boolean points.

Text

STRING A one	Character alphanumeric.
STRING_20 A 20	Character alphanumeric string.
STRING_8 An 8	Character alphanumeric string.
STRING_80 An 80	Character alphanumeric string.

The OPC Client supports all CIMPLICITY data types. The target OPC Server must also support the same data types. Since some servers implement strict type checking, it is recommended that you check the OPC Server documentation for data types supported by any OPC Server.

Points may be single elements or array points if the OPC Server supports that configuration.

Note:

- Review Data Access 3 specification documentation to determine if Data Access 3 provides for passing arrays of data. Data Access 2.0 specification does not provide for passing arrays of data. It does not define how an OPC Server should pass array data to an OPC Client.
- Some OPC Servers provide a way of defining Visual Basic (VB) style arrays (usually by appending an element count onto the Item ID). How arrays are supported, if they are supported, is specific to the OPC server. Consult your OPC Server vendor for details.

If an OPC Server does support VB style arrays, they can be configured in the CIMPLICITY OPC Client by simply setting the number of elements under Point Properties to match the number of elements defined in the Item ID (the CIMPLICITY point address).

Note that these values must match.

Example

If the array is defined by the syntax MyItem_20, indicating a twenty-element array, then the **Elements** property of the Point Properties in CIMPLICITY must also be set to 20 or the array will not be successfully retrieved from the OPC Server.

OLE passes the array within a single Variant value. Therefore, it is not possible to read or write a single element. The entire array must be passed when reading from the OPC Server or writing to the OPC Server.

OPC Client Unsolicited Data

Unsolicited data is supported for the OPC Client interface.

OPC Client Diagnostic Points

The following three point addresses can be used in conjunction with the **Diagnostic** checkbox on the Device tab of the Point Properties dialog box. The point ID can be anything that makes sense. The point address should be the values provided below, and the **Diagnostic** checkbox must be checked. These points may be configured for each CIMPLICITY device configured.

$OPC_SERVER_STATUS	**Read Only** Must be configured as SINT or USINT	Will be one of the following values:

		OPC_STATUS_RUNNING OPC_STATUS_FAILED OPC_STATUS_NOCONFIG OPC_STATUS_SUSPENDED OPC_STATUS_TEST	= 1 = 2 = 3 = 4 = 5
$OPC_FORCE_REFRESH_NOW	**Read / Write** Must be configured as BOOL	Writing a non-zero value to this diagnostic point instructs the OPC Client to call the Refresh method for all groups in the OPC server. This instructs the server to send all point values (changed or not). The value displayed will: • Not reflect the written value • Always display zero. An entry is written to the status log indicating that the command was received.	
$OPC_FORCE_DISCONNECT_NOW	**Read / Write** Must be configured as BOOL	Writing a non-zero value to this diagnostic point instructs the OPC Client to abort the connection with the OPC server then re-establish it. **Important:** Use of this diagnostic is not recommended! Well-behaved OPC Servers simply do not need it. Using this diagnostic can leave the OPC Server in a bad state. After using this diagnostic, it is likely that the OPC server will not shut down properly once the project is stopped.	

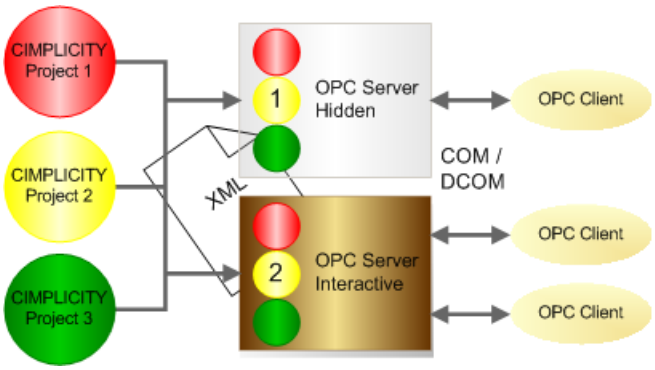
Chapter 16. CIMPLICITY OPC Server

About the CIMPLICITY OPC Server

The CIMPLICITY OPC Server is an out-of-process server compliant with the OPC Data Access V2.0x and V3.0 specifications. The OPC Server allows OPC clients to read, write, and subscribe to changes in CIMPLICITY runtime database points.

The CIMPLICITY OPC Server is an extremely flexible and robust communications tool.

The OPC Server:



	Stores data in an XML file. **Important:** Open the OPC Server at least one time before you attach clients and confirm that the projects from which data is to be served are configured as needed.
	Runs as an interactive or hidden process, depending on how it was started.
	Restarts collecting data when a project stops and restarts, without having to stop and restart.
	Provides an OPC Server interactive window with features that include monitoring a project's state and an embedded Point Control Panel.

Note:

- The CIMPLICITY OPC Server requires a 75 IO point license or higher to collect serve data from CIMPLICITY projects. running on other computers.
- The 7 Active X control © Bennet-Tec Information Systems, Inc was used to build the Tree View *(page 221)* display in the OPC Server

OPC Server Data Storage

Note: Open *(page 214)* the OPC Server window at least one time before allowing clients to attach to the OPC Server.

Because of a change in how OPC Server data is stored beginning with CIMPLICITY 7.0, opening the OPC Server enables you to determine if legacy project/log in data was read from the Registry or if you need to add projects *(page 215)* to the OPC Server configuration.

By opening the OPC Server at least one time, you are either:

- Enabling the OPC Server to perform a configuration upgrade, provided there is registry based configuration from a previous OPC server version.
- Doing a first time configuration, if no registry configuration was found.

1. OPC Server information is stored, as follows.

CIMPLICITY Version	OPC Server Information is Stored in:
7.0 and higher	An XML file: • Named CimOPCServer.xml. • Located in the $bsm_root\data directory. The default location is: c:\Program Files\Proficy\Proficy CIMPLICITY\data.
Lower than 7.0	The computer Registry.

2. When the CIMPLICITY 7.0 and higher OPC Server is started for the first time the OPC Server:
   a. Searches the Registry for configuration data from an OPC Server in CIMPLICITY versions that are lower than 7.0.
   b. Does the following based on whether or not it finds data in the Registry.

Data in the Registry is:	The OPC Server
Found	a. Reads the Registry configuration. b. Saves the configuration in the XML file.
Not Found	Creates a default XML configuration file.

   a. Once the XML file is successfully saved, any configuration data found in the registry will be removed.

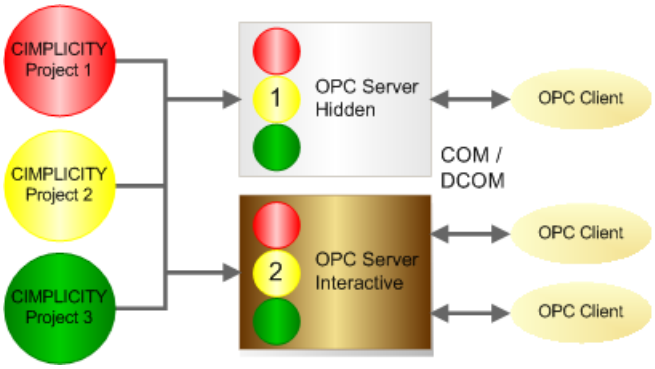
OPC Server Setup and Operation

The CIMPLICITY OPC Server can keep running and communicate data even when projects stop and start.

The CIMPLICITY OPC Server can be run as a hidden process or as an interactive process.

- Hidden vs. Interactive Process

- A project stops and restarts



Hidden vs. Interactive Process

1. When an OPC Client launches the OPC Server, the OPC Server will run as a **hidden process**.

   The OPC Server window will not open and take up desktop space.

   The process will be listed in the Windows Task Manager.

2. Authorized users can open an **interactive OPC Server window** *(page 213)* through the Windows Start menu.

   When the OPC Server window is opened it starts an instance of the OPC Server.

   When an instance of the OPC Server is running, other OPC clients with the same privileges can attach themselves to the running instance.

   Multiple instances of the OPC Server are supported. The DCOM configuration determines if an OPC Server will launch a new instance or attach to an existing instance of the OPC Server.

   The OPC Server:

   - Can run as a hidden or interactive process
   - Includes the ability to view item data that can be served through the OPC Server.

   Note:

   - The OPC Server uses memory and resources. If your system needs to conserve memory and resources you may decide to add an Interactive *(page 253)* user to your DCOM configuration.
   - Microsoft Windows DCOM determines whether an OPC Client can attach to a running OPC Server or if it needs to launch a new instance.
   - Consult Microsoft documentation for in depth details about DCOM.

   A Project Stops and Restarts

The CIMPLICITY OPC Server can serve data from a project running on either the local or a remote computer.

Following is a brief overview of how it operates.

3. When the OPC Server is initiated if a:Viewer or project is not running, the CIMPLICITY Router is started.
4. When at least one project is running, the OPC Server serves data to the OPC Clients for items that were successfully added.

   Note: When a project shuts down, the OPC Server will indicate that items served from the shutdown project are dirty . If the project is restarted, the items will be updated.

5. If the CIMPLICITY Router starts to shut down, the OPC Server notifies its OPC Clients of a shutdown request.
6. The CIMPLICITY Router shuts down when the last project shuts down.
7. To resume data collection, the OPC Clients will need to reconnect to the OPC Server and re-add groups and items when the OPC Server is in a state where valid items can be added, i.e. at least one project is again running.

   Note: The OPC Server window reports whether or not an attached project is running *(page 221)*.

CIMPLICITY OPC Server Interactive Window

CIMPLICITY OPC Server Interactive Window

1 *(page 214)*	Display the OPC Server window.
2 *(page 215)*	Attach projects to the OPC Server.
3 *(page 222)*	Select attributes shown to OPC browsers
4 *(page 226)*	Review namespaces in the OPC Server window Tree View.

5 *(page 231)*	Review points in the OPC Server window List View.
6 *(page 235)*	Close the OPC Server window.

1. Display the OPC Server Window

CIMPLICITY projects do not need to be running to work with the CIMPLICITY OPC Server. The OPC Server will start the router even if no projects are running.

1. Click Start on the Windows task bar.
2. Select (All) Programs> HMI SCADA - CIMPLICITY version>OPC Server.

The CIMPLICITY OPC Server window opens.

When the OPC Server is opened, projects and user names that the OPC Server found in the Windows Registry and wrote to CimOPCServer.xml *(page 210)* are listed.

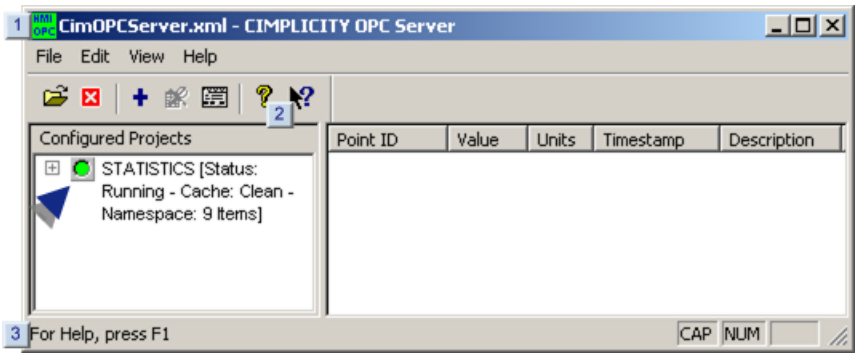
If no projects were found one entry, STATISTICS, displays in the OPC Server left pane

The STATISTICS project:

- Is an internal project
- Is always available.
- Displays some statistic information about the OPC Server, e.g. how many clients are attached.
- Cannot be removed.

The STATISTICS project name space is always showing to browsers.

CAUTION: The project name **STATISTICS** is reserved for the OPC Server. Make sure no other project has that name in order to avoid problems with the OPC Server operation.



Note:

1	The title bar identifies the CIMPLICITY OPC Server.
2	Documentation is available through the Help menu and the Contents button or Topics button .
3	Options to display or hide the OPC Server window toolbar and/or Status bar are on the View menu.

2. Attach Projects to the OPC Server

2. Attach Projects to the OPC Server

Important: The OPC server will not accept points from a project unless the project is listed in the OPC Server configuration.

- Cached namespaces.
- Steps to attach projects to the OPC Server.

Cached Namespaces

Note: Caching name spaces *(page 219)* for a project and reconciling the project on start up *(page 220)* are optional.

If both are selected, when the OPC server attaches to a running project for the first time, if it reads all of the points from the project. Once this is done, the OPC server writes this information to a cache file.

On subsequent startups,

The OPC server will first read in the cache. This improves the startup time dramatically. It also insures that clients attaching to the OPC server will not have added items rejected, even if the project is not yet running.

OPC Browser clients will still see the project's namespace, even though the project is not running. Once the project starts, or if it is already running, the OPC server begins a background task that reconciles the cache file against the actual project points, then it re-writes the corrected cache back to disk. This insures the cache always remains current.

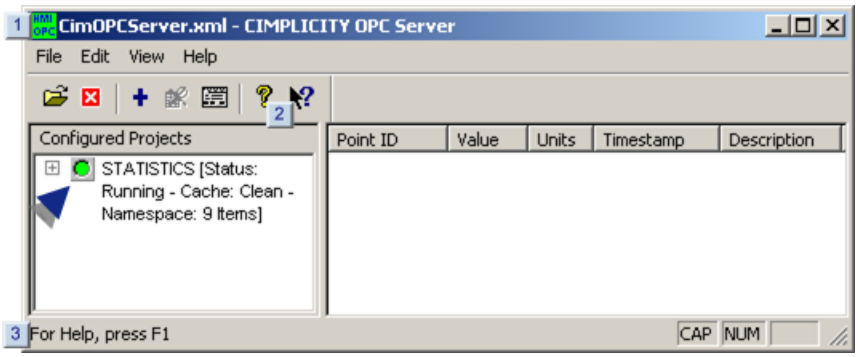
Steps to Attach Projects to the OPC Server

2.1	Open the OPC Server Configuration dialog box.
2.2	Select projects and users for the OPC Server.
2.3	Review the project state reported in the OPC Server window.

2.1. Open the OPC Server Configuration Dialog Box

The security settings for the projects are configured through the OPC Server Configuration dialog box.

Do one of the following in the OPC Server window.



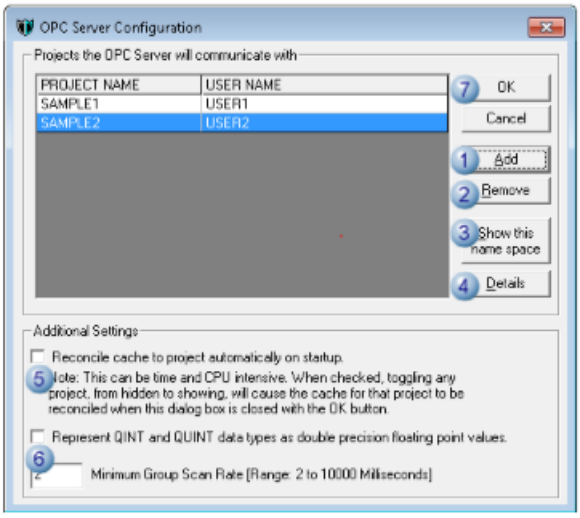
1	Click Edit>Server Configuration on the CIMPLICITY OPC Server window menu bar.
2	Click the Server Configuration button.
3	Press Ctrl+Enter on the keyboard.

Result: The OPC Server Configuration dialog opens.

2.2. Select Projects and Users for the OPC Server

The OPC Server Configuration dialog box lists the projects with authorized users that are attached to the OPC Server.

Do any of the following to modify the list.



rect 275, 88, 298, 107 *(page 217)*
rect 280, 112, 302, 129 *(page 218)*
rect 282, 135, 301, 155 *(page 219)*
rect 281, 170, 303, 188 *(page 219)*
rect 14, 225, 37, 246 *(page 220)*
rect 16, 277, 35, 295 *(page 221)*
rect 282, 47, 300, 64 *(page 221)*

1 *(page 217)*	Add.
2 *(page 218)*	Remove.
3 *(page 219)*	Show/Hide name space.
4 *(page 219)*	Details.
5 *(page 220)*	Reconcile cache to project.
6 *(page 221)*	Minimum Scan Rate
7 *(page 221)*	OK / Cancel.

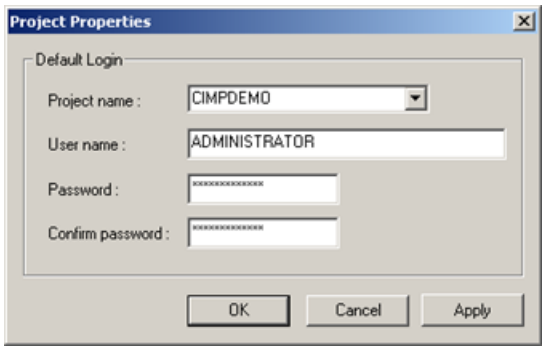
1	Add

Important: The OPC server will only accept points from a project if the project is listed in the configuration

1. Click Add.

   The Project Properties dialog opens.

2. Enter the information in the fields as follows.



Field	Description
Project name	Project that will be added to the OPC Server configuration. **Note:** Names of running projects are available in the drop down list. You can add a project name that is not running. However, the project will not be available to the OPC Server until it is started.
User name	Authorized CIMPLICITY project login user name.
Password	A CIMPLICITY password that is valid for the specified user. The OPC server uses the password in conjunction with project user name to determine user authorization.
Confirm password	Verification check. The **Password** and **Confirm password** must be the same. If they are not the same, the project settings cannot be saved.

3. Click one of the following.

Button	Result
OK	Save project settings and return to the OPC Server Configuration dialog box, if there are no errors.
Cancel	Cancel the settings and return to the OPC Server Configuration dialog box.
Apply	Apply the latest entries. The names will be added to the OPC Server list, if there are no errors. Continue adding names and passwords in the Project Properties dialog box.

   Result: All correctly added project names and user names will be listed in the OPC Server Configuration dialog box when the OPC Project Properties dialog box is closed.

2	Remove

Note: The Remove button is enabled only when no OPC Clients are attached to the instance *(page 212)* of the OPC Server that is being configured.

Any project can be disconnected from the OPC Server.

4. Select a project in the list.
5. Click Remove.

   Result: The selected project is removed from the list.

3	Show/Hide name space

The OPC Server can show or hide the points and data for any listed project..

The Show the name space/Hide the name space button specifies the following.

Button Label	Description
Show this name space	When Show this name space is selected: The project's: • Name space can be browsed from an OPC Client. • Cache, e.g. points and data are cached. An OPC Client can successfully add items whether or not the project is running if the item is found in the cache. **Note:** In the Configuration Server dialog box: • Projects are listed in a bold font when the name space is shown. • The button label will display as Hide this name space when the project is selected in the OPC Server Configuration dialog box.
Hide this name space	Default When Hide this name space is selected. The project has: • A hidden name space. • No cache. Items cannot be added when the project is not running An OPC Client browse will not see items in the project. **Note:** In the Configuration Server dialog box: • Projects are listed in a normal font when the name space is hidden. • The button label will display as Show this name space when the project is selected in the OPC Server Configuration dialog box.

4	Details

The user name and/or password can be modified at any time for a selected project.

6. Select a project in the list.
7. Click Details.

The project's Project Properties dialog box containing the selected project's user name and password opens.

8. Make any necessary changes.
9. Accept or cancel the changes when you close the dialog box.

5	Reconcile cache to project

Reconcile cache to project specifies if the OPC Server, on startup, should reconcile its cache with the project server .

The options are as follows.

Reconcile	Description
Check	The OPC Server, on startup, will reconcile the cache for the project and up-date the cache with any changes that have been made in the project configuration. Reconciling the cache on startup: • Important for projects that change frequently, e.g. points are added or deleted. • Causes the OPC Server to take more time on startup, particularly for large projects. **Note:** When there is a change in the project or OPC Server configuration, • An instance of the OPC Server that started after the change will read the latest version of the .xml *(page 210)* file on startup; its cache will be reconciled. • The cache for instances that are already running will contain the configuration data from when the obsolete .xml version was read. Those instances can be reconciled while running or stopped and re-started to update their cache with the latest .xml data.
Clear	The OPC Server will not reconcile the cache for the project. Not reconciling the cache on startup: • Is appropriate for projects that have stable configuration. If the configuration has not changed, differences will not be found so reconciling will not change anything in the cache. • Speeds up the OPC Server startup.

Tip: If Reconcile cache to project is clear and the configuration for a project whose name space is in the OPC Server has changed, a reconcile can be forced as follows.

10. Check Reconcile cache to project.
11. Click the Show the name space button.

    The project will be listed in bold font. When the project is selected in the OPC Server Configuration dialog box, the button label will display as Hide the name space.

12. Click OK.

    The OPC Server Configuration dialog box closes.

    The project data is hidden.

13. Re-open the OPC Server Configuration dialog box.
14. Click Hide the name space.
15. Click OK.

    The OPC Server Configuration dialog box closes. The OPC Server cache is reconciled.

16. To turn or reconciliation again:
    a. Open the OPC Server Configuration dialog box
    b. Clear Reconcile cache to project.

6	Minimum Group Scan Rate

You can set the OPC Minimum group scan rate in this fieeld. CIMPLICITY OPC Server will use this value as the default minium group scan rate in cases where OPC Clients connecting to this server has not specified any value for group scan rate. If an OPC client specifies a group scan rate then CIMPLICITY OPC Server will use the value specified by OPC Client. If an OPC client specifies an invalid group scan rate, or if its value is less or greater than the range between 2 to 10000 milliseconds, then the value provided to this configuration will be used as the group scan rate.

7	OK / Cancel

Click one of the following.

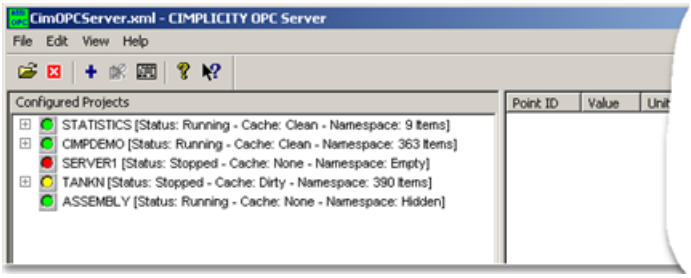
Button	Result
OK	The OPC Server Configuration dialog box closes. All changes made while the dialog box was open are made in the OPC Server. **Note:** CIMPLICITY Login dialog boxes for configured projects may open when the OPC Server Configuration dialog box closes.
Cancel	Cancel all changes and return to the OPC Server window.

2.3. Review the Project State Reported in the OPC Server Window

The tree view contains the projects the OPC Server is configured to access. All configured projects appear as top-level nodes with a plus (+) sign and an icon to indicate the status of the project (i.e. Running or Stopped).

There is a “parent” node for each configured project.

Project state is indicated by the following icons:



Icon	Description
	Green is a running project. • The project is running, and the cache is clean, i.e. it matches the actual project configuration. When the project node is expanded, the project's points are displayed according to the hierarchy described above. When the points are expanded, the final leaf nodes are the attributes selected in the Select Attributes Shown to OPC Browsers *(page 222)* dialog box. Any of the entities in the Tree View (left pane) can be dragged and dropped onto the List View (right pane). Running projects that are shown can be expanded to display their points and point attributes. • The project is running but there is no cache; the name space is hidden *(page 219)*.
	Yellow can be one of the following. This project is: • Not running, but a cached namespace is available • Running and the OPC server is in the process of reconciling the cache. In other words, a cached namespace is available, but it is dirty, possibly does not match the project configuration.
	Red indicates that the project is not running and there was no cache available for this project. For example, a newly configured project that is not running when it is connected *(page 216)* to the OPC Server will display red until it is started. Name space is empty. Projects that are not running are considered off line and cannot be expanded.

If the OPC Server is launched when no associated projects are running,

- The tree view shows the configured projects.
- The only points available to view are built-in statistic or diagnostic type items.
- Any OPC Client trying to browse the OPC Server in this case will see only the built-in items' state, the namespace for this project is empty.

3. Select Attributes Shown to OPC Browsers

3. Select Attributes Shown to OPC Browsers

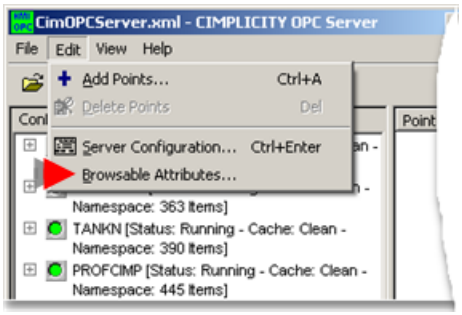
Point attributes, in projects accessed by the OPC Server, that OPC Clients can browse are selected in the Select Attributes Show to OPC Browsers dialog box.

There are 37 available attributes per point.

3.1 *(page 223)*	Open the Select Attributes Shown to OPC Browsers dialog box.
3.2 *(page 224)*	Select attributes to display.

3.1. Open the Select Attributes Shown to OPC Browsers Dialog Box

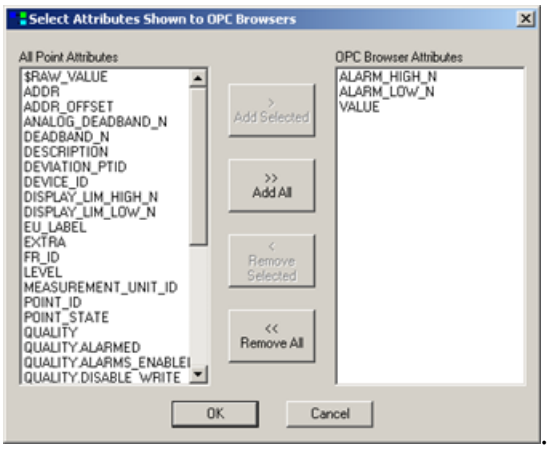
Click Edit>Browsable Attributes on the OPC Server window menu bar.



The Select Attributes Shown to OPC Browsers dialog box opens

The attributes that OPC Clients can browse are selected/de-selected by moving them to/from the **All Point Attributes** column to/from the **OPC Browser Attributes** column.

Columns and buttons are as follows.



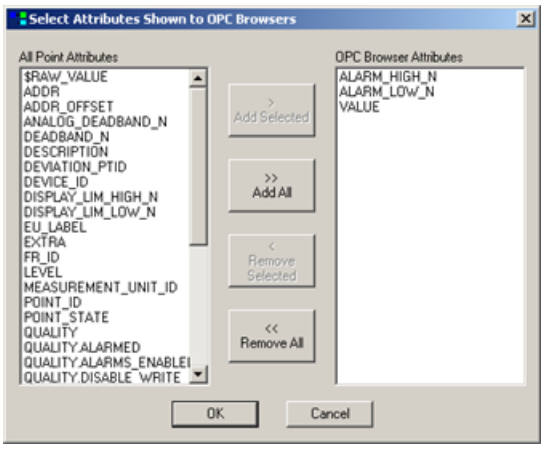
Column/ Button	Description
All Point Attributes	Displays a full list of the point attributes that can be made available to OPC Clients for browsing.
OPC Browser Attributes	Displays the selected point attributes that OPC Clients can browse.

	Default	Value
Add Selected	Moves the selected point attribute(s) to the OPC Browser Attributes list.	
Add All	Moves all of the point attributes to the OPC Browswer Attributes list.	
Remove Selected	Moves the selected point attribute(s) to the All Point Attributes list.	
Remove All	Moves all of the point attributes to the All Point Attributes list, except for the Value attribute. This is the default point attribute that is always available for OPC Clients to browse.	
OK	Saves the point attribute settings and returns to the CIMPLICITY OPC Server dialog box if there are no errors.	
Cancel	Returns to the CIMPLICITY OPC Server dialog box without saving the changes.	

3.2. Select Attributes to Display

The OPC Server provides 37 available attributes per point.

Select the attributes to be displayed.

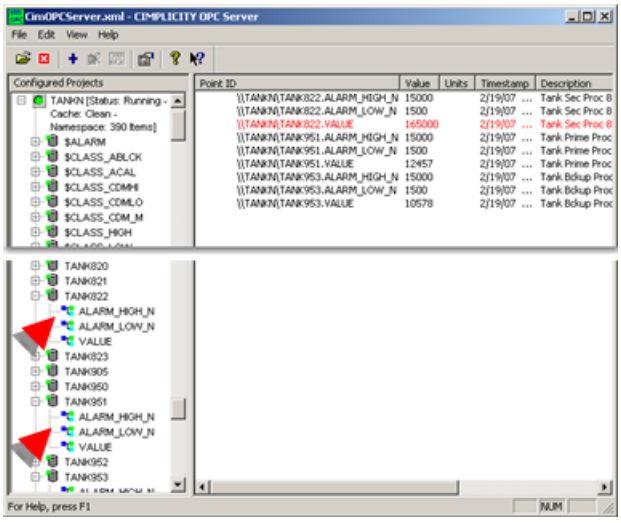


- $RAW_VALUE
- ADDR
- ADDR_OFFSET
- ALARM_HIGH_N
- ALARM_LOW_N
- ANALOG_DEADBAND_N
- DEADBAND_N
- DESCRIPTION
- DEVIATION_PTID
- DEVICE_ID
- DISPLAY_LIM_HIGH_N
- DISPLAY_LIM_LOW_N
- EU_LABEL

- EXTRA
- FR_ID
- LEVEL
- MEASUREMENT_UNIT_ID
- POINT_ID
- POINT_STATE
- QUALITY.ALARMED
- QUALITY.ALARMS_ENABLE
- QUALITY.DISABLE_WRITE
- QUALITY.IS_AVAILABLE
- QUALITY.IS_IN_RANGE
- QUALITY.LAST_UPD_MAN
- QUALITY.MANUAL_MODE
- QUALITY.STALE_DATA
- RANGE_HIGH_N
- RANGE_LOW_N
- SETPOINT_HIGH_N
- SETPOINT_LOW_N
- SETPT_CHECK_PTID
- TIMESTAMP
- USER_FLAGS
- VALUE
- WARNING_HIGH_N
- WARNING_LOW_N

Result: The selected attributes are listed for each point in the OPC Server window left pane.

The attributes are now available to the OPC Client.



4. Review Namespaces in the OPC Server Window Tree View

A namespace is the collection of acceptable Item ID's that an OPC server publishes.

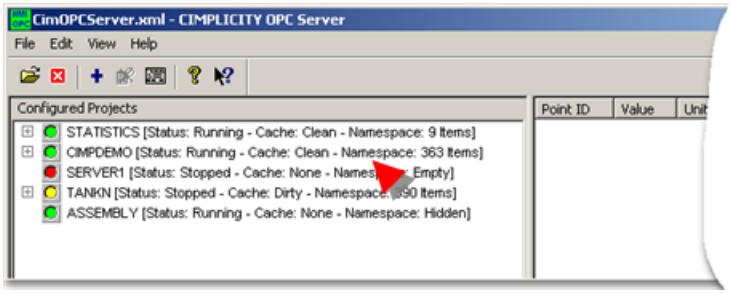
The OPC Server window's Tree View (left pane) provides extensive information about namespace status.

Namespaces display in the OPC Server Window left pane as follows.

- Namespace cache.
- User defined namespace
- Namespace behavior.
- Dynamic namespaces
- Browsable namespaces.
- Maximum number of nodes allowed in Tree View.

Namespace Cache

- The OPC Server creates a namespace cache for each project it is configured to access *(page 216)*.



- If the cache for a particular project is found when the OPC Server starts, it is read in to the namespace immediately, even if the project is not running.
- If an OPC Client requests an item from a project that is not running, but a cache exists for that project, the OPC Client will not be rejected by the OPC Server. This allows a project to be started after the OPC Server and OPC Clients do not have to request those items again after the project starts.
- If points were added, deleted, or modified while a project was stopped, the cache becomes obsolete.

However, when the OPC Server reads the namespace cache, and the project is later started if reconcile cache to project *(page 220)* in the OPC Server Configuration window is:

Checked	The cache is reconciled against the project and is automatically updated.

Not checked	The cache will be out-of-date until reconcile is enabled.

User Defined Namespace

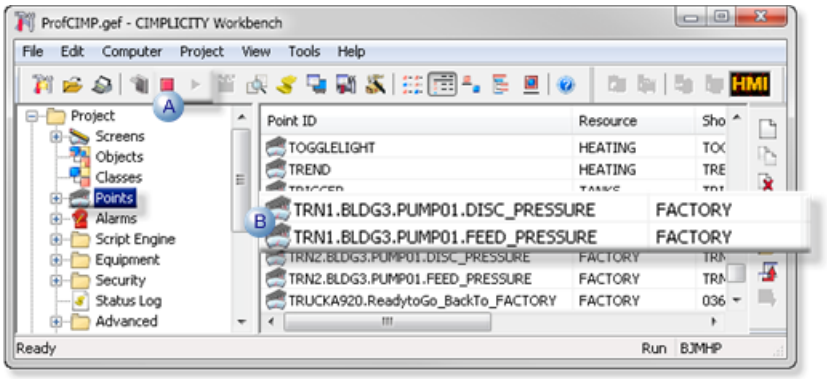
The hierarchy presented in the OPC Server window left pane and in the Browser namespace is defined by the project's Point IDs.

The new OPC Server honors any **dot delimited** naming convention used when the project points were defined.

This is a very powerful feature that enables a user to define the browse hierarchy presented by the OPC Server for each configured project.

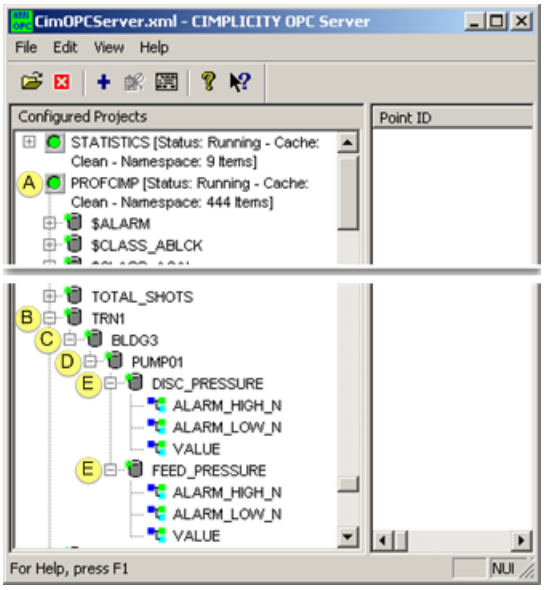
Example

1. Open a project's Workbench.
2. Do the following.



A	Create two points in an OPC Server connected project as follows. TRN1.BLDG3.PUMP01.FEED_PRESSURE TRN1.BLDG3.PUMP01.DISCH_PRESSURE
B	Start the project.

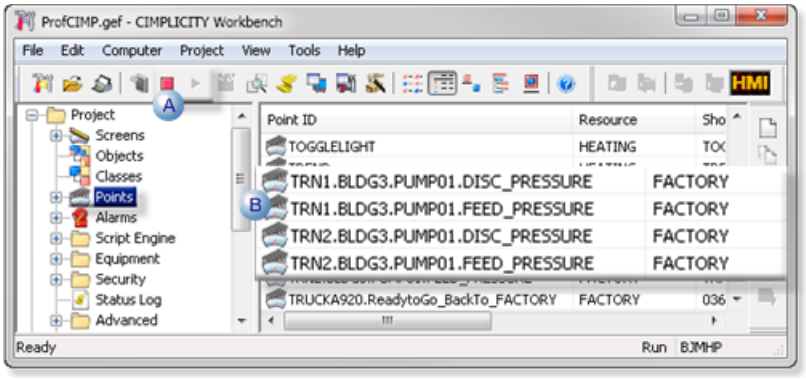
3. Open the OPC Server window.
4. Do the following.



A	Expand the project namespace. **Note:** If the project is not connected, connect it.
B	Expand TRN1. **Note:** Even though there are two TRN1 points, there is only one TRN1 in the tree.
C	Expand BLDG3.
D	Expand PUMP01.
E	The two children, DISC_PRESSURE and FEED_PRESSURE display.

5. Display the project's Workbench again.

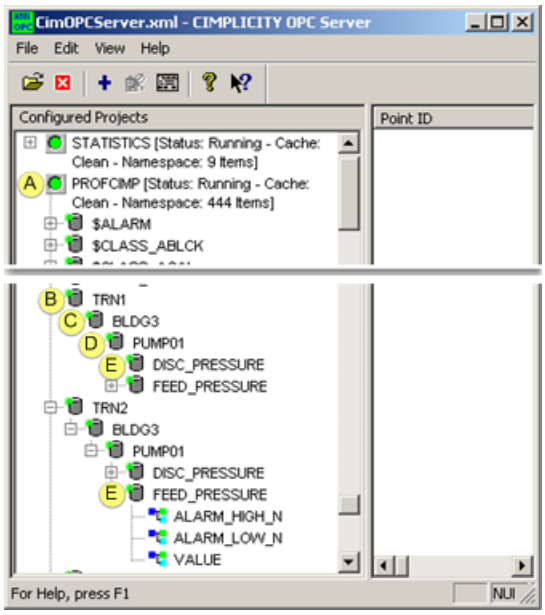
6. Do the following.



A	Create two more points in an OPC Server connected project as follows. TRN2.BLDG3.PUMP01.FEED_PRESSURE TRN2.BLDG3.PUMP01.DISCH_PRESSURE
B	Make sure the project is running.

7. Open the OPC Server window.

8. Do the following.

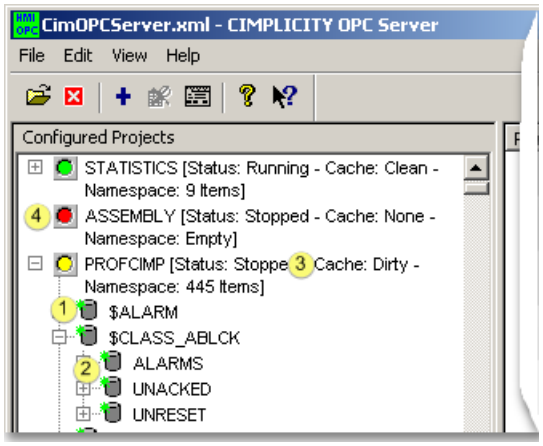


A	Expand the project namespace. **Note:** If the project is not connected, connect it.
B	Expand the TRN1 group. **Note:** Even though there are two TRN1 points, there is only one TRN1 in the tree.
C	The two children, DISC_PRESSURE and FEED_PRESSURE display.
D	Expand the TRN2 group. **Note:** Even though there are two TRN2 points, there is only one TRN2 in the tree.
E	The two children, DISC_PRESSURE and FEED_PRESSURE display.

Result: This tree hierarchy continues for all dot delimited points.

Namespace Behavior

If a project stops the namespace behaves as follows.



1	The namespace icon changes from green to yellow,
2	Namespace points are not immediately removed from the namespace.

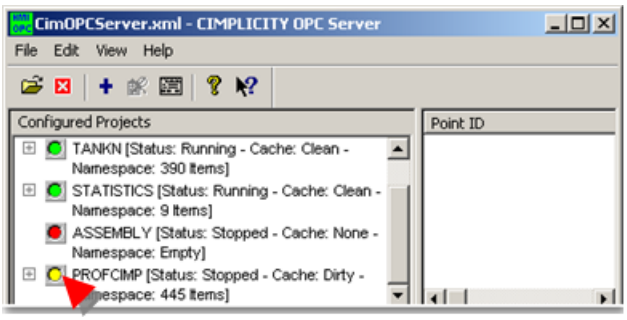
3	An OPC Client can still add items from that project. Clients that have already added items from this project will continue to see the points although the Quality *(page 268)* will be **Dirty**.
4	When the OPC Server detects that there are no clients using points from the stopped project it will: • Remove the project points from the namespace. • Change the icon color to Red. A project that is removed from the configuration (i.e. no longer accessed by the OPC Server), will behave the same as if the project is stopped.

Dynamic Namespaces

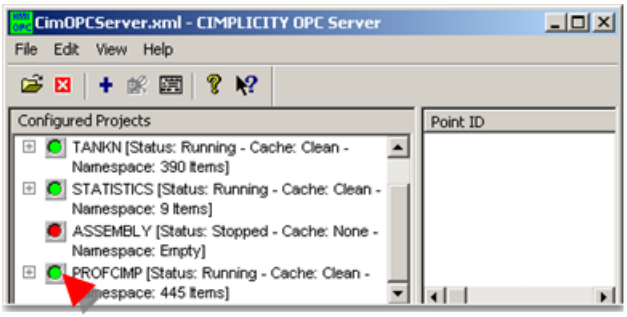
- When a project starts after the OPC Server starts, the OPC Server queries for the project points and adds them to the namespace dynamically without having to stop and restart the OPC Server.

Note: The icon color will change from:

- Red or Yellow



- To Green.



- If a new project is configured to be accessed by the OPC Server, and is already running, its points are immediately added to the namespace.

Browsable Namespace

The namespace that can be browsed is only a subset of the items that are supported by the OPC Server. Therefore, when the OPC server reads point information from a project, not all legitimate points are reported.

9. If a client adds an item that is not displayed in the namespace, the OPC Server checks the project's Point Manager to validate the point.

10. If Point Manager recognizes the item, the OPC Server will accept the client's **Add** request.

    Example

    Although delay load, On Demand, and Point by Address points from a CimView screen are not displayed in the namespace, PTMAP recognizes them and allows them to be referenced by an OPC Client.

    Note: Items added in this manner will not show up in either the OPC Server window left pane or in the Browser namespace, the namespace an OPC Browser sees.

    Maximum Number of Nodes allowed in Tree View

    The only limitation for Tree View nodes is your system's available memory.

5. Review Points in the OPC Server Window List View

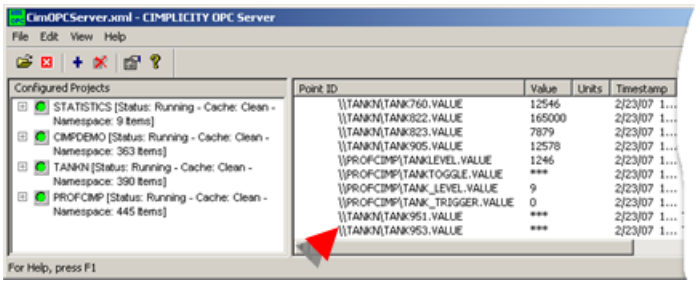
5. Review Points in the OPC Server Window List View

- List View overview.
- List View functionality.

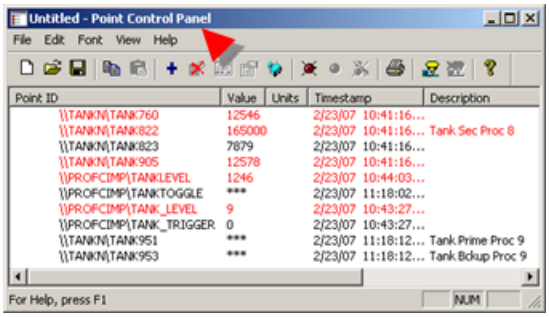
List View Overview

- The List view displays point data using the same layout as the CIMPLICITY Point Control Panel.

CIMPLICITY OPC Server Window List View



CIMPLICITY Point Control Panel



List View Functionality

Do either of the following in the OPC Server window List View.

5.1 *(page 232)*	Add points to the List View.
5.2 *(page 234)*	Delete points from the List View.

5.1. Add Points to the List View

Note: As items are added to the List View, they are added to an OPC Group in the server, and the current value is updated.

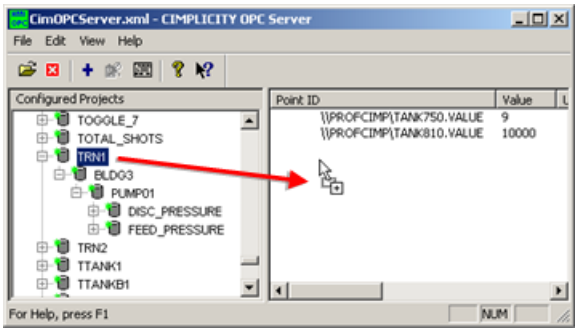
- Drag points from the Tree View.
- Add points from the Select a Point browser.

Drag Points from the Tree View

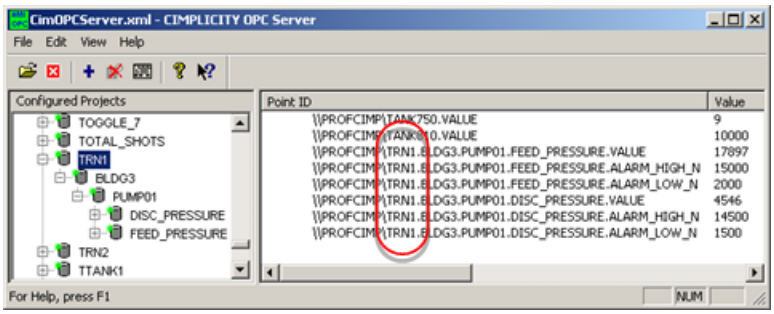
As items are added to the List View, they are added to an OPC Group in the server, and the current value is updated.

Drag a selected point from the OPC Server window left pane to the right pane.

Note: The point may be a delimited point.

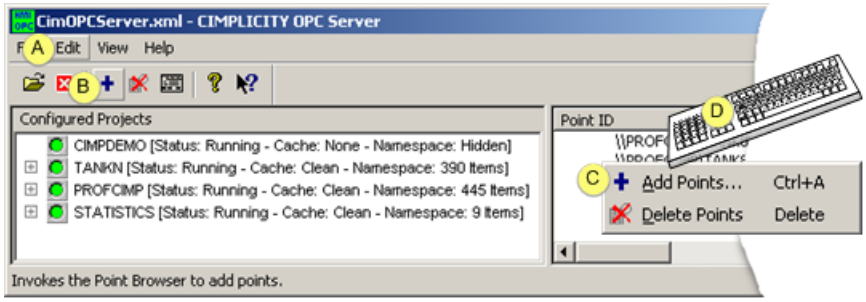


Result: The point (or points) and details are listed in the OPC Server window right pane.



Add Points from the Select a Point Browser

1. Do one of the following.



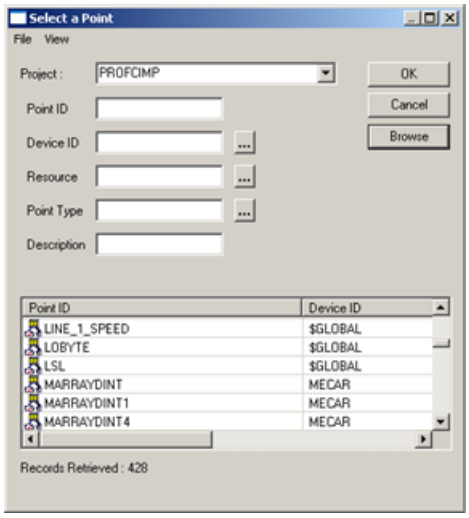
A	Click Edit>Add Points on the OPC Server window menu bar.
B	Click the Add Points button on the OPC Server window toolbar.
C	a. Click the right mouse button in the OPC Server window right pane. b. Select Add Points on the Popup menu.
D	Press Ctrl+A on the keyboard.

The Select a Point browser opens when you use any of these methods.

Projects that are available to browse for points are:

- Attached to the CIMPLICITY OPC Server.
- Running.

2. Click the right mouse button in the OPC Server window right pane.
3. Select Add Points on the Popup menu.
4. Select a project.
5. Browse for points the same way you browse for any CIMPLICITY points.



The selected qualified point names are added to the View List; the project.

Example

\\profcimp\TANK750.value

Where

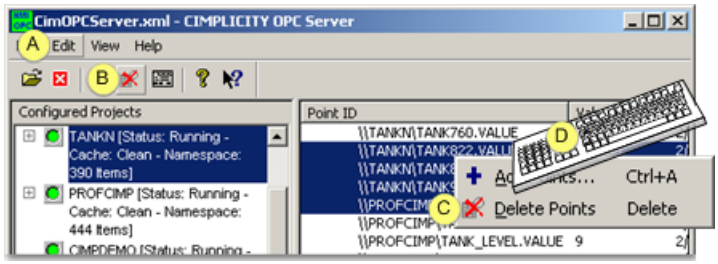
\\profcimp is the project.

TANK750 is a point in profcimp.

value is the attribute *(page 222)*

5.2. Delete Points in the List View

1. Select the point(s) to delete from the OPC Server window.
2. Do one of the following.



A	Click Edit>Delete Points on the OPC Server window menu bar.
B	Click the Delete Points button on the OPC Server window toolbar.

C	a. Click the right mouse button in the OPC Server window right pane. b. Select Delete Points on the Popup menu.
D	Press Delete on the keyboard.

3. Click the right mouse button in the OPC Server window right pane.
4. Select Delete Points on the Popup menu.

6. Close the OPC Server Window

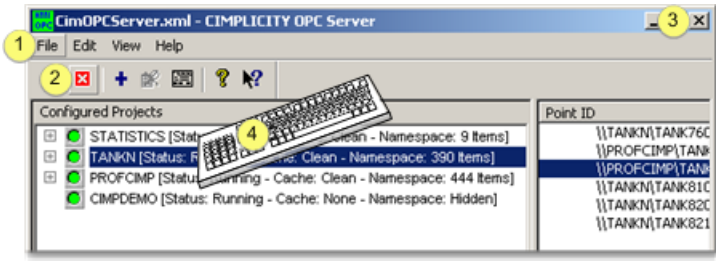
- Stop all attached CIMPLICITY projects.
- OPC Server window exit procedures.

Stop all attached CIMPLICITY Projects

When all attached CIMPLICITY projects stop running, the OPC Server window will automatically shut down.

OPC Server Window Exit Procedures

Do one of the following.



1	Click File>Exit on the OPC Server window menu bar.
2	Click the Exit button on the OPC Server window toolbar.
3	Click the standard Windows Exit button.
4	Press Alt+F+X on the keyboard.

Result: The exact behavior depends on whether or not OPC clients have attached to your interactive OPC server.

- OPC clients are atttached.

1. A warning message box displays informing clients that the OPC Server is about to shut down.
2. A Shutdown event is sent to all attached clients.

3. The OPC server:
    a. Waits 10 seconds for clients to exit
    b. Shuts down.
    • OPC clients are not attached.

4. The OPC Server window closes.

5. The instance of the OPC Server shuts down.

    When running as a hidden process, the OPC server will shut itself down when the last client exits, provided all clients release their references to objects in the server, as they should (per the OPC and OLE specifications).

Troubleshoot OPC Connections

Troubleshoot OPC Connections

1 *(page 236)*	OPC Server diagnostic tools.
2 *(page 248)*	Possible solutions when an OPC Client cannot connect to the OPC Server.
3 *(page 253)*	Limit OPC Server instances.

1. OPC Server Diagnostic Tools

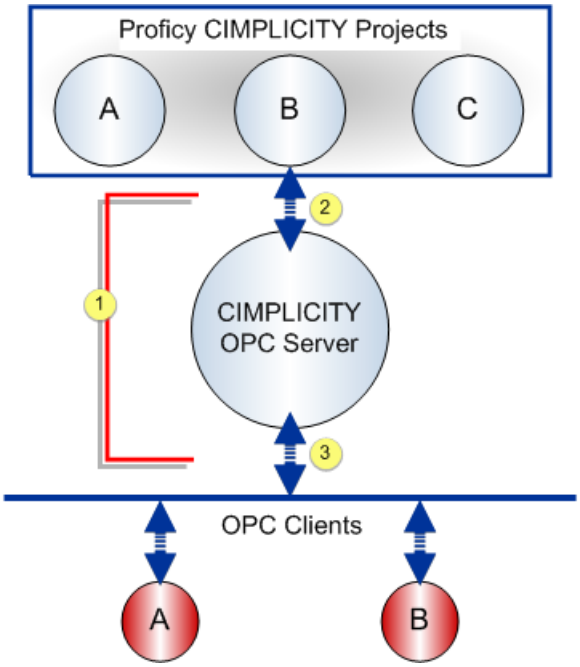
1. OPC Server Diagnostic Tools

The CIMPLICITY OPC Server provides troubleshooting tools to help a user diagnose specific aspects of a conversation between an OPC client and a point in the CIMPLICITY runtime database. There are three tools provided with the OPC server. Use of the tool outputs are discussed in this section. A fourth tool, the CIMPLICITY OPC Server data cache dump is intended for providing detailed diagnostic information for technical support personnel. It is not described here.

- Diagnostic information illustrated.
- Diagnostic tool options.

Diagnostic Information Illustrated

Diagnostic information generated by each tool is illustrated as follows. The output from each tool and how to use it is described in the following sections.



1	Use runtime statistics to monitor through put problems.
2	Use CIMPLICITY project connection logging to diagnose problems between a CIMPLICITY project and the CIMPLICITY OPC Server.
3	Use OPC connection logging to diagnose problems between the CIMPLICITY OPC Server and OPC Clients.

Diagnostic Tool Options

Option 1 *(page 237)*	Set up OPC Connection Trace Logging
Option 2 *(page 246)*	Examine CIMPLICITY Project Connection Logging
Option 3 *(page 243)*	Use Runtime Statistics

Option 1. Set up OPC Connection Trace Logging

Option 1. Set up OPC Connection Trace Logging

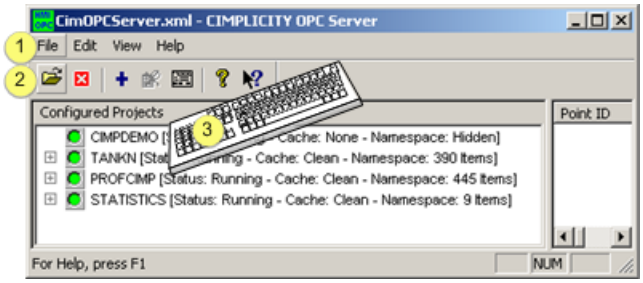
Trace logging monitors the state of a connection between an OPC client(s) and the CIMPLICITY OPC Server.

Information is captured using the Trace Logging diagnostic tool. This tool is used to log information about an OPC conversation between a client and a server to a text file.

1 *(page 238)*	Set the Trace file path.
2 *(page 239)*	Select what will be logged.
3 *(page 239)*	Trace and interpret connections

Set the Trace File Path

1. Do one of the following.



1	Click File>Set Trace Filename on the OPC Server window menu bar.
2	Click the Set Trace File Path button.
3	Press Alt+F+T on the keyboard.

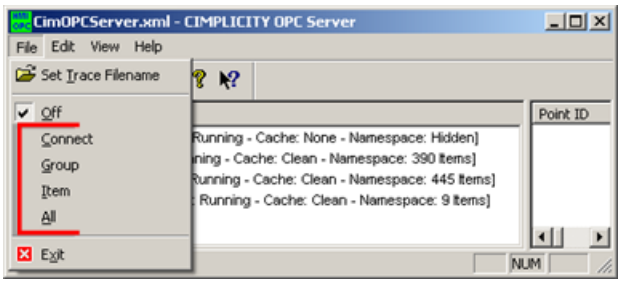
An Open dialog box opens.

1. Select the Trace file location.
2. Enter the name of the trace text file, e.g. CimOPCServerTrace.txt.
3. Click Open.

Result: The Open dialog box closes; logging will be written to the selected file in the selected location.

Select what will be logged

1. Click File on the CIMPLICITY OPC Server menu bar.
2. Check one of the following.



Option	Log
Off	Turn off trace logging
Connect	Server activation events and client connect / disconnect events.
Group	OPC group creation, deletion, and modification events.
Item	OPC item transaction events (read, write, subscription updates).
All	All events: connect, group, item.

Trace and interpret connections

Option 1.1 *(page 239)*	Use the Trace Log.
Option 1.2 *(page 239)*	Interpret the Trace Log.

Note: Messages written to the trace log assume that the user is familiar with the OPC Foundation Data Access specification.

Option 1.1. Use the Trace Log

Information about the interaction between the OPC Server and a CIMPLICITY project is logged to the selected CIMPLICITY Trace Log.

Following is a sample list of problems that can be diagnosed with the Status log. The list is not exhaustive. It is intended as a guide only.

- Client connection problems. Use the log to verify if the OPC Server received the client's request to connect. This request may have been blocked by DCOM security.

- DCOM security authorization problems. This can occur if a client can connect to the OPC Server but cannot access server objects. For example, the client can connect to the OPC Server but cannot create an OPC group.
- Validate the OPC items requested by a client and verify the item ID syntax.
- View the sequence of OPC interface requests to verify the correct operation of a client.
- Verify that a client "gracefully" disconnects.
- Troubleshoot subscription problems (i.e. callbacks into the client by the OPC Server when data changes are reported). In this case, the client is able to perform synchronous and asynchronous read and write requests but cannot receive subscription updates. This may be due to a DCOM security authentication problem on the client machine. The client is unable to authenticate the OPC server.
- See how a client organizes OPC groups and OPC items within groups.

Option 1.2. Interpret the Trace Log

The trace log is composed of a series of messages. Each message logs a single OPC client – server event. A message is prefaced with a local machine date and time stamps and the source of the trace message. The date/time and source are not shown in the following sample trace log for the purposes of clarity.

The sample trace log illustrates the sequence of messages logged with the trace level set to All. The OPC client initiated the following sequence of events (note that the OPC client used to generate the log is Data Access 1.0A compliant).

1. The OPC client connected to the CIMPLICITY OPC Server.

2. The client created an OPC group and called it Group1. The client created two advise sinks for Group1 for:
    - Receiving subscription notification callbacks (i.e. unsolicited updates from the OPC Server for all items in Group1).
    - Receiving asynchronous write complete callbacks (i.e. notification from the OPC Server when an asynchronous write operation completed).

3. The client create a second OPC group called Group2 with similar advise sinks as Group1.

4. The client added an OPC item to Group1 called \\CIMPDEMO\DEMO_COSINE.VALUE. This references the current value of the point DEMO_COSINE in the project CIMPDEMO.

5. The OPC Server immediately began reporting data change notifications to the client (via a callback into the client).

6. The client proceeded to add two more points: \\CIMPDEMO\DEMO_COUNTER.VALUE and \ \CIMPDEMO\DEMO_RANDOM.VALUE.

7. Note that immediately after each new item was added, the number of items reported in the callback to the client increased. This is because the values are changing in CIMPLICITY and being reported to the client at the requested OPC group update rate.

8. The client then removed the OPC group, Group2. Prior to doing this, it disconnects the advise sinks previously set up.

9. The client then deletes the OPC group, Group1. First, it removes the item references from the group and then it disconnects the advise sinks.

10. Finally, the client disconnects from the OPC Server.

```
OPC Client connected
Added OPC Group 'Group1'
Group 'Group1': client connected OPCSTMFORMATDATATIME V1.0 advise sink
Group 'Group1': client connected OPCSTMFORMATWRITECOMPLETE V1.0 advise
 sink
Added OPC Group 'Group2'
Group 'Group2': client connected OPCSTMFORMATDATATIME V1.0 advise sink
Group 'Group2': client connected OPCSTMFORMATWRITECOMPLETE V1.0 advise
 sink
Group 'Group1': added item '\\CIMPDEMO
\DEMO_COSINE.VALUE' (handle=18155968)
Group 'Group1': Invoked V1.0 data change callback (with timestamps) for
 1 item(s)
Group 'Group1': Invoked V1.0 data change callback (with timestamps) for
 1 item(s)
Group 'Group1': added item '\\CIMPDEMO
\DEMO_COUNTER.VALUE' (handle=18157088)
Group 'Group1': Invoked V1.0 data change callback (with timestamps) for
 2 item(s)
Group 'Group1': Invoked V1.0 data change callback (with timestamps) for
 2 item(s)
Group 'Group1': added item '\\CIMPDEMO
\DEMO_RANDOM.VALUE' (handle=18158672)
Group 'Group1': Invoked V1.0 data change callback (with timestamps) for
 3 item(s)
Group 'Group1': Invoked V1.0 data change callback (with timestamps) for
 3 item(s)
Group 'Group2': client disconnected V1.0 OPCSTMFORMATDATATIME advise
 sink
Group 'Group2': client disconnected V1.0 OPCSTMFORMATWRITECOMPLETE
 advise sink
Removed OPC Group 'Group2'
Group 'Group1': Invoked V1.0 data change callback (with timestamps) for
 3 item(s)
Group 'Group1': Invoked V1.0 data change callback (with timestamps) for
 3 item(s)
Group 'Group1': removed item '\\CIMPDEMO
\DEMO_COSINE.VALUE' (handle=18155968)
```

```
Group 'Group1': removed item '\\CIMPDEMO
\DEMO_COUNTER.VALUE' (handle=18157088)
Group 'Group1': removed item '\\CIMPDEMO
\DEMO_RANDOM.VALUE' (handle=18158672)
Group 'Group1': client disconnected V1.0 OPCSTMFORMATDATATIME advise
 sink
Group 'Group1': client disconnected V1.0 OPCSTMFORMATWRITECOMPLETE
 advise sink
Removed OPC Group 'Group1'
OPC Client disconnected
```

Option 2. Use Runtime Statistics

Option 2. Use Runtime Statistics

STATISTICS that display in the CIMPLICITY OPC Server window:

- Report runtime performance for for the OPC Client - OPC Server interactions.
- Are internal data provided by the NDI toolkit.
- Can be used to diagnose computer node performance problems and to tune an OPC client's use of the CIMPLICITY OPC Server resources.

Use the statistics to identify the general problem and then use the OPC Connection Trace Logging to identify the specific problem.

Option 2.1 *(page 242)*	Display runtime statistics.
Option 2.2 *(page 243)*	Review runtime statistics.

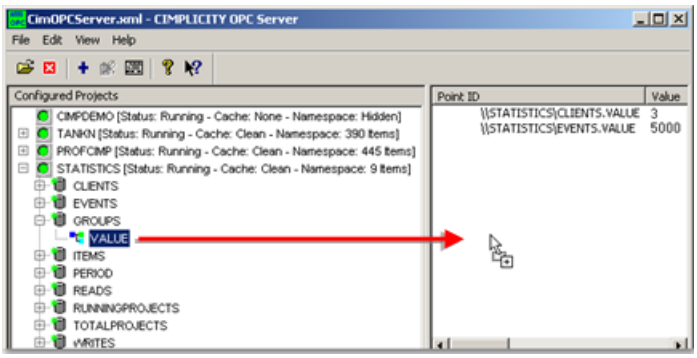
Option 2.1. Display Runtime Statistics

1. Expand STATISTICS in the OPC Server left pane.

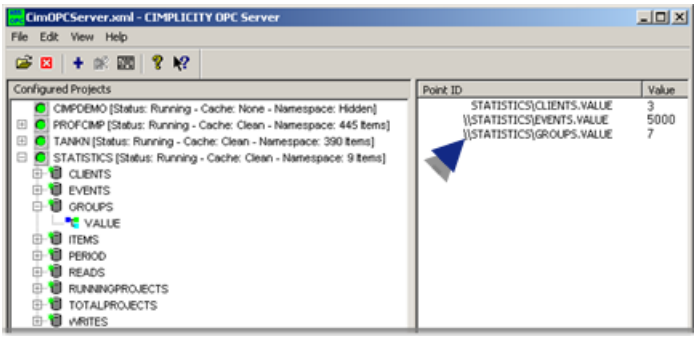
   The statistics categories are listed.

2. Expand a category.

3. Drag VALUE from the OPC Server window left pane to the right pane.

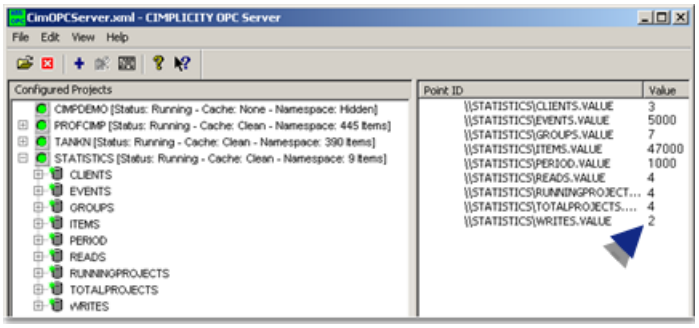


An associated internal OPC Server point ID and details display in the right pane.



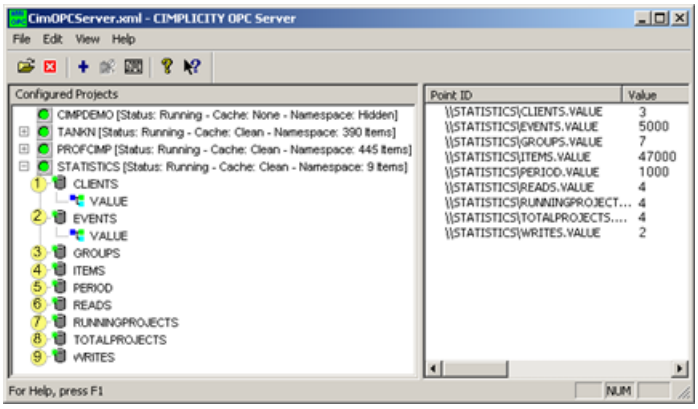
4. Continue dragging the VALUE for each category to be analyzed.

The right pane displays the internal data that can be used for analysis.



Option 2.2. Review Runtime Statistics

OPC Server runtime statistics are as follows.



rect 15, 103, 103, 127 *(page 245)*
rect 270, 60, 422, 72 *(page 245)*
rect 13, 212, 101, 229 *(page 246)*
rect 271, 137, 412, 149 *(page 246)*
rect 13, 202, 101, 214 *(page 246)*
rect 271, 127, 409, 139 *(page 246)*
rect 13, 192, 101, 204 *(page 246)*
rect 13, 192, 101, 204 *(page 246)*
rect 270, 117, 414, 129 *(page 246)*
rect 13, 180, 101, 192 *(page 246)*
rect 277, 107, 418, 119 *(page 246)*
rect 13, 170, 101, 182 *(page 245)*
rect 272, 97, 427, 109 *(page 245)*
rect 13, 160, 101, 172 *(page 245)*
rect 269, 87, 426, 99 *(page 245)*
rect 13, 150, 101, 162 *(page 245)*
rect 268, 81, 421, 89 *(page 245)*
rect 14, 126, 102, 151 *(page 245)*
rect 268, 71, 421, 82 *(page 245)*

1 *(page 245)*	CLIENTS
2 *(page 245)*	EVENTS
3 *(page 245)*	GROUPS
4 *(page 245)*	ITEMS
5 *(page 245)*	PERIOD

6 *(page 246)*	READS
7 *(page 246)*	RUNNINGPROJECTS
8 *(page 246)*	TOTALPROJECTS
9 *(page 246)*	WRITES

1	CLIENTS

Point ID	\\STATISTICS\CLIENTS.VALUE
Description	Number of OPC clients currently connected to the CIMPLICITY OPC Server.

2	EVENTS

Point ID	\\STATISTICS\EVENTS.VALUE
Description	Number of value updates have been sent to attached OPC clients in the last PERIOD (second).

3	GROUPS

Point ID	\\STATISTICS\GROUPS.VALUE
Description	Number of OPC Group objects have been created by all attached clients. **Note:** Some OPC client applications initially create a large number of OPC groups and disable the subscription updates until needed. While this will not cause CPU loading problems, it could cause the initial connection and setup time with the CIMPLICITY OPC Server to be slow or for a large amount of memory to be used by the OPC Server.

4	ITEMS

Point ID	\\STATISTICS\ITEMS.VALUE
Description	Total number of items (points) the groups contain. **Important** • ITEMS is not the number of items available from the OPC server. It is the number of items that OPC clients have requested the values for. • The same item may appear in more than one group (or be referenced by more than one OPC client). Each reference is counted in this statistic.

5	PERIOD

Point ID	\\STATISTICS\PERIOD.VALUE
Description	How often the Toolkit updates the statistics 1000 = 1 second.

6	READS

Point ID	\\STATISTICS\READS.VALUE
Description	Number of read requests have been made by attached clients in the last sample PERIOD (second). A read transaction is composed of one or more items and may be a cache read or device read. **Note:** Cache reads are very efficient and do not typically cause significant CPU loading problems.)

7	RUNNINGPROJECTS

Point ID	\\STATISTICS\RUNNINGPROJECTS.VALUE
Description	Number of projects attached to the OPC Server that are running.

8	TOTALPROJECTS

Point ID	\\STATISTICS\TOTALPROJECTS.VALUE
Description	Total number of projects attached to the OPC Server, running and not running.

9	WRITES

Point ID	\\STATISTICS\WRITES.VALUE
Description	Number of write requests (set points) that have been made by attached clients in the last sample PERIOD (second).

Option 3. Examine CIMPLICITY Project Connection Logging

- CIMPLICITY OPC Server and the Status Log.
- Messages common to normal interactions.

CIMPLICITY OPC Server and the Status Log

The CIMPLICITY OPC Server interacts with the CIMPLICITY runtime database on behalf of the OPC clients. Warning and failure messages generated by this interaction are logged to the CIMPLICITY Status Log . Informational messages (or success messages) are also logged.

Success Messages	Indicate normal state changes between the runtime database and server interactions.
Warning Messages	Indicate that a runtime database / server interaction was not successful but that the problem will not affect future interactions.
Failure Messages	Indicate that a runtime database / server interaction has failed and that future interactions will most likely fail.

Whenever possible, the OPC Server logs error messages generated by the runtime database (referred to as PTMAP in error messages). When a PTMAP error message is available, the message is logged to the Status Log after the message generated by the OPC Server, creating a two-part message.

Note: The CIMPLICITY OPC Server defines several item attributes that are recognized only by the OPC Server. These attributes are internally maintained by the OPC Server. Thus, some two-part warning and failure messages display an item ID (as requested by an OPC client) that differs from the point reference in the message generated by the runtime database. This is not an error.

Messages Common to Normal Interactions

Following are messages common to normal interactions between the CIMPLICITY runtime database and the OPC server.

Note: If other messages are logged, consult with CIMPLICITY customer support . The second part of two-part messages are not shown.

OPC Server connected to PTMAP.

A success message indicating that the CIMPLICITY OPC Server has created a connection with the CIMPLICITY runtime database.

OPC Server disconnected from PTMAP.

A success message indicating that the CIMPLICITY OPC Server has successfully disconnected from the CIMPLICITY runtime database.

OPC client request to connect failed. CIMPLICITY OPC Server is not licensed.

A warning message indicating that the running OPC server has not been licensed for use. No OPC client connections are permitted.

The CIMPLICITY PTMAP IPC system is inactive. OPC client connections refused.

A warning message indicating that the sub-system supporting the communications between the CIMPLICITY runtime database and the OPC Server is no longer running. This can occur when a server is active (with OPC clients connected) and the last CIMPLICITY project on the same node as the OPC Server is stopped. OPC clients must disconnect before the OPC Server automatically terminates (or is deactivated via the user interface).

Item \\project\point.attribute add failed. See next error.

A warning message indicating that a request to add a new point to the OPC Server's internal cache has been rejected by the runtime database. Refer to the following message for more information on why the request was rejected.

Item \\project\point.attribute write failed. See next error.

A warning message indicating that a point write request by the OPC Server has been rejected by the runtime database. Refer to the following message for more information on why the request was rejected.

Item \\project\point.attribute read failed. See next error.

A warning message indicating that a point read request by the OPC Server has been rejected by the runtime database. Refer to the following message for more information on why the request was rejected.

Item \\project\point.attribute add on change request failed. See next error.

A warning message indicating that a registration for change notifications for a point (used by the server for subscriptions and cache reads/writes) has been rejected by the runtime database. Refer to the following message for more information on why the request was rejected.

2. Possible Solutions when an OPC Client cannot Connect to the OPC Server

2. Possible Solutions when an OPC Client cannot Connect to the OPC Server

Start your OPC client application.

Each client application will provide a unique method of referencing OPC servers. One common method is to display a list of OPC servers that are visible to the OPC client. If this is the case, pull up this list. The CIMPLICITY OPC Server reference you are looking for may be one of the following:

- CIMPLICITY.HMI.OPCServer
- CIMPLICITY OPC Server
- {B01241E8-921B-11d2-B43F-204C4F4F5020}

Note: The OPC Server user interface is not visible when you connect to it via an OPC Client.

Check the following possible solutions if you are having difficulty connecting with the OPC Server, using any of the server objects, or receiving subscription updates,

Solution 1 *(page 249)*	Set up the OPC Client.
Solution 2 *(page 250)*	Register an OPC Server on an OPC Client machine.

Solution 3 *(page 251)*	Identify a remote OPC Server on an OPC Client machine.

Note: These options are valid only if the OPC client does not programmatically configure the DCOM security settings. Consult with the OPC client documentation to determine if DCOM security settings are set up by the application directly.

Solution 1. Set up the OPC Client

1. Open the My Computer Properties dialog box on the Windows XP\2003 Client node as follows.
   a. Click Start on the Windows task bar.
   b. Select Run on the Start menu.

   The Run dialog box opens.

   a. Type `dcomcnfg` in the **Open** field.

2. Double-click **Component Services** in the Component Services left pane.

   Component Services expands.

3. Double-click **Computers**.

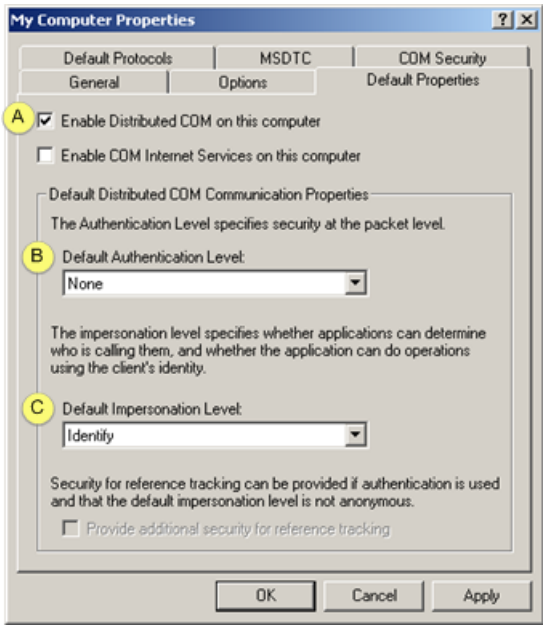
4. Right-click **My Computer**.

   The My Computer Properties dialog box opens.

5. Select Properties on the Popup menu.

6. Select the Default Properties tab.

7. Select the following.



	Option	Description
A	Enable Distributed COM on this computer	Check
B	Default Authentication Level	Select None.
C	Default Impersonation Level	Select Identify.

Solution 2. Register an OPC Server on an OPC Client Machine

Remote OPC client applications (i.e. client applications that run on a computer node other than the one running the CIMPLICITY OPC Server) do not have to have CIMPLICITY software installed on a remote machine in order to access the CIMPLICITY OPC Server.

If the OPC Client cannot connect to the OPC Server, a solution may be to update the registry on the remote client machine with information specific to the CIMPLICITY OPC Server.

A remote OPC client registry entry file (CIMOPCServerClient.reg) ships with the OPC server. These registry entries will point the OPC client to the CIMPLCITY OPC Server across a network.

CIMOPCServerclient.reg is located in the following directory on the OPC Server:

C:\Program Files\Proficy\Proficy CIMPLICITY\dc\OPCServers\Redist

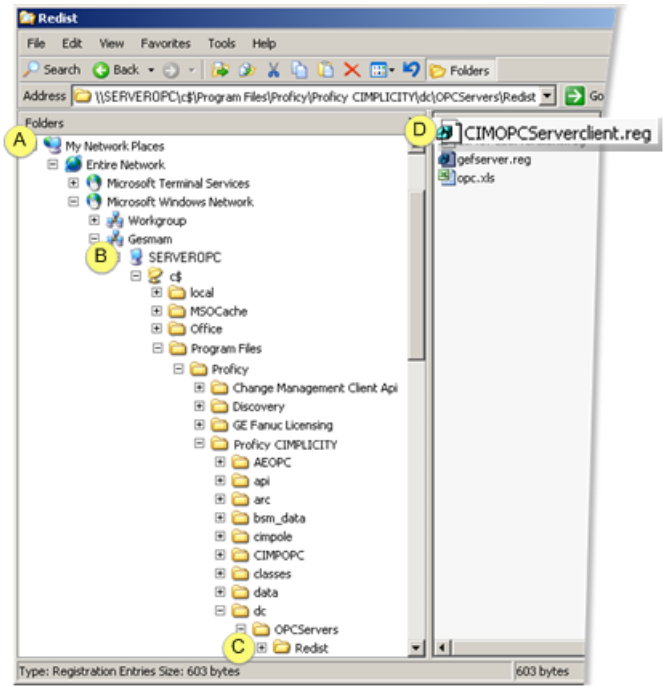
Where

C:\Program Files\Proficy\Proficy CIMPLICITY... is the default location on the CIMPLICITY server.

1. Log onto the remote client node using an account with administrator privileges.

2. Open Windows Explorer.

3. Do the following.



A	Expand My Network Places
B	Find the CIMPLICITY OPC Server node.
C	Open the C:\Program Files\Proficy\Proficy CIMPLICITY\dc\OPCServers\Redist folder on the CIMPLICITY OPC Server node.
D	Double-click CIMOpcServerClient.reg.

Result: The registry on the remote client computer is automatically updated. This registers the OPC server on the client node so that it can be found by OPC client programs.

Note: You can also:

4. Copy CIMOpcServerClient.reg from the OPC Server to, for example, a CD or a flash drive.

5. Insert and double-click it at the OPC Client machine.

Solution 3. Identify a Remote OPC Server on an OPC Client Machine

A possible solution, if the OPC Client cannot find the CIMPLICITY OPC Server, is to identify the OPC Server in the CIMPLICITY OPC Server properties dialog box.

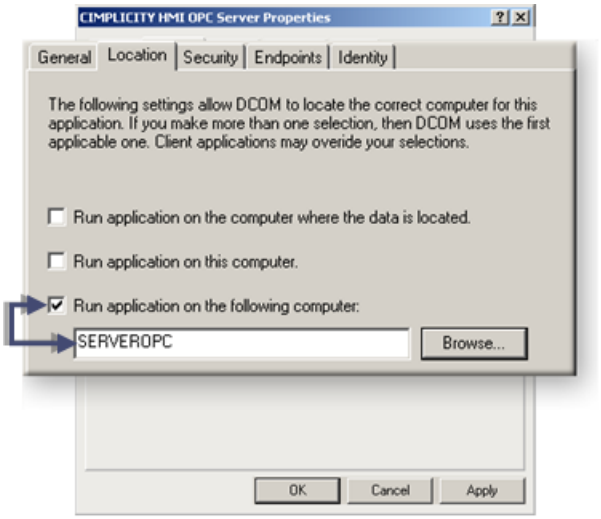
1 *(page 252)*	Client Location
2 *(page 252)*	Client General

Client Location

Note: If you do not see CIMPLICITY OPC Server in the DCOM Config folder, reboot your computer before completing this procedure.

1. Open the CIMPLICITY OPC Server Properties dialog box on the Windows XP or Windows 2003 client node.
2. Select the Location tab.
3. Check Run application on the following computer.
4. Enter the computer name that is the CIMPLICITY OPC Server in the **....following computer** field.

Tip: Click Browse to find the computer.



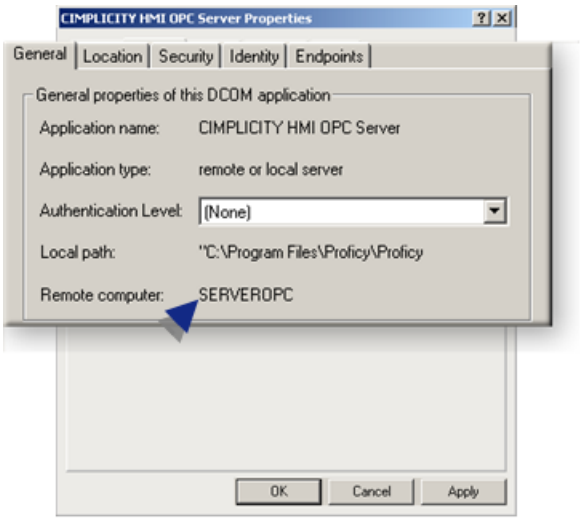
1. Click Apply.

Client General

1. Select the General tab in the CIMPLICITY OPC Server Properties dialog box on the client node.

- The General tab confirms that:
- The CIMPLICITY OPC server object is registered on the client node.

Another node is specified for activation/access of the object.



1. Click OK.

Result: The CIMPLICITY OPC Server Properties dialog box closes.

A client application on the remote node should now be able to reference the CIMPLICITY OPC Server (given that DCOM security issues have been addressed).

ID's are:

ProgID of the OPC Server	CIMPLICITY.HMI.OPCServer
CLSID	{B01241E8-921B-11d2-B43F-204C4F4F5020}

3. Limit OPC Server Instances

3. Limit OPC Server Instances

Microsoft DCOM configuration security determines what instance of the OPC Server a client can attach to.

As a default, each time an OPC client launches the OPC server, the server is launched using the privileges and permissions of the user currently logged on.

If clients have different privilege levels DCOM will open separate instances of the OPC Server to accommodate the different privileges.

Beginning with CIMPLICITY 7.0 there is no inherent problem with running more than one instance of the OPC Server.

However, each instance does take up memory and resources.

If you need to conserve your Windows XP or Windows 2003 system memory and resources you can configure DCOMCNFG identity settings to run as an Interactive User.

Step 3.1 *(page 254)*	Open the CIMPLICITY HMI OPC Server Properties dialog box.
Step 3.2 *(page 255)*	Set DCOMCNFG settings for the CIMPLICITY OPC Server.
Step 3.3 *(page 257)*	Select DCOM access for the selected user.

Step 3.1. Open the CIMPLICITY HMI OPC Server Properties Dialog Box

1. Click Start on the Windows task bar.
2. Select Run on the Start menu.

   The Run dialog box opens.

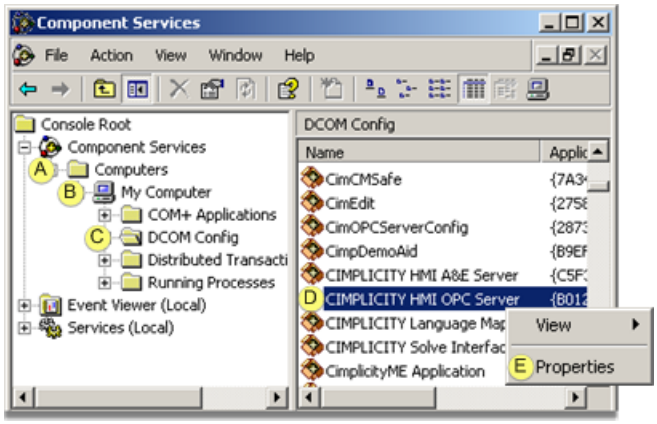
3. Type `dcomcnfg` in the **Open** field.

   The Component Services window opens.

4. Double-click **Component Services** in the Component Services left pane.

   Component Services expands.

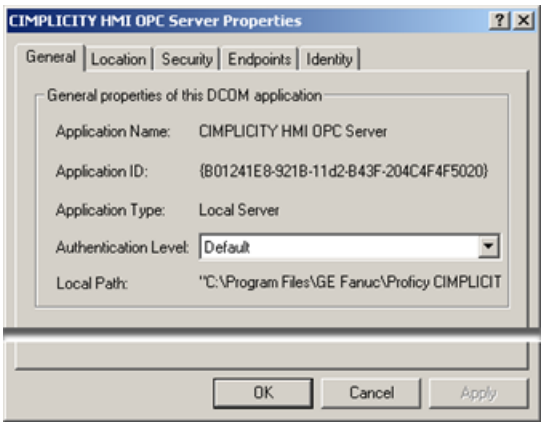
5. Do the following.



A	Double-click **Computers**.

B	Expand **My Computer**.
C	Select **DCOM Config**.
D	Right-click CIMPLICITY HMI OPC Server in the Component Services window right pane.
E	Select Properties on the Popup menu.

The CIMPLICITY HMI OPC Server Properties dialog box opens.



Step 3.2. Set DCOMCNFG Settings for the CIMPLICITY OPC Server

Set DCOMCNFG setttings for the OPC Server on a Windows XP or Windows 2003 machine as follows.

1 *(page 255)*	Identity tab.
2 *(page 257)*	Security tab.

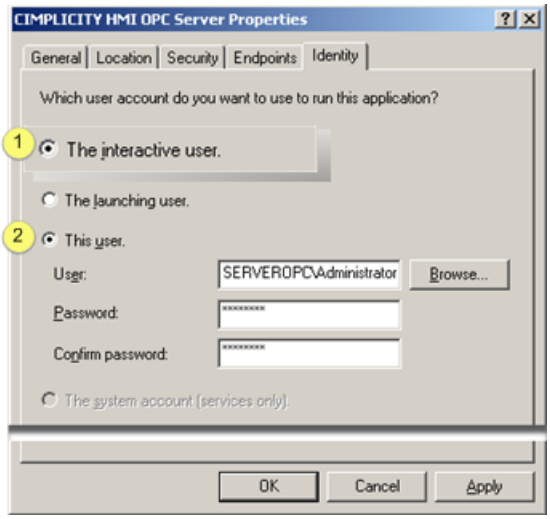
Identity tab

Important:

- The OPC server will take on the identity (and privileges and permissions) of the user account when activated by an OPC client. The chosen user account (either The interactive user or This user) must be authenticated on a remote OPC client node in order for some OPC interactions to occur (e.g. subscription notifications).
- You must have identical user accounts with identical passwords on the OPC server and remote node in order for the remote node to connect with the server.

1. Select the Identity tab in the CIMPLICITY HMI OPC Server Properties dialog box.

2. Check either of the following.



1 *(page 256)*	The interactive user
2 *(page 256)*	This user

Important: Regardless of the method chosen, the account must be part of the USER group, at a minimum.

1	The interactive user

Checking The Interactive user does the following.

Description	The OPC Server: • Is launched with access to the user interface. • Allows other interactive clients to attach to the server.
Limitation	The server will shut down after a log out and data collection will be stopped. However, CIMPLICITY may still be running.
Use for	Tasks such as troubleshooting or initial setup

2	This user

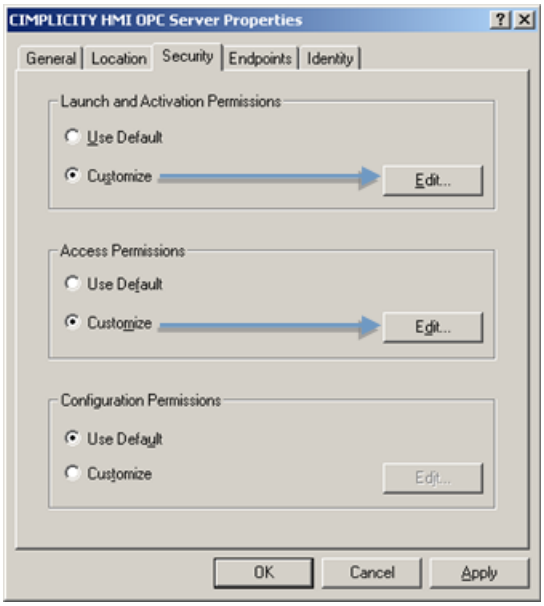
Checking This user does the following.

User/ Password	User and Password fields are enabled. Enter a valid **User** and **Password**. Examples • Administrator account. • The current interactive user. • A special account set up for the OPC server.
Description	The OPC Server • Runs as a background process, as if the entered user was logged in. • Allows other interactive clients to attach to the server and continues to run after log out; data collection continues.
Use for	Continuous activity. Most often, the CIMPLICITY OPC Server is configured to run as a background process (i.e. non-interactive mode) with the OPC Server starting and stopping as OPC clients connect and disconnect.

Security tab

1. Select the Security tab in the CIMPLICITY HMI OPC Server Properties dialog box.
2. Select **Launch and Activation** and **Access** permissions for the interactive or selected user.

Consult Microsoft documentation for details about permissions options.

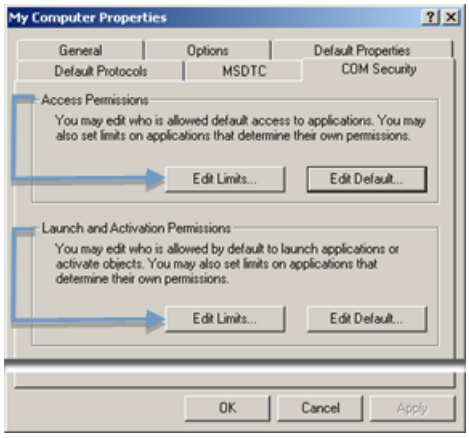


Note: Typically, check Use Default for Configuration Permissions.

Step 3.3. Select DCOM Access for the Selected User

1. Open *(page 249)* the My Computer Properties dialog box.

2. Select the COM Security tab.
3. Add the user selected on the Identity tab.



Note: Consult Microsoft documentation for details about DCOM configuration.

CIMPLICITY OPC Server Technical Notes

CIMPLICITY OPC Server Technical Notes

- DCOM security overview
- CIMPLICITY OPC specifications.
- CIMPLICITY project security.
- CIMPLICITY project point Item ID syntax.
- Point by Address item ID syntax.
- Data types
- Timestamps
- Quality

DCOM Security Overview

It is very rare that DCOM security for the CIMPLICITY OPC Server will remain off. Most environments require controlled access to systems interacting with a manufacturing process.

When communications between the OPC Server and your OPC client application are satisfactory, you will want to consider enabling DCOM security.

The degree of security enabled is dependent upon the policies at the site where the OPC Server is installed.

The topic of DCOM security (and Windows security for that matter) is extensive and can be confusing. There are several books dedicated to these topics alone. In addition, detailed information about DCOM security can be found at the OPC Foundation Web site, www.OPCFoundation.org, and the Microsoft Web site, www.Microsoft.com.

Brief definitions of DCOM security settings for CIMPLICITY OPC Server / OPC client interactions are as follows.

- Authentication security
- Domain authentication
- Authorization security
- Activation security

Dcomcnfg.exe and 64-bit Applications Accessed by Remote Clients

On x64 operating systems before Windows 7 and Windows Server 2008 R2, the 64-bit version of DCOMCNFG.EXE does not correctly configure 32-bit DCOM applications for remote activation. The workaround is to use the 32-bit version of DCOMCNFG using the following command line to register the 32-bit DCOM applications for remote activation:

C:\WINDOWS\SysWOW64>mmc comexp.msc /32

Authentication security

Authentication security ensures that the interaction between an OPC client and the CIMPLICITY OPC Server is legitimate. Authentication security for DCOM is an extension of the standard Windows operating system security (which itself is layered upon secured RPC (remote procedure call)). Authentication poses the question "Is the OPC client who it says it is?" and "Is the OPC server who it says it is?". The user configures the level of authentication required, which specifies how often this question is posed. Each more secure level places extra processing overhead on communications between the OPC client and the OPC server. A client and server negotiate to the highest level of authentication when the configured authentication levels differ.

For example, authentication can be required only at OPC client connection time to a server (level = Connect). Once a client is connected (and is authorized to use the OPC Server), all interactions are performed without further authentication. As another example, authentication can be required at the packet level (level = Packet Privacy), with each packet being fully encrypted. The choice of the authentication level is dependent on the security policies of the user.

In a multi-node computing environment, the security system on the computer node running the OPC server must be able to verify that the security ID of the OPC client is valid. In a domain environment,

domain accounts must be validated. In peer-to-peer environments, matching local user accounts must be configured.

Authentication of an OPC client must be satisfied before authorization and activation permissions are checked. If a client cannot be authenticated, permission checking for the requested action is not performed.

Domain authentication

A domain authentication architecture provides the lowest cost solution (from a maintenance perspective) for DCOM security. If you are using a domain, then follow (or ask your network administrator to follow) these general setup guidelines:

- Create a new domain group. Users who are part of this group will be allowed to launch the CIMPLICITY OPC Server and access its objects.
- Add the new group to the launch permissions and access permissions for the CIMPLICITY OPC Server using the DCOMCNFG utility.
- Make sure all user accounts that run an OPC client application are part of this new group.

Authorization security

Authorization security occurs once an OPC client transaction has been authenticated. At that time DCOM security must determine if that OPC client is authorized to perform call-level interactions with the OPC server. (COM/DCOM technology allows OPC client applications to make programmatic calls across process and computer node boundaries.) This determination is made by looking at the ACL (access control list) for the OPC server COM object. This ACL (or list of users and / or user groups) for the OPC server is configured using the DCOMCNFG utility supplied with the Windows operation system.

If the OPC client's user identity is listed on the OPC server's access permissions ACL (as a user or group member), then the OPC client can access CIMPLICITY OPC Server objects.

To restrict access of OPC clients to a CIMPLICITY OPC Server that is already running (authorization security), modify the access control list (ACL) of the OPC server by editing the custom access permissions with DCOMCNFG.

Activation security

Activation security is unique to DCOM. The DCOM framework provides the ability for an OPC client to access the CIMPLICITY OPC Server object. If the OPC server object is installed on another computer node, then the framework launches (or activates) the OPC server (if it is not already running) on behalf of the client. Activation permission checking works the same as authorization permission checking. An authenticated client's user identity is checked against the OPC server's ACL for launch permissions. Activation permissions for the CIMPLICITY OPC Server are set up using DCOMCNFG.

Enable CIMPLICITY OPC Server activation security by editing the custom launch permissions for the OPC Server and specifying known users and / or groups.

Note: As a general rule, the activation security should always be more restricted than the authorization security. This prevents the situation where an OPC client can activate the CIMPLICITY OPC Server, but cannot use the OPC Server objects.

CIMPLICITY OPC Specifications

Refer to the OPC Foundation specification documents for more information on details presented in this section.

- OPC Specification Compliance.
- COM Program ID.
- Supported Data Access Custom Interfaces.
- Supported Automation Interfaces.
- OPC Group Object Percent Dead Band.
- Blobs.

OPC Specification Compliance

The CIMPLICITY OPC Server is compliant with the following OPC standards:

- Data Access Custom Interface Standard V3.0, V2.0, and V1.0a.
- Data Access Automation Interface Standard V3.0 and V2.0.

Note: For specific information about the differences between 2.x and 3.x OPC specifications, refer to the OPC Foundation web site.

COM Program ID

Once installed, the CIMPLICITY OPC Server is typically referenced by an OPC client by its ProgID (program ID). The ProgID for the CIMPLICITY OPC Server is:

CIMPLICITY.HMI.OPCServer

The OPC Server's unique CLSID (class ID) is:

`{B01241E8-921B-11d2-B43F-204C4F4F5020}`

Supported Data Access Custom Interfaces

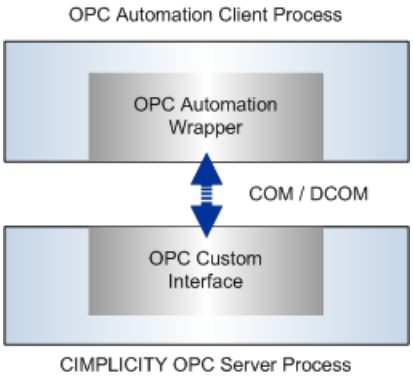
The CIMPLICITY OPC Server implements the following COM interfaces. Note that standard COM interfaces are not listed.

IOPCServer	IOPCBrowseServerAddressSpace
IOPCItemProperties	IConnectionPointContainer
IOPCCommon	IOPCGroupStateMgt
IOPCASyncIO2	IOPCAsyncIO
IOPCItemMgt	IConnectionPointContainer
IOPCSyncIO	IDataObject
EnumOPCItemAttributes	IEnumOPCItemAttributes

Supported Automation Interfaces

The CIMPLICITY OPC Server relies on the standard automation wrapper supplied by the OPC Foundation.

The automation wrapper interacts with the automation client (e.g. Visual Basic for Applications script) and the custom interface of the CIMPLICITY OPC Server.



Note:

- The automation wrapper executes in-process with the automation client.
- Selection of the type of interface to use, custom or automation, depends on the goals of the client application developer.

Adhoc client applications written in Microsoft Visual Basic (for example) typically use the automation interface.

- Automation interfaces are easy to use in the VB (and VBA) development environments.

However, the automation interface is slower at execution time.

- Applications written in Microsoft Visual C++ (for example) use the custom interface.

This is the most efficient interface, but it is more complex to use.

OPC Group Object Percent Dead Band

Percent dead-banding is not supported directly by the CIMPLICITY OPC Server when a client configures an OPC group.

The CIMPLICITY runtime database provides support for dead band change notifications.

Dead banding is configured when a point is added to a CIMPLICITY project.

If a client specifies a percent dead-band value for a group, the value is ignored.

Change notifications are reported to an OPC client based on the dead-banding configured for a point in the CIMPLICITY project.

Blobs

The CIMPLICITY OPC Server does not support the use of blobs. Refer to the Data Access specification for more information on blobs.

CIMPLICITY Project Security

CIMPLICITY project security is extended to all clients of the CIMPLICITY OPC Server. The OPC Server acts as a proxy for an OPC client, granting secure access to a CIMPLICITY project point using the username and the password attributed to the OPC Server.

Security settings are attributed to a server using the Security Dialog of the OPC Server interface.

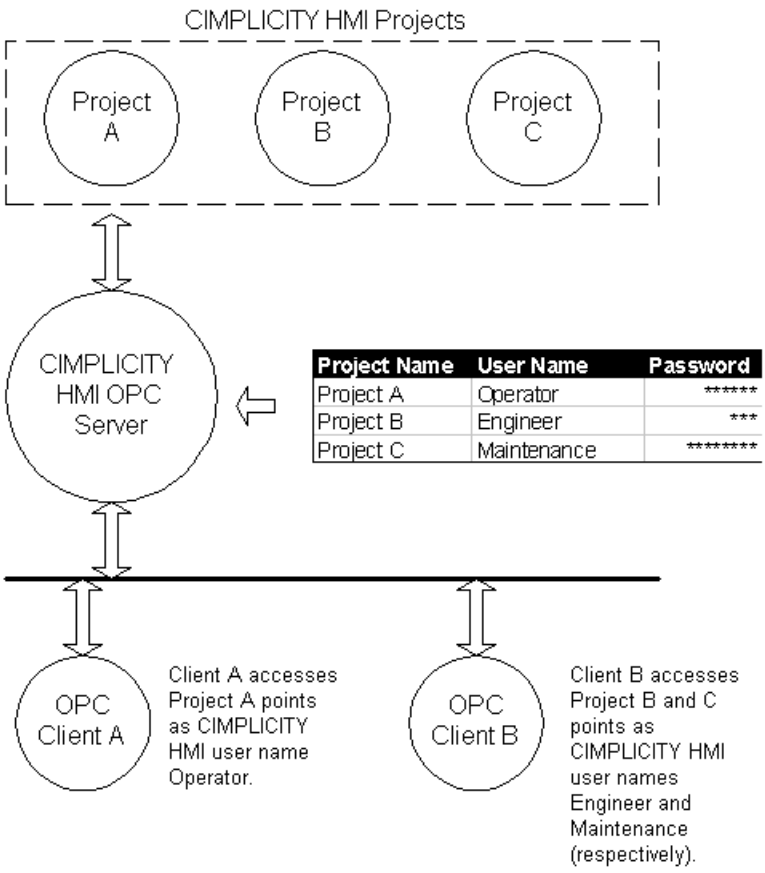
The OPC Server always runs on the same computer node as a CIMPLICITY project(s).

In order for an OPC client to successfully access points of a project, a username and a password must be specified for each CIMPLICITY project that:

- Is visible on this computer node, and
- Must be made accessible to an OPC client.

When an OPC client references a point in a project, the OPC Server connects to the project using the username and the password specified for the project.

The following diagram illustrates the security settings granted to two OPC client applications accessing three CIMPLICITY projects via the OPC server.



CIMPLICITY Project Point Item ID Syntax

Points configured in a CIMPLICITY project are referenced for read and / or write operations using the following syntax. Each field is described below.

\\PROJECT\POINT.ATTRIBUTE

Field	Description	
PROJECT	Required	A CIMPLICITY project name under which the reference is made. **Important:** When using point by address for the item IDs used by an OPC Client, the project ID must be included.
POINT	Required	The name of a CIMPLICITY project point.
ATTRIBUTE	Required	A server-defined string specifying the type of information associated with the point. A point has several attributes.

Point by Address Item ID Syntax

Point by address Item IDs allow an OPC client to explicitly refer to device registers for devices that currently communicate with a CIMPLICITY project. A CIMPLICITY project point does not necessarily need to be configured.

Note: Point by Address Item IDs do not appear during an OPC client browse session. These Items IDs must be manually entered in an OPC client application.

The Point by Address Item ID syntax is shown below where `[ ]` indicates an optional keyword. Valid keywords (required and optional) are described below.

\\PROJECT\@DEVICE=xxx|ADDR=yyy|[TYPE=zzz]

Keyword/ Field	Description		
PROJECT	Required	A CIMPLICITY project name under which the reference is made. **Important:** When using point by address for the item IDs used by an OPC Client, the project ID must be included.	
DEVICE	Required	Any valid CIMPLICITY device identifier.	
ADDR	Required	A valid device address for the specified device.	
TYPE	Optional	Any valid CIMPLICITY point type.	
		Default	`INT`
SCAN	Optional	Multiple of the device scan rate at which the data will be collected.	
		Default	1
OFFSET	Optional	Bit offset for the address of BOOL, BYTE or WORD points	
		Default	0
ACCESS	Optional	Either READ or WRITE.	
		Default	`READ`
ELEM	Optional	The number of elements (for an array).	
		Default	1
ORIGIN	Optional	The point's origin - use one of the following:	
		ORIGIN	Point Type
		DEV	Device .
		DIA	Diagnostic
		ALW	Ethernet Global Data
		Default	`DEV`

Data Types

- Data type table
- Data type coercion
- Array support
- Dynamic configuration mode

Data Type Table

The CIMPLICITY OPC Server represents CIMPLICITY point values in a canonical (or baseline) format. This format, or data type, is compatible with Microsoft COM/DCOM technology and is called a VARIANT data type.

Each Item ID attribute has a pre-defined canonical data type. The canonical data types for VALUE and RAW_VALUE attributes are dependent on the CIMPLICITY point type. The following table maps the CIMPLICITY point type to the canonical form. The table is grouped by CIMPLICITY point class.

CIMPLICITY Point Type	OPC Server Canonical Form
Analog	DINT
	INT
	REAL
	SINT
	UDINT
	UINT
	USINT
Boolean	BOOL
	BYTE
	WORD
	DWORD
Text	STRING
	STRING_20
	STRING_8
	STRING_80

Data Type Coercion

To ensure the highest throughput of point values through the CIMPLICITY OPC Server to an OPC Client, the client should always request the canonical data type of an attribute.

For example, if an OPC client wants to subscribe to changes in a CIMPLICITY analog class point configured as a real point type, the fastest throughput is achieved by requesting the value (when added to a group) as an eight byte real value (VT_R8).

By requesting a point in canonical form, the OPC Server does not have to coerce (or convert) between the data type stored internally and the data type requested by the OPC client.

The OPC Server provides coercion support for all non-array OPC items. The OPC Server utilizes standard Microsoft coercion support routines. A drawback to relying on coercion is the penalty of extra processing overhead required for each transaction. However, relying on coercion in the OPC Server may simplify the OPC client or provide the user with the ability to select the data type most applicable.

Array Support

The CIMPLICITY OPC Server supports arrays of all CIMPLICITY point types (with the exception of the STRING type). The OPC Server does not support coercion of array items. OPC clients must request an array item in canonical form when adding items to an OPC group.

Access to arrays is best done in canonical form as the overhead imposed by coercion could increase proportionally by the size of the array.

Dynamic Configuration Mode

The data type of a point can be changed dynamically (i.e. while a CIMPLICITY project is running) using the Dynamic Configuration mode in CIMPLICITY Workbench. If an OPC client is accessing a point while the point type changes, these changes are reflected in the state of the OPC item maintained by the OPC Server.

For non-array points, changes to the point type are transparent to OPC client as the OPC Server automatically coerces the new canonical data type to the data type requested when the client added the item to an OPC group.

For array points, the OPC client must be prepared to accept array information in a new data type form (and possibly with a new number of array elements).

Timestamps

OPC Item Timestamps

Associated with each OPC item value is the time at which the value last changed or the value was refreshed. This is known as the OPC item timestamp.

The CIMPLICITY OPC Server synchronizes OPC item timestamps with point timestamps stored in the CIMPLICITY runtime database. When timestamps are not available from CIMPLICITY (as in the case where a CIMPLICITY project is stopped while OPC clients are connected), the OPC Server generates a timestamp based on the current computer node time.

Universal Coordinated Time

The OPC Server returns all timestamps to an OPC client in universal coordinated time (UTC). An OPC client must convert the timestamp to local time as required.

Quality

OPC Item Quality

Associated with each OPC item value is an indicator of the quality of that value. This is known as the OPC item quality. The quality of an item is based on point status information from a CIMPLICITY project and the state of communications between the OPC Server and the CIMPLICITY project.

The CIMPLICITY OPC Server supports a subset of quality flags specified in the OPC Foundation Data Access standard. The supported quality statuses and sub-statuses are listed below.

OPC Status	OPC Sub-Status	Description
Good	N/A	The quality of the OPC item value is good.
Bad	Last Known Value	Communications with CIMPLICITY project have failed. The OPC item value is the last known value.
	Comm Failure	Communication with the CIMPLICITY project has failed. The OPC item value is invalid.

Chapter 17. CIMPLICITY OPC UA Client

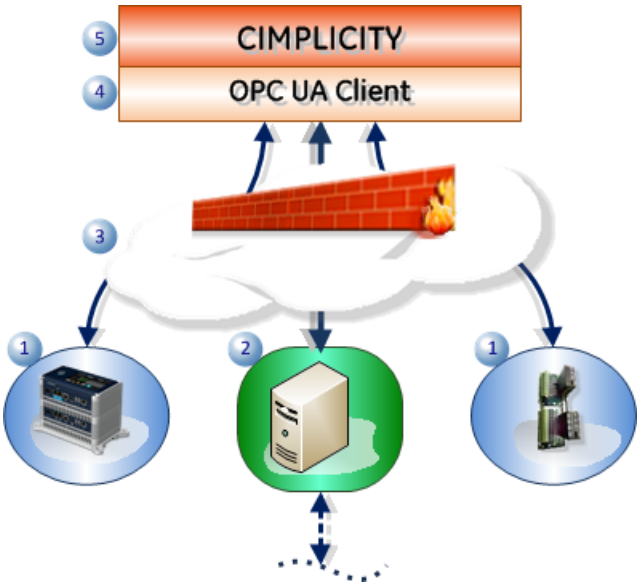
About the CIMPLICITY OPC UA Client

CIMPLICITY OPC UA (OPC Unified Architecture) Client is a device communications module that can connect to OPC UA Servers and collect data from UA Variables.

OPC UA is a standard that is rapidly gaining acceptance among vendors of process automation products in response to the complexities of inter-operability among industrial control vendors. Developed by the OPC Foundation and meant to be platform independent, OPC UA can provide lower costs and increased productivity for end-users, systems integrators and process control vendors alike by focusing communications issues on a single technology and strategy. Like CIMPLICITY®, OPC UA builds on the success and strength of common industrial standards.

OPC UA defines a platform independent communication system that has a useful and adaptive Information Model for both industrial and business application needs.

Basic components for the CIMPLICITY OPC UA Client can include the following.



1	GE Control systems (including OPC UA Servers)
2	OPC UA Server connected to other control systems
3	Network/Internet behind a firewall
4	OPC UA Client
5	CIMPLICITY

Note: CIMPLICITY OPC UA Client supports the basic numeric, BOOLEAN and string data types for OPC UA Data Access

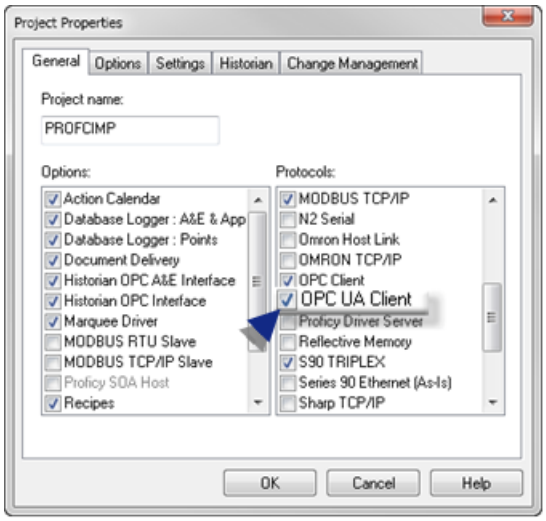
Enable the OPC UA Client

1. Click Project>Properties on the Workbench menu bar.

   The Project Properties dialog box opens.

2. Select the General tab.

3. Check OPC UA Client in the Protocols box.



4. Click OK.

The Project Properties dialog box closes; the OPC UA Client protocol is enabled.

OPC UA Client Port

OPC UA Client Port

- OPC UA Client Port: New
- OPC UA Client Port: Configure

OPC UA Port: New

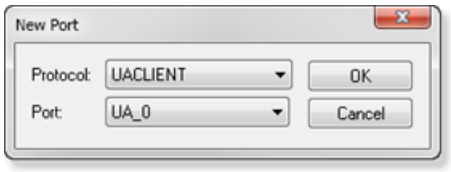
CIMPLICITY provides many ways to open a New Port dialog box.

1. Select Ports in the Workbench left-pane.

2. Do any of the following.
   - Double click **Ports**.
   - Right-click **Ports**; select New on the Popup menu.
   - Click File>New>Object on the Workbench menu bar.
   - Click the New Object button on the Workbench toolbar.
   - Press Ctrl+N on the keyboard.

   A New Port dialog box opens when you use any method.

3. Select the following.



Protocol	Select UACLIENT from the drop down list
Port	Select an OPC UA client port from the drop down list.

4. Click OK.

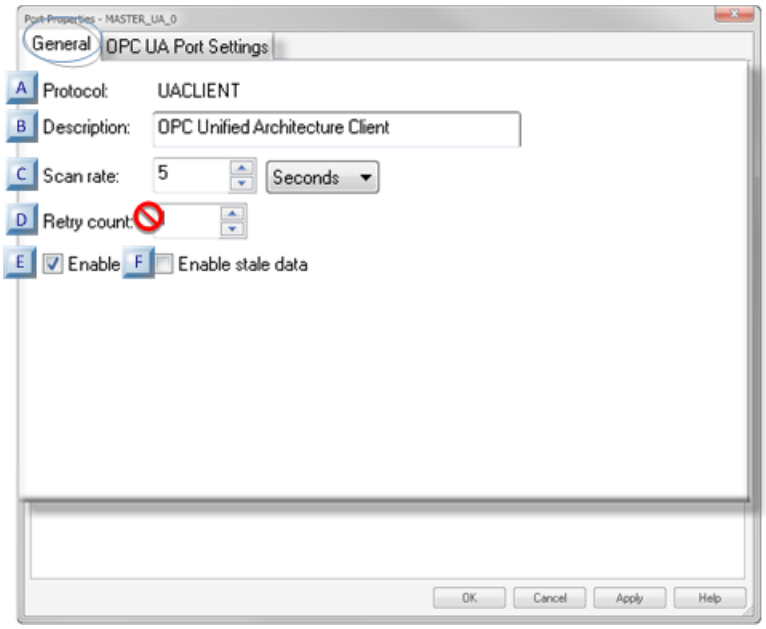
A Port Properties dialog box opens for the OPC UA client.

OPC UA Port: Configure

1 *(page 271)*	OPC UA Client Port: General Tab.
2 *(page 274)*	OPC UA Client Port: Settings Tab.

1. OPC UA Client Port: General Tab

Select the General tab in the Port Properties dialog box.



rect 69, 147, 95, 174 *(page 274)*
rect -2, 147, 24, 174 *(page 273)*
rect -3, 119, 23, 146 *(page 273)*
rect -2, 91, 24, 118 *(page 273)*
rect -2, 61, 24, 88 *(page 273)*
rect -3, 40, 23, 62 *(page 272)*

A *(page 272)*	Protocol
B *(page 273)*	Description
C *(page 273)*	Scan rate
D *(page 273)*	Retry count
E *(page 273)*	Enable
F *(page 274)*	Enable stale data

A	Protocol

(Read-only) `UACLIENT` protocol selected when the port was created.

B	Description

Explanatory text about the port.

Maximum Length	40 characters

C	Scan rate

The basic timer for points monitored from this port.

The rate at which points are polled is a multiple of the Scan Rate.

Configurable units can be set in any of the following units.

- Ticks (hundredths of seconds)
- Seconds
- Minutes
- Hours

Note: This option is applicable only for polled points.

D	Retry count

Note: OPC UA Client does not use this option..

If communication is interrupted, the OPC UA Client will retry to reconnect infinitely.

E	Enable

Do one of the following.

Check	Enable communications on this port.
Clear	Disable communications on this port.

guide: When a port is dynamically disabled, communication to all devices associated to that port will stop.

By default, when the port is dynamically disabled:

- The associated devices will be marked **Down** and their associated points will be marked **Unavailable**.
- Setpoints and processing of unsolicited data will not be allowed for the associated devices.

Tip: As an alternative to this default action, you can configure a Global Parameter, `ALLOW_UPDATE_WHEN_DISABLED`, to keep the associated devices alive.

If `ALLOW_UPDATE_WHEN_DISABLED` is configured, then when the port is dynamically disabled:

1. The device is not marked **Down**.
2. Polling stops.
3. Setpoints and unsolicited messages will still be processed.

F	Enable stale data

Do one of the following.

Check	Keep the point available in most circumstances that would have made it unavailable. However the point value will be stale, meaning it is the last known value and may or may not have changed in the device.
Clear	Leave the point unavailable in all circumstances the would make it unavailable.

Tip: Use QUALITY.STALE_DATA (Attribute) to report if the point value is stale.

The following table displays a point's availability when events occur and **QUALITY.STALE_DATA** is configured as on or off.

	Point Availability when Stale Data is Configured
Event	On
Poll Failure	Available
Process Shutdown	Available
System Shutdown	Not Applicable
Ind. Point Unavailable	Available
Out of PTMRP Range Limits	Range Bit / Available
DC Sends Device State	Available
DynCfg Disable Point/Device	Not Applicable

2. OPC UA Client Port: Settings Tab

OPC UA Client Port: Settings Tab

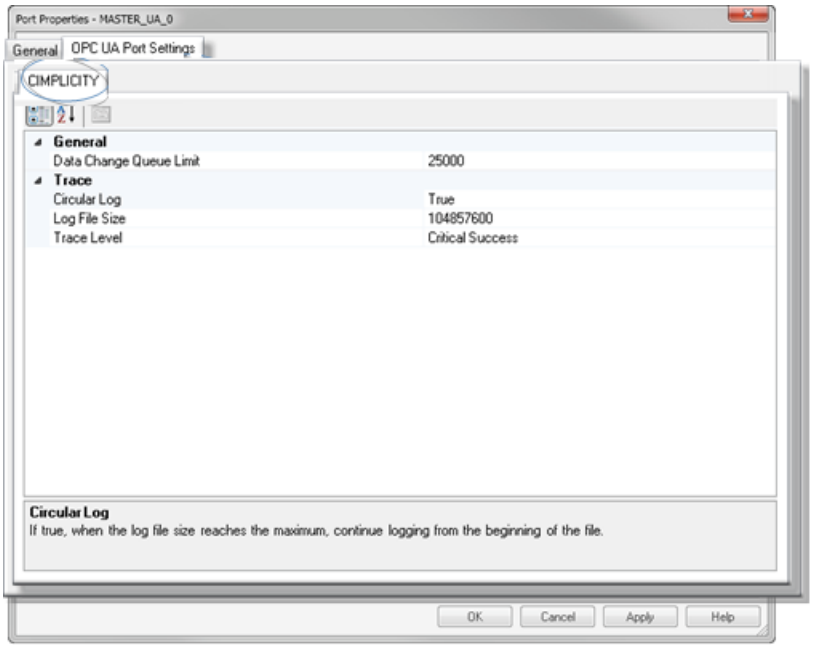
The OPC UA Port Settings tab provides the following.

	OPC UA Client Port Settings: CIMPLICITY Tab *(page 274)*

OPC UA Client Port Settings: CIMPLICITY Tab

The CIMPLICITY tab holds the configurable items that are available within CIMPLICITY.

Items include the following.



Feature	Description
Circular Log	Options for whether or not the log will continue to add new data, are as follows. • Tracing to be set to a circular file. • When the log file size reaches the maximum Log File Size, logging wraps to the beginning of the file. • Logging goes to the output file. True
Log File Size	(Bytes) Maximum circular log file size before logging wraps to the beginning of the file. 104857600 (bytes).
Trace Level	Trace level records that are created depend on the selected option. Available levels are: • Always • Critical Errors • All Errors • Critical Success • All Success • Point Level Details • Debug

OPC UA Client Device

OPC UA Client Device

OPC UA device configuration enables you to:

- Configure the connection to the UA server.
- Define a user identity that will be associated with the UA Session.
- Add/modify subscriptions and common configurable items available to CIMPLICITY.

- OPC UA Device: New
- OPC UA Device: Configure

OPC UA Device: New

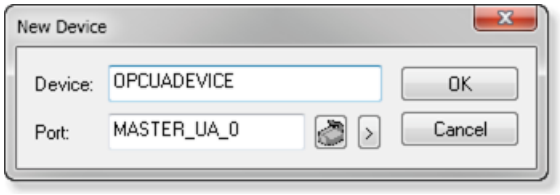
CIMPLICITY provides many ways to open a New Port dialog box.

1. Select Devices in the Workbench left-pane.

2. Do any of the following.
   - Double click Devices.
   - Right-click **Devices**; select New on the Popup menu.
   - Click File>New>Object on the Workbench menu bar.
   - Click the New Object button on the Workbench toolbar.
   - Press Ctrl+N on the keyboard.

   A New Device dialog box opens when you use any method.

3. Enter/select the following



Device	Name for the OPC UA Client device.
Port	An OPC UA Client port from the drop down list of configured ports.

4. Click OK.

The Device dialog box opens.

OPC UA Device: Configure

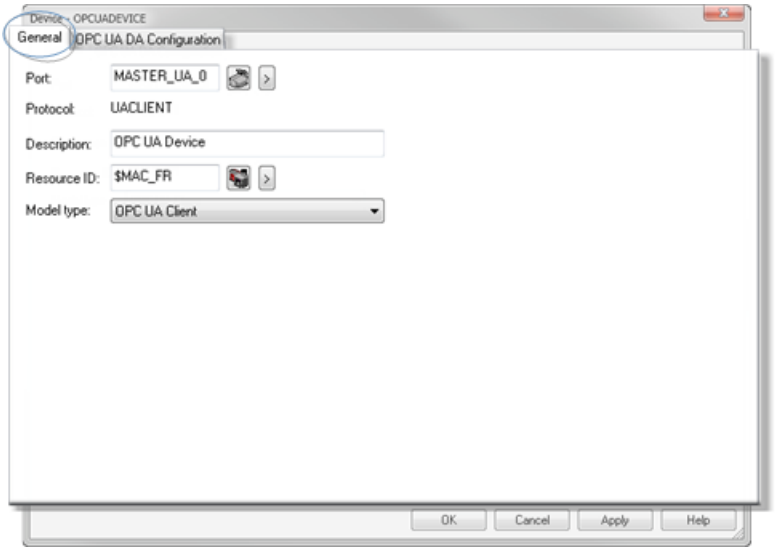
1 *(page 277)*	OPC UA Client Device: General Tab

2 (page 278)	OPC UA Client Device: OPC UA DA Configuration Tab

1. OPC UA Client Device: General Tab

Select the General tab in the Device dialog box.

Options are as follows.



Field	Description
Port	OPC UA Client port that the device will use. Buttons to the right of the Port field enable you to do the following.
	Button
	
	
	Default
Protocol	(Read-only) Device's protocol
Description	Explanatory text about the device, up to 40 characters.
Resource ID	Name of the device's resource. **Note:** Only the users that are assigned this resource will be able to see device alarms. Buttons to the right of the Resource ID field do the following.
	Button
	

				
Model Type	OPC UA Client is the only type for an OPC UA Client device.			

2. OPC UA Client Device: OPC UA DA Configuration Tab

2. OPC UA Client Device: OPC UA DA Configuration Tab

Select the OPC UA DA Configuration tab in the OPC UA Client Device dialog box.

OPC UA DA configuration will help you configure the CIMPLICITY UA device to communicate with OPC UA server.

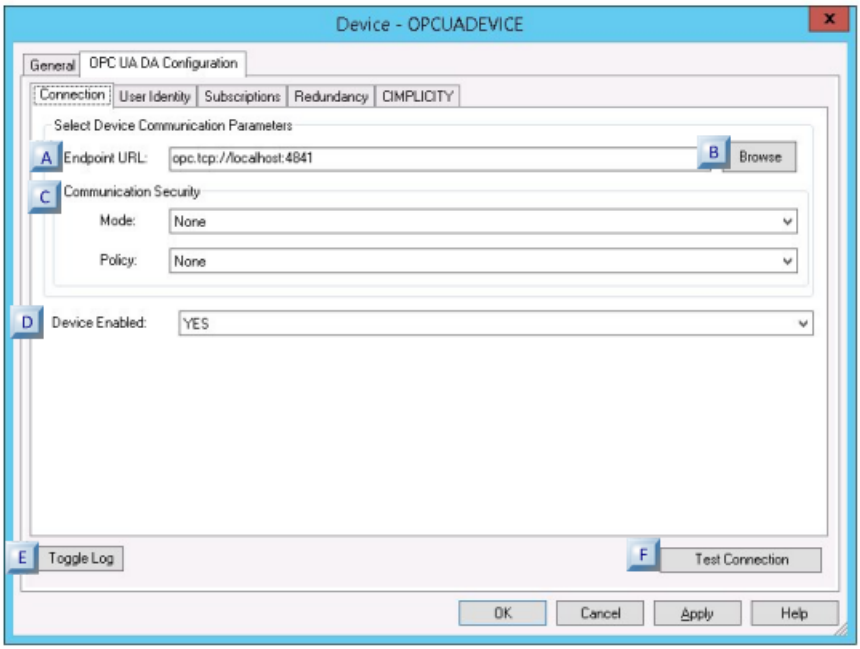
The OPC UA DA Configuration tab includes the following tabs.

2.1 *(page 278)*	OPC UA DA Configuration: Connection
2.2 *(page 292)*	OPC UA DA Configuration: User Identity
2.3 *(page 296)*	OPC UA DA Configuration: Subscriptions
2.4 *(page 301)*	OPC UA DA Configuration: Redundancy
2.5 *(page 305)*	OPC UA DA Configuration: CIMPLICITY

2.1. OPC UA DA Configuration: Connection

The OPC UA device Connection tab defines the necessary information to communicate with an OPC UA server

Selections include the following.



rect 17, 86, 42, 116 *(page 279)*
rect 433, 79, 467, 120 *(page 280)*
rect 15, 117, 40, 142 *(page 291)*
rect 0, 193, 26, 222 *(page 291)*
rect 0, 343, 24, 373 *(page 291)*
rect 394, 335, 419, 365 *(page 292)*

A *(page 279)*	Endpoint URL
B *(page 280)*	Browse Button/UA Server Discovery
C *(page 291)*	Communication Security
D *(page 291)*	Device Enabled
E *(page 291)*	Toggle Log
F *(page 292)*	Test Connection

A	Endpoint URL

A network location that OPC UA Client applications can use to find and connect to an OPC UA Server.

Note:

- An endpoint is a physical address available on a network that allows clients to access one or more services provided by a server.
- An OPC UA Endpoint URL (Uniform Resource Locator) is a formatted text string that consists of three or four parts (substrings):

1. Network protocol ((must be opc.tcp (case sensitive)).
2. Host name or IP address.
3. Port number.
4. (Optional) File or resource location

For the OPC UA specific URL, it shows as:

opc.tcp://hostname:4841/<file or resource location>

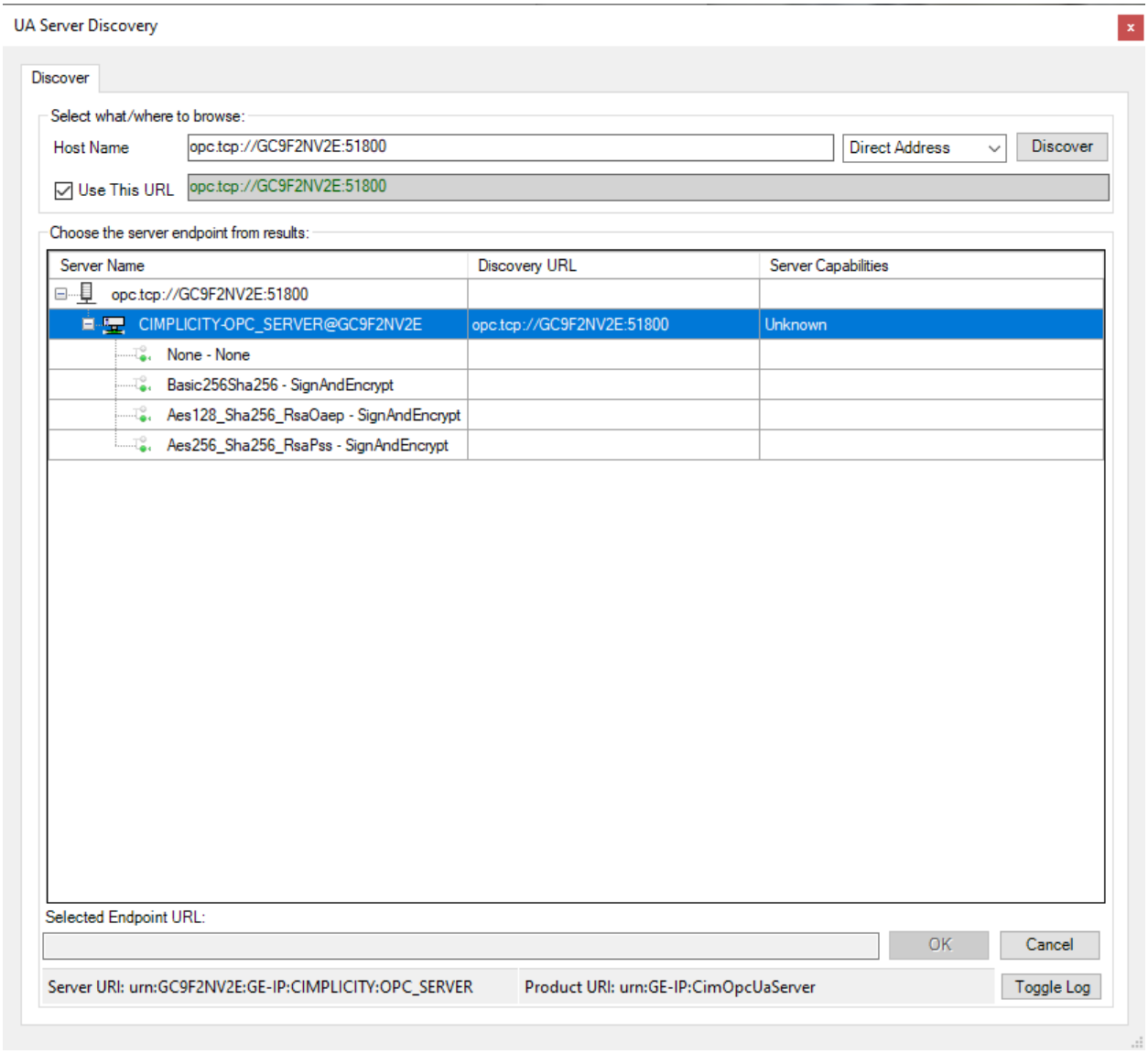
B	Browse Button/UA Server Discovery

Click the Browse button to open a UA Server Discovery browser. The browser enables you to find and select the Endpoint URL that is best for your system.

Discovery modes are as follows.

- Direct Address - Discovery Tab
- Global Directory

Direct Address



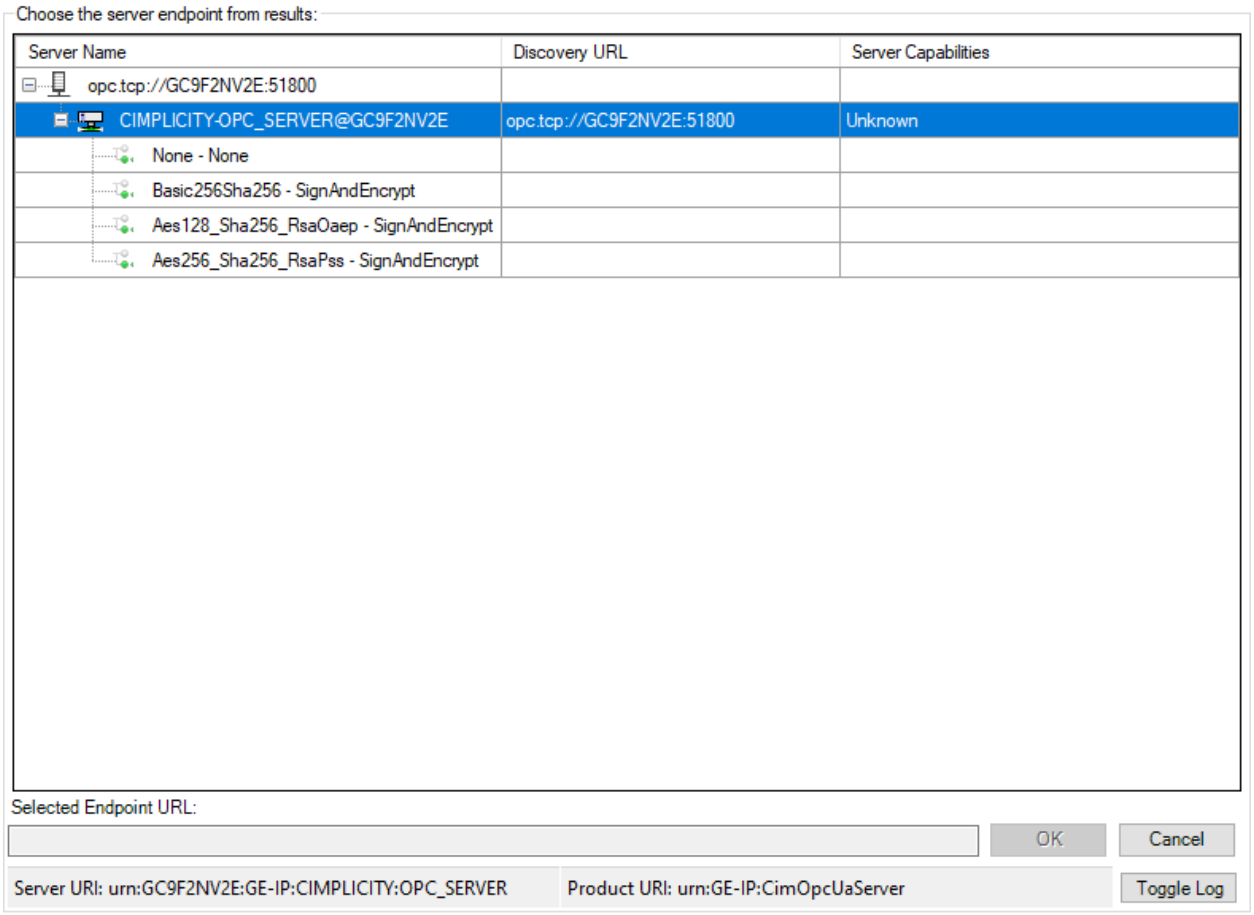
1 *(page 281)*	URL Settings
2 *(page 282)*	Choose the server endpoint from results
3 *(page 283)*	Selected Endpoint URL

1	URL Settings

URL settings include the following.

	Item	Description
A	Direct Address	Select Direct Address from the drop down menu. The search will be for either a: • Local discovery server. • Direct UA server.
B	Use This URL	The Server name that will be browsed. The host name can be either a • Local discovery server • Direct UA server
C	Use this Url	Provides the following.
		Checked
		Clear
D	Discover button	Discover

2	Choose the server endpoint from results



Tree View levels display available UA server(s) and available endpoint(s)
Column
Server Name
Discovery URL
Server Capabilities

3	Selected Endpoint URL

	Field	Description
A	Selected Endpoint URL	Read only)If a valid Endpoint is selected, the endpoint's URL will display in the **Selected Endpoint URL** field. Note: Click OK to select/confirm the selection.
B	Server URI	if a UA server is selected, the Server URI display.
C	Product URI	Product URI: if an UA server is selected, the product URI display..

	Field	Description
D	OK/Cancel	Click either button to do the following.
		OK
		Cancel
E	Toggle Log Button	Toggle Log

Important:

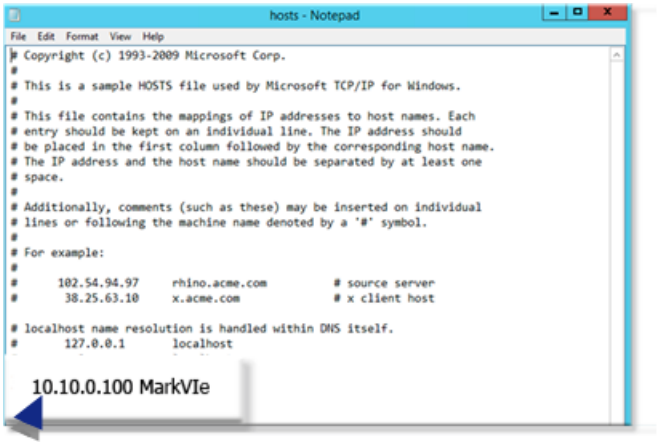
Some OPC UA endpoints use a:

- Network IP opc.tcp:\\192.168.0.100:4841.
- Domain name opc.tcp:\\MarkVIe:4841.

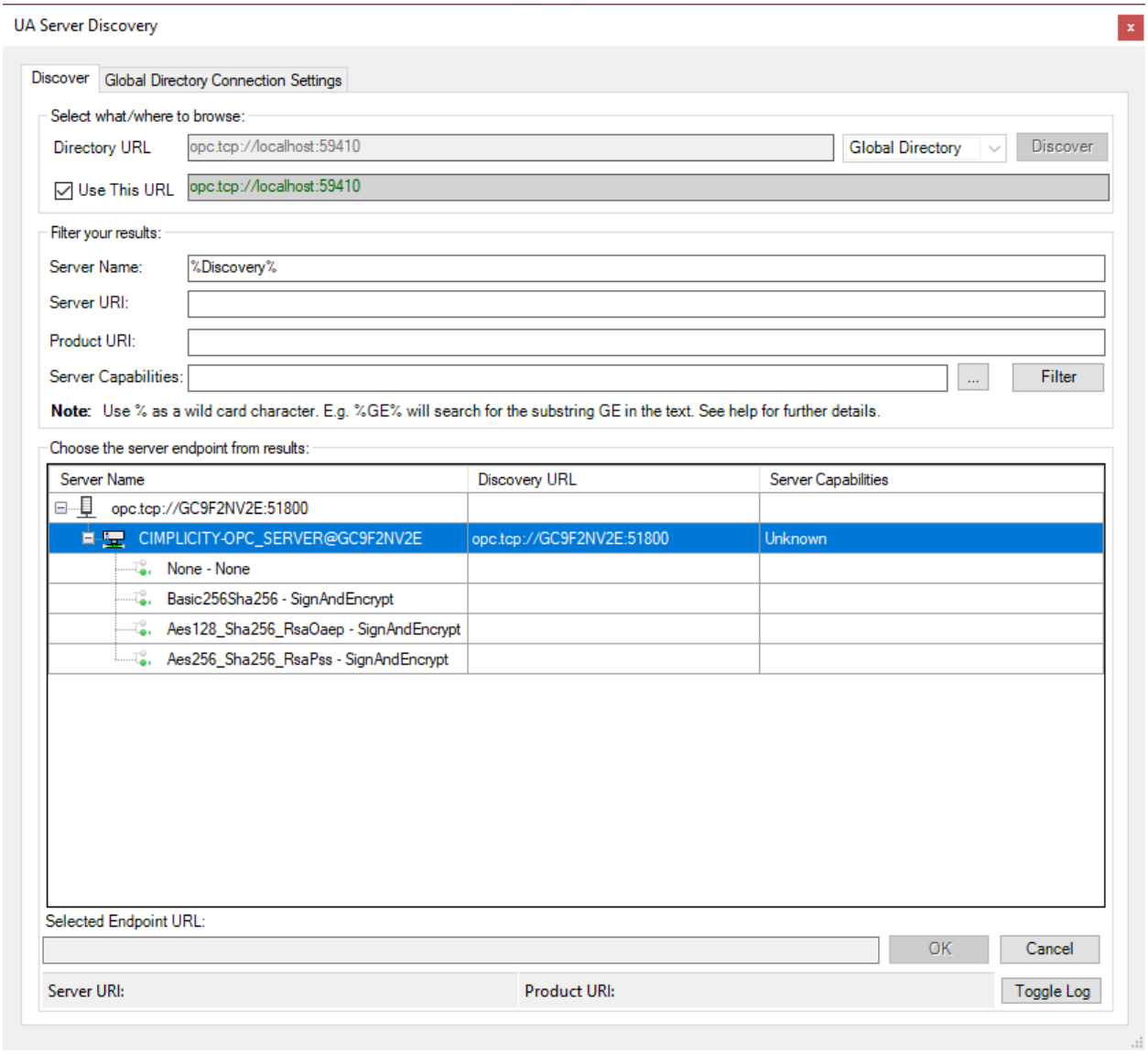
If an endpoint uses a Domain name and there is no Domain server to resolve it; then the domain name can be added by hand to the ..\system32\drivers\etc\hosts file.

Example

```
10.10.0.100 MarkVIe
```



Global Directory Discover Tab



1 *(page 286)*	URL Settings
2 *(page 286)*	Filters
3 *(page 288)*	Choose the server endpoint from results
4 *(page 289)*	Selected Endpoint URL

1	URL Settings

	Item	Description
A	Global Directory	Select Global Directory from the drop down menu. The search will use a Server that maintains a directory of other Servers.
B	Directory URL	A Server, which can be a Global Discover Server that will be browsed.
C	Use this URL	Provides the following.
		Checked
		Clear
D	Discover Button	Discover

2	Filters

Filters enable you to limit the number of servers that are returned based on your specifications.

- Global Discovery searches can be for:
- Exact matches to the entry if available wildcard characters are not used.
- Keywords based on the entry and what wildcard characters are used.
- Searches are case sensitive.

Example

If “Foundation” is the keyword (with no wildcard characters).

The filter:

- Will return only:

Foundation.

- Will not return:

foundation

UA server FoundationDemo

- Wildcard characters
- Filter Entries

Wildcard Characters

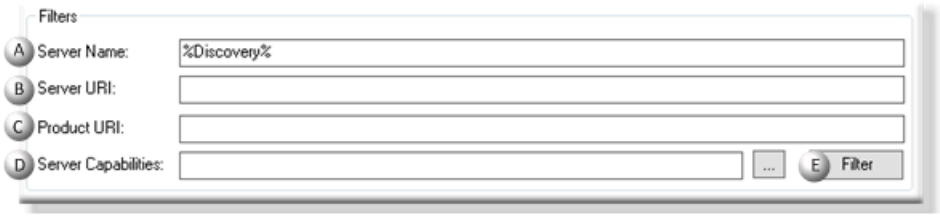
Character	Description
%	Any string of zero or more characters, as follows
	main%
	%en%
	If a % sign is intend in a string the list operand can be used.
	5[%]
_	Any single character (i.e. „_ould“ would match“). If the „_“ is indended in a string then the list operand can be used (i.e. would match “).
	_ould
	If the _ is indended in a string then the list operand can be used.
	5[_]
\	Escape character that allows literal interpretation
	\\
	\%
	_
[]	Any single character in a list
	abc[13-68]
	xyz[c-f]
[^]	Not any single character in a list. **Important:** The ^ must be the first character inside on the [].
	ABC[^13-5]
	xyz[^dgh]

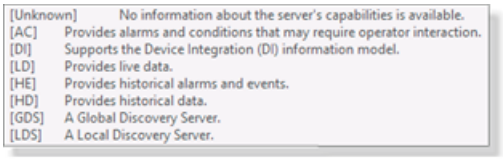
Note: = in the Wildcard Characters table indicates Match

UP *(page 286)*

Filter Entries

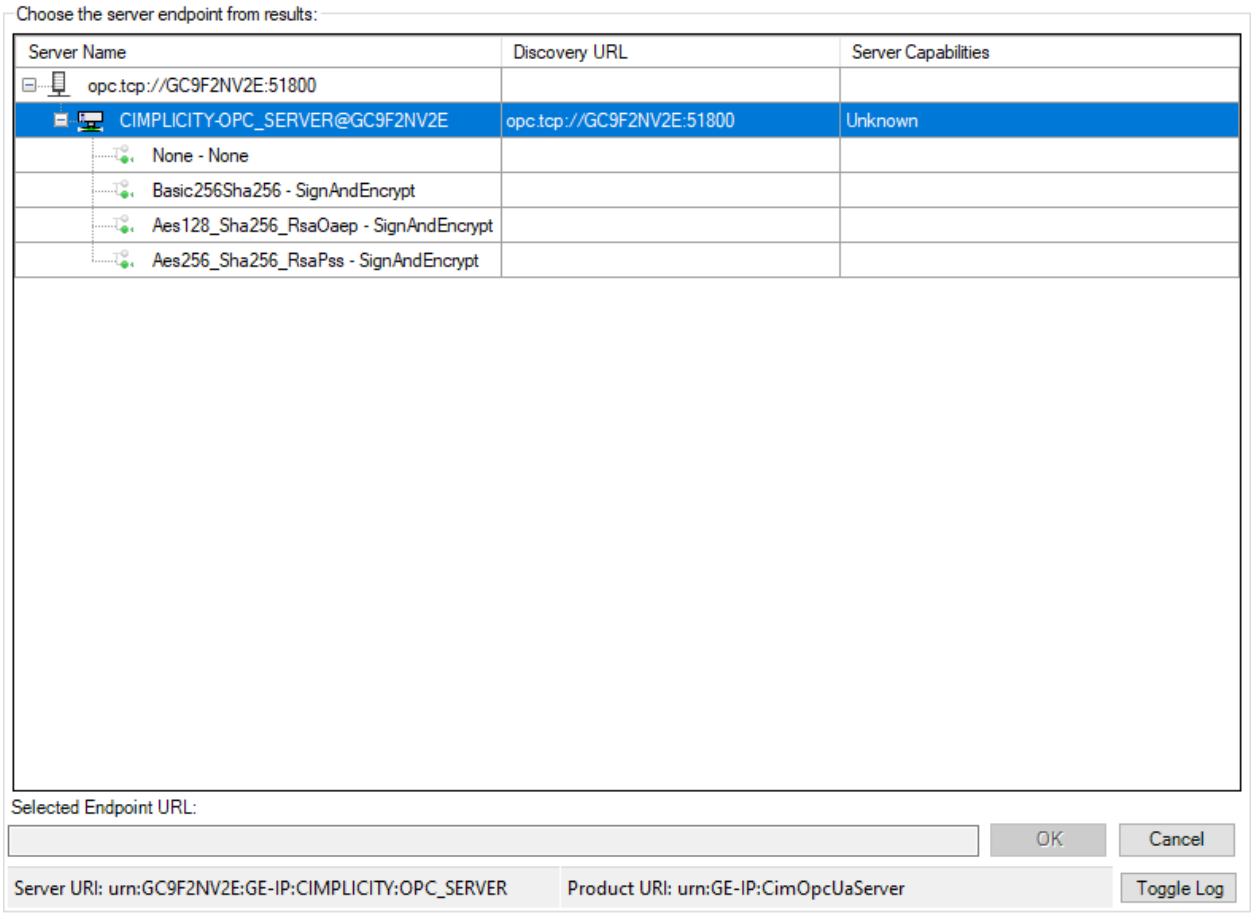
Important: All searches are case sensitive.



	Field	Results
A	Server Name	Name of the server(s) to be found. Results are based on:
		No special characters
		Special characters
B	Product URI	Name of the product URI
		No special characters
		Special characters
C	Server Capabilities	What the UA server can do. Click the Open button to the right of the Server Capabilities field.
		
		Note: Once the mouse moves over the column, a tool tip will display abbreviation/descriptions for each possible capability. 
D	Filter Button	

Result: The list of found servers will reflect the search criteria.

3	Choose the server endpoint from results



Tree View levels display available UA server(s) and available endpoint(s)
Column
Server Name
Discovery URL
Server Capabilities

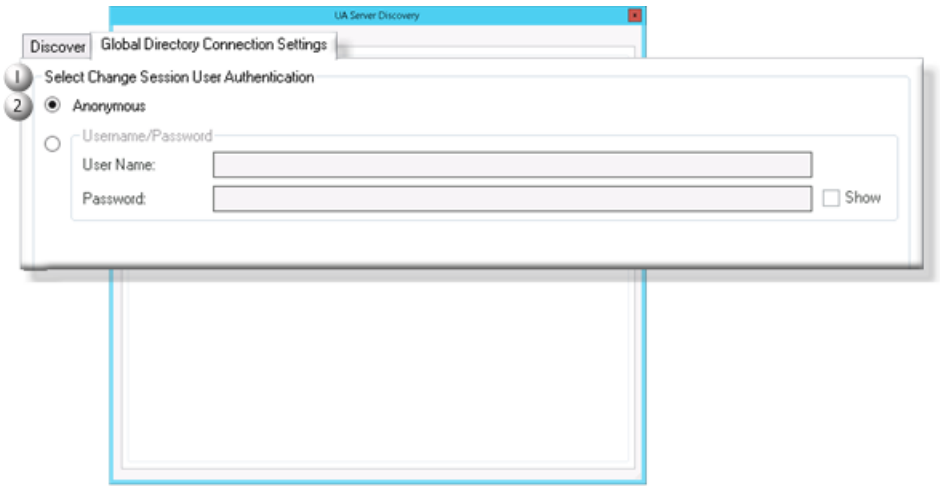
4	Selected Endpoint URL

	Field	Description
A	Selected Endpoint URL	Read only)If a valid Endpoint is selected, the endpoint's URL will display in the **Selected Endpoint URL** field. Note: Click OK to select/ confirm the selection.

	Field	Description
B	Server URI	if a UA server is selected, the server URI will be displayed. The server URI can be used as a reference to filter the discovery result.
C	Product URI	Product URI: if an UA server is selected, the product URI will be displayed. The product URI can be used as a reference to filter the discovery result.
D	OK/Cancel	Click either button to do the following.
		OK
		Cancel
E	Toggle Log Button	Toggle Log

Global Directory: Connection Settings Tab

The Global Directory Connection Settings tab enables you to select user authentication that is either of the following.



A	Anonymous

An `AnonymousIdentityToken` indicates that the client has no user credentials.

The allowed access rights for this token depend on the OPC UA server. Some OPC UA servers will refuse this type of the token.

B	User Name/Password

A `UserNameIdentityToken` passes a simple user name/password credentials to the UA Server.

Note: Both the user name and password are stored in the configuration file in encrypted form.

The password is encrypted when it is sent to the OPC UA Server in encrypted form; it is also stored in the memory as a plain text.

Note: Check Show to display the password.

C	Communication Security

Communication Security includes selecting the mode and the policy.

Mode

Options are:

Mode	Description
None	No security is applied.
Sign	Encryption is still used in the initial handshake so this mode may not be appropriate when legal requirements prohibit the use of encryption.
SignAndEncrypt	All messages are signed and encrypted.

Policy

Policy	Description
None	No security is applied.
Basic256Sha256	For configurations that require high security.
Aes128_Sha256_RsaOaep (Fastest)	For configurations that require high speed with average security.
Aes256_Sha256_RsaPss (Most Secure)	For configurations that require very high security.

Note: Basic128Rsa15 and Basic256 are deprecated due to vulnerability and theoritical issues.

D	Device Enabled

Select either of the following.

Yes	The device is ready to be used.
No	The device cannot be used until Yes is selected.

E	Toggle Log

Shows or hides a Connection Test Log window that displays a report created when the Test Connection button is clicked.

F	Test Connection

Click Test Connection to make sure the entries are valid.

Result: A Connection Test Log window opens reporting:

- Whether or not the connection succeeded.
- Detailed information to diagnose any connection issues.

2.2. OPC UA DA Configuration: User Identity

The User Identity tab defines the user identity token that will be used to activate an OPC UA session. The OPC UA server uses the user identity token it received to accept or reject the communication.

- User Identity: Overview
- User Identity: Configuration

User Identity: Overview

- The OPC UA Server uses the User Identity token it received to
- Authenticate a user.
- Conduct further authorization (e.g. grant different level of access rights to its services depending on the user).

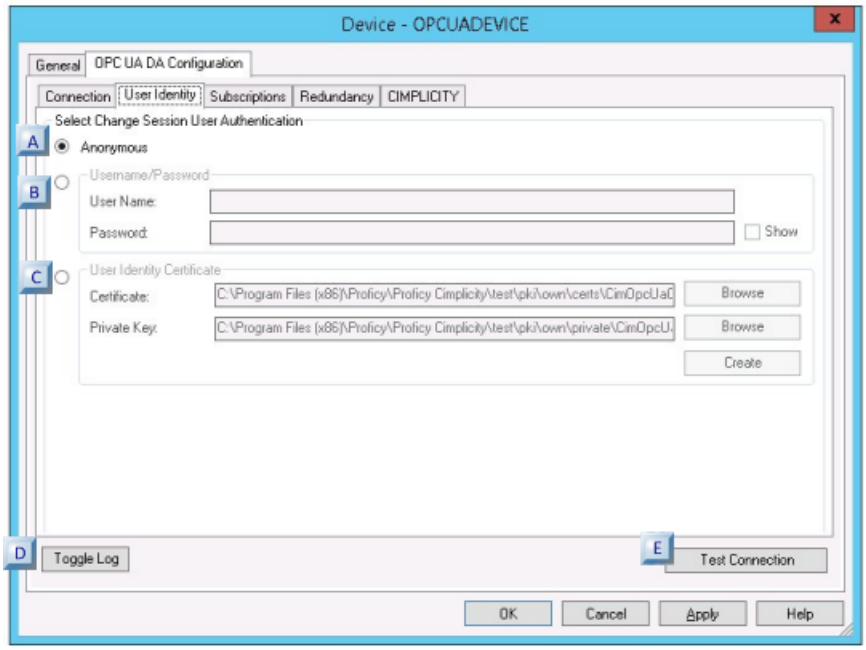
If provided user credentials are not valid, the connection to the UA Server will fail.

- The CIMPLICITY OPC UA Client supports the following three (of four possible) types of User Identity tokens.
- Anonymous;
- User Name / Password;
- Certificate.

Note: The OPC UA Server might support only some of these token types. For example, some OPC UA Server can support only the User Name/Password type.

User Identity: Configuration

User identity options are as follows.



rect 391, 330, 437, 377 *(page 296)*
rect -4, 341, 20, 372 *(page 295)*
rect 9, 161, 34, 193 *(page 294)*
rect 3, 109, 28, 141 *(page 294)*
rect 6, 74, 31, 106 *(page 293)*

A *(page 293)*	Anonymous
B *(page 294)*	User Name Password
C *(page 294)*	Certificate/Private Key
D *(page 295)*	Toggle Log
E *(page 296)*	Test Connection

A	Anonymous

An `AnonymousIdentityToken` indicates that the client has no user credentials.

The allowed access rights for this token depends on the OPC UA server. Some UA servers will refuse this type of the token.

B	User Name/Password

A `UserNameIdentityToken` passes a simple user name/password credentials to the UA Server.

Note: Both the user name and password are stored in the configuration file in encrypted form.

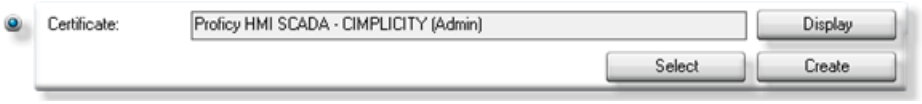
The password is passed to the UA Server in encrypted form; it is never stored in the memory as a plain text so it is always protected.

Note: Check Show to display the password.

C	Certificate

An `X509IdentityToken` passes an **X509 v3 Certificate** that is used to authenticate the user.

Example



Field	Description
Certificate	User Identity Certificate's common name.
	(Read-only) Text
	HMI SCADA - CIMPLICITY (User)
	Opens a Certificate dialog box that provides details about the certificate that is currently in the **Certificate** field. **Notes** If a certificate has not been selected, a message box will open reporting that there is no certificate selected. Once a certificate is selected and applied its Common Name will remain in the **Certificate** field. However, • It will only be used if Certificate is checked. • Another certificate can be selected.
	Select
	Create

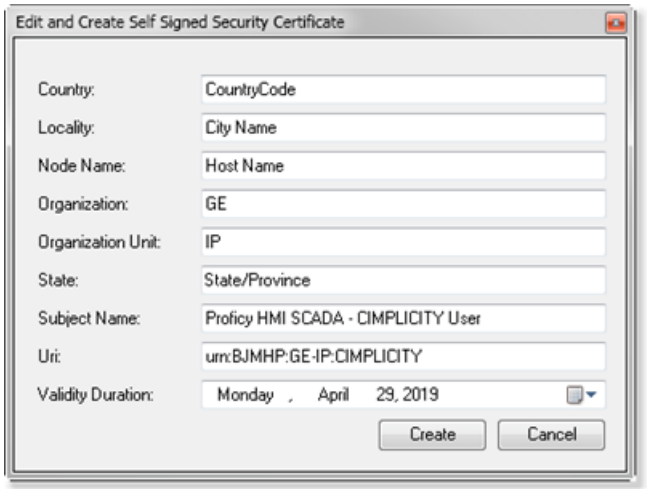
Create Self Signed User Identity Certificate Dialog Box

OPC UA enables you to create a self-signed user certificate.

1. Click the Create Self Signed Instance Certificate button on the Device dialog box>OPC UA DA Configuration tab>User Identity tab.

An Edit and Create Self Signed Security Certificate dialog box opens. All information entered in the dialog box will be used to create the certificate.

2. Edit and enter information to accommodate your OPC UA client requirements



Field	Description
Country	Automatically filled In based on your computer's settings.
Locality	City in which the OPC UA Client certificate is issued.
Node Name	Required
Organization	Organization that owns the application.
Organization Unit	Organization's division/department to which the certificate is attached.
State	State/province in which the OPC UA Client certificate is issued.
Subject Name	Subject name to show in the certificate
	`HMI SCADA - CIMPLICITY User`
Uri	Required
	`urn:[`NodeName`]:GE_IP:CIMPLICITY` Where `[`NodeName`]` is the host name of the computer.
Validity Duration	Certificate's expiration date.

3. Click Create or Cancel.

The certificate is created or cancelled, based on what you click; the dialog box closes.

D	Toggle Log

Displays/hides a Connection Test Log window that displays a report created when the Test Connection button is clicked.

F	Test Connection

Click Test Connection to make sure the entries are valid.

A Connection Test Log window opens reporting:

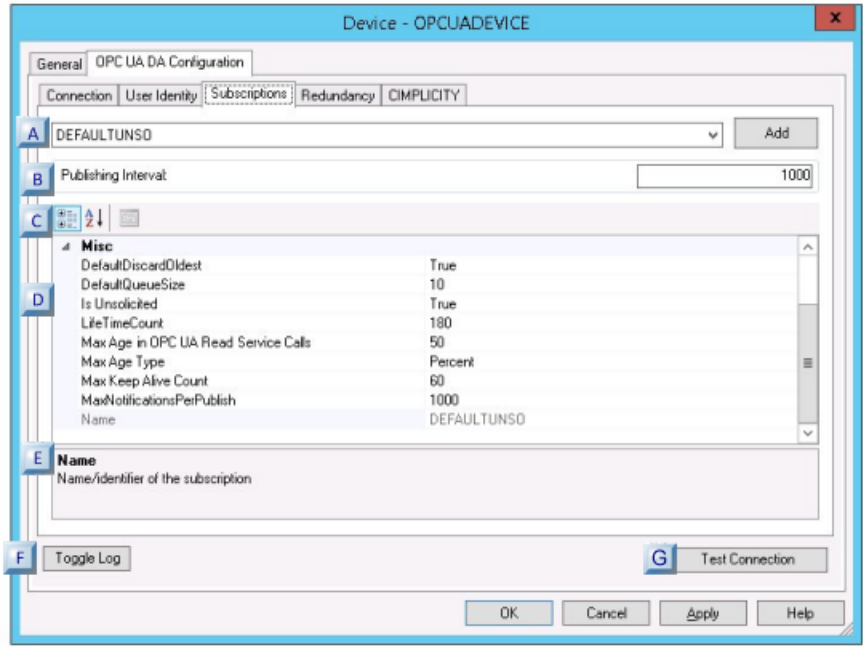
- Whether or not the connection succeeded.
- Detailed information to diagnose any connection issues.

2.3. OPC UA DA Configuration: Subscriptions

The Subscriptions tab enables you to create, delete or modify the default or custom subscription configurations (referred to as subscriptions).

Note: Default *(page 297)* subscriptions cannot be deleted.

Subscription features and options are as follows.



A *(page 297)*	Subscriptions
B *(page 298)*	Publishing Interval
C *(page 298)*	Toolbar Buttons

D *(page 298)*	Properties List
E *(page 301)*	Descriptions
F *(page 301)*	Toggle Log
G *(page 301)*	Test Connection

A **Subscriptions**

A device can have multiple subscriptions; all subscriptions listed in the dropdown menu are available to the device.

Subscriptions can be added, deleted or modified; at any time, only one subscription can be modified.

Note:

- (Unsolicited subscription configurations only) there will be at least one OPC UA Subscription created on the UA Server.
- The actual number of OPC UA Subscriptions depends on limitations, such as the Max Number of Monitored Items *(page 299)* . If the limits are exceeded, more subscriptions are created with the same properties.

The selected subscription can have the following properties.

Subscription	Description
Default Groups	Created by the OPC UA Client, the default groups cannot be deleted. They are: Allows unsolicited updates Updates points when polled
Custom Groups	1. Click the **Add** button. A Create New Subscription dialog box opens. 2. Enter a name for the new subscription. 3. Click **OK**. Result: The subscription is added to the dropdown menu list. **Note:** The Del (Delete) button is enabled when a custom description is selected.

B **Publishing Interval**

Cyclic rate at which the selected subscription executes.

Each time it executes, the subscription attempts to send a notification message to the Client. Default 1000ms

Note: Notification messages contain information that has not yet been reported to Client.

C **Toolbar Buttons**

The Subscription tab includes a toolbar with the following available buttons.

Sort features by category.

Sort features alphabetically.

D **Properties List**

Subscription properties are sorted into two categories.

- Limitations
- Misc

Limitations

Properties that can place limits on the OPC UA Client activity as follows.

Limitation	Description
Allow Mixed Sampling Intervals	Options are as follows.
	True
	False
	Default

Max Number of Monitored Items	Maximum number of monitored items per subscription. Note: If the number of monitored items assigned to a subscription exceeds this limit, additional subscriptions are created with the same settings. Default
Maximum domain length	(Bytes) Maximum value for the total size of data values of all monitored items. Note: If this value is exceeded, more subscriptions are created with the same settings. Default

Up *(page 298)*

Misc

Miscellaneous subscription properties are as follows.

Misc	Description
DefaultDiscardOldest	A BOOLEAN parameter that specifies the discard policy when the queue is full and a new notification needs to be queued. It has the following values: The oldest (first) Notification in the queue is discarded. The new Notification is added to the end of the queue. The new Notification is discarded. The queue is unchanged
DefaultQueueSize	The number of processed values that should be kept on the OPC UA Server. Notifications are queued on the server for transfer to the client by subscriptions. When the queue is full and a new notification is received, the Server either discards the oldest Notification and queues the new one, or it simply discards the new one. 10
IsUnsolicited	Options are as follows. Enables unsolicited updates

	Uses polling for updates. Depends on the subscription.
LifeTimeCount	A requested value that contains the number of consecutive publishing timer expirations without Client activity before the Subscription is terminated. Note: The lifetime count shall be a minimum of 3 times the keep-alive count. 180
Max Age in OPC UA Read service calls	(Milliseconds or Percent of Max Age Rate) A maxAge parameter directs the Server to access the value from the underlying data source (e.g. a device) if its copy of the data is older than that which the maxAge specifies. If the Server cannot meet the requested max age, it returns its best effort value rather than rejecting the request. 50
Max Age Type	Type of number entered for Max Age in OPC UA Read service calls. Options are: Percent of minimum sampling rate Milliseconds Percent
MaxKeepAliveCount	Requested maximum keep-alive count. When the publishing timer has expired the specified number of times without having any notification message to be sent, the Subscription sends a keep-alive message to the Client. 60
MaxNotificationsPerPublish	The maximum number of notifications that the client wishes to receive in a single publish response. Note: A value of zero indicates that there is no limit. 1000
Name	(Read-only) Name of the selected subscription. The following options are requested values and can be revised and used by the OPC UA Server. • PublishingInterval • LifeTimeCount • MaxKeepAliveCount The revised values are logged in the device log file.

Up *(page 298)*

E **Descriptions**

(Read-only) Brief description of the selected property

F **Toggle Log**

Displays/hides a Connection Test Log window that displays a report created when the Test Connection button is clicked.

G **Test Connection**

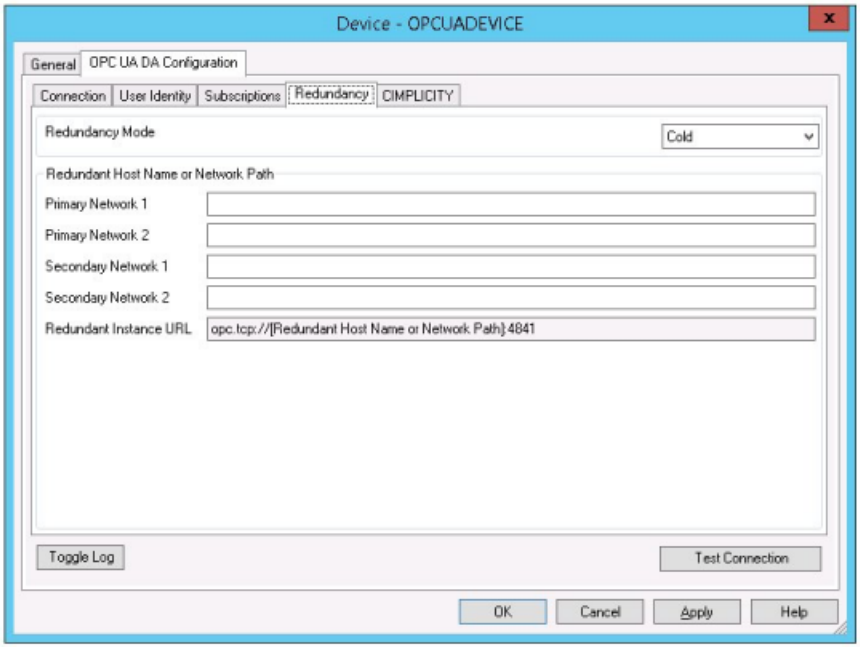
Click Test Connection to make sure the entries are valid.

Result: A Connection Test Log window opens reporting:

- Whether or not the connection succeeded.
- Detailed information to diagnose any connection issues.

2.4. OPC UA DA Configuration: Client Redundancy

To configure redundancy, access the Redundancy tab in the OPC UA DA Configuration dialog box as shown in the following figure.



The following fields are available on the Redundancy tab:

Item	Description

Mode	The CIMPLICITY OPC UA Client driver supports cold, warm, and hot redundant OPC UA Servers with or without cabling redundancy. With cold redundancy, no stand-by server subscription for all the monitored items is created. With hot redundancy, two parallel sessions are created with the redundant servers. All monitored items are created and are set to sample; however, the stand-by server subscription has publishing disabled. When a failover occurs, publishing is enabled on the standby and disabled (if possible) on the current active. With warm redundancy, two parallel sessions are created with the

Primary Network 1	Host name or network path for primary network 1 for Redundancy feature.
Primary Network 2	Host name or network path for primary network 2 for Redundancy feature.
Secondary Network 1	Host name or network path for secondary network 1 for Redundancy feature.
Secondary Network 2	Host name or network path for secondary network 2 for Redundancy feature.

Redundant Instance URL	The URL used to connect to a specific redundancy server instance. When running with redundancy the client makes multiple connections to multiple servers which may be used to recover communication when there is a network or device failure. The Redundant Instance URL describes how those URLs are constructed from the information entered in this dialog box. For example: opc.tcp://[Redundant Host Name or Network Path]:484

When a failover occurs sampling is enabled for all monitored items on the standby and disabled (if possible) on the current active.

If cabling redundancy is enabled, the OPC UA Client driver needs a list of alternate network paths for each redundant server. This information may be provided by the ServerNetworkGroups property of the ServerRedundancy object. However, this information can also be provided as part CIMPLICITY UA Client configuration.

When a network failure occurs the UA Client immediately attempts to re-connect via the alternate network for the current active. If this fails, it switches to the stand-by server.

The UA client subscribes to the ServiceLevel property provided by each redundant Server. This value indicates which instance is the current active (any number >200 is assumed to be the active). If this ServiceLevel drops below 200, a failover is initiated if the standby server has a greater ServiceLevel.

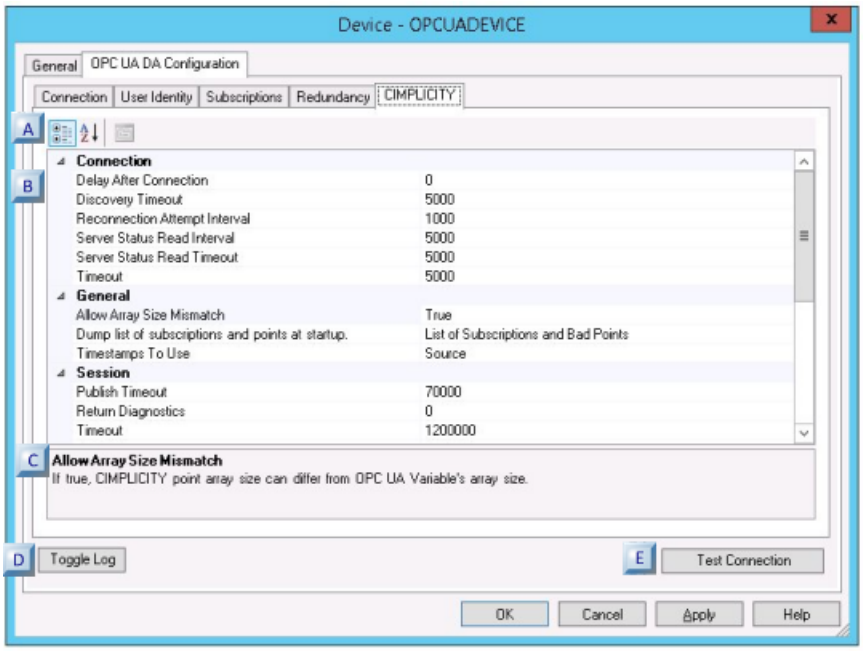
The following OPC UA Diagnostic points can be used for troubleshooting:

Item	Diagnostic Point Address	Data Type
Current Active Server	$OPCUA_CURRENT_MASTER	STRING_80
Redundant Server Failover Count	$OPCUA_FAILOVER_COUNT	UDINT
Primary Network 1 Server Status	$OPCUA_SERVER_STATUS_PRIMARY1	STRING_80
Primary Network 2 Server Status	$OPCUA_SERVER_STATUS_PRIMARY2	STRING_80
Secondary Network 1 Server Status	$OPCUA_SERVER_STATUS_SECONDARY1	STRING_80
Secondary Network 2 Server Status	$OPCUA_SERVER_STATUS_SECONDARY2	STRING_80

2.5. OPC UA DA Configuration: CIMPLICITY

The CIMPLICITY tab Provides the configurable items that are available within CIMPLICITY.

The CIMPLICITY tab features and options are as follows.



rect -4, 61, 27, 91 *(page 306)*
rect -4, 96, 27, 126 *(page 307)*
rect 4, 272, 36, 302 *(page 310)*
rect 380, 334, 411, 363 *(page 310)*
rect -3, 336, 29, 365 *(page 310)*

A *(page 306)*	Toolbar Buttons
B *(page 307)*	Properties List
C *(page 310)*	Descriptions
D *(page 310)*	Toggle Log
E *(page 310)*	Test Connection

A	Toolbar Buttons

The Subscription tab includes a toolbar with the following available buttons.

	Sort features by category.
	Sort features alphabetically.

B	Properties List

CIMPLICITY properties are sorted into five categories.

- Connection
- General
- Redundancy
- Session
- Trace

Connection

Connection	Description
Delay After Connection	(milliseconds) Amount of delay after the connection between the OPC UA Client and Server is established before the OPC UA Client sends requests to the OPC UA Server.
Discovery Timeout	(milliseconds) Time out for OPC UA Client discovery service calls to the OPC UA Server. Example A discovery call can retrieve a server certificate in secured communication mode.

Reconnection Attempt Interval	(milliseconds) Interval between consecutive re-connection attempts.
Server Status Read Interval	(milliseconds) Frequency of reading the server node's state from the OPC UA Server.
Server Status Read Timeout	(milliseconds) Time out between OPC UA Client service calls to read the OPC UA Server node's state (e.g. if the node is online).
Timeout	(milliseconds) Time allowed for the OPC UA Client to connect to the OPC UA Server before the attempt times out.

General

General	Description
Allow Array Size Mismatch	CIMPLICITY point array size. Can differ from the OPC UA variable's array size. Cannot differ from the OPC UA variable's array size. False.
Dump list of subscriptions and points at startup	Defines if list of subscriptions and/or points that should be logged to the file when the OPC UA Client starts up. [PortId]_[DeviceId]_dump.json Example UA_0_D1_dump.json Where `UA_0` is [Port Id] `D1` is [Device Id] ..\<project location>\log folder List of subscriptions and bad points

Timestamps to use	Timestamp source for data changes. Options are: • Local • Server • Source
White Array Using Index Element	Enables OPC UA Client to use index range and write data to a single array element. Writes data to single array element. Writes data to complete array. True

Session

Connection	Description
Publish Timeout	Timeout hint for Publish requests.
Return Diagnostics	Bit mask that identifies the types of vendor-specific diagnostics to be returned in `diagnosticInfo` response parameters. Refer to the OPC Foundation's OPC UA Specification, part 4 for details. 0
Timeout	(milliseconds) Maximum number of milliseconds that a session should remain open without activity. **Note:** If the Client fails to issue a service request within this interval, the Server will automatically terminate the Client session. 0 - 1200000
Timeout Hint	(milliseconds) Does the following on the Client or the Server side. The communication stack sets the time out on a per-call basis. The hint can be used to cancel long running operations in order to free resources. **Notes** If the Server detects a time out, it: • Can cancel the operation by sending the Service result: `Bad_Timout`. • Will wait, at the minimum, the specified Timeout period after receiving the request and before cancelling the operation. 0-3000

Trace

Connection	Description
Circular Log	Options for whether or not the log will continue to add new data, are as follows.
Log File Size	(Bytes) Maximum circular log file size before logging wraps to the beginning of the file.
Trace all activities	**Note:** All activities can be logged using this one command or by selecting True for every listed activity. All activities:
Trace Connection	All activities related to connecting to and disconnecting from the OPC UA Server:
Trace data changes	Data changes: Are logged, Are not logged, True.
Trace demand reads	Are logged, Are not logged, True.
Trace Events	Activities related to point update 'events fired by the OPC UA server: **Note:** When events occur, the OPC UA client puts the new value into the point update queue, which is then emptied as the new values are passed into CIMPLICITY. Trace events: Are logged, Are not logged, True.
Trace Item Activity	Activities related to OPC Items; adding them to the group, removing them from the group, adding them to the OPC Server, etc: Is logged, Are not logged, True.
Trace Level	Trace level records that are created depend on the selected option. Available levels are: • Always • Critical Errors • All Errors • Critical Success • All Success • Point Level Details • Debug
Trace Subscription Level Activity	Activities at the subscription level such as: Are logged, Are not Logged, True.
Trace when data changes are dequeued and passed to CIMPLICITY	When data that has changed leaves the point update queue and is sent to CIMPLICITY: Are logged, Are not logged, True.
Trace write operations	Activities related to writing a new value to a point in the OPC UA server from CIMPLICITY: Are logged, Are not Logged, True.

C	Descriptions

(Read-only) Brief description of the selected property

D	Toggle Log

Displays/hides a Connection Test Log window that displays a report created when the Test Connection button is clicked.

E	Test Connection

Click Test Connection to make sure the entries are valid.

Result: A Connection Test Log window opens reporting:

- Whether or not the connection succeeded.
- Detailed information to diagnose any connection issues.

OPC UA Client Points

Once you have attached a device to a server and you can create points to communicate back and forth for monitoring and control purposes.

Points need to be created/configured based on what is available on the OPC UA Server. CIMPLICITY OPC UA client enables you to select a **Node ID** from an OPC UA Address Space browser.

- OPC UA Client: New Point
- OPC UA Client: Point Device Properties

OPC UA Client: New Point

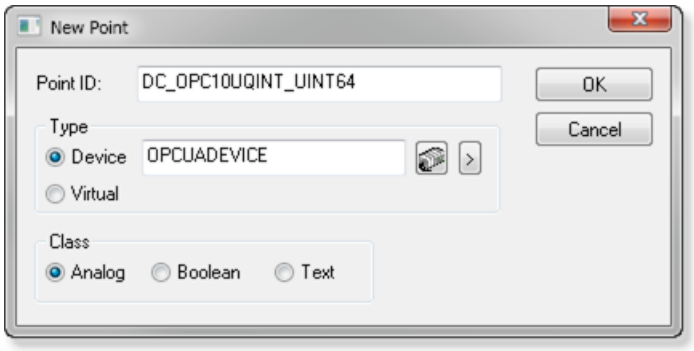
CIMPLICITY provides many ways to open a New Point dialog box.

1. Select **Points** in the Workbench left-pane.

2. Do any of the following.
   - Double click **Points**.
   - Right-click **Points**; select New on the Popup menu.
   - Click File>New>Object on the Workbench menu bar.
   - Click the New Object button on the Workbench toolbar.
   - Press Ctrl+N on the keyboard.

   A New Point dialog box opens when you use any method.

3. Enter/select the following.



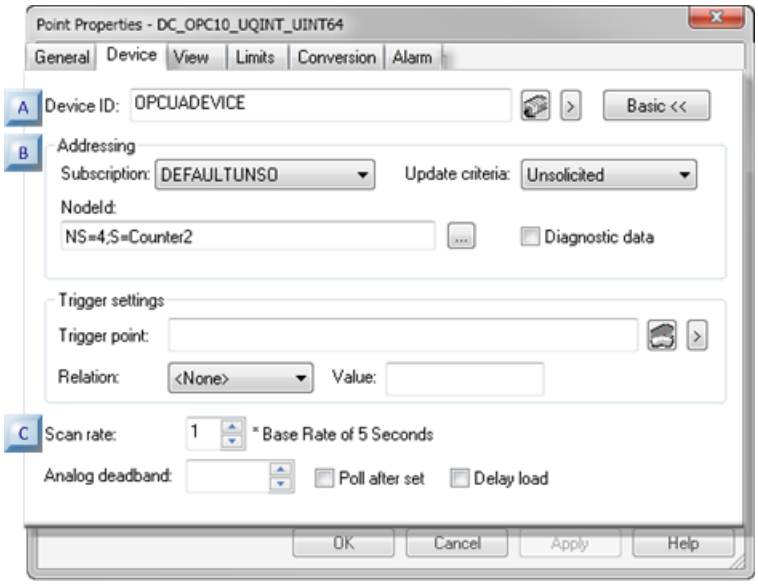
Option	Description
Point ID	Unique name that identifies the point.
	Maximum Length
Type	Do the following.
	Check
	Select
Class	Selection depends on the point requirements. Options are: • Analog • Boolean • Text

Review the Points section in the CIMPLICITY documentation for details about creating points.

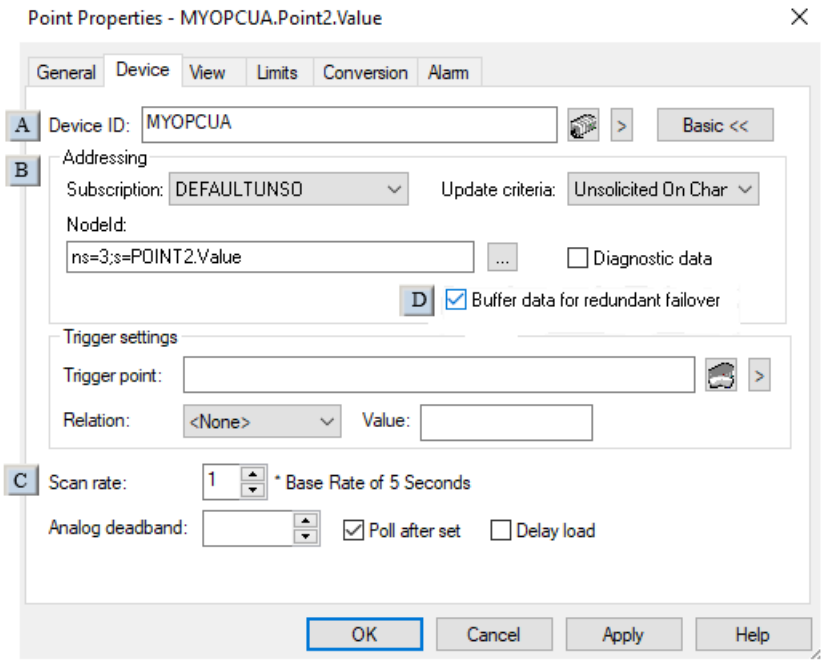
4. Click OK.

OPC UA Client: Point Device Properties

An OPC UA Client point requires the following entries in the Point Properties dialog box>Device tab.



rect 1, 51, 25, 77 *(page 314)*
rect 1, 78, 25, 104 *(page 314)*
rect 1, 253, 25, 279 *(page 316)*



A *(page 314)*	Device ID
B *(page 314)*	Addressing

C *(page 316)*	Scan Rate
D *(page 316)*	Buffer data for redundant failover

A	Device ID

Enter the name of a configured OPC UA Client device.

Buttons to the right of the Device ID field enable you to do the following.

Button	Name
	Point Browser
	Popup Menu

B	Addressing

Addressing for an OPC UA Client point requires the following.

- Subscription/Update Criteria
- Node ID
- OPC Address Space Properties

Subscription/Update Criteria

Select a subscription that was created and/or configured in the selected OPC UA Client's Device dialog box> OPC UA DA Configuration tab>Subscriptions tab.

The subscriptions that are available in the dropdown list correspond to the Update criteria selection.

Default subscriptions are:	• DEFAULTUNSO • DEFAULTPOLL

Custom descriptions display with related update criteria based on whether `IsUnsolicited` is True or False on the Device dialog box>OPC UA DA Configuration>Subscriptions tab.

IMPORTANT:	OPC UA servers running on devices such as PLCs may not support the **Unsolicited** update criteria. If this is the case, the points will not update in the PCP, even though it is possible to browse the server from the Device configuration dialog and the **Test** button reports success. Changing the update criteria to **On Demand Poll Once** or **On Demand On Scan** should allow data collection to start. Note that these update criteria are less efficient and should only be used when there are a small number of points.

Node ID

The node ID comes from the OPC UA Server.

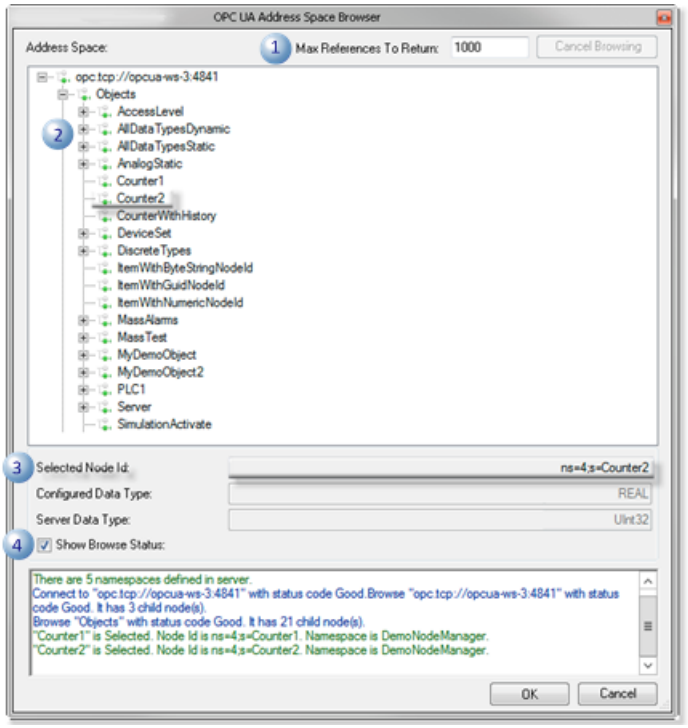
Do either of the following.

- Manually enter the node ID.
- Click the OK button after selecting a **node ID** in the OPC UA Address browser to have CIMPLICITY enter the node ID.

Click the Browse button ... to the right of the **NodeId** field.

An OPC UA Address Space Browser opens listing the available server nodes

Features/options include the following.

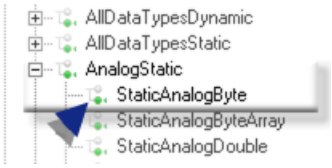


	Feature/Option	Description
1	Max References To Return	Maximum number of nodes/child nodes to be listed in the **Address Space** tree.
		Default
2	Address Space	Tree listing: • Nodes. • Objects associated with a node. • Child nodes associated with an object.
3	Basic information	Read-only fields report the following for a selected node. • Selected Node ID • Configured Data Type • Server Data Type
4	Show Browse Status	Check to display diagnostic messages about the OPC UA Server

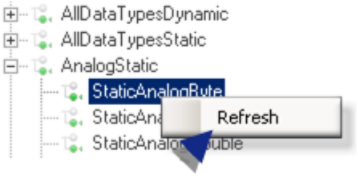
OPC Address Space Properties

The OPC UA Address Space browser may not list all available nodes. If you think a node should have property nodes, but they are not available do the following.

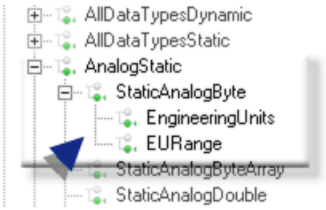
5. Select a node that you believe is missing properties in the tree.



6. Right-click the node; select Refresh.



Properties that were not available before are now available.



C	Scan Rate

If the **Update Criteria**, corresponds to:

- A **Polled** subscription, **Scan Rate** entries can schedule how often the point will read values from the OPC UA Server.
- An **Unsolicited** subscription (recommended), the Scan Rate is ignored.

The Publishing Interval is used to control how often the OPC UA Server will report when a value has changed.

D	Buffer data for redundant failover

To enable PTMRP to buffer the data from OPC DA and UA, select Buffer data for redundant failovercheck box in Point Device Properties. In case of a failover, the data changes are buffered to avoid any data loss during the switch over. When the secondary server starts working, the relevant Alarms and Events will be generated based on the buffered data. Set the value of

REDUND_BUFFER_TIMESPAN property based on time required for the secondary server to switch over. For more information, refer REDUND_BUFFER_TIMESPAN in Global Parameters.

OPC UA Client Technical Reference

OPC UA Client Technical Reference

- OPC UA Data Types Mapped to CIMPLICITY Data Types
- UA Applications in the Windows Certificate Store (MMC)
- Guidelines: Server/Instance Certificate Export/Import
- Guidelines: User Identity Certificate Export/Import

OPC UA Data Types Mapped to CIMPLICITY Data Types

- Data Mapping Notes
- Data Mapping Table

Data Mapping Notes

- Array Support

1. Only one-dimensional arrays are supported.
2. It is recommended to map OPC UA arrays to CIMPLICITY data types with the Elements property equal to the UA array size.

Note: The number of elements in a CIMPLICITY array point can be selected on the Point Properties dialog box>General tab>**Elements** field.

- If array elements are configured as not equal (or, if later, a UA variable array size is changed by the UA Server), then, by default, reading these points will fail.
- It is possible to allow a mismatch between a UA array size and CIMPLICITY point **Elements**, for the device level configuration; the property Allow Array Size Mismatch should be set to True.

Note: Even in this case the CIMPLICITY point **Elements** cannot be less than the UA Variable array size; truncation is not allowed.

1. It is possible to write to the range of array elements via CimScript API.

2. Scalar OPC UA types must be mapped to CIMPLICITY data types with number of **Elements** = 1

- 3D_BCD and 4D_BCD Types

3D_BCD and 4D_BCD types are not supported.

Data Mapping Table

Data Type Class	Data type	Description	
Analog	SINT	1 byte signed integer Range from -128 to 127	
	1 byte unsigned integer Range from 0 to 255		
	2 byte (16-bit) signed integers Range from -32,768 to 32,767		
	2 byte (16-bit) unsigned integers Range from 0 to 65,535		
	4 byte (32-bit) signed integers Range from -2,147,483,648 to 2,147,483,647		
	4 byte unsigned integers Ranging from 0 to 4,294,967,295		
	8 byte signed integers Range from -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807		
	8 byte unsigned integers Range from 0 to 18,446,744,073,709,551,615		
	4 byte or 8 byte floating point number. The length depends on the value in the • Project configuration file: ..\<project folder>/master/OpcUaOptions.json • **FloatsAre8bytes** field. The length, based on the value, is:		
		True	8 bytes
		False	4 bytes
		Default	True
		Note: Manual editing of the configuration file is required; the value cannot be changed through the User Interface.	
BOOL	BOOL	A one digit BOOLEAN point with a value of 0 or 1	
	An 8-bit array of BOOLEAN points		

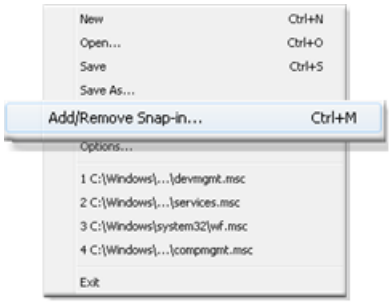
	A 16-bit array of BOOLEAN points		
	A 32-bit array of BOOLEAN points		
STRING	STRING	A character string of 1 element	
	A character string of 8 elements		
	A character string of 20 elements		
	A character string of 80 elements		

UA Applications in the Windows Certificate Store (MMC)

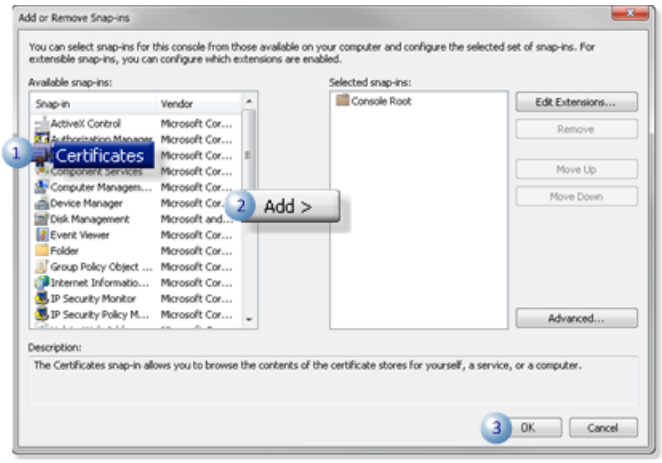
The Windows Certificates Store:

- Stores certificates by default in the Windows Certificate store **UA Applications** for the local machine.
- Uses the Certificates snap-in provided by Microsoft Windows, if you need to export or import certificates.

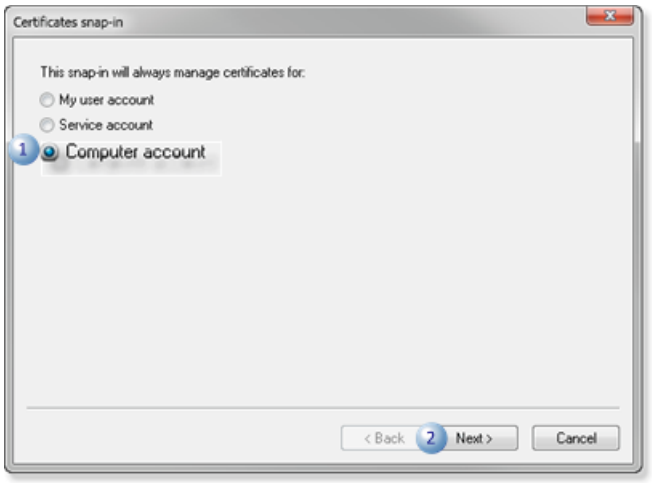
1. Open the Microsoft Console by typing mmc.exe at a Cmd window and pressing Enter.
2. Click File>Add/Remove Snap-in on the Console menu bar to run the Certificates Snap-in.



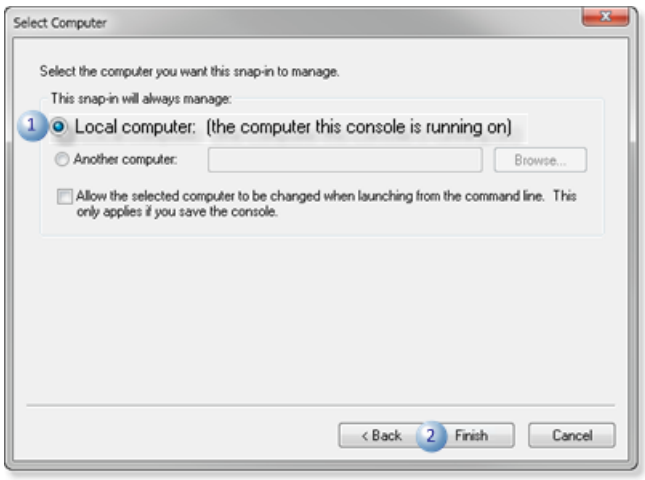
3. From the Add or Remove Snap-ins dialog box, select Certificates (1), click Add (2), and then click OK (3).

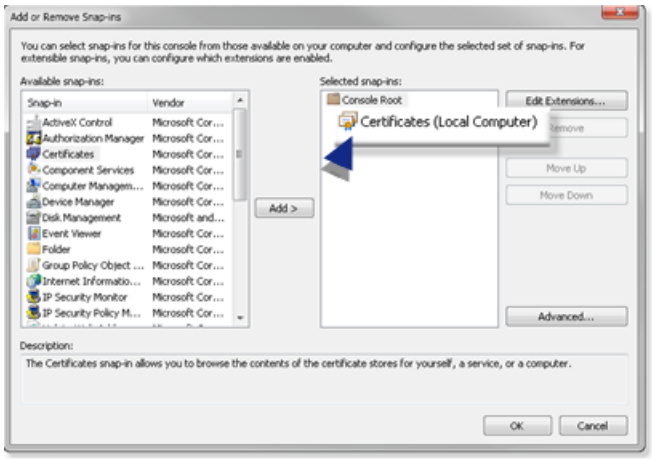


4. From the Certificates snap-in dialog box, check Computer account and click Next.



5. From the Select Computer dialog box, check the Local Computer (1), and click Finish (2).





6. Click OK.

   Result: The CIMPLICITY UA application certificates will be available for export/import.

   Run as Administrator

   Access to the Windows Certificate Store

   A user who creates the certificates must have proper access rights to the Windows Certificate store in order for the store to accept the certificates from the Workbench.

   Access rights can vary from one machine to another, depending on the:

   - Windows Operating System.
   - Local and/or domain security policies.

   In certain cases access can occur when the:

   - Workbench process is granted write access to the Windows Certificates store because the user is a member of the local machine's Administrators group.
   - User starts the Workbench as Administrator.

   Ultimately, it is the machine's administrator's job to configure access rights. CIMPLICITY OPC UA Client does not do any access rights configuration.

   Tip: To run the Workbench as administrator, right-click the Workbench in the Windows Start menu>HMI SCADA - CIMPLICITY v10.0section. Select Run as administrator on the Popup menu, and open the OPC UA Client project through the Workbench. If you use this method frequently, create a Workbench shortcut on your desktop.

   Existing Certificates

   Once certificates are created, no write access is required.

Therefore in order to read existing certificates, in most cases, there is no need to run the Workbench as Administrator.

Exceptions include the following.

- The certificates created in CIMPLICITY always includes public key and private key; the private key has the access rights settings that depend on the operating system. As a result, on some machines or in some operating systems, the Workbench user, even an Administrator, cannot access the private key. A user will still need to run the Workbench as Administrator in order to read, configure and test the projects.
- If the system administrator decides to run a `DevCom` process under a user account that is different from the default system account, then the user should be granted access rights to the Windows Certificates store.

If the user is not granted access rights the Workbench will have to be Run as Administrator.

By default, the CIMPLICITY OPC UA Device Communication process (CimOpcUaClient.exe) runs under a Windows system account, which has access rights to the Windows Certificates Store,

As a result, after a project starts it can successfully, you can read certificates, communicate with the UA Server, and retrieve point values.

Non-Secured Mode

Access to the certificates store is required only when write/read certificates are needed.

If a:

- Device is configured to connect in non-secured mode, an Instance certificate is not needed.
- UA Session user identity is Anonymous, then there is no need to encrypt or decrypt user name/password.

Default settings are non-secured communication, with an Anonymous user

By default: No certificates or access is required. There is no need to Run as Administrator.

Guidelines: Server/Instance Certificate Export/Import

- CIMPLICITY OPC UA Client Instance Certificate
- UA Server Instance Certificate

CIMPLICITY OPC UA Client Instance Certificate

In order for the OPC UA Server to trust the CIMPLICITY OPC UA client, the client Instance certificate must be in a Trusted folder on the OPC UA server. The certificate can be transferred from the OPC UA Client to the server by either of two methods.

Automatic Transfer

During an attempt to connect the client to the server, the client Instance certificate will be sent to the server directly.

The server's administrator can decide to trust it or reject it.

Export to a *.cer/*.der File

If the OPC UA Server system administrator requests a certificate do the following.

1. Right-click the Instance certificate in the Windows Certificate Store>UA Applications>Certificates pane.

   The Certificate Export Wizard opens.

2. Select the following as you go through the Wizard

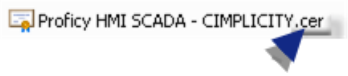
Screen	Select
Private Key	No, do not export the private key.
File Format	DER encoded binary X.509 (.CER)
File name	Name assigned to the .cer file. **Notes** • The name does not have to match the Instance certificate name. However, the name should make it clear what certificate is being used. • Click the Browse button that is on the screen to open a Windows browser and select the location/ enter the name to be applied.

   The new file will be available in the specified location after you exit the Certificate Export Wizard.

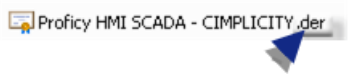
3. Find the *.cer file that was just created.

4. (In many instances) rename the file extension from:

*.cer



to *.der.



Note: Many OPC UA Servers only recognize the .der extension.

Result: The file is ready to send to the UA Server Administrator.

UA Server Instance Certificate

Important:

The default PKI root folder location is C:\ProgramData\CIMPLICITY.., which is a hidden folder.

Set Windows Explorer to display hidden folders.

The OPC UA Client will trust the OPC UA Server after the CIMPLICITY OPC UA Client is configured to trust the Server Instance certificate directly, by storing it in the location designated for trusted certificates.

Note: If the server certificate is not self-signed, it is enough to save the issuer's certificate in the trusted location, but also possible to store the Server certificate directly.

The certificate may come from a server that is trusted already, if:

- The certificate itself or one of its issuer's certificate is in the trusted certificates folder.
- Other issuers' certificates from the chain that are in the issuers certificates folder.

Note: This is not applicable for self-signed certificates.

When a user:

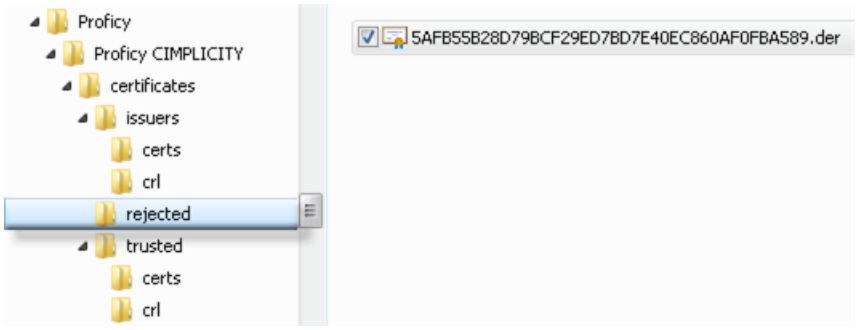
5. Selects a secured communication mode in Device dialog box>OPC UA DA Configuration>Connection tab>Communication Security *(page 291)* section.
6. Clicks the Test Connection button in the OPC UA Client Device dialog box.

   The connection attempt will fail because the:

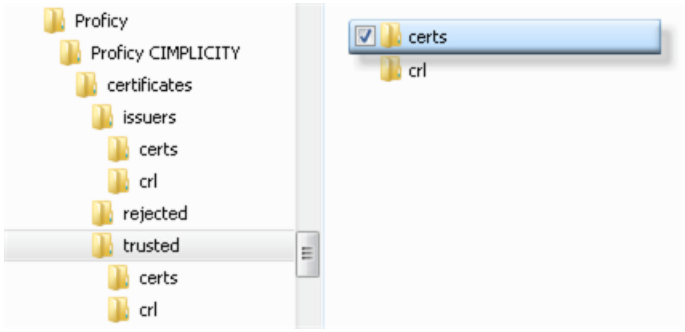
   a. CIMPLICITY OPC UA Client, initially, is not configured to trust the UA Server's certificate.
   b. Client side rejects the UA Server's certificate.

The OPC UA Server's certificate file (*[Thumbprint].der, where [Thumbprint] is the certificate thumbprint) will be stored in the following folder.

C:\ProgramData\Proficy\Proficy CIMPLICITY/certificates/rejected



7. Moves (cut/paste) the certificate to the following folder.
   C:\ProgramData\Proficy\Proficy CIMPLICITY/certificates/trusted/certs



The CIMPLICITY OPC UA Client will now trust the associated OPC UA server.

Note: A UA Server certificate can be issued by a certificate authority, which in turn can be issued by another higher level certificate authority.

As a result it can contain a chain of certificates.

If this is the case, the system administrator will need to determine which certificate should be placed in the **trusted>certs** folder and which others should be placed n the **issuers>certs** folder.

Guidelines: User Identity Certificate Export/Import

- Export/Import Overview
- User Identity Certificate: Export
- User Identity Certificate: Import

Export/Import Overview

Important:

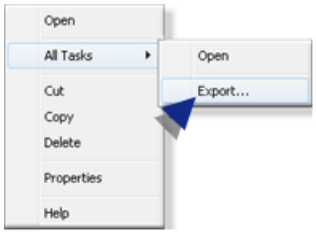
- The Windows Certificate Store must be available on the target machine in addition to the local machine.
- On the local machine, when the certificate is exported:
- Windows exports the certificate and private key in Personal Information Exchange –PKCS #12 format.
- The extension of the file is *.pfx by default and will be protected during Certificate Export Wizard.

For more specific to your system, ask your system administrator or security expert.

- Identity Certificate: Export
- Identity Certificate: Import

User Identity Certificate: Export

1. Right-click the Instance certificate in the Windows Certificate Store>UA Applications>Certificates pane.
2. Select Export on the Popup menu.



   The Certificate Export Wizard opens.

3. Select the following as you go through the Wizard

Screen	Select
Private Key	Yes, export the private key
File Format	Personal Information Exchange - PKCS #12 (.PFX)
Password	Password that a user will have to enter when importing the exported certificate file into the Windows Certificate Store on the target machine.
File name	Name assigned to the .pfx file. **Note:** Click the Browse button that is on the screen to open a Windows browser and select the location/enter the name to be applied.

When you exit the Certificate Export Wizard, the new *.pfx file will be available in the specified location.

4. Copy/paste the *.pfx file to the target machine.

   Result: The *.pfx file can be imported into the Windows Certificate Store.

   User Identity Certificate: Import

5. Do either of the following on the target machine.
   - Double-click the *.pfx file that was just pasted in a target machine folder.
   - Right click UA Applications or UA Applications>Certificates in the Windows Certificate Store; select All Tasks>Import on the Popup menu.

   A Certificate Import Wizard opens when you use either method.

6. Select the following as you go through the Wizard.

Screen	Select
Welcome	Local Machine

File to Import	*.pfx file that was just pasted on the target machine. **Note:** Click the Browse button that is on the screen to open a Windows browser and select the file. Make sure to select the Personal Information Exchange format.
Private Key Protection	Mark this key as exportable. This will allow you to back up or transport your keys at a later time

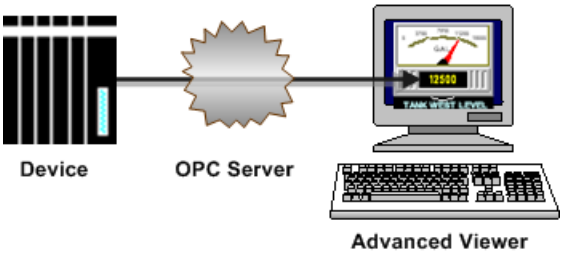
Certificate Store	Check Place all certificates in the following store. Keep the UA Applications entry that displays in the **Certificate Store** field.

When the Certificate Import Wizard is exited, Windows will import the certificate and private key from the *.pfx file; the certificate will be available in the Windows Certificate Store on the target machine.

Chapter 18. CIMPLICITY Advanced Viewer OPC Client

About the Advanced Viewer

A CIMPLICITY Advanced Viewer option is an OPC Client that can display point values that it collects from an OPC Server through CimView screens and the Point Control Panel without a running CIMPLICITY project.



Note: If the OPC Server communicates with software, e.g. the CIMPLICITY OPC server, rather than hardware, a CIMPLICITY project may be required to run. However, that is a server requirement, not the Advanced Viewer requirement. The Advanced Viewer offers the project not running benefit when point values are reported by an OPC Server that talks with hardware. The Advanced Viewer will not work in demo mode.

Steps to set up and use the Advanced Viewer are as follows.

Step 1 *(page 330)*	Set up the OPC Server for the Advanced Viewer.
Step 2 *(page 344)*	Configure user interfaces for the Advanced Viewer.
Step 3 *(page 364)*	Set up each Viewer.

Step 1. Set up the OPC Server for the Advanced Viewer

Step 1. Set up the OPC Server for the Advanced Viewer

The following steps are required on the OPC Server side for the Advanced Viewer to work.

Step 1.1 *(page 331)*	Distribute the OPC Server files.
Step 1.2 *(page 331)*	Enter specification in the ptopc_config.xml file.
Step 1.3 *(page 344)*	Specify log in requirements.

Step 1.1 Distribute the OPC Server Files

You can use any OPC Foundation Compliant OPC server for the Advanced Viewer.

Make sure, and confirm that the OPC Server you plan to use is:

- Installed correctly and can be accessed by an OPC client.
- Registered *(page 258)* so Windows can find it when an OPC client asks for it by name.

Note: Windows looks up the OPC Server location in the registry.

The OPC Server directory can be on the same node as the Advanced Viewer or a different node; the node can be referenced in the Ptopc_config.xml file, which is also part of the OPC Server setup.

Step 1.2. Enter Specifications in the ptopc_config.xml File

Step 1.2. Enter Specifications in the ptopc_config.xml File

The ptopc_config.xml file provides the Advanced Viewer with the specifications it needs to collect data from an assigned OPC Server based on your system's requirements.

Step 1.2.1 *(page 331)*	Open the distributed ptopc_config.xml file.
Step 1.2.2 *(page 334)*	Enter Advanced Viewer specifications in the ptopc_config.xml file.

Step 1.2.1. Open the Distributed ptopc_config.xml File

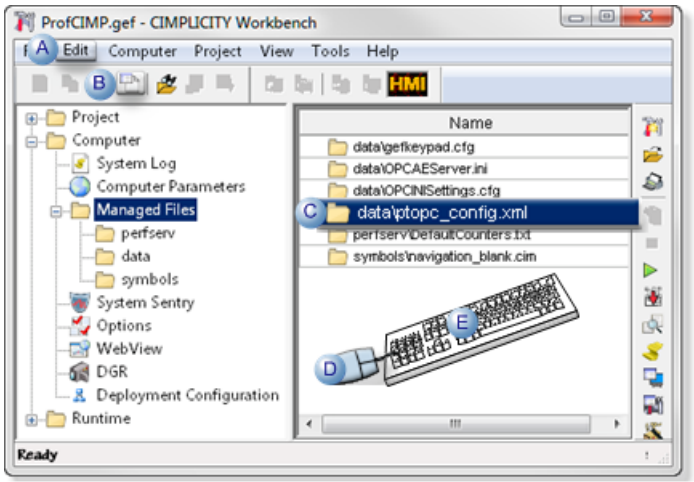
Proficy CIMPLICITY includes a ptopc_config.xml file that is ready to be filled in with your system specifications.

Use either method to open the ptopc_config.xml file.

- Workbench Computer Managed files.
- Window Explorer

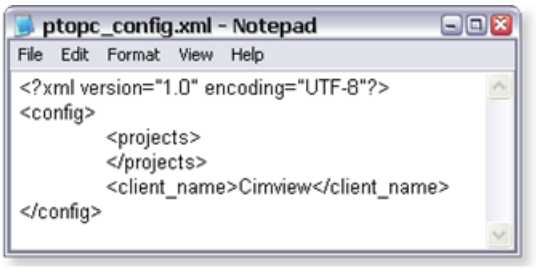
Workbench Computer Managed Files

1. Open a CIMPLICITY Workbench.
2. Select Computer>Managed Files in the Workbench left pane.
3. Select data\ptopc_config.xml in the Workbench right pane.
4. Do one of the following.



A	Click Edit>Properties on the Workbench menu bar.
B	Click the Properties button on the Workbench toolbar.
C	Right-click data\ptopc_config.xml; select Edit on the Popup menu.
D	Double-click data\ptopc_config.xml.
E	Press Alt+Enter on the keyboard.

   a. Result: The ptopc_config.xml file opens in Notepad. The first time it opens, it is ready for your system's specifications.



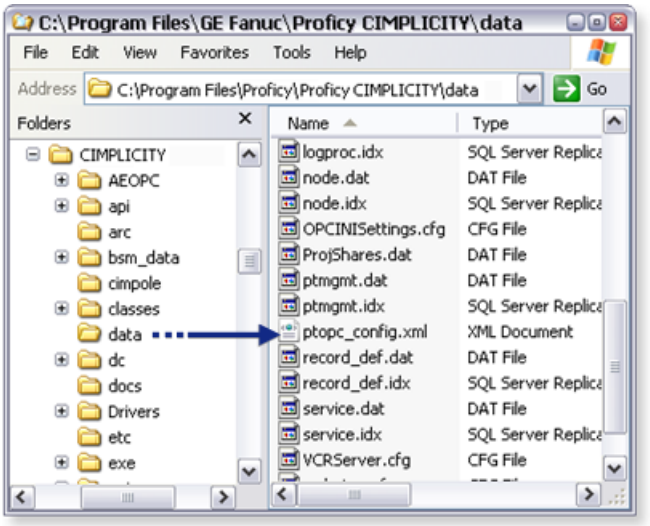
Windows Explorer

5. Open Windows Explorer.
6. Select C:\Program Files\Proficy\Proficy CIMPLICITY\data

   Where

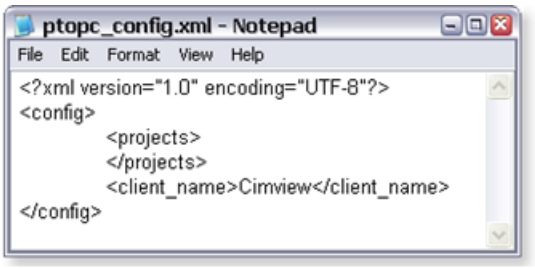
   C:\Program Files\Proficy\Proficy CIMPLICITY is the default installation directory.

7. Find ptopc_config.xml in the Windows Explorer window right pane.



8. Open ptopc_config.xml in an .xml editor or text editor, e.g. Notepad.

The ptopc_config.xml file opens in the selected text editor. The first time is opens, it is ready for your system's specifications.



Important:

- If you do not plan to use the Advanced Viewer, leave ptopc_config.xml in the ..\data directory.
- Make sure the ptopc_config.xml file is in the ..\data directory on every computer that will use the Advanced Viewer.

Step 1.2.2. Enter Advanced Viewer Specifications in the ptopc_config.xml File

Step 1.2.2. Enter Advanced Viewer Specifications in the ptopc_config.xml File

The ptopc_config.xml file includes three basic sections.

Note: The following sample file includes more lines than are required. The descriptions on the next pages identify required lines from optional lines.

1 (page 336)	1	`<?xml version="1.0" encoding="UTF-8"?>`
	2	`<config>`
	3	
2 (page 337)	4	
	5	
	6	
	7	
	8	
	9	

	10	
	11	
	12	
	13	
	14	
	15	
	16	
	17	
	18	
	19	
	20	
	21	
	22	
	23	
	24	
	25	
[1 *(page 336)*](#)	26	
[3 *(page 341)*](#)	27	
	28	

	29	
	30	
	31	
	32	
	33	
	34	
	35	
	36	
1 *(page 336)*	37	`</config>`

Step 1.2.2.1 *(page 336)*	Ptopc_config.xml file: Framework
Step 1.2.2.2 *(page 337)*	Ptopc_config.xml file: Project definitions
Step 1.2.2.3 *(page 341)*	Ptopc_config.xml file: Global specifications

Step 1.2.2.1. Ptopc_config.xml File: Framework

Lines that are included in the distributed ptopc_config.xml file and make up the file framework are as follows.

Code Snippet from Sample File

1 *(page 337)*	`<?xml version="1.0" encoding="UTF-8"?>`
2 *(page 337)*	`<config>`

3 *(page 337)*	
	`...`
26 *(page 337)*	
	`...`
37 *(page 337)*	`</config>`

Snippet Line Description

1	`<?xml version="1.0" encoding="UTF-8"?>`
	(Required)
	Included in the distributed file.

2/37	< config >...</config>
	(Required)
	Identifies the beginning and end of the configuration. Included in the distributed file.

3/26	< projects >...</projects>
	(Required)
	• Identifies the section for defining pseudo-projects. • At least one pseudo-project must be defined. • Included in the distributed file.

Step 1.2.2.2. Ptopc_config.xml File: Project Definitions

Lines in the ptopc_config.xml file that define individual projects are as follows.

- The number of lines in the list of projects depends on your system requirements.
- This sample ptopc_config.xml file lists three projects. Each project includes the required lines and zero or more additional lines.

Note: If optional lines are not added and there is a default value, the default value will be used.

Code Snippet from Sample File

```
...
4 (page 339)  <ATLSIM>
5 (page 339)      <ptcom_prog_id>PtComOpc</ptcom_prog_id>
6 (page 339)      <opc_prog_id>NDI.OPCATLSimSvr.1</opc_prog_id>
7 (page 340)      <opc_server_node_name>NodeName.1</opc_server_node_name>
8 (page 340)      <dcom_timeout>10000</dcom_timeout>
9 (page 340)      <dcom_threshold>20000</dcom_threshold>
10 (page 340)     <ping_interval>15000</ping_interval>
11 (page 340)     <reconnect_interval>15000</reconnect_interval>
12 (page 340)     <hang_protection_timeout>120000</hang_protection_timeout>
13 (page 340)     <item_property_subscriptions>TRUE</item_property_subscriptions>
14 (page 340)     <RemoveInvalidItems>TRUE</RemoveInvalidItems>
15 (page 341)     <RemoveItemsOnRemoveGroup>FALSE</RemoveItemsOnRemoveGroup>
16 (page 341)  </ATLSIM>

               <CIMPOPC>
```

		<ptcom_prog_id>PtComOpc</ptcom_prog_id>
		<opc_prog_id>NDI.OPCATLSimSvr.1</opc_prog_id>
	</CIMPOPC>	
	<KEPLOCAL>	
		<ptcom_prog_id>PtComOpc</ptcom_prog_id>
		<opc_prog_id>NDI.OPCATLSimSvr.1</opc_prog_id>
		<opc_server_node_name>NODE3< /opc_server_node_name>
	</KEPLOCAL>	
	...	

Snippet Line Description

4	< project name>
	(Required)
	• <Project name>/</Project name> identify the section that defines specifications for an individual pseudo-project. • At least one pseudo-project is required, as follows. • For each OPC Server. • For each node that an OPC Server is on, if the OPC Server is on more than one node. **Note:** You can create two pseudo-projects that point to the same OPC Server on the same node. However, this is not necessary. • A name that is assigned to a pseudo-project. • Must not have the same name as any standard CIMPLICITY server projects, either locally or anywhere on the network. • Name is limited to the same length as a standard CIMPLICITY server project name. • The pseudo-project includes a number of point ID's does not have to exist as a CIMPLICITY project. • Each OPC Server can have more than one pseudo-project. • If the OPC Server is running on more than one node, there must be a pseudo-project for each node.
5	<ptcom_prog_id>PtComOpc</ptcom_prog_id>
	(Required) Identifies the program ID. **Important:** PTComOpc must be the PtCom program ID.
6	<opc_prog_id>NDI.OPCATLSimSvr.1</opc_prog_id>

	(Required)
	Program ID in the registry that identifies the OPC server to be used. This is the same Program ID used by the OPC client communications device in the **Network Address** field of an OPC device. Example NDI.OPCATLSimSvr.1
7	`<opc_server_node_name>`NodeName.1`</opc_server_node_name>`
	(Optional)
	Node Name is an off-node OPC Server node name. The entry identifies the node where the OPC server resides. It can be a DNS (WINS) resolvable node name or an IP address. If you do not specify a node name, the local node is assumed. Example NodeName.1
8	`<dcom_timeout>`10000`</dcom_timeout>`
	(Optional)
	The amount of time the client must wait for a valid connection to the OPC server. When any action is performed that involves the OPC server, those actions happen on a separate thread. The main thread will wait the DCOM timeout for the other thread to complete the call to the OPC server. If the separate (worker or pool) thread does not return in that amount of time, the main thread is allowed to continue (with an error noted).
9	`<dcom_threshold>`20000`</dcom_threshold>`
	(Optional)
	How long the OPC Client waits to abort the connection to the OPC server after detecting that the server has not responded to a ping. Enter the DCOM timeout threshold in milliseconds (ms). Default: 20000 (milliseconds)
10	`<ping_interval>`15000`</ping_interval>`
	(Optional)
	How often the OPC client pings the OPC server in milliseconds. Default: 15000 (milliseconds)
11	`<reconnect_interval>15000</reconnect_interval>`
	(Optional)
	Time between reconnection attempts (i.e. RestartDelay). This time does not include the time it takes Microsoft's COM (DCOM) engine to determine if it is able to launch the OPC server application. Default: 15000 (milliseconds)
12	`<hang_protection_timeout>`120000`</hang_protection_timeout>`
	(Optional)
	Time before the Advanced Viewer times out. An error is written to the core log reporting that the OPC Server appears to be hung. Default: 120000 (milliseconds)
13	`<item_property_subscriptions>`TRUE`</item_property_subscriptions>`
	(Optional- Available with OPC Servers that implement the IOPCBrowse interface as part of the OPC DA 3.0 Specifications)
	TRUE
	FALSE
14	`<RemoveInvalidItems>TRUE</RemoveInvalidItems>`
	(Optional)
	TRUE

	FALSE
	Default
15	`<RemoveItemsOnRemoveGroup>FALSE</RemoveItemsOnRemoveGroup>`
	(Optional)
	TRUE
	FALSE
	Default
16	`<`/project name>
	(Required) • <Project name>`/<`/Project name`>` identify the section that defines specifications for an individual pseudo-project. • `<`/project name`>` is the end of specifications for the selected project.

Step 1.2.2.3. Ptopc_config.xml File: Global Specifications

Lines in the ptopc_config.xml file that define specifications for all projects are as follows.

The number of lines in the list depends on your system requirements.

Note: If optional lines are not added and there is a default value, the default value will be used.

Code Snippet from Sample File

...
27 *(page 342)*
28 *(page 342)*
29 *(page 342)*
30 *(page 342)*
31 *(page 342)*
32 *(page 343)*

33 *(page 343)*
34 *(page 343)*
35 *(page 343)*
36 *(page 343)*
37 *(page 343)*

Snippet Line Description

27	`<client_name>`CimView`</client_name>`
	(Optional)
	Included in the distribution file. This line allows the Advanced Viewer to identify itself to the OPC Server via the IOPCCommon::SetClientName function. Because the Advanced Viewer is designed to work with external OPC servers (not the OPC Server that comes with CIMPLICITY) you might have to make use of IOPCCommon::SetClientName so the OPC Server can perform special logic based on what type of client is connecting to it. You enter the Advanced Viewer name as a string between the XML tags. Example In the following string: `<client_name>AdvancedViewer</client_name>` `AdvancedViewer` will get passed as a parameter to the SetClientName function. This allows the Advanced Viewer (which is an OPC client) to identify itself as a particular type of client to the external OPC Server. **Note:** The OPC Foundation Web site provides more information about the IOPCCommon::SetClientName function in its OPC DA 3.0 Specification.
28	`<default_update_rate>`1000`</default_update_rate>`
	(Optional)
	How often the OPC server updates the items in this group with fresh device data. It also determines how often to send Default: 1000 (milliseconds) Caution: CPU usage increases when updates are set to occur more frequently.
29	`<update_rates>`
	(Optional)
	Starts update rates for selected groups that are different from the default update rate.
30/31	`<MY2SECGROUP>2000<_MY2SECGROUP>/<MY2SECGROUP>2000<_MY2SECGROUP>`
	(Optional)

	List of groups with assigned update rates that are different from the default rate.
32	`</update_rates>`
	(Optional)
	Ends the list of groups with their own assigned update rates.
33	`<max_string_length>80</max_string_length>`
	(Optional)
	Sets the maximum string length that will be accepted from the OPC Server. If the actual string length is higher than the entered value, it will be truncated/ Default: 80
34	`<max_array_string_length>80</max_array_string_length>`
	(Optional)
	Sets the maximum array string length that will be accepted from the OPC Server. If the actual string length is higher than the entered value, it will be truncated/. Default: 80
35	`<property_id_display_format>5003</property_id_display_format>`
	(Optional)
	If your OPC server supports a Property ID Display Format attribute and uses a different attribute number from 5003, you can enter the number your OPC server uses with the above line in the ptopc_config.xml file. Default: 5003
36	`<trace_flags>`0`</trace_flags>`
	A Trace flags utility is required for the `<trace_flags>` setting. If you experience problems contact your support representative.
37	`</config>`
	(Required)
	Ends the ptopc_config.xml file.

Other Optional Lines

The Advanced Viewer used the following two standard OPC property ID values assigned by the OPC Foundation.

If your OPC Server uses different values, include these lines in the global section of the **ptopc_config.xml** file..

`<property_id_display_limit_high>102</property_id_display_limit_high>`	(Optional)
	Assigned OPC property ID value for the Property ID Display Limit High attribute. Default: 102
`<property_id_display_limit_low>102</property_id_display_limit_low>`	(Optional)

	Assigned OPC property ID value for the Property ID Display Limit Low attribute. Default: 103

Step 1.3. Specify Log In Requirements

Your OPC Server determines whether or not login is required.

If log in is required, log in for the Advanced Viewer is as follows.

- Login will only be requested one time for the current user.
- The Login Panel will list the log in for the OPC pseudo-project.

You can add saved logs in the Login Panel for an Advanced Viewer project.

- When cancelling a login, items display as unavailable.
- Log out Items display as unavailable and cannot be set.
- Users can log into more than one OPC Server at a time. A user can log out of one Server; Servers that the user is still logged into will continue to display their values; the values can be set.
- If you are using the Advanced Viewer in a Vista operating system and are configuring DCOM *(page 199)* in a domain, set the Advanced Viewer `ptopc` service (ptopc.exe) to login as a specific domain user.

Step 2. Configure User Interfaces for the Advanced Viewer

Step 2. Configure User Interfaces for the Advanced Viewer

The Advanced Viewer can use several CIMPLICITY user interfaces even when a CIMPLICITY project is not running. The interfaces you use, of course, depend on your system requirements. The exact entries depend on what your OPC Server will accept.

Following are some options and examples to demonstrate the flexibility and possibilities for user interfaces with the Advanced Viewer.

Option 2.1 *(page 345)*	Start the Advanced Viewer.
Option 2.2 *(page 347)*	Create one or more CimEdit/CimView screens.

Option 2.3 *(page 360)*	Create a Point Control Panel for Advanced Viewer Points.
Option 2.4 *(page 363)*	Create a Quick Trends chart.

Option 2.1. Start the Advanced Viewer

If you want to test your configuration you must start a Viewer.

You wait to start the Viewer when you test your first runtime application. However, you can also start the Viewer in the CIMPLICITY® Options dialog box.

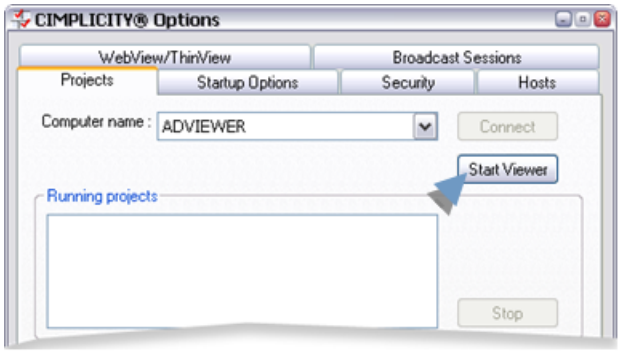
- Use the CIMPLICITY® Options dialog box.
- Open a CimEdit, CimView screen or the Point Control Panel

Use the CIMPLICITY® Options Dialog Box

1. Open the CIMPLICITY® Options dialog box.
2. Select the General tab.

   The local computer name displays in the **Computer name** field.

3. Click Start Viewer.



   Result: When the Viewer is started the Router starts running.

   Note: The Advanced Viewer processes do not start. They will be started when either a CimView screen or Point Control Panel is opened.

   Open a CimEdit, CimView Screen or the Point Control Panel

if a CIMPLICITY project is not running, do the following when Advanced Viewer applications are opened in several CIMPLICITY runtime user interfaces.

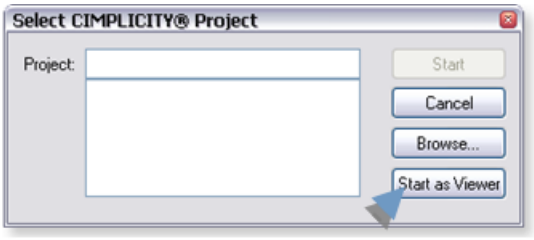
4. Open one of the following.
   - CimEdit screen.
   - CimView screen.
   - Point Control Panel

   A Select CIMPLICITY® Project dialog box opens.

5. Do one of the following.

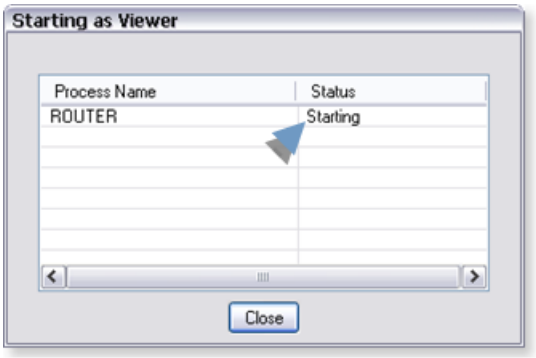
   If the Viewer is not running

   Click Start as Viewer.



The following processes start, if the ptopc_config.xml file was configured correctly.

- CIMPLICITY Router.
- OPC Communications program.
- OPC Server that was identified for the Advanced Viewer.

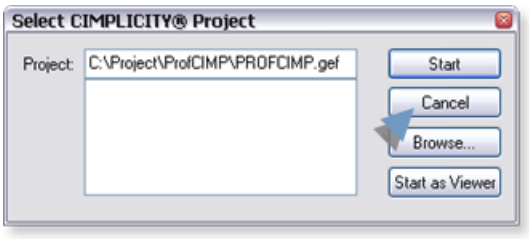


When the Viewer and the associated Advanced Viewer processes are running, your configured CimView screens and Point Control Panels can be tested and/or used.

If the Viewer is running.

A Select CIMPLICITY® Project dialog box may still open.

Click Cancel.



The Advanced Viewer processes remain available to run your configured CimView screens and Point Control Panels.

Option 2.2. Create One or More CimEdit/CimView Screens

Option 2.2. Create One or More CimEdit/CimView Screens

The Advanced Viewer can display point values on a CimView screen, even if a CIMPLICITY project is not running, wherever CimEdit accepts point values.

- Guidelines
- Examples of CimEdit features configured for the Advanced Viewer.

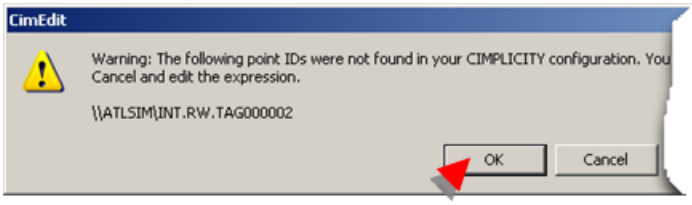
Guidelines

CimEdit configuration for the Advanced Viewer is basically the same as configuration for any CimEdit/CimView screen.

However:

- When you leave a field in which you entered an item ID that is not a point in a CIMPLICITY project, a message box will open warning you that the "...following point IDs were not found in your CIMPLICITY configuration."

Click OK.



The message box will close.

- Make sure you observe standard syntax rules when you enter item ID's, in addition to the Advanced Viewer syntax.

Example

A group tag is used in a color animated expression.

'\\ATLSIM\<NORTH>INT.RW.TAG000001' EQ 200

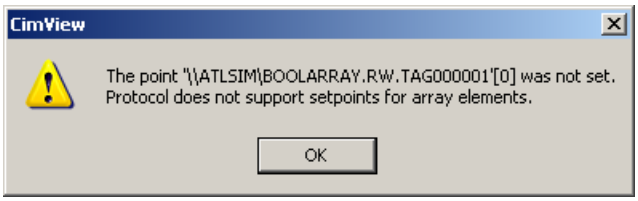
The item ID must be enclosed in single quotes because of the <> around the group name.

- The Advanced Viewer does not support array set points in CimView.

An error message will display if a user attempts to set the array point in CimView, reporting that

The point <point name> was not set. Protocol does not support setpoints for array elements.

Example



Note: Array set points are supported in the Point Control Panel.

Examples of CimEdit Features Configured for the Advanced Viewer

The configuration for several examples of CimEdit features that can be used with the Advanced Viewer are based on an OPC Server named ATLSIM.

ATLSIM OPC Server accepts:

- Hierarchical item ID's.

The hierarchy is delineated by periods in the point syntax.

- Groups.

The groups are identified with <> in the item ID syntax.

- Read-write capability.

- Several point types, including:
- INT
- BOOL
- STRING

Examples include the following.

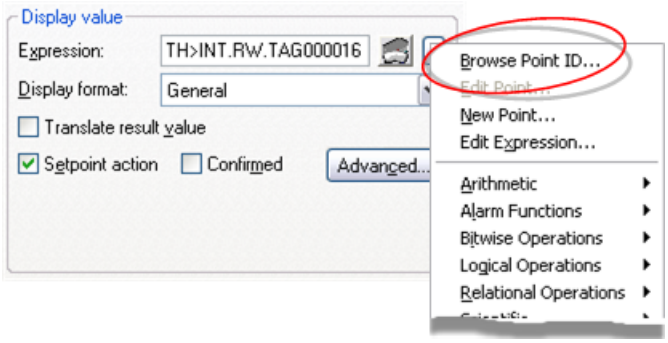
Example	2.2.1 *(page 349)*	CimEdit: Point Browser.
Example	2.2.2 *(page 350)*	CimEdit: Variables.
Example	2.2.3 *(page 351)*	CimEdit: Text objects with set points.
Example	2.2.4 *(page 352)*	CimEdit: Text button objects with procedures.
Example	2.2.5 *(page 355)*	CimEdit: Animation.
Example	2.2.6 *(page 358)*	CimEdit: Script..
Example	2.2.7 *(page 359)*	CimEdit: Point View and Expression View.

Example 2.2.1. CimEdit: Point Browser

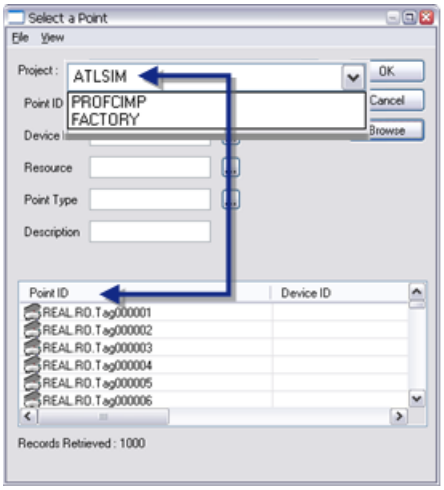
1. Select a field that allows a point as a value.
2. Use any of CimEdit's methods to open a Point Browser.

   Example

   Browse Point ID can be selected on a Popup menu to the right of an **Expression** field.



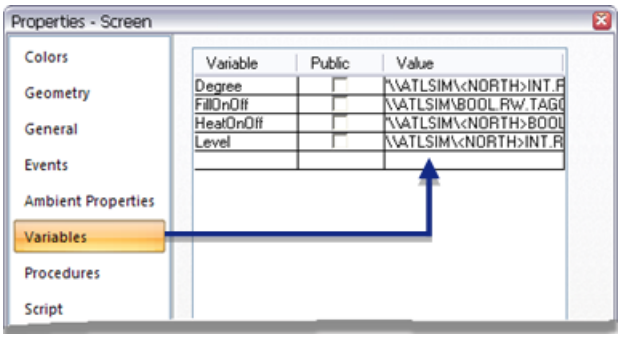
A Select a Point browser opens. You can select points from the OPC Server or any available CIMPLICITY projects.



Example 2.2.2: CimEdit Variables

The CimEdit screen includes some variables to facilitate configuration when the same item ID needs to be available for more than one entry.

The Variables tab in the Properties - Screen dialog box lists the following variables whose values are supplied by Advanced Viewer points.



Variable	Value

Degree	'\\ATLSIM\<NORTH>INT.RW.TAG000006'
FillOnOff	\\ATLSIM\<NORTH>BOOL.RW.TAG000001
HeatOnOff	\\ATLSIM\<NORTH>BOOL.RW.TAG000010
Level	\\ATLSIM\<NORTH>INT.RW.TAG000002

Note: The variable Degree will be used in a color animation expression. Therefore the value is entered in single quotes.

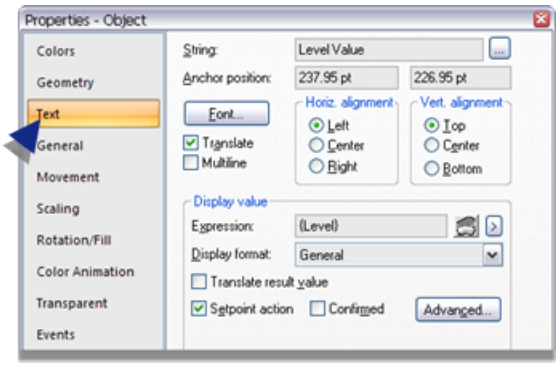
Example 2.2.3. CimEdit: Text Objects with Set Points

Text on the screen displays tag values for tags that can be set at the Viewer.

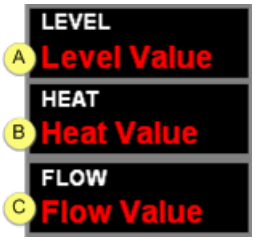
The Text tab is selected in the Point Properties - Object dialog box for each value.

- Text configuration example.
- Text runtime example.

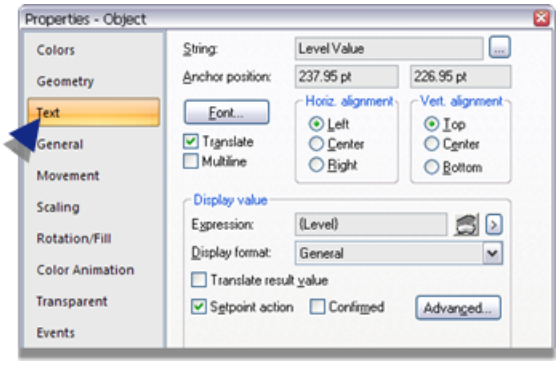
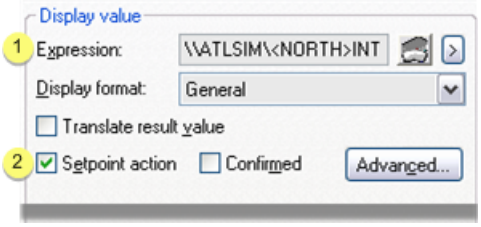
Text Configuration Example



The following values are entered in the **String** field.

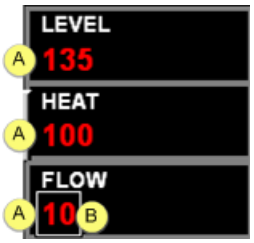


A	Level Value The Level Value text object uses the following configuration

	
B	Heat Value The Heat Value text object uses the following configuration.
	
C	Flow Value The Flow Value uses the item ID for the tag that reports the flow rate and allows a setpoint.

Text Runtime Example

During runtime:



A	The text objects display the Advanced Viewer point values.
B	The values can be set.

Example 2.2.4. CimEdit: Text Button Objects with Procedures

The CimEdit screen has two text buttons. Both buttons are configured with a Mouse Up event with a Toggle set point procedure.

- Text button configuration example.
- Text button runtime example.

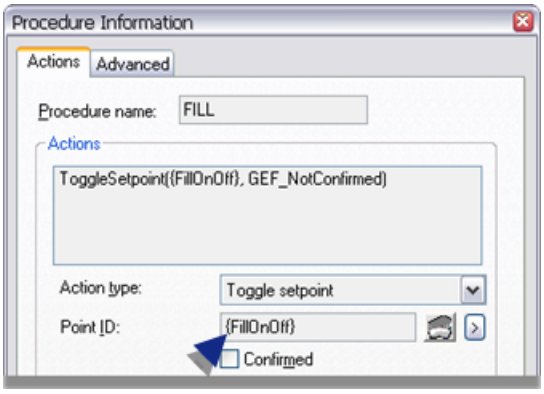
Text Button Configuration Example

The buttons are as follows.

FILL *(page 353)*	Operators can start and stop the water flow into the reservoir.
HEAT *(page 353)*	Operators can turn heat on and off.

FILL

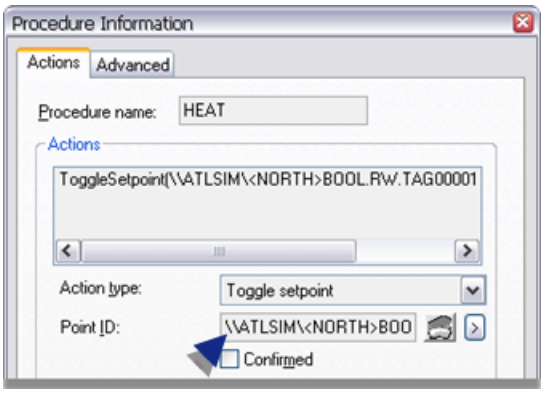
The FILL Toggle setpoint procedure requires a value in the **Point ID** field.



Field	Value
Point ID	{FillOnOff} (One of the variables that was defined at the screen level)

HEAT

The HEAT Toggle setpoint procedure requires a value in the **Point ID** field.

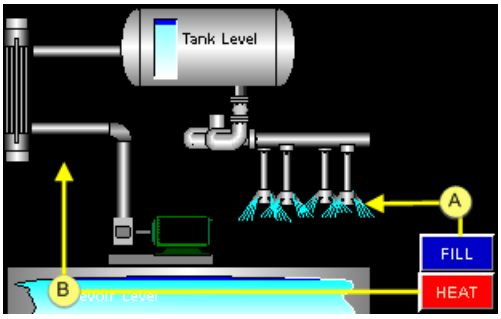


Field	Value
Point ID	\\ATLSIM\<NORTH>BOOL.RW.TAG000010 A BOOLEAN group item ID

Text Button Runtime Example

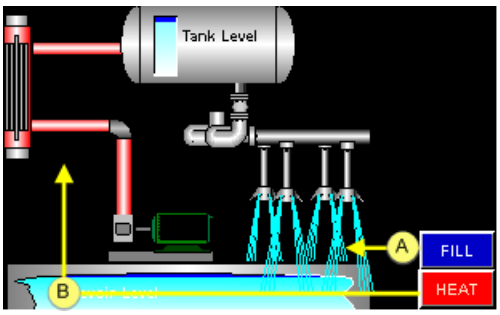
During runtime, the text buttons turn the fill and heat:

Off



A	The FILL button toggles the water valve closed.
B	The HEAT button toggles the heat off.

On



A	The FILL button toggles the water valve open.

B	The HEAT button toggles the heat on.

Example 2.2.5. CimEdit: Animation

The Advanced Viewer can display CimEdit/CimView animations.

- Animation configuration example.
- Animation runtime example.

Animation Configuration Example

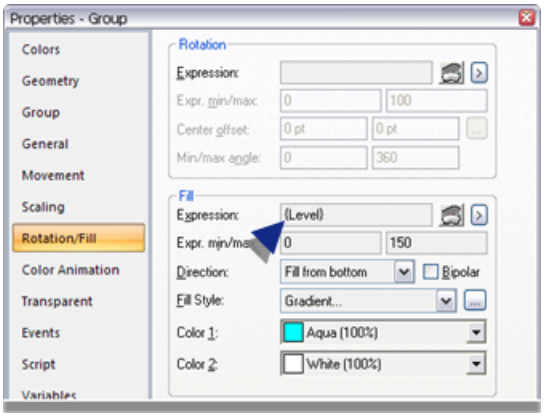
The Overview screen uses the following animations.

- Fill
- Scaling
- Color animation

Fill

A reservoir on the screen fills or empties to report the tag that displays the level.

The Fill/Movement tab is selected for the fill object's Properties dialog box.



The variable `{Level}` that was defined at the screen level is used to direct the fill.

The variable definition is:

\\ATLSIM\<NORTH>INT.RW.TAG000002

Where

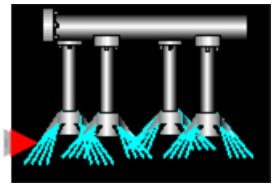
\\ATLSIM is the project name

<NORTH> is the group

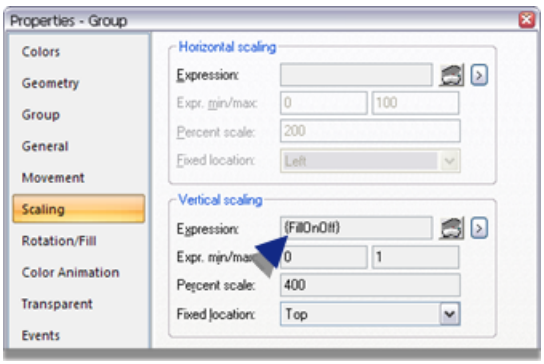
INT.RW.TAG00000S is the integer tag ID.

Scaling

Water flowing into the CimEdit screen's reservoir graphically shows if the flow is turned on or off.



The Scaling tab is selected for the fill object's Properties dialog box.



The variable `{FillOnOff}` that was defined at the screen level is used to expand and contract the flow lines into the reservoir.

The variable definition is:

\\ATLSIM\<NORTH>BOOL.RW.TAG000001

Where

\\ATLSIM is the project name

<NORTH> is the group

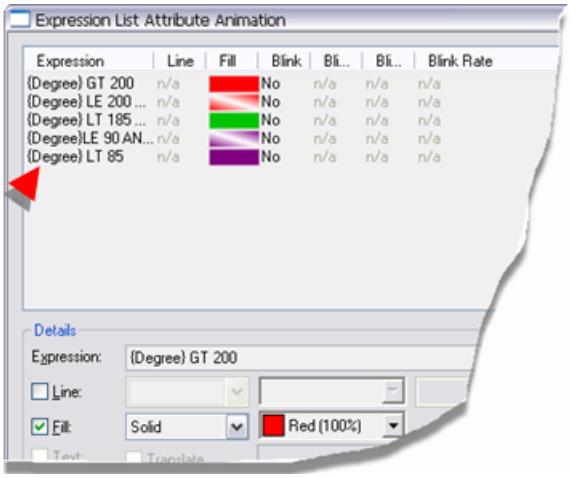
BOOL.RW.TAG000001 is the BOOLEAN tag ID.

Color Animation

A light is used to report the temperature status.

Color animation is used to display colors that signal the temperature status.

Note: The animation is applied to the rectangle object in the light group.



The expressions for the color animation use the variable `{Degree}`, which was defined at the screen level.

The variable definition is:

'\\ATLSIM\<NORTH>INT.RW.TAG000006'

Where

' ' protects the group enclosures <> from being evaluated in the expression.

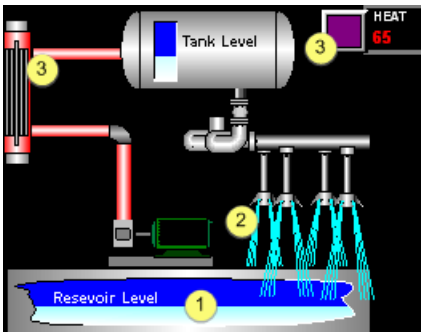
\\ATLSIM is the project name

<NORTH> is the group

INT.RW.TAG000006 is the integer tag ID.

Animation Runtime Example

The several animation configurations operate based on the configuration for the real point values.



1	Fill.
2	Scaling.
3	Color animation.

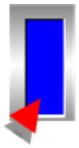
Example 2.2.6. CimEdit: Script

Important: The Advanced Viewer currently only supports CimEdit scripts; it does not support EMRP scripts.

The Advanced Viewer Overview screen uses an `OnMouseUp` event and script to display message boxes that report temperature, level and flow values from the WEST area.

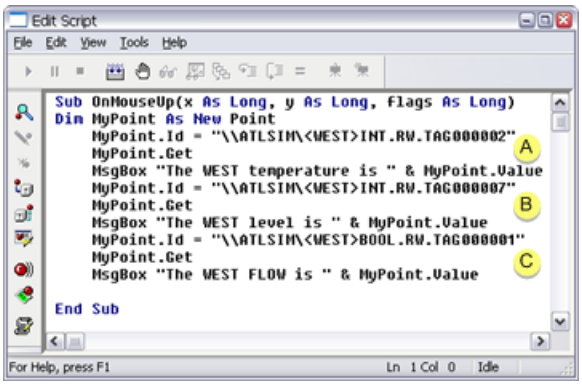
- Script configuration example.
- Script runtime example.

Script Configuration Example



The script for this example gets point values from item ID's in the ATLSIM Server's WEST group.

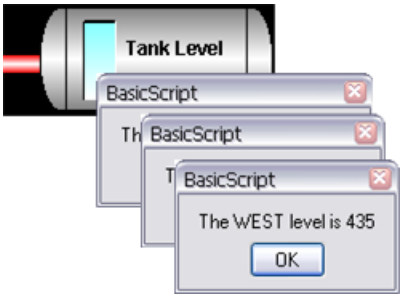
The item ID's are as follows.



A	\\ATLSIM\<WEST>INT.RW.TAG000002 Where \\ATLSIM is the project <WEST> is the group INT>RW.TAG000002 is the tag ID.
B	\\ATLSIM\<WEST>INT.RW.TAG000007
C	\\ATLSIM\<WEST>BOOL.RW.TAG000001

Script Runtime Example

During runtime when an operator clicks the tank cutout Basic Script message boxes report the heat, level and flow values for the ATLSIM OPC Server's WEST group.



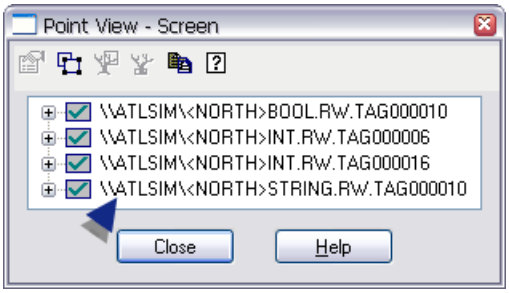
Example 2.2.7. CimEdit Point View and Expression View

The Advanced Server points and expressions can be reviewed in the Point View and Expression View window.

- Point View
- Expression View

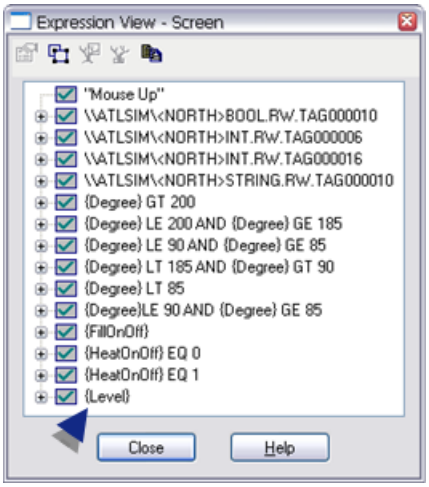
Point View

When the Point View window is opened, it displays the points associated with the selected) Advanced Viewer Overview screen.



Expression View

When the Expression View window is opened, it displays the expressions associated with the (selected) Advanced Viewer screen.



Option 2.3. Use a Point Control Panel for Advanced Viewer Points

The Advanced Viewer

- Can display point values from the OPC Server through the Point Control Panel.
- Enables you to use the Select a Point browser to find and select points from the OPC Server.

The Point Control Panel can be opened to display points using a:

- CimEdit/CimView screen.
- Point Control Panel (.ppl) file.
- List a Point browser.
- Point Control Panel file sections.

CimEdit/CimView screen

A Point Control Panel can be opened for the Advanced Viewer the same way it is opened for a CIMPLICITY project through a CimEdit or CimView screen.

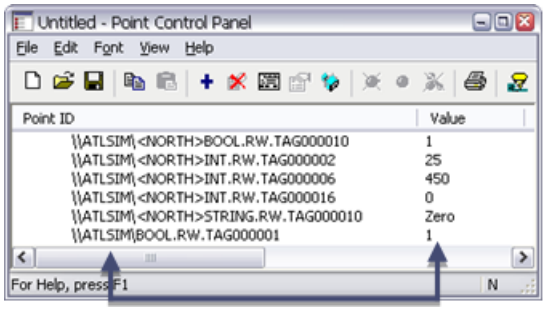
1. Right-click the CimEdit or CimView screen.
2. Select Point Control Panel on the Popup menu.

   Result: The Point Control Panel opens.

   • The router is running:

   Item ID's being used by the selected object are listed in the Point Control Panel.

   The tag values display.

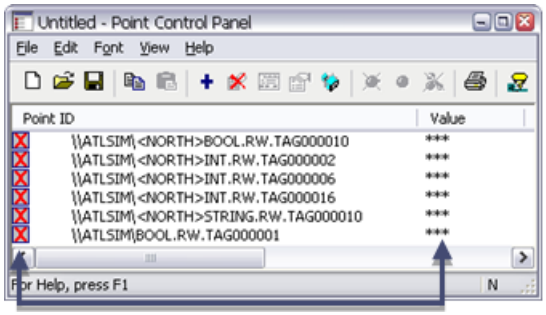


   Note: The Advanced Viewer supports array point set points in the Point Control Panel; they are not supported in CimView.

   • The router is not running.

   Item ID's associated with the selected object are listed in the Point Control Panel.

   The tag values do not display.



   Point Control Panel File

   You can open a Point Control Panel file (.ppl) to view Advanced Viewer points.

   The .ppl file:

- Defines the Point Control Panel
- Can optionally include specified point ID's.

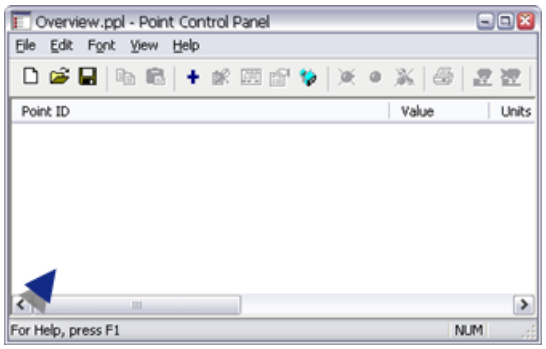
Note: You can enter the point ID's manually or save a Point Control file that displays the points you want.

Double-click a Point Control Panel file (.ppl) when the Router is running.

Result: Point ID's that are listed in the .ppl file display when the Point Control Panel opens.

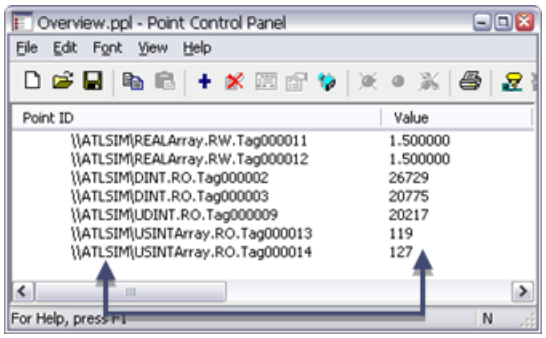
- No point ID's are listed.

The Point Control Panel is initially blank.



- Point ID's are listed in a Point ID section.

The Point Control Panel initially displays the points that are listed in the .ppl file.

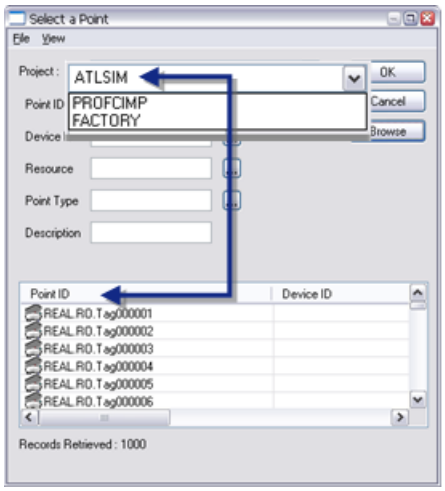


List a Point Browser

3. Click the Open Browser button on the Point Control Panel toolbar.

   The List a Point browser opens.

   Note: If the OPC Server does not display in the Project field, you can select it in the drop down list.



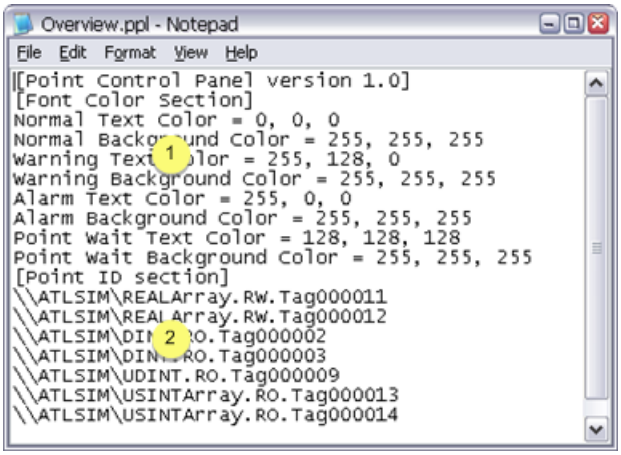
4. Select the point ID's that you need.

   Note: The Select a Point browser operates for the Advanced Viewer the same as it does for point ID's in any CIMPLICITY project.

The selected points display in the Point Control Panel; you have the option to save the .ppl file for future use.

Point Control Panel File Sections

The .ppl file sections are as follows.



1	Basic Point Control Panel display definition.
2	Point ID section. The entry syntax is: \\<Project Name>\<item ID> Where <Project Name> is a project name entered in the ptopc_config.xml file. <item ID> is an item ID accepted by your OPC Server.

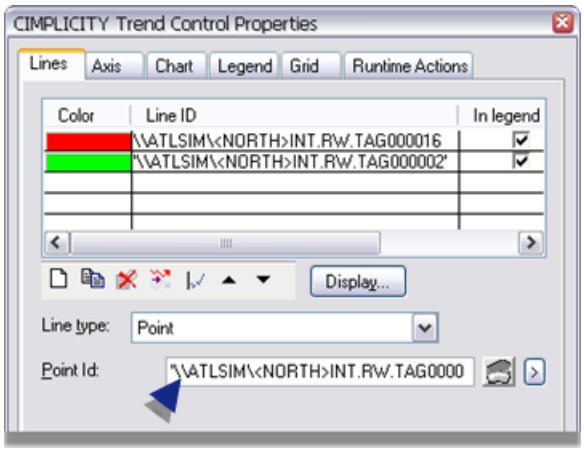
Option 2.4. Create a Quick Trends Chart

1. Right-click a point value in the Point Control Panel or on a CimView screen.
2. Select Quick Trends on the Popup menu.

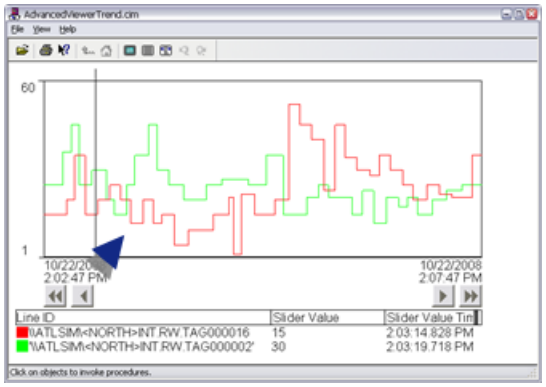
   Results.

3. A Quick Trend window opens.
4. Values for the selected point are trended.
5. Additional points can be added.

   You must enter the exact item ID. The points are not available through the Point browser since they are not in a project.



6. The added points are now also trended on the Quick Trend chart.



Step 3. Set up each Viewer

Distribute the Advanced Viewer files

When your user ptopc_config.xml files and user interface files are ready, distribute them to each Viewer that will use them.

- Make sure that the OPC Server is installed and registered *(page 258)* correctly for the node entered in the ptopc_config.xml file.
- Replace the empty ptopc_config.xml file with the version configured for that Viewer in the viewer's ..\Proficy CIMPLICITY\data directory.
- Copy the Advanced Viewer CimEdit/CimView files (.cim) and Point Control Panel files (.ppl) to the Viewer.

Start the Advanced Viewer on the Viewer

Start *(page 345)* the Advanced Viewer on the viewer or viewers on which it should be running.

Result: The Advanced Viewer will collect and display the selected item ID's from the selected OPC Server.

Technical Reference

- Advanced Viewer processes.
- Point properties.
- Item/group deletion.
- Advanced Viewer limitations.

Advanced Viewer Processes

The following processes need to be running for the Advanced Viewer to work. If there is an issue you can check the Windows Task Manager to see if the following files are running.

Process	File Listed in Task Manager
CIMPLICITY Router	w32rtr.exe
OPC Communications program	ptopc.exe
OPC Server selected for the Advanced Viewer	Depends on selected OPC Server. Example ATLSimServer.exe

Point Properties

The Advanced Viewer can dynamically change the point properties that are available through the Advanced Viewer, in response to changes in the OPC Server.

Important: Dynamically changing point properties requires an OPC Server that implements the IOPCBrowse interface as part of the OPC DA 3.0 specifications.

The OPC DA 3.0 server must implement `IOPCBrowse::GetProperties` and populate the `ItemID` member of the `OPCITEMPROPERTY` structure with the Name of an Item whose value is the value of the property.

Please refer to sections **4.2.14**, **4.3.6.2** and **6.7.7** of the "OPC Data Access Custom Interface Standard Version 3.00" for more details.

The OPC Server standard descriptions and property ID's that correspond to the CIMPLICITY properties are as follows.

CIMPLICITY	OPC Server		
Property	Standard Description	ID	Comments
Description	Item Description	101	Fixed
Display Format	No concept in OPC	-	Configurable
High Display Limit	High EU	102	Configurable
Low Display Limit	Low EU	103	Configurable
EU Label	EU Units	100	Fixed
Enumeration Strings	EU Information	8	Fixed

The following line *(page 340)* in the PTopc_config.xml file enables this functionality.

```
<item_property_subscriptions>TRUE</item_property_subscriptions>
```

Note: `FALSE` disables this functionality.

Item/Group Deletion

Items can now be deleted from a group before the group is removed.

The following line *(page 341)* in the PTopc_config.xml file enables this functionality.

```
<RemoveItemsOnRemoveGroup>TRUE</RemoveItemsOnRemoveGroup>
```

Note: `FALSE` or no entry does not delete items before the group is removed. When set to False, items are not explicitly removed on the client side; in that case the assumption is that the server will remove them.

Advanced Viewer Limitations

The Advanced Viewer currently has the following limitations.

- Security is the responsibility of your OPC server.
- The following are not currently provided with the Advanced Viewer.
- Logging for values collected through the Advanced Viewer.
- Alarming for these items.
- You cannot use Advanced Viewer item ID tags in EMRP scripts; they can be used in CimEdit scripts.
- Array set points are not supported in CimView.
- You must manually configure the ptopc_config.xml *(page 331)* file.
- The Advanced Viewer system does not recover if the Advanced Viewer service crashes. You must stop and restart the Viewer.

Note: You can stop the Viewer in the CIMPLICITY Options dialog box or stop the Router in the Task Manager; you may have to stop your OPC Server.

Chapter 19. CIMPLICITY OPC UA Security Configuration

About the OPC UA Security Configuration Tool

A major feature of OPC UA communications is the ability to enable security for all communications, which makes the system resistant to many forms of cyber-attacks.

OPC UA security uses public key cryptography to ensure secure communication channels.

In order to allow OPC UA Clients to securely communicate with OPC UA Servers, it is necessary for both the:

1. Client application to accept the Server as a legitimate Server
2. Server to accept the Client as a legitimate Client.

   The OPC UA Security Configuration Tool enables you quickly set up a secure communication between CIMPLICITY and OPC UA Clients and Servers.

   CIMPLICITY acts as an OPC UA Client when an OPC UA client device is added to a CIMPLCITY project.

   CIMPLICITY acts as an OPC UA Server when the OPC UA server is enabled in the Project Properties.

   For secured communication, the OPC UA security configuration is done on both the OPC UA Server and Client in a CIMPLICITY project.

   The OPC UA security configuration is specific to a computer. If CIMPLICITY installed on another computer, then security must be re-configured even if the project directory is copied.

   Important: The CIMPLICITY server name can be no more than 15 characters long to use secured OPC UA communications.

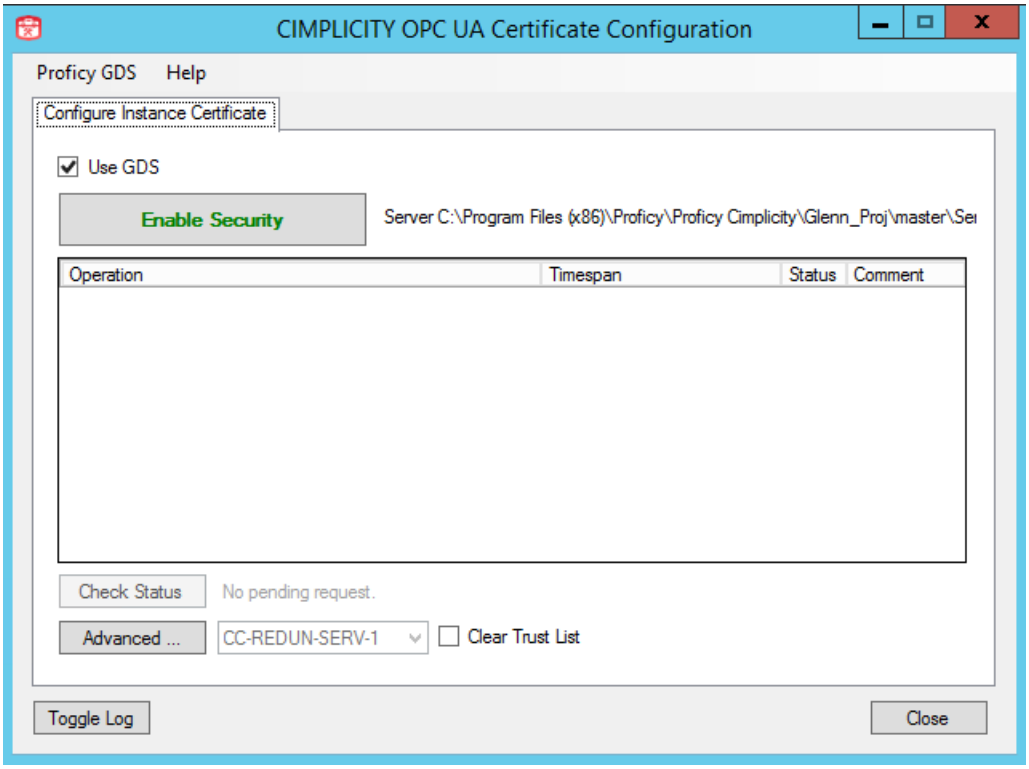
1	Start Certificate Configuration
2 *(page 369)*	Configure a Self-Signed Certificate
3 *(page 371)*	Configure the Global Discovery Server

4 (page 372)	Configure a GDS-signed Certificate

OPC UA Server Certificate Configuration

Configure a Self-Signed Certificate

After starting the OPC UA Security Configuration from the CIMPLICITY Workbench, the CIMPLICITY OPC UA Certificate Configuration dialog box appears.



The elements in this dialog box are described in the following table:

Field	Description
Use GDS	This check box is selected by default. For a self-signed certificate, clear this check box. For a GDS Certificate, leave this check box selected.
Enable Security	This button sets up secure OPC UA communications using certificates. The CIMPLICITY serverconfig.xml file location is displayed next to this button.
Display area	Shows the progress and displays the results when the Enable Security button is clicked.
Check Status	When using GDS, click to check the status of pending requests.

Field	Description
Advanced	This button opens another screen to view and select configuration options.
Redundancy Selection Box	Used to specify whether you are configuring the certification for the primary or secondary node. This option is available only for projects with Server Redundancy enabled.
Clear Trust List	This check box allows you to clear the current Trust List. If unchecked, the GDS-assigned Trust List is added to the existing Trust List.
Toggle Log	Used to toggle the Configuration Log window on and off.
Close	Closes this page.

Self-signed Certificate Configuration

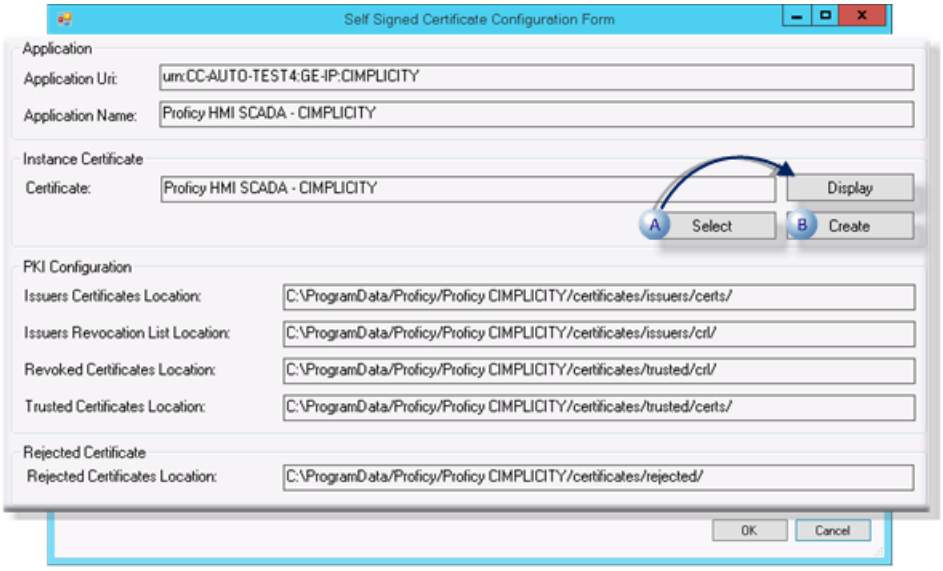
1	Clear the Use GDS check box.
2	Click Enable Security.

Result: A self-signed certificate that can be used by the CIMPLICITY OPC UA Client and Server is created. By default, the duration of the certificate is five years.

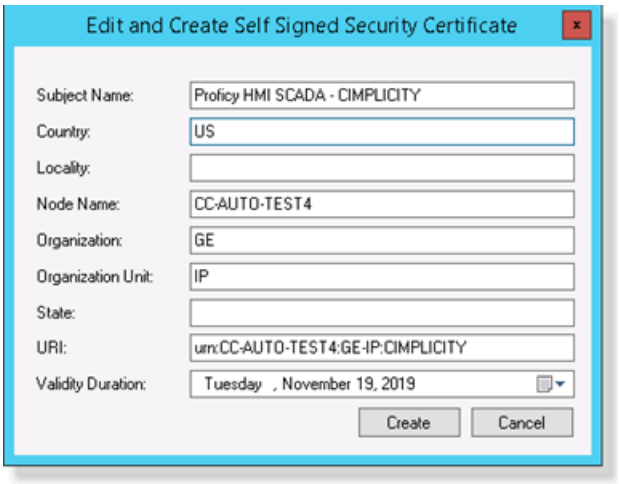
Advanced Configuration

Click Advanced to see more detailed information about the certificate settings and to directly select or create specific certificate instances.

A Self Signed Certificate Configuration Form opens enabling you to do the following.



A	Select>Display an existing certificate.
B	Create a new certificate.



Configure the Global Discovery Server

You can create a signed certificate on the Global Discovery Server to communicate more securely between CIMPLICITY and other OPC UA applications. These signed certificates can communicate with any client or server that also has its certificate signed by the GDS.

Important: Interactions with the Global Discovery Server require certain privileges. Please check the GDS Configuration Panel>Server Configuration tab to find the GDS user group settings for GDS privileges.

1. Do the following in the CIMPLICITY OPC UA Certification Configuration dialog box to begin a GDS Server quick start configuration.

A	Check Use GDS.
B	Click **GDS > Configure GDS** on the CIMPLICITY OPC UA Certification Configuration menu bar.

   The security tool checks:

Does the Agent have Connection information.	Verifies if there are already credentials.
Get Configured Endpoint.	Opens a connection to the GDS.

   A Global Discovery Server Settings dialog box opens.

2. Enter valid credentials in the Global Discovery Server Settings dialog box. If the parameters have been saved there is no need to login again.

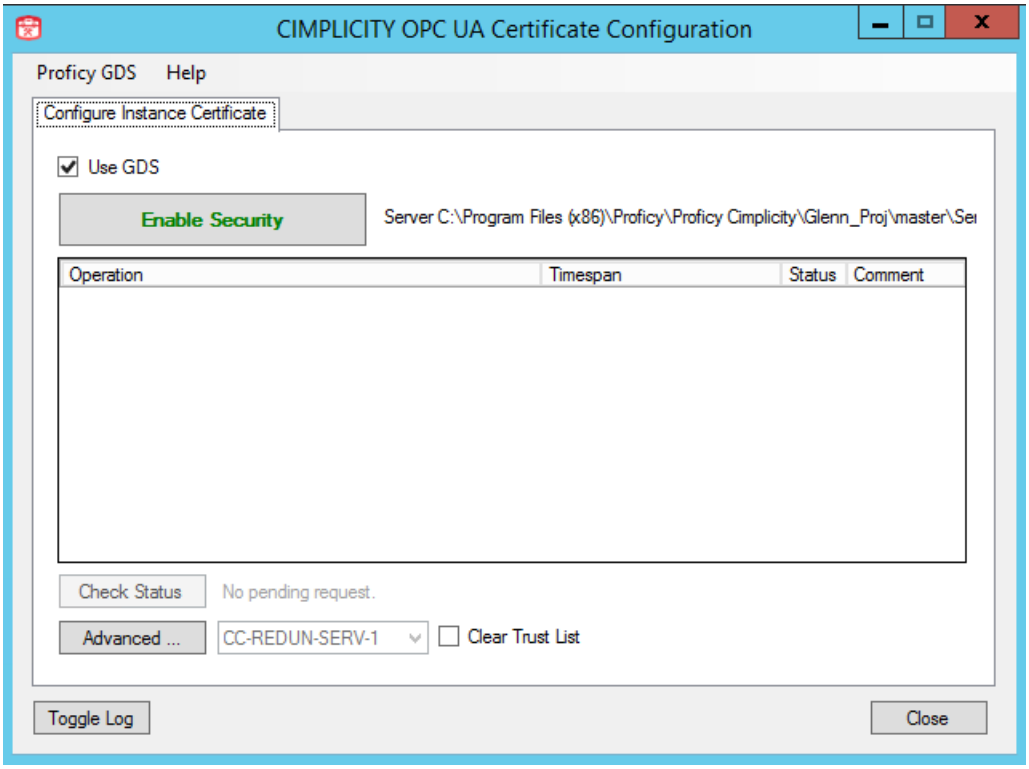
GDS Endpoint	The format is: `opc.tcp://` `MachineNameOrIPAddress` `:59410`
User Name/Password	Valid administrator user name and password for the GDS.

3. Click OK.

Configure a GDS-signed Certificate

OPC UA Certificate Configuration dialog box

After starting the OPC UA Security Configuration from the CIMPLICITY Workbench, the CIMPLICITY OPC UA Certificate Configuration dialog box appears.



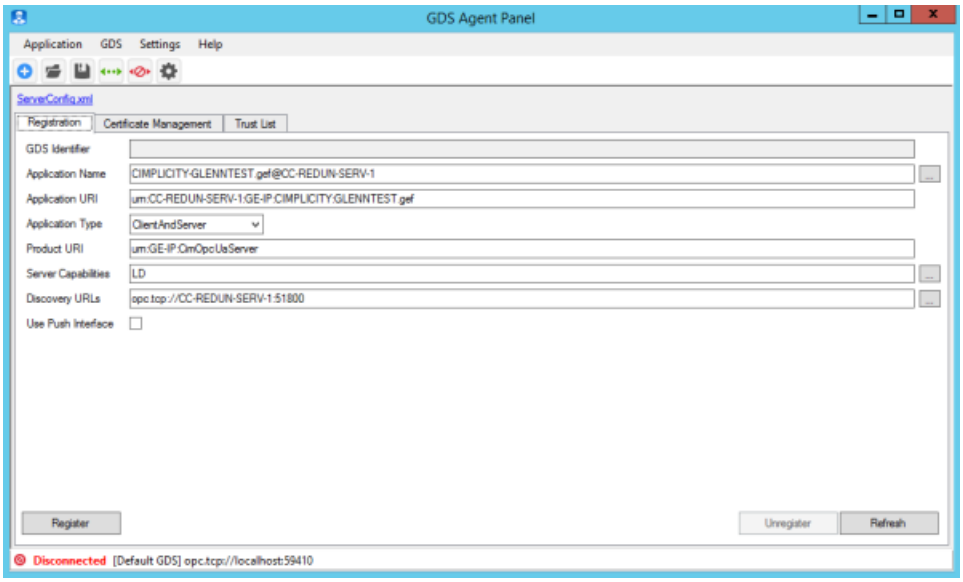
The features are described in the following table:

Field	Description
Use GDS	This check box is selected by default. If you want to create a GDS-signed certificate, leave this check box selected.
Enable Security	This button sets up secure OPC UA communications using certificates. **Note:** Enable security only when the CIMPLICITY project is not running, because the certificates cannot be updated when the project is running.
Display area	Shows the progress and displays the results when the Enable Security button is clicked.
Check Status	When using GDS, click to check the status of pending requests.

Field	Description
Advanced	This button opens another screen to view and select configuration options.
Server Host Name Box	Used to select the host name that is being configured. This option is available only for projects with Server Redundancy enabled.
Clear Trust List	This check box allows you to clear the current Trust List. If unchecked, the GDS-assigned Trust List is added to the existing Trust List.
Toggle Log	Used to toggle the Configuration Log window on and off.
Close	Closes this page.

Advanced Configuration

If you need to view or modify registration, certificate, or trust list information, click Advanced to open the GDS Agent Panel.



The button bar along the top of the page is used to create an application with default values, load an application from a file, save the current application file, connect to the GDS, disconnect from the GDS, and edit the settings for the agent.

The Registration tab is used to view and select information that allows you to register the current application with the current GDS.

The Certificate Management tab is used to view and select information that allows you to create a certificate for the current application.

The Trust List tab displays information for the locally cached Trust List and the GDS assigned Trust List.

Remotely Manage CIMPLICITY OPC UA Server Certificate

The OPC UA Server in CIMPLICITY supports remote OPC UA security configuration. The following steps are to use a GDS agent to push a GDS signed certificate to a CIMPLICITY OPC UA Server and update the OPC UA server trust list with GDS trust list.

1. In the Project Properties dialog box, enable the OPC UA Server component.
2. Ensure at least one CIMPLICITY user has the OPC UA server admin permissions in its role.
3. Start the CIMPLICITY project.
4. Launch GDS Agent, and create a new application by selecting the CIMPLICITY server.
5. Register the application.
6. Check the ‘Use Push Interface’ button.
7. Switch to the Certificate Management tab.
8. Click the Sign Certificate button.
9. Enter the username/password of a CIMPLICITY user with OPC UA server admin permissions.
10. If the GDS agent isn’t trusted by the OPC UA server, there will be a message saying the push operation failed due to the OPC UA server does not trust the GDS agent. To solve this, the GDS agent certificate needs to be moved from <CIMPLICITY project folder>/pki/rejected to <CIMPLICITY project folder>/pki/trusted/certs to make the OPC UA server to trust the GDS agent.
11. Click the Push certificate button to push signed publish key to OPC UA server.

    Note: When the push certificate operation is completed, the OPC UA server will be restarted remotely, which may cause the communication to be disrupted in a short period of time.

12. Switch to the Trust List tab.
13. Click the Replace with GDS button.
14. Click the Push Trust List button.

    The CIMPLICITY OPC UA server should now have a CA issued certificate and allow secure connections from any other application with a certificated issued from the same CA.

Create GDS-signed Certificate

With the Use GDS check box selected, click Enable Security on the CIMPLICITY OPC UA Certification Configuration dialog box.

Result: The OPC UA Security tool performs the necessary steps for interacting with GDS.

The OPC UA Security tool:

1. Registers CIMPLICITY with the GDS.
2. Creates a self-signed certificate for CIMPLICITY.
3. Requests that the GDS sign the certificate.

4. Replaces or updates the existing trust list.

Success: CIMPLICITY can now talk to any other OPC UA applications that have signed certificates and are trusted by GDS.

Issues: There could be many reasons for not succeeding. If so, do the following:

1. Click either of the following to view a Log file for data about the operation.
   - Toggle Log button.
   - Log hyper link after each action.

2. If you login with Super User credentials vs. Administrator credentials, then the certificate request will require an Administrator to approve the request on the GDS server. In this case, all the steps are not completed until the request is approved. Click Check Status to see if the request has been accepted or rejected.

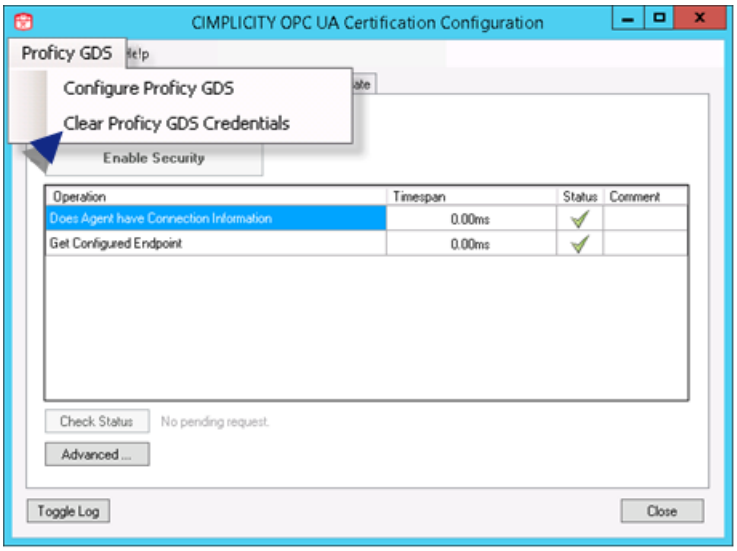
Note: Clicking the Advanced button will open the GDS Agent Panel, where you can find detailed help about the Global Discovery Server and Global Discovery Agent.

Clear GDS Credentials

For security purposes, you can clear out GDS credentials so they will not be saved on the CIMPLICITY disk.

Select GDS>Clear GDS Credentials on the CIMPLICITY OPC UA Certification Configuration dialog box menu bar.

Result: The GDS credentials will be cleared from the CIMPLICITY machine; you will need to provide credentials the next time the CIMPLICITY connects to the Global Discovery Server.


CIMPLICITY OPC UA Certification Configuration
Proficy GDS
Help
Configure Proficy GDS
Clear Proficy GDS Credentials
Enable Security
Operation
Timespan
Status
Comment
Does Agent have Connection Information
0.00ms
Get Configured Endpoint
0.00ms
Check Status
No pending request.
Advanced ...
Toggle Log
Close

Chapter 20. CIMPLICITY OPC UA Server

Overview

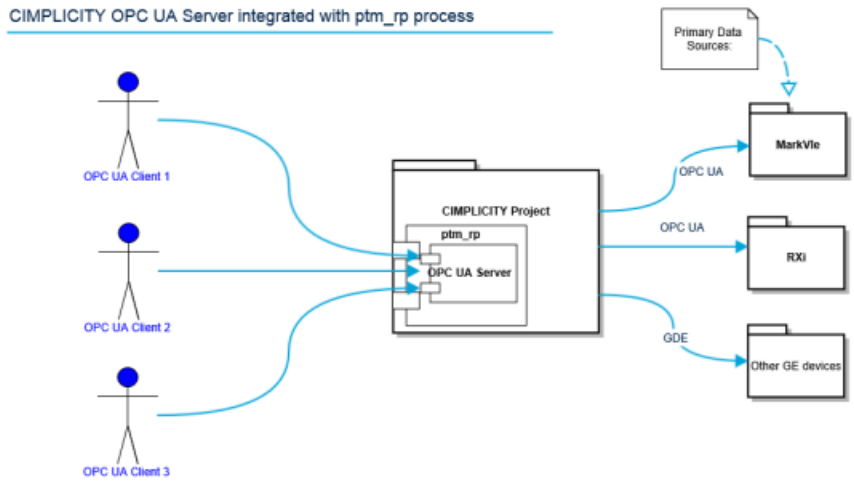
The OPC UA Server implements OPC UA, which is a secure, scalable multi-platform (including portable ANSi C, Java and .NET) communication protocol that you can use for information modeling of your data and events.

The OPC UA Server offers the following features:

- Redundancy support
- Heartbeat for connections in both directions (to indicate whether the other end is "alive"). This means that both server and client recognize interrupts.
- Data buffering and transmitted data acknowledgements so that data isn't permanently lost just because the connection is lost. Lost datagrams can be re-fetched.

The CIMPLICITY OPC UA Server is embedded into the CIMPLICITY core and is compliant with OPC UA v1.03. The OPC UA Server allows OPC UA clients to read, write, and subscribe to changes in CIMPLICITY runtime database points.

The following illustration shows a high level architecture of the OPC UA Server, integrated with point management process (PTMRP). PTMRP provides efficient access to real-time data, avoiding the additional overhead of accessing it from another process.



Getting Started

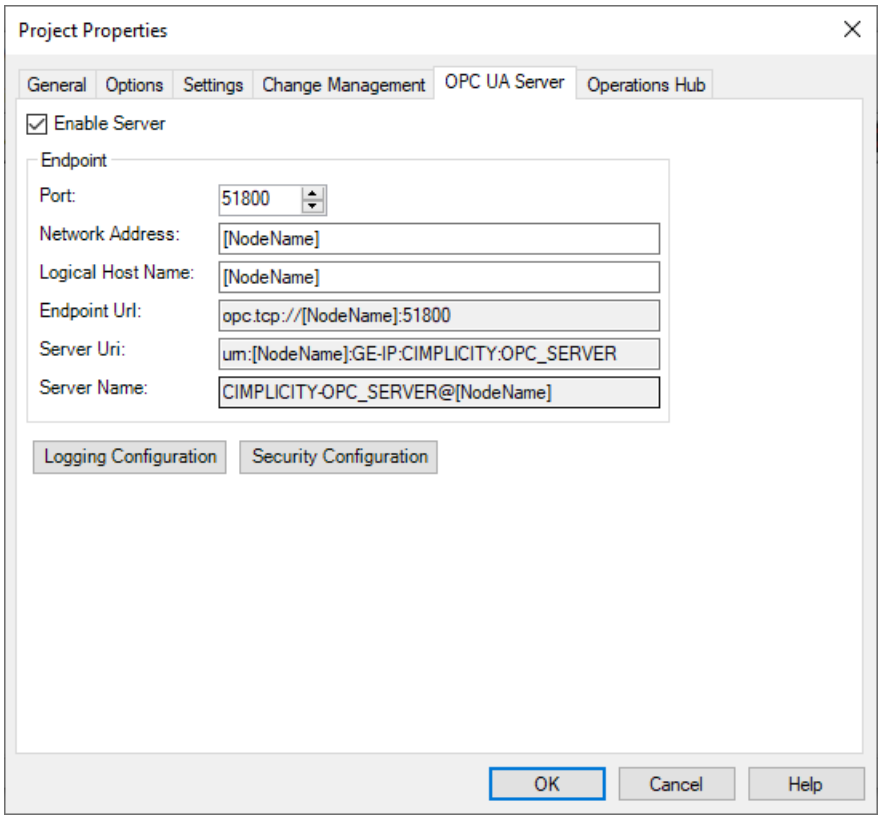
Configuration User Interface

1. Do one of the following.
   - Click Project>Properties on the Workbench menu bar.
   - Click the **Project Properties** button on the Workbench toolbar.

   The Project Properties dialog box opens.

2. Select the **OPC UA Server** tab.

3. Select or clear the Enable Server check box. Be aware that enabling the OPC UA server uses more system resources and has impact on CIMPLICITY performance. This option is disabled by default.



UA Endpoints Configuration

4. Under Endpoint, enter or confirm your information in the Port, Network Address, and Logical Host Name fields.

5. Once the data is entered, you can view how these fields affect the data in the Endpoint URL, Server URI, and Server Name fields.

   The port field is configurable from 1025-49151 and is reflected in the Endpoint URL as the value changes.

   The Network Address accepts the machine name, an IPv4 address, or an IPv6 address. If you enter "[NodeName]," then the field will be replaced with the machine name at server runtime. This field only affects the corresponding place holder in the Endpoint URL.

   The Logical Host Name must be of valid DNS hostname syntax but doesn't need to be a machine that is actually online. This field affects the corresponding place holders in the Server URI and Server Name.

   If any of the fields have incorrect values, a message box will appear and explain the syntax error.

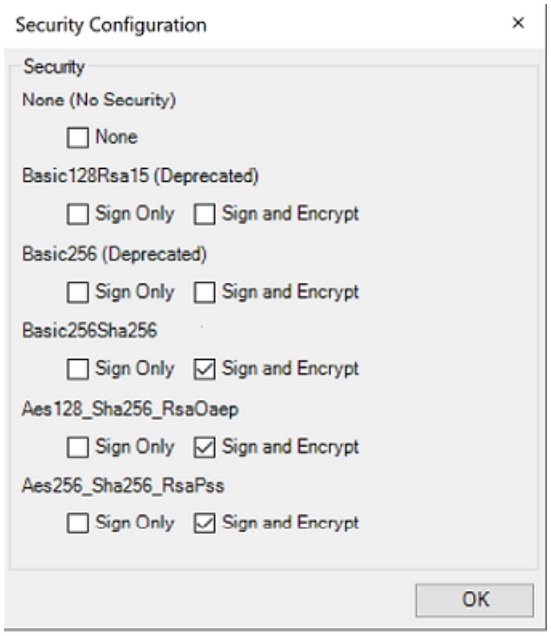
   Any subsequent changes made to the project name will be reflected in the three fields without manual modification.

6. Click **OK** to save the data to the ServerConfig.xml file.

   Security Configuration

   Select one or more security configurations from which you can choose when you configure your endpoints.

7. From the **OPC UA Server tab** of the **Project Properties** dialog box, click **Security Configuration** to open the Security Configuration dialog box.

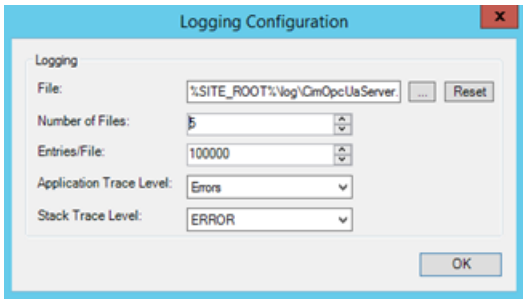


Note: Basic128Rsa15 and Basic256 are deprecated due to security and theoretical issues.

8. Check the boxes for the security policies and modes you want the server to support. The default selections are , Basic256Sha256 – Sign and Encrypt, Aes128_Sha256_RsaOaep – Sign and Encrypt, Aes256_Sha256_RsaPss – Sign and Encrypt. If you do not check any of the boxes, a message box will appear indicating that at least one of the boxes must be checked.

9. Click **OK** to save the data to the ServerConfig.xml file.

   Logging Configuration



   The logging UI enables you to modify the following nodes in ServerConfig.xml:

   - UaAppTraceMaxEntries: The number of lines per log file (range is from 1-500000)
   - UaAppTraceMaxBackup: The number of files for log backups (range is from 1-255)
   - UaAppTraceFile: Location of the log file to be used
   - UaStackTraceLevel: Possible values NONE, ERROR, WARNING, SYSTEM, INFO, DEBUG, CONTENT, ALL

- UaAppTraceLevel: Possible values NoTrace, Errors, Warning, Info, InterfaceCall, CtorDtor, ProgramFlow, Data

Note the following:

- You can enter a path to a log file manually. In this case, the file will be generated by the OPC UA SDK automatically, given the proper permissions, and the path to the file will be created. If the file cannot be created for whatever reason, no log file will be used during runtime. You can also enter "%SITE_ROOT%\log" as a directory prior to the file if you would like the log file to be placed in the project's "log" directory.
- The Reset button sets the path of the log file to the last saved path.
- The trace levels (log levels) are ranked in the list by ascending log level; in other words, ERROR will create fewer log entries than ALL for the Stack Trace Level.
- These changes are made visible in ServerConfig.xml when you click OK on the Project Properties dialog box (not when you click **OK** on the current dialog box).

Refer to the Troubleshooting section for more information about Trace Levels.

User Authentication

Mapping of CIMPLICITY user to UA Session

You can use the user name/password authentication setting when connecting to the CIMPOPCUAServer.

If you are enabled in CIMPLICITY and the password supplied is correct, you are logged in.

User Credentials	Value
User	KYLE
Password	MYPASSWORD
Role	CUSTOMROLE
Resources	$MAC_FR, $PTM_FR, $SYSTEM
Level	17
Enabled	Y

UA Client passed credentials

User: Kyle

Password: myPassword

UA Session Points

UaNodeId	Value
$ROLE.Value	CUSTOMROLE
$ROLE.LEVEL.Value	17
$USER.Value	KYLE

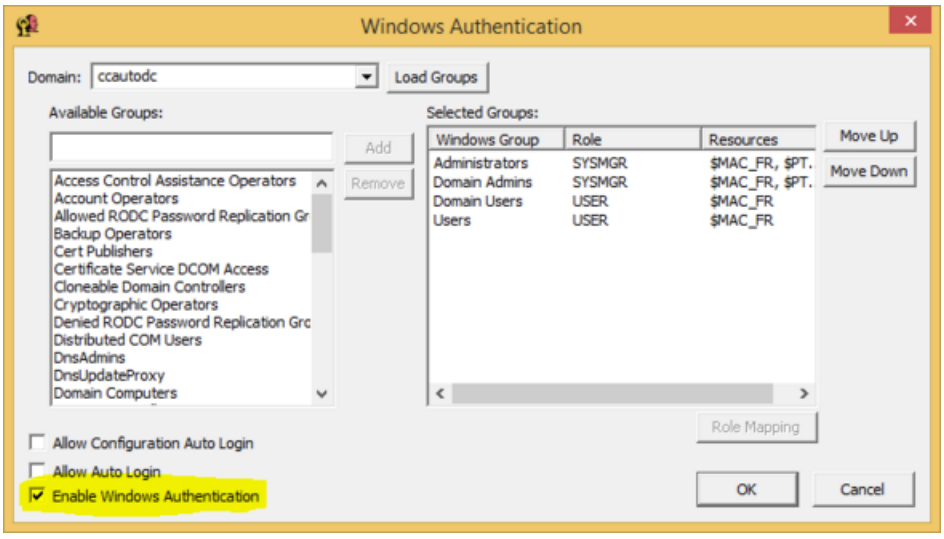
These values are be specific to the current session and will not be visible to other sessions.

The user will not be connected if any of the following apply:

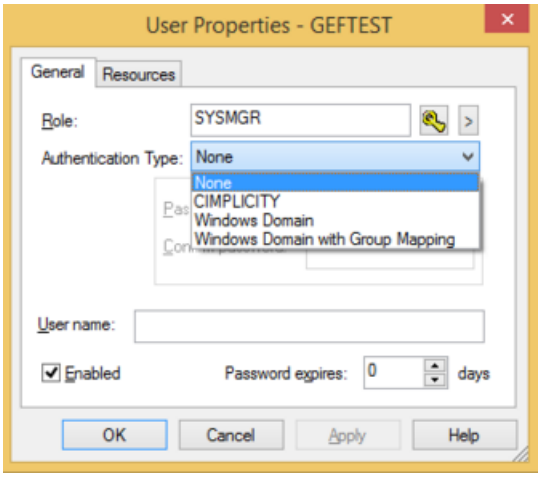
- User is disabled
- Password is incorrect
- User does not exist in database

Windows Authentication

The security section of the CIMPLICITY Workbench allows an administrator to set up Windows Authentication under the domain settings. This authentication setting is enabled via the "Enable Windows Authentication" check box as shown below.



With this authentication enabled the administrator has access to the Authentication types "Windows Domain" and "Windows Domain with Group Mapping" as shown below.



If "Windows Domain" is chosen, then the password and user name supplied to the server from an OPC UA client must match those on the domain being used by the CIMPLICITY project.

The same is true if "Windows Domain with Group Mapping" is chosen. In this case, the role and resources of the user will be updated to match those delegated by the Windows authentication panel. In the example above, a user belonging to the Domain Users Windows group will inherit a role of USER and the resource $MAC_FR. If a user is in multiple groups, the group with the highest priority will be used to determine the role and resources. If the user is not in any of the groups listed, then the user will be rejected.

In the case that the CIMPLICITY user does not exist, the server will attempt to use Windows authentication with the credentials supplied from the OPC UA Client. In this case, if the user is found in the groups specified above, a CIMPLICITY user will be created with the authentication type "Windows Domain with Group Mapping" and inherit the role and resources of that group.

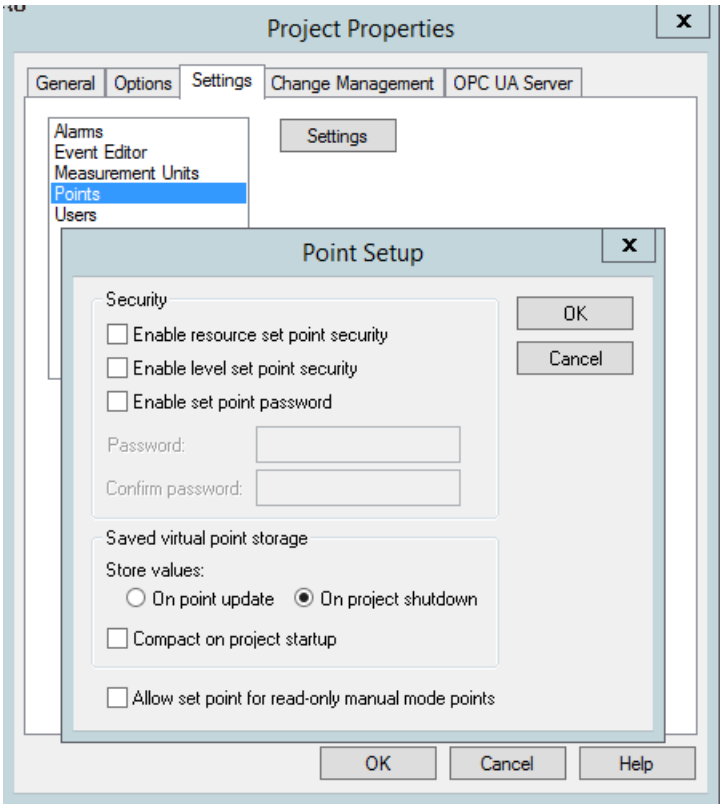
The user name provided from the client must be in the form of username@domain or domain \username, where the domain is a case insensitive match to the domain chosen in the project's Windows Authentication page.

Level Set Point Security

You can enable level set point security as follows:

1. From the Settings tab on the Project Properties dialog box, highlight Points, and click Settings.

The Point Setup dialog box appears.



1. Select the "Enable level set point security" check box.

If this is set, the logged-in user has only write access to those points with a level less than or equal to the level of the user. The level of the user is determined by the role that is assigned in the security section of the CIMPLICITY project.

User	User's Level	Point	Point's Level	Writable
KYLE	17	VALVE1.Value	16	Yes
KYLE	17	VALVE1.Value	17	Yes
KYLE	17	VALVE1.Value	18	No

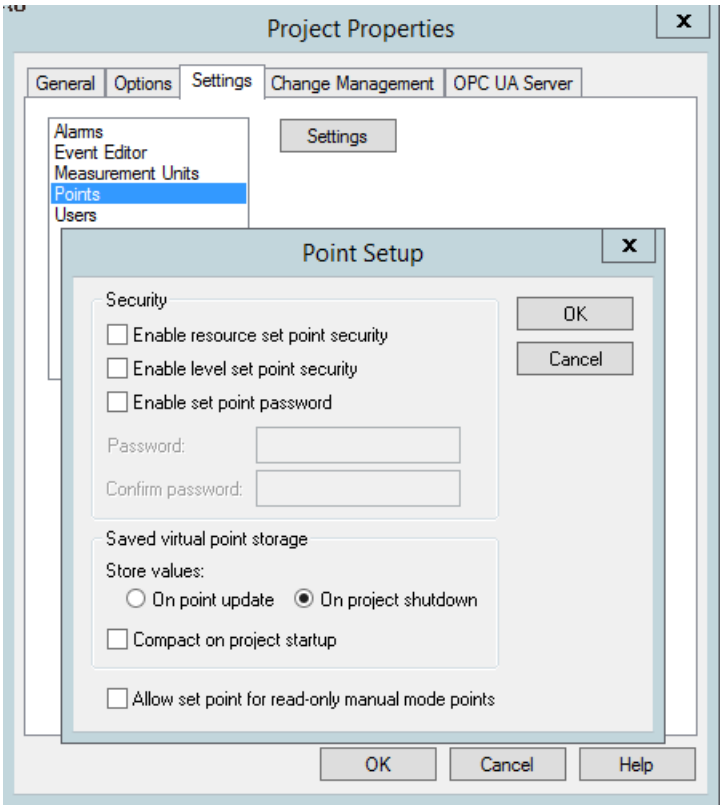
If the "Enable level set point security" option is not selected, then all users will retain equal write access (neglecting other security options such as resource set point security).

Resource Set Point Security

You can enable level set point security as follows:

1. From the Settings tab on the Project Properties dialog box, highlight Points, and click Settings.

The Point Setup dialog box appears.



1. Select the "Enable resource set point security" check box.

If this option is checked, users will only have write access on points with a resource that is also in their view. A point's resource is determined in its properties. Resources in a user's view can be configured from the user's properties or from the resource's properties.

User	User's Resources	Point	Point's FR_ID	Writable
KYLE	$SYSTEM, PUMPS	PUMP1.Value	PUMPS	Yes
BOB	$SYSTEM, VALVES	PUMP1.Value	PUMPS	No

If the "Enable resource set point security" option is not selected, then all users will retain equal write access (neglecting other security options such as level set point security).

Browsing and Availability of Address Space

Browsing

Namespace URIs

0 - standard namespace URI: http://opcfoundation.org/UA/ (This cannot be changed)

1 -the same as serverURI: urn:[NodeName]:GE-IP:CIMPLICITY:[ProjectName]

2 - http://ge.com/UA/CIMPLICITY - common CIMPLICITY specific type definitions

3 - http://ge.com/UA/CIMPLICITY/[ProjectName] – CIMPLICITY Objects and Points

4 - http://ge.com/UA/CIMPLICITY/[ProjectName]/project – CIMPLICITY Classes

Examples of NodeIDs:

ns=3;s=Compressor1.Pump1.Temperature;

Create Instance Nodes at startup (for points and objects)

UA nodes for all CIMPLICITY points and objects are created at CIMPLICITY project startup.

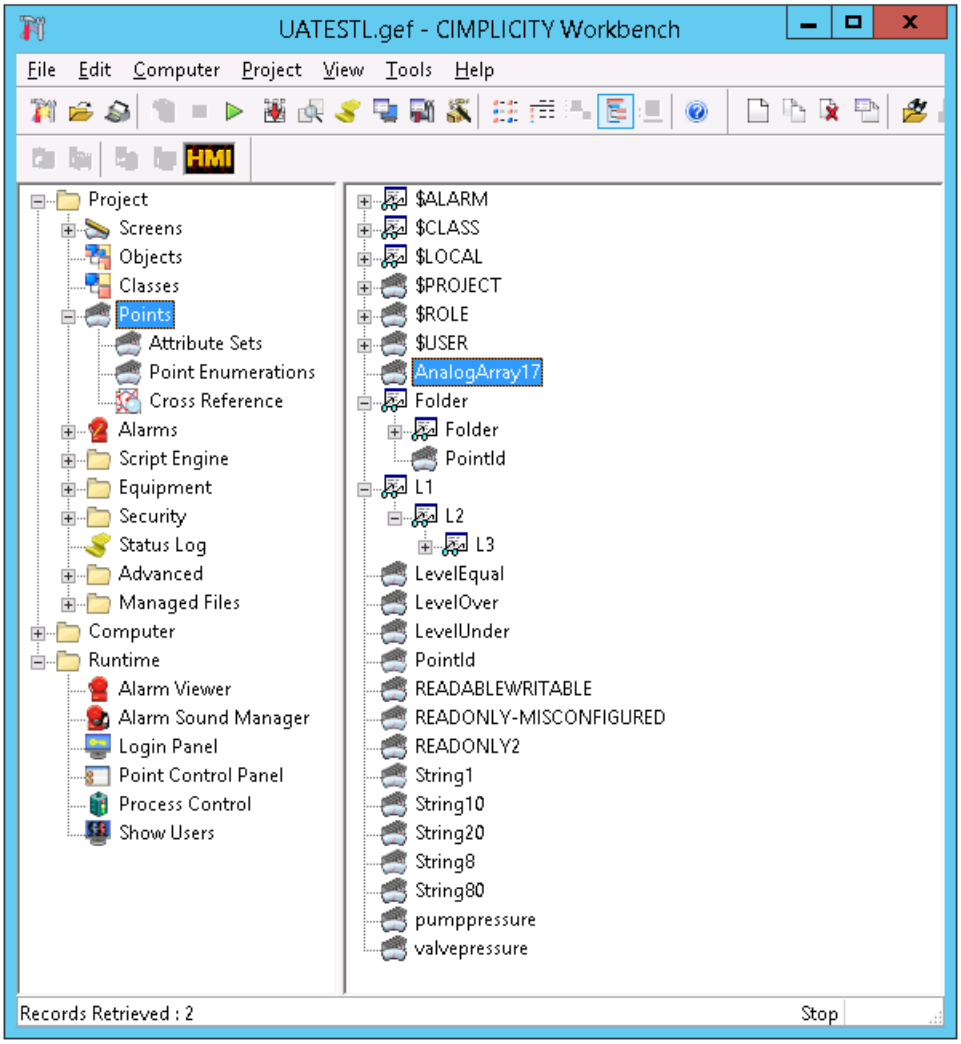
The default CIMPLICITY OPC UA Server endpoint is opc.tcp//<cimplicity SERVER Machine Name>:51800, i.e. opc.tcp://CC-AUTO-TEST10:51800, where CC-AUTO-TEST10 is the machine name running the CIMPLICITY Project.

By default, the CIMPLICITY OPC UA Server is disabled at CIMPLICITY Project startup. However, the OPC UA server can be enabled in project settings.

Availability of Address Space

CIMPLICITY OPC UA Server Address Space

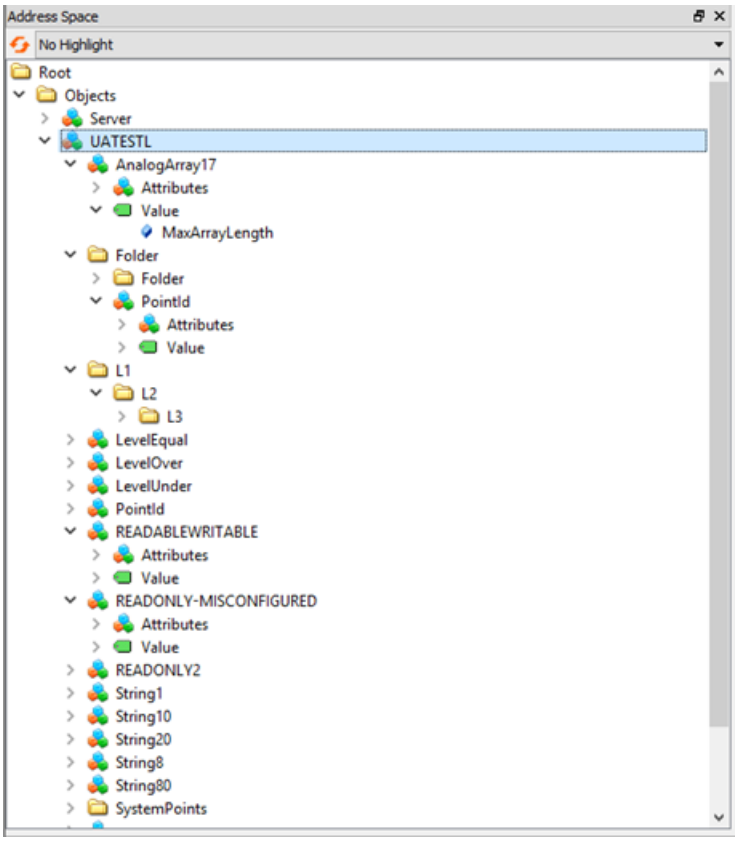
Point IDs are defined using a delimiter, which allows hierarchical representation of the address space in the CIMPLICITY workbench. The same structure should be created in the OPC UA Server.



A folder representing the CIMPLICITY Project root is created under the Objects folder. This helps easily distinguish CIMPLICITY project's object and variable instances from other generic OPC UA Server objects and variables.

The data type of that folder is a new defined type - CIMProjectType.

The project root folder has all points under it, including statistical points. The statistical points will be organized under their sub-folder.



NamespaceURIs

0 - standard namespace URI: http://opcfoundation.org/UA/ (This cannot be changed)

1 - the same as serverURI: urn:[NodeName]:GE-IP:CIMPLICITY:[ProjectName]UaServer

2 - http://ge.com/UA/CIMPLICITY - common CIMPLICITY specific type definitions

3 - http://ge.com/UA/CIMPLICITY/[ProjectName] – CIMPLICITY Objects and Points

4 - http://ge.com/UA/CIMPLICITY/[ProjectName]/project – CIMPLICITY Classes

Examples of NodeIDs:

ns=3;s=Compressor1.Pump1.Temperature;

Communication

System Points

The following CIMPLICITY system points are exposed by CIMPLICITY OPC UA Server.

UA NodeID

$ALARM.ACKED.Value

$ALARM.ACTIVE.Value

$ALARM.TOTAL.Value

$ALARM.UNACKED.Value

$CLASS_HIGH.ALARMS.Value

$CLASS_HIGH.UNACKED.Value

$CLASS_HIGH.UNRESET.Value

$CLASS_LOW.ALARMS.Value

$CLASS_LOW.UNACKED.Value

$CLASS_LOW.UNRESET.Value

$CLASS_MED.ALARMS.Value

$CLASS_MED.UNACKED.Value

$CLASS_MED.UNRESET.Value

$CLASS_SYS.ALARMS.Value

$CLASS_SYS.UNACKED.Value

$CLASS_SYS.UNRESET.Value

$PROJECT.COMPUTER.Value

$PROJECT.DATE.AMPM.Value

$PROJECT.DATE.DAY.Value

$PROJECT.DATE.DAYOFWEEK.Value

$PROJECT.DATE.DAYOFTEAR.Value

$PROJECT.DATE.HOUR.Value

$PROJECT.DATE.HOUR12.Value

$PROJECT.DATE.MINUTE.Value

$PROJECT.DATE.MONTH.Value

$PROJECT.DATE.SECOND.Value

$PROJECT.DATE.SECONDOFDAY.Value

$PROJECT.DATE.WEEK.Value

$PROJECT.DATE.YEAR.Value

$PROJECT.DATETIME.Value

$PROJECT.DEVICES.Value

$PROJECT.USERS.Value

$PROJECT.Value

$ROLE.Level.Value

$ROLE.Value

$USER.Value

Read Point Attributes

All Points in the CIMPLICITY project are mapped to UA Objects. The Value of the point can be read or subscribed to by selecting the "Value" component of the Point object. The NodeId of the Point object is the CIMPLICITY Point unique identifier. The NodeId of the Value component is constructed by appending the text ".Value" to the Point ID.

The CIMPLICITY Attributes associated with a Point are properties of the 'Attributes' component. Only a subset of CIMPLICITY Attributes are exposed via OPC UA. The subset was chosen based on the CIMPLICITY Attributes currently exposed by the OPC UA Server.

The following table defines the mapping for all CIMPLICITY Attributes:

CIMPLICITY Attribute	Mapping	Attribute Name	Access	Rule
DESCRIPTION	UA Description Attribute	-	Static	Always Present
DEVICE_ID	CIM Attribute	DEVICE_ID	Static	Device Points Only
ADDR	CIM Attribute	ADDR	Static	Device Points Only
ADDR_OFFSET	CIM Attribute	ADDR_OFFSET	Static	Device Points Only

ANALOG_DEADBAND	CIM Attribute	ANALOG_DEADBAND	Static	Device Points Only and If Specified
FR_ID	CIM Attribute	FR_ID	Static	Always Present
PT_ENABLE_POINT	CIM Attribute	POINT_STATE	Static	Always Present
PT_ACCESS_FLAG	UA AccessLevel Attribute	-	Static	Always Present
SETPT_CHECK_PTID	CIM Attribute	SETCHECK_PTID	Static	If Specified
DISPLAY_LIM_HIGH_N	UA EURange Property	-	Static	Realtime
DISPLAY_LIM_LOW_N	UA EURange Property	-	Static	Realtime
RANGE_HIGH_N	UA InstrumentRange Property	-	Static	Realtime
RANGE_LOW_N	UA InstrumentRange Property	-	Static	Realtime
EU_LABEL	UA EngineeringUnits Property	-	Static	If Specified
DEVIATION_PTID	CIM Attribute	DEVIATION_PTID	Static	If Specified
ALARM_HIGH_N	UA HighHighLimit Property	ALARM_HIGH_N	Static	If Specified
WARNING_HIGH_N	UA HighLimit Property	WARNING_HIGH_N	Static	If Specified
ALARM_LOW_N	UA LowLowLimit Property	ALARM_LOW_N	Static	If Specified
WARNING_LOW_N	UA LowLimit Property	WARNING_LOW_N	Static	If Specified
DEADBAND_N	CIM Attribute	DEADBAND	Static	If Specified
SETPOINT_LOW_N	CIM Attribute	SETPOINT_LOW_N	Static	If Specified
SETPOINT_HIGH_N	CIM Attribute	SETPOINT_HIGH_N	Static	If Specified
MEASUREMENT_UNIT_ID	N/A	-	Static	If Specified
QUALITY	CIM Attribute	QUALITY	Realtime	Always Present
USER_FLAGS	CIM Attribute	USER_FLAGS	Realtime	Always Present
QUALITY.MANUAL_MODE	CIM Attribute	MANUAL_MODE	Realtime	Always Present
QUALITY.IS_IN_RANGE	CIM Attribute	IS_IN_RANGE	Realtime	Always Present
QUALITY.IS_AVAILABLE	CIM Attribute	IS_AVAILABLE	Realtime	Always Present
QUALITY.LAST_UPD_MAN	CIM Attribute	LAST_UPD_MAN	Realtime	Always Present

QUALITY.STALE_DATA	CIM Attribute	STALE_DATA	Realtime	Always Present
QUALITY.ALARMS_ENABLE	CIM Attribute	ALARMS_ENABLED	Realtime	Always Present
QUALITY.DISABLE_WRITE	CIM Attribute	DISABLE_WRITE	Realtime	Always Present
QUALITY.ALARMED	CIM Attribute	ALARMED	Realtime	Always Present
PTM_ATTR_TIMESTAMP	SourceTimestamp	-	Realtime	Always Present
EXTRA	CIM Attribute	EXTRA	Static	Always Present
$RAW_VALUE	CIM Attribute	$RAW_VALUE	Realtime	Always Present
LEVEL	CIM Attribute	LEVEL	Static	Always Present
PT_POINT_ID	UA NodeID Attribute	-	Static	Always Present
EXTENDED_USER_FLAGS_LOW	CIM Attribute	USER_FLAGS	Realtime	Always Present
EXTENDED_USER_FLAGS_HIGH	CIM Attribute	USER_FLAGS	Realtime	Always Present

In some cases, the CIMPLICITY attribute maps to a concept defined by OPC UA. The mapping column specifies the details for each mapping. If an attribute is mapped to a 'CIM Attribute,' it displays as a property of the Attributes component for the Point object. The name of the property is specified by the Attribute Name column.

In many cases, CIMPLICITY attributes provide information that is set at configuration time. This information can change, but only if the CIMPLICITY project supports dynamic configuration. Realtime attributes are associated with the current value and are read whenever the current value is read.

The Mapping for CIMPLICITY points described in the diagram linked above uses four possible Variable Type for points: DataItemType (the default), AnalogType, TwoStateDiscreteType, and MultiStateValueDiscreteType.

A CIMPLICITY Point is mapped to a TwoStateDiscreteType if it has a Boolean DataType and an Enumeration set with exactly two values.

A CIMPLICITY Point is mapped to a MultiStateValueDiscreteType if it has a numeric DataType and an Enumeration set, and it is not a TwoStateDiscreteType.

A CIMPLICITY Point is mapped to a AnalogType if it has a has a numeric DataType and the RANGE_HIGH or DISPLAY_HIGH attribute set.

Any other CIMPLICITY Point is mapped to DataItemType.

The AttributeSets defined for the Project can be found by browsing the subtypes of CIMPointUserAttribType.

The NodeId of an AttributeSet is <project name>.AS.<set name>.

The PointEnumerations defined for the CIMPLICITY Project can be found by browsing the subtypes of CIMPointEnumerationType.

The NodeId of a PointEnumeration is <project name>.PE.<enumeration name>.

Device Points may have a conversion defined. If they do the $RAW_VALUE property contains the original value received from the Device and the Value contains the value with the conversion applied. If a conversion is applied, the formula specified in the CIMPLICITY configuration is specified by the Definition property.

CIMPLICITY Alarms

All OPC UA Alarm clients can receive CIMPLICITY active alarm details, which can be used by other applications including Operations Hub.

The CIMPLICITY data in the OPC UA Alarm fields matches alarm number and state with the CIMPLICITY alarm viewer. Users can subscribe to different levels in the Browse hierarchy, as well as non-point alarms, and users can also filter alarms as necessary. Alarm acknowledgements are also supported.

Redundancy

Configuring Redundancy

Server Configuration

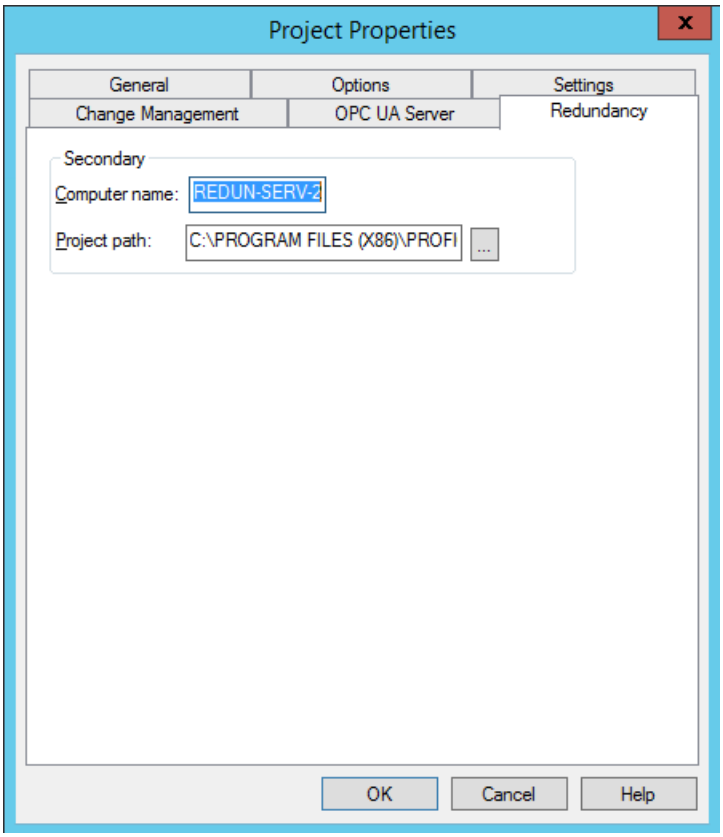
To configure redundancy:

1. Select Properties from the Projects menu on the Workbench menu bar.

The Project Properties dialog box opens.

1. Select the General tab.
2. Check Server Redundancy in the Options box.

A Redundancy tab is added to the Project Properties dialog box and opens.



1. Enter the Secondary Computer Name and Project Path.
2. Click OK to save the information.

Server Redundancy

The CIMPLICITY OPC UA Server detects if a project is configured for redundancy. If it is configured for redundancy:

- Generates (if they do not already exist) separate certificates for the Primary and Secondary machines. These certificates are saved in the <project>/pki/own folder and have the host name appended to the name. When the OPC UA starts, it replaces the default certificate and key with the certificate and key matching the current host.
- Updates the Server/ServerRedundancy Object. The RedundancySupport property is set to 'HOT'. The ServerUriArray property is populated with the URIs of the Primary and Secondary CIMPLICITY UA Servers. The ServerNetworkGroups property is populated with the IP addresses for the Primary and Secondary hosts which are saved in the CimHosts file.
- Sets the ServiceLevel to 255 for the current active and 190 for the current standby. The CIMPLICITY OPC UA Server detects when the active is changed and updates the ServiceLevel to reflect the current state. Clients use this property to determine which server must be used for writes.

The OPC UA server network address needs to use [NodeName] for the primary and the secondary nodes to have different end points.

If GDS agent is used to sign OPC UA server certificate, GDS agent needs to be launched using advanced button in the OPC UA security configuration tool.

Supported Data Types

Data Types

The CIMPLICITY OPC UA server exposes the following data types.

3D_BCD	3-digit binary coded, 2 byte (16 bits) unsigned integer ranging from 0 to 999.
4D_BCD	4-digit binary coded, 2 byte (16 bits) unsigned integer ranging from 0 to 9999.

CIMPLICITY Data Type	OPC UA Datatype	OPC UA Node Id
BOOL	Boolean	1
BYTE	Byte	3
DWORD	UInt32	7
WORD	UInt16	5
DINT	Int32	6
INT	Int16	4
QINT	Int64	8
REAL	Double	11
SINT	SByte	2
UDINT	UInt32	7
UINT	UInt16	5
UQINT	UInt64	9
USINT	Byte	3
3D_BCD	UInt16	5
4D_BCD	Uint16	5

These are very similar to the data types described in the OPC UA Client data type help in the following topics:

- Communications Equipment

- Native Device Communications
- CIMPLICITY OPC UA Client
- OPC UA Client Technical Reference
- OPC UA Data Types Mapped to CIMPLICITY Data Types

However, there are a few differences:

- The CIMPLICITY Real Data type is exposed as an OPC UA Double.
- The 3D_BCD data type is exposed as an OPC UA UInt16, with maximum limits and setpoint limits of 0 - 999.
- The 4D_BCD data type is exposed as an OPC UA UInt16, with maximum limits and setpoint limits of 0 - 9999.
- The user has the ability to set these limits, but if they don't, the OPC UA Server will limit the values to the described limits.
- The limits are exposed in the OPC UA Analog Item Type Address Space properties EURange and InstrumentRange.
- Setpoints are exposed in the OPC UA item address space attributes as SETPOINT_LOW and SETPOINT_HIGH.

The BCD values are describe in the help under the following topics:

- Points
- Device Points
- Step 2. Enter Device Point General Properties
- Step 2.1 Enter Device Point Basic General Properties

Troubleshooting

OPC UA Server log files are saved to the default location "%SITE_ROOT%\log \CimOpcUaServer.Log.txt" unless you specify a different location in the Logging Configuration dialog box in the OPC UA Server tab of the Project Properties dialog box.

Logging Trace Level Explanations

Stack Trace Level

ERROR - Critical errors (unexpected and/or requiring external actions) which require attention

WARNING - Non-critical faults which should not go unnoticed but are handled internally

SYSTEM - Rare major events (good cases) like initializations, shut-down, etc.

INFO - Regular good case events like connects, renews

DEBUG - Used for debugging purposes

CONTENT - Used to add additional content (i.e. whole message bodies) to debug traces

Application Trace Level

NoTrace - No Trace

Errors - Unexpected errors

Warning - Unexpected behavior that is not an error

Info - Information about important activities like connection establishment

InterfaceCall - Calls to module interfaces

CtorDtor - Creation and destruction of objects

Frequently Asked Questions

Q: Why can't I connect to the OPC UA Server after I start my project?

A: Open the Project Properties dialog box and check that the OPC UA Server is enabled and that the OPC UA Client is using the correct port number.

Note: Each project that needs to run on the same machine at the same time must have a different port assigned. If two projects have the same port they will conflict with each other.

Q: Why can't I connect even though the OPC UA Server is enabled and the URL is correct?

A: Check the Status Log from Workbench for error information. Look for details in the CimOpcUaServer.Log.txt file.

Q: Why does the log say that the OPC UA Client Certificate is not trusted?

A: Could be an issue with your certificate. Copy the OPC UA Client Certificate from the "%SITE_ROOT%\pki\rejected" to "%SITE_ROOT%\pki\trusted\certs".

Note: You must manually verify any Certificate copied to the trusted directory by looking at the Certificate properties. If it does not belong the OPC UA Client that you are currently configuring, you must delete the Certificate from the rejected directory.

Q: Why do I get an error when I subscribe to a Point in my OPC UA Client?

A: A CIMPLICITY Point is represented as an Object. If you wish the current value of the Point you must select the Value Variable which is a component of the Point Object.

Chapter 21. Driver Server Client

About Driver Server Client

The Driver Server is a multi-protocol server for process data.

Driver Server Required Documentation

Additional documentation is available in the Driver Configuration.

CIMPLICITY Configuration for the Driver Server

CIMPLICITY Configuration for the Driver Server

You will be asked for some information that is specific to Driver Server when you:

Step 1 *(page 398)*	Enable the Driver Server.
Step 2 *(page 399)*	Configure the CIMPLICITY Driver Server port.
Step 3 *(page 402)*	Configure the CIMPLICITY Driver Server device.
Step 4 *(page 415)*	Configure the CIMPLICITY Driver Server points.

Step 1. Enable the Driver Server

1. To create a new project and enable Driver Server, do one of the following:

   Method 1

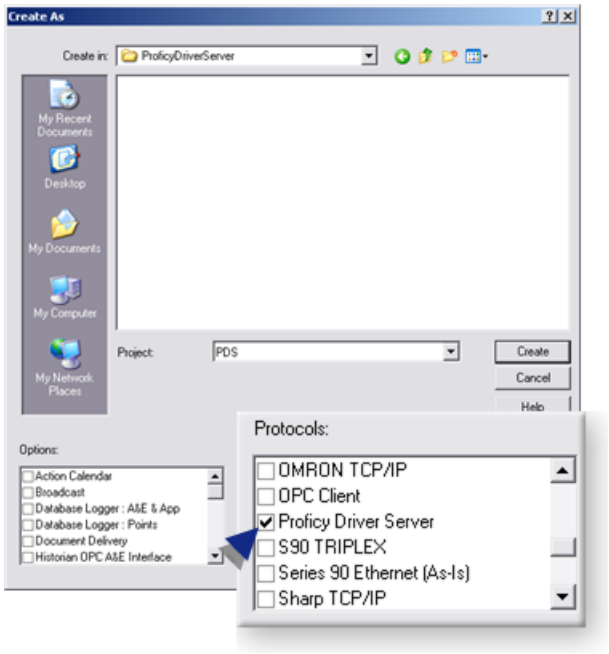
   Click File>New>Project on the Workbench menu bar.

Method 2

Click the Project button in the Workbench toolbar.

The Create As dialog box opens when you use either method.

2. Enter specifications to create a new CIMPLICITY project.
3. Check Driver Server in the **Protocols** box.



4. Click **Create** to create the project with the Driver Server protocol.

   The Project Properties dialog box opens.

5. Do one of the following,
   - Edit your previous selections in the General tab.
   - Click **OK** to continue.

A project is created in CIMPLICITY with Driver Server enabled.

Note: Clearing Driver Server in the Protocols field of the General tab will disable Driver Server for CIMPLICITY.

Step 2. Configure the CIMPLICITY Driver Server Port

Step 2. Configure the CIMPLICITY Driver Server Port

Step 2.1 *(page 400)*	Create a new CIMPLICITY Driver Server Port.
Step 2.2 *(page 400)*	Set Port General Properties.
Step 2.3 *(page 401)*	Set Port Settings Properties.

Step 2.1. Create a New CIMPLICITY Driver Server Port

1. Select the Equipment>**Ports** icon Ports in the Workbench left pane.
2. Do one of the following.

   Method 1

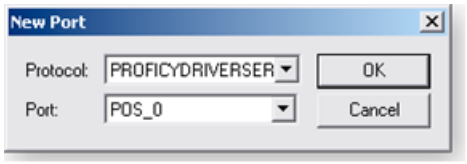
   Right-click the **Ports** icon; select New on the Popup menu.

   Method 2

   Click File> New>Object on the Workbench menu bar.

   The New Port dialog box opens when you use either method.

3. Select the following.



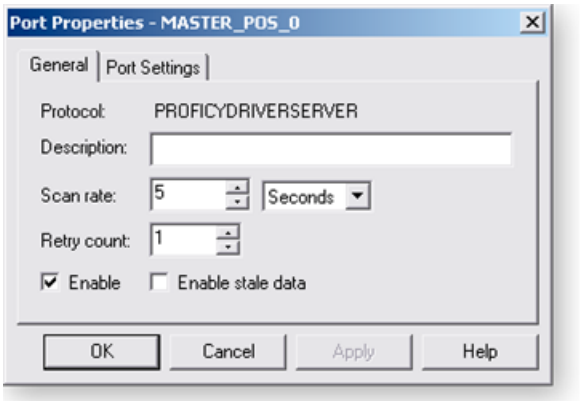
Field	Select
Protocol	PROFICYDRIVERSERVER
Port	One of the ports in the dropdown list.

4. Click OK.

The Port Properties dialog box for the protocol opens.

Step 2.2. Set Port General Properties

The OPC protocol you select in the New Port dialog box appears in the General tab of the Port Properties dialog box. Enter information that pertains to Driver Server in this tab.



Enter the following information in the General tab of the Port Properties dialog box.

Description	(Optional) The description of the port.	
Scan Rate	The base scan rate for the port. Point scan rates will be multiples of the base rate. You can specify a scan rate in Ticks (100 ticks = 1 second), Seconds, Minutes, or Hours. **Notes:** • This scan rate applies only to points that are not configured with an unsolicited or unsolicited on change update criteria. • This value does not define how often the Driver Server polls its hardware. It defines only how often CIMPLICITY polls the Driver Server. See the Scan Rate variable on the Group Settings tab of the Device dialog box for more information.	
Retry Count	If communications cannot be established with a Driver Server on this port, the Driver Server is considered to be down. For polled communication only, this is the number of scans until polled communications is attempted in the absence of the receipt of unsolicited data.	
Enable	Check	The port is enabled when the project starts.
	Clear	The port will not be enabled when the project starts, and points will not be available the Driver Server device points for devices on the port
Enable stale data	Check	Point values will remain available under most circumstances that would have made them unavailable. Under these conditions, the values displayed will be the last known value for the point. Since the current values are unknown, the values may or may not represent the current values on the device.
	Clear	Point values are no longer available under circumstances that make them unavailable.

Step 2.3. Set Port Settings Properties

Value in ticks (1/100 of a second).`BatchTimeToLive` determines how long Driver Server waits for a completion notification that a dynamic batch addition has finished.When `BatchDynamicAdditions` is True:

1. Driver Server waits until it receives a completion notification that the dynamic additions have completed.

2. Upon receiving this notification, CIMPLICITY adds the batch of items to the Driver Server.

In heavily loaded systems, it is possible for the completion notification to be delayed or missed. In these cases, a timer (that is, the BatchTimeToLive property) causes the batched points to be added without waiting any longer for the completion notification.

Step 3. Configure the CIMPLICITY Driver Server Device

Step 3. Configure the CIMPLICITY Driver Server Device

To create a new Client Device configuration for the Driver Server, you perform the following steps:

Step 3.1 *(page 402)*	Create a new Driver Server device.
Step 3.2 *(page 403)*	Set device general properties.
Step 3.3 *(page 404)*	Set device settings properties.
Step 3.4 *(page 414)*	Set Device group settings properties.

Step 3.1. Create a new CIMPLICITY Driver Server Device

1. Select the Equipment>**Devices** icon Devices in the Workbench left pane.
2. Do one of the following.

   Method 1

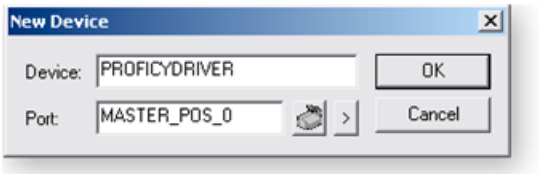
   Right-click the **Devices** icon; select New on the Popup menu.

   Method 2

   Click File> New>Object on the Workbench menu bar.

   The New Device dialog box opens when you use either method.

3. Select the following.



Field	Select
Device	Name to identify the device.
Port	Driver Server port that will be used for the device in the dropdown list

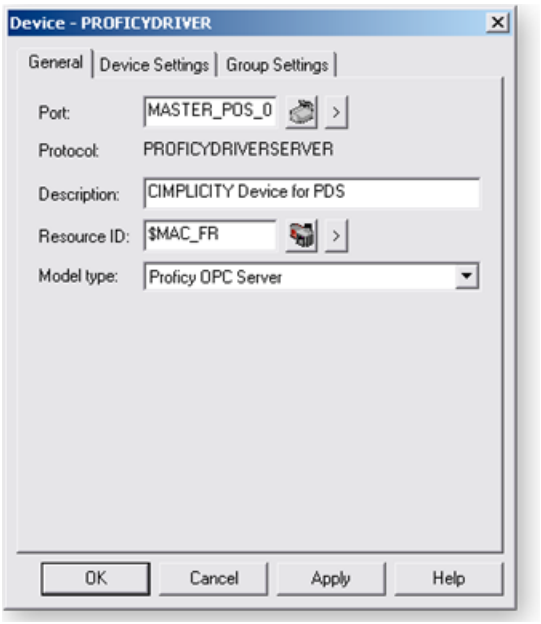
4. Click OK.

The Device dialog box opens.

Step 3.2. Set Device General Properties

When you create a new device, the Device dialog box appears displaying the Driver Server associated with the port you selected in the New Device dialog box.

Enter information on the General tab of the Device dialog box as follows:



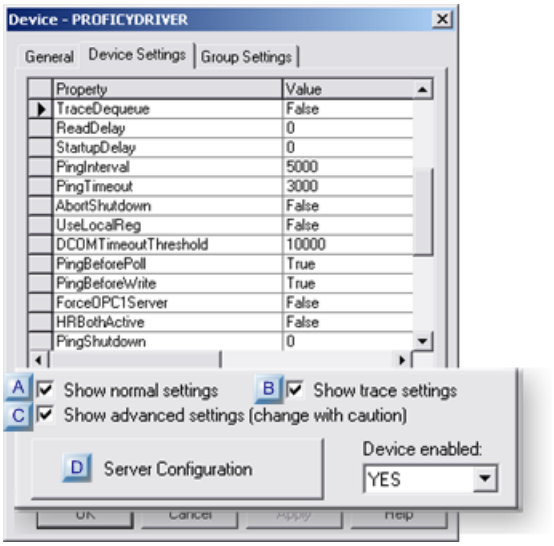
Field	Description	
Port	Select the port for this device. Click the buttons to the right of the Port field to select the port, as follows.	
		Display the list of ports and select one.
		Create a new port, edit the current port, or select a port from the list of ports.
Description	Enter an optional description to help you identify the device.	
Resource ID	Enter a resource that can be associated with this device for alarm generation. Click the buttons to the right of the **Resource** field to select the resource, as follows.	
		Display the list of resources and select one.

	>	Create a new resource, edit the current resource, or select a resource from the list of resources.
Model Type	Always select OPC Server.	

Step 3.3. Set Device Settings Properties

Step 3.3. Set Device Settings Properties

The Device Settings tab in the Driver Server Device dialog box provides the following features.



rect 36, 277, 58, 298 *(page 409)*
rect 0, 246, 22, 267 *(page 407)*
rect 156, 229, 178, 250 *(page 406)*
rect 0, 226, 22, 247 *(page 404)*

A *(page 404)*	Show normal settings.
B *(page 406)*	Show trace settings.
C *(page 407)*	Show advanced settings.
D *(page 409)*	Server configuration.

A	Show normal settings

Normal settings are required device properties.

Normal Settings	Value	Description
PingInterval		(Milliseconds) Determines how often CIMPLICITY pings the Driver Server.
	Default	5000
PingTimeout		(Milliseconds) Determines how long Driver Server waits for a response from the Driver Server. A communication error is declared if this timeout period passes with no response.
	Default	3000
UseLocalReg	True	Enables CIMPLICITY to access the local registry information to find the Driver Server ID first. If CIMPLICITY fails to find the server information in the local registry, it logs a warning message and searches the information in the remote node. Setting this parameter to True resolves the multi-sessions issue.
	False	CIMPLICITY does not access the local registry information to find the Driver Server ID first.
	Default	False
DCOMTimeoutThreshold		(Milliseconds) Determines how long CIMPLICITY waits to abort the connection to the Driver Server after detecting that the server has not responded to a ping.
	Default	10000
PingBeforePoll	True	Causes CIMPLICITY to ping the Driver Server before sending a poll request. Pinging the Driver Server immediately before the poll eliminates most potential DCOM timeouts.
	False	Can cause all device communications to hang until the poll times out.
	Default	True
PingBeforeWrite	True	Causes CIMPLICITY to ping the Driver Server before sending a write request. Pinging the Driver Server immediately before the write eliminates most potential DCOM timeouts.
	False	Can cause all device communications to hang until the write request times out.
	Default	True
ReAddAsEmptyOnBadType	True	Enables CIMPLICITY to automatically re-add a point to the Driver Server that was rejected by the server based on the point's data type. CIMPLICITY changes the data type of the rejected point to VT_EMPTY. The Driver Server will not reject the point based on this type.
	False	Enables the Driver Server to reject points based on the configured data type. You will need to investigate and determine why the point type(s) are rejected by the Driver Server.
	Default	True
RestartDelay		(Milliseconds) Determines how long CIMPLICITY waits before attempting to re-establish a lost connection to the Driver Server. Driver Server attempts to reconnect until it succeeds.

	Default	10000
ReconnectInterval		(Milliseconds) Determines the time between reconnection attempts (i.e. RestartDelay). This time does not include the time it takes Microsoft's COM (DCOM) engine to determine if it is able to launch the OPC server application.
	Default	1000
RemoveItemsOnGroupRemove	True	Deletes items from a group before the group is removed.
	False	Deletes items at the same time the group is removed.
	Default	True
ItemAccessPathEnable	True	Enables access paths for a given device.
	False	Access paths are not enabled.
	Default	False
DeviceReadAfterSet	True	Requests that the Driver Server read the data from the device.
	False	Forces the Driver Server to read from its cache. **Note:** False is the default when DEFAULTUNSO is selected as the configured group *(page 414)*.
	Default	True

B	Show trace settings

Trace settings are runtime trace diagnostics properties.

Trace Settings	Value		Description
TraceSeparate	True		Trace records from this device are written to a separate log file. For a detailed description of the Trace Separate property, refer to Enable Tracing *(page 429)*.
	False		Trace records from this device are written to the common log file.
	Default		True
TraceLevel			The Trace Level determines how much trace data is written to the log file. Options are:
		1	BIGERRORS
		2	ALLERRORS
		3	BIGSUCCESSES
		4	ALLSUCCESSES
		5	ITEMDETAILS
		6	DEVELOPER LEVEL 1
		7	DEVELOPER LEVEL 2
	Important: : If the TraceLevel property is not set, nothing is written to the log file. For detailed descriptions of trace levels, refer to Enable Tracing *(page 429)*.		

	Default	1
TraceAll	True	Traces all communications between CIMPLICITY and the Driver Server.
	False	All communications between CIMPLICITY and the Driver Server are not being traced, but setting TraceAll to False does not prevent any other tracing.
	Default	True
TraceConnection	True	Traces connections.
	False	Connections are not traced.
	Default	False
TraceGroupActivity	True	Traces group activity.
	False	Group activity is not traced.
	Default	False
TraceItemActivity	True	Traces item activity.
	False	Item activity is not traced.
	Default	False
TracePinging	True	Traces pinging activity.
	False	Pinging activity is not traced.
	Default	False
TracePolling	True	Traces polling activity.
	False	Polling activity is not traced.
	Default	False
TraceEvents	True	Traces event activity.
	False	Event activity is not traced.
	Default	False
TraceWriting	True	Traces writing activity.
	False	Writing activity is not traced.
	Default	False
TraceDequeue	True	Traces dequeuing activity, the removal of the item id updates from the Server.
	False	Dequeuing activity is not traced.
	Default	False

C	Show advanced settings (change with caution)

Advanced settings are advanced properties.

Note: These properties should not be changed unless they resolve a specific issue.

For assistance with changing the advanced properties, contact GE Intelligent Platforms technical support.

Advanced Settings	Value	Description
StartupDelay		(Milliseconds) Provides the Driver Server extra time to start in order to process points being loaded by CIMPLICITY. This delay occurs after the Driver Server starts, but before any points are added by CIMPLICITY.
	Default	0
ReadDelay		(Milliseconds) Provides the Driver Server extra time to get the initial values from the points added by the client. This delay occurs after CIMPLICITY adds points to the Driver Server, but before performing the first poll. If a poll occurs too soon after the Driver Server starts, values of Bad Quality are provided.
	Default	0
AbortShutdown	True	Shuts down CIMPLICITY communications with the Driver Server when the CIMPLICITY project shuts down, without releasing any references to objects in the Driver Server. It is not recommended that you set this property to True, as it is against the rules of OLE. If you set this property to True, the Driver Server does not shut down because it does not know that CIMPLICITY has disconnected.
	False	Driver Server does not shut down when the project shuts down.
	Default	False
ForceOPC1Server	NA	Data Access 1 specification is not supported.
	False	The client establishes a connection using OPC DA 2.0 methods if the server supports it. OPC DA 1.0 methods are used if the server does not support OPC DA 2.0.
	Default	False
HRBothActive	True	Both servers in a Host Redundant system are active. Setting this property to True results in both servers advising the client, causing more traffic and a slower transition time. By default, in Host Redundancy environments, the acting active server advises the points that are defined to be: • Unsolicited. • Unsolicited on change. • Poll once. • Poll once on change (i.e. the server polls the data and notifies the client only of those items that have changed). Some servers may be slow in collecting the data, resulting in longer transition times for unsolicited data. However, all data is advised and refreshed (unless refresh is disabled) on host transition.

	False	Any redundant servers are inactive. Some servers may be slow in collecting the data, resulting in longer transition times for unsolicited data. There will be less traffic and faster transition times with this property set to False.
	Default	False
PingShutdown		(Milliseconds) Determines how long it takes to shut down pinging. This property is useful if the Driver Server is slow or has heavy traffic. If set to zero (0), the Ping shutdown rate is equal to twice the defined PingTimeout value.
	Default	0
DeviceShutdown		(Milliseconds) Determines how long it takes to shut down the device. This property is useful if the Driver Server requires more time to clean up large configurations.
	Default	10000
DetectPingHang	True	This prevents Driver Server from being "hung". The DetectPingHang parameter will log that the OPC Server is down and will begin trying to reconnect to the OPC Server.
	False	If the Driver Server, pings the OPC Server and the server has crashed or the connection is broken, then Driver Server will wait or "hang" for an unspecified time.
	Default	False
ChangeGroupStateOnDeviceTransition	True	Changes the state of the group associated with the device to match the state (enabled/disabled) of the device when it changes.
	False	The state of the group does not change when the state of the device changes.
	Default	True
ChangeGroupStateOnHostTransition	True	Changes the state of the group associated with the device when the servers in a Host Redundancy system transition between primary and secondary.
	False	Does not change the state of the group associated with the device when the servers in a Host Redundancy system transition between primary and secondary.
	Default	False
ItemAccessPathDelimiter		Defines the token to use for the access path if the ItemAccesspathEnable *(page 406)* property is set to True. A space cannot be assigned as a delimiter.
	Default	Semi-colon (;)

D	Server Configuration

The Device Settings tab in the Device dialog box provides direct access to the Driver Configuration window.

Option		Description

Server Configuration	Opens the restricted Driver Configuration window. **Note:** The physical device is configured through the Driver Configuration window.	
Device Enabled	Select YES or NO in the **Device Enabled** dropdown list.	
	YES	Enables the selected CIMPLICITY device when the CIMPLICITY project is started.
	NO	Disables the selected CIMPLICITY device when the CIMPLICITY project is started.

Configuration for the Driver Server project that is created as a companion for the CIMPLICITY project includes the following.

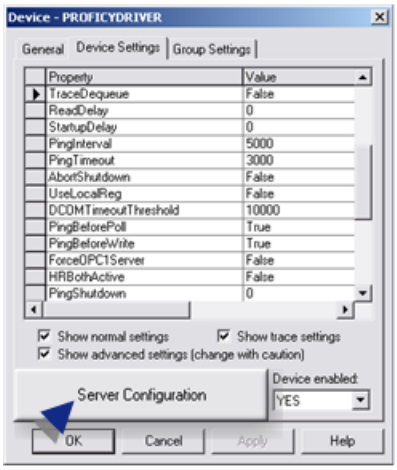
3.3.1 *(page 410)*	Driver Configuration window.
3.3.2 *(page 411)*	Driver Server associated project configuration.

3.3.1. Driver Configuration Window

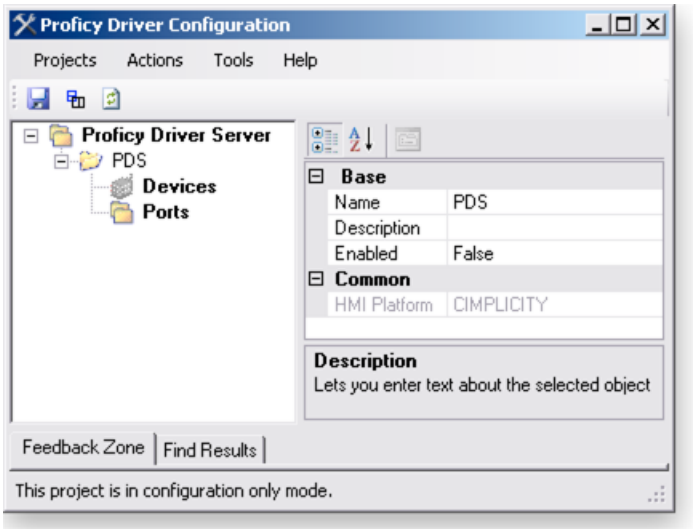
Note:

- The Server Configuration button is disabled when the CIMPLICITY project is running in dynamic mode.
- Consult Driver Server documentation for in depth information about configuration in the Driver Configuration window.

Click the Server Configuration button on the Device dialog box>Device Settings tab.

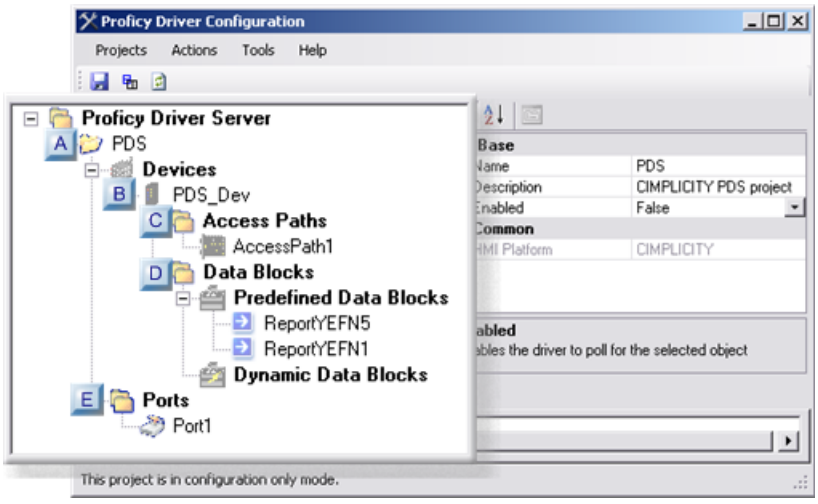


Result: The Driver Configuration window opens in configuration mode for the Driver Server project.



3.3.2. Driver Server Associated Project Configuration

Basic entries for a Driver Server project that is associated with a CIMPLICITY project are as follows.



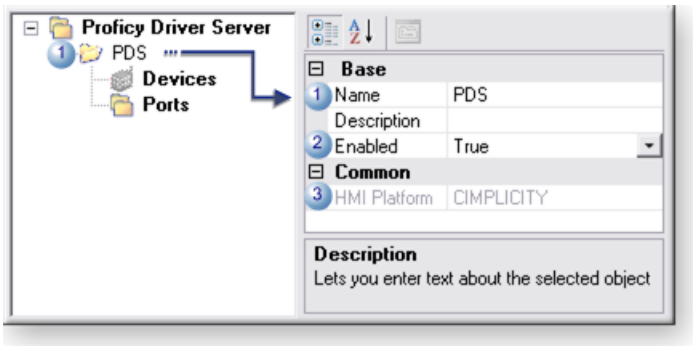
rect 43, 235, 65, 258 *(page 414)*
rect 84, 158, 106, 181 *(page 413)*
rect 84, 123, 106, 146 *(page 413)*
rect 60, 106, 82, 129 *(page 412)*
rect 24, 76, 46, 99 *(page 412)*

A *(page 412)*	Driver Server project
B *(page 412)*	Driver Server device.

C *(page 413)*	Driver Server access path.
D *(page 413)*	Driver Server data blocks.
E *(page 414)*	Driver Server ports.

A	Driver Server Project

The associated Driver Server project displays as follows when the Driver Configuration window opens in configuration mode.

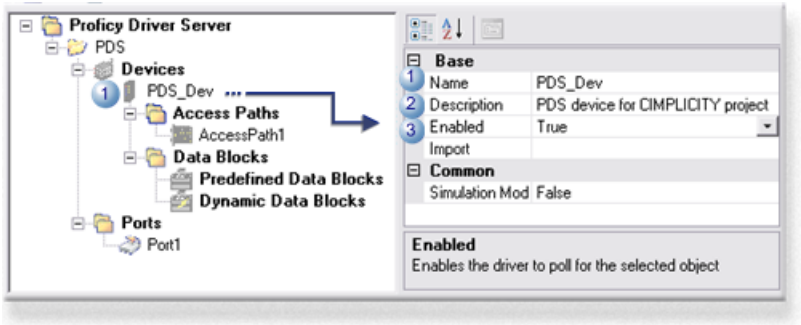


1	Name	The CIMPLICITY project name is assigned to the Driver Server project.
2	Enabled	The Driver server project is initially disabled. Set **Enabled** to True before starting the CIMPLICITY project.
3	HMI Platform	(Read-only) CIMPLICITY is the **HMI Platform**.

Review Driver Server documentation for detailed information about Driver Server projects.

B	Driver Server Device

The following needs to be entered in the Driver Configuration window.

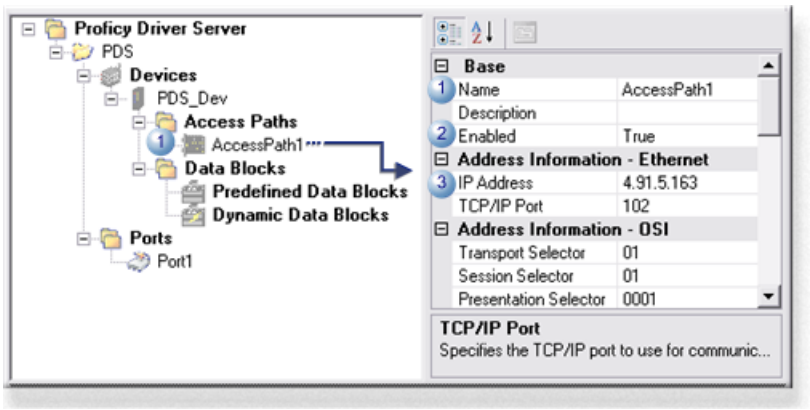


1	Name	A physical device (e.g. IED) must be added to the Driver configuration. Result: The tree expands to include the following. • Access Paths, which includes AccessPath1 • Data Blocks, which includes • Predefined Data Blocks • Dynamic Data Blocks • Port1 under Ports **Note:** • The Driver Server device communicates directly with the physical device. • The name of the Driver Server device that is added in the Driver Configuration window will be included in the CIMPLICITY device point address *(page 418)*.
2	Description	A description helps identify the Driver Server device will display when configuring an address for a CIMPLICITY point.
3	Enabled	Selecting **True** before starting the CIMPLICITY project will enable the device. **Note:** Make sure the Driver Server project is also enabled *(page 412)*.

Review Driver Server documentation for detailed information about Driver Server devices.

C	Driver Server Access Path

Communication through the CIMPLICITY project requires the following basic entries.



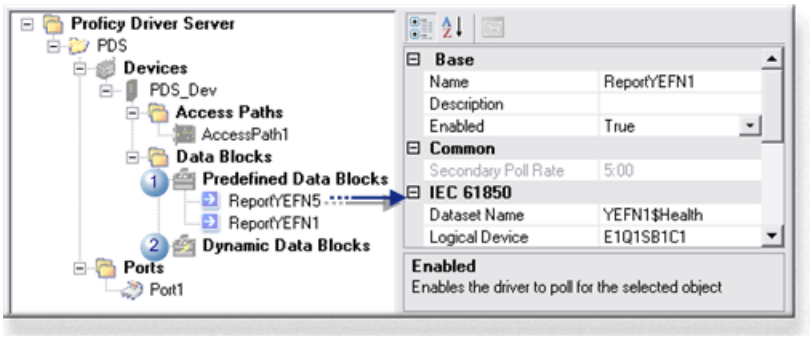
1	Name	An access path is added to the tree when the device is added. **Note:** (Optional) The name can be changed.	
		Default	AccessPath<n> Where n is a random number.
2	Enabled	Set **Enabled** to True before starting the CIMPLICITY project	
3	IP Address	Make sure the access path has a valid **IP address**.	

Review Driver Server and protocol documentation for detailed information about access paths.

D	Driver Server Data Blocks

Driver Server data blocks can be predefined or dynamic.

When the Driver Server project is associated with a CIMPLICITY project, CIMPLICITY does the following.

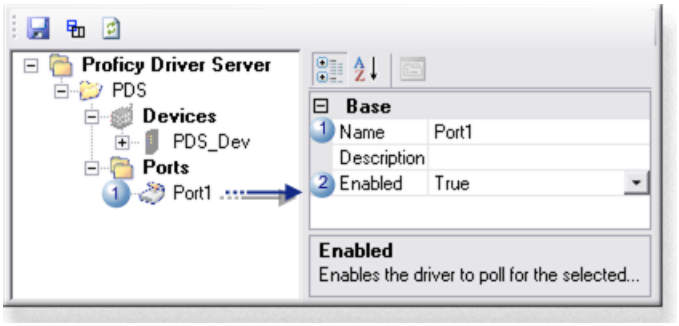


1	Predefined Data Blocks	Configuration for predefined data block data attributes that are CIMPLICITY point addresses impact how point values are retrieved from the device.
2	Dynamic Data Blocks	When the CIMPLICITY project starts the Driver Server will create dynamic data blocks for any CIMPLICITY point addresses that are not included as attributes in predefined data blocks.

Review Driver Server and protocol documentation for detailed information about access paths.

E	Driver Server Ports

When a device is added to the Driver Server project, a port is automatically added.



1	Name	A port is added to the tree when the device is added. **Note:** (Optional) The name can be changed.	
		Default	Port<n> Where n is a random number.
2	Enabled	Set **Enabled** to True before starting the CIMPLICITY project	

Review Driver Server and protocol documentation for detailed information about ports.

Step 3.4.: Set Device Group Settings Properties

1. The system is fast enough relative to data retrieval requirements for the server to poll devices every 500 ms. and retrieve all values.
2. CIMPLICITY update requirements can accommodate the possible slight variations in timing that might occur and cause a slight delay in updates.
3. The server requires more than 500 ms. to retrieve all values from the device, causing data overruns.
4. Updates are lost instead of being sent to CIMPLICITY.
5. The server can retrieve all values from the device within a 10,000 ms. interval.
6. CIMPLICITY does not require fast updates; if some are delayed by up to 20,000 ms. it is not a problem.
7. The client timer was 250 ms. ahead of the server.
8. The server:
   a. Did not retrieve the updated value from the device for 9,000 ms. after the actual update.
   b. Had sent updates to the client before it retrieved the updated value into its cache.
   c. Waited until the next scan was triggered to send the value to the OPC client.
9. Over 19,000 ms. passed before the OPC client received the updated value.
   - Server retrieval of all values from the device requires up to 10,000 ms.

   Any updated values that were retrieved from the device in between the poll/scan rates will not be sent to the client until the next scan is triggered, which can be up to 10,000 ms.
10. Has time to successfully poll the device and retrieve values.
11. Will send any required updates that it finds in the cache over to the OPC Client every 1,000 ms.

Step 4. Configure the CIMPLICITY Driver Server Points

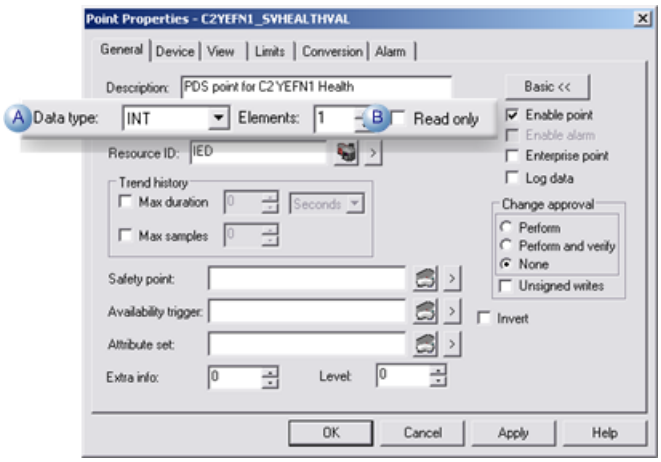
Step 4. Configure the CIMPLICITY Driver Server Points

Configure CIMPLICITY point properties specific for the Driver Server in the Point Properties dialog box:

Step 4.1 *(page 416)*	Set point general properties.
Step 4.2 *(page 416)*	Set point device properties.

Step 4.1. Set Point General Properties

When you configure a new point in the Point Properties dialog box, entries on the General tab that apply particularly to the Driver Server configuration include the following.

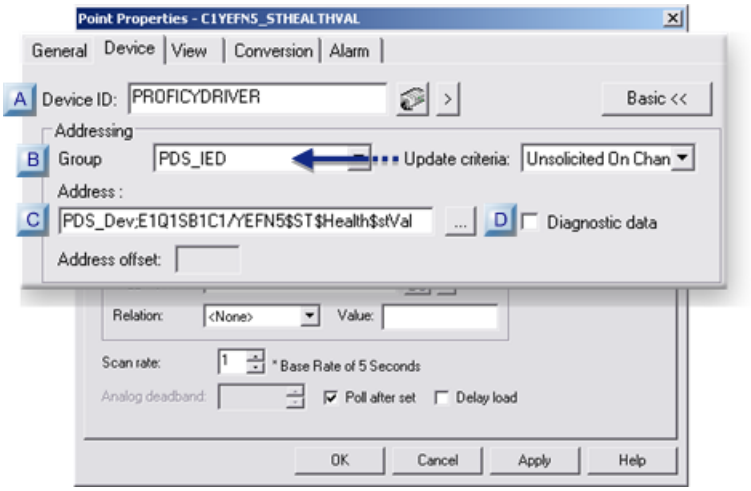


	Field	Description	
A	Data type	When you select the **Data Type** for the point, the Driver Server determines the validity of that type for the selected point. If the point is rejected by the Driver Server because the requested type is not valid, CIMPLICITY will: • Declare the point as invalid. • Log a message to the Status Log indicating that the point was rejected because the point type was invalid. **Note:** In some Driver Server protocols, points may be any of several allowed point types.	
B	Read only	Points can be configured for Read-only or Read/Write	
		Check	Read-only
		Clear	Read/Write

Note: Review the Driver Server documentation for the protocol you are configuring. The more familiar you are with your particular protocol, the easier it will be to configure points successfully.

Step 4.2. Set Point Device Properties

The protocol that is



rect 303, 126, 322, 145 *(page 425)*
rect 7, 125, 26, 144 *(page 418)*
rect 7, 89, 26, 108 *(page 417)*
rect -1, 49, 18, 68 *(page 417)*

Fill in the following information on the Device Tab of the Point Properties dialog box:

A *(page 417)*	Device ID
B *(page 417)*	Group/Update criteria
C *(page 418)*	Address
D *(page 425)*	Diagnostic data

A	Device ID

Select a Driver device.

Note: The Device Browser button  to the right of the Device ID field opens the Select a Device browser.

B	Group/Update criteria

The **Update criteria** selection compared to the group's **isPolled** setting on the Device dialog box>Group Settings tab determines what groups in the **Group** field are available for the selected point.

The isPolled value must be as follows for a selected update criteria.

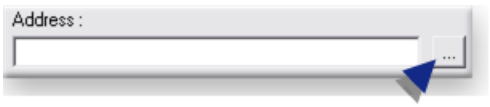
Update Criteria	isPolled must be:
Unsolicited	False
Solicited on Change	False
On Change	True
On Scan	True
Poll Once	True
Poll Once on Change	True
On Demand On Scan	True
On Demand On Change	True

Important: You cannot use an unsolicited point as a trigger point .

C	Address

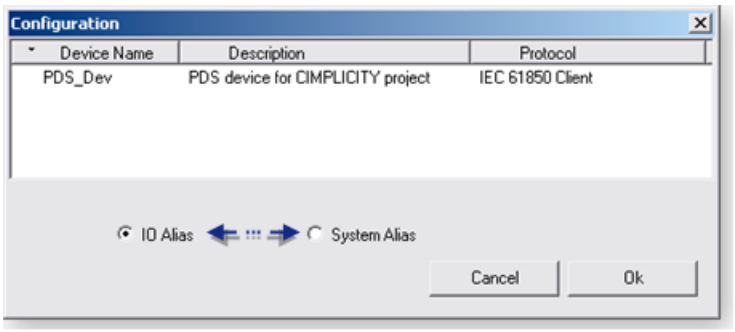
- The **Address** field may contain up to 256 characters.
- The information in the Configuration dialog box is coming from the Driver Server.
- An address can be entered directly in the field or (recommended) through in the Configuration dialog box.

Click the Browse button to the right of the **Address** field.



Result: The Configuration dialog box opens.

The address can be one of the following.



rect 190, 133, 260, 155 *(page 424)*
rect 62, 133, 126, 155 *(page 419)*

- IO alias: Live point data.
- System alias: Diagnostic point data.
- Addressing examples.

IO Alias: Live Point Data

When IO Alias is checked the associated device that was entered and configured in the Driver Configuration window for the associated Driver Server project is listed with its description and selected protocol.

When the device is selected the Configuration window expands to display fields to enter an address for the selected protocol; what fields display depend on the protocol.

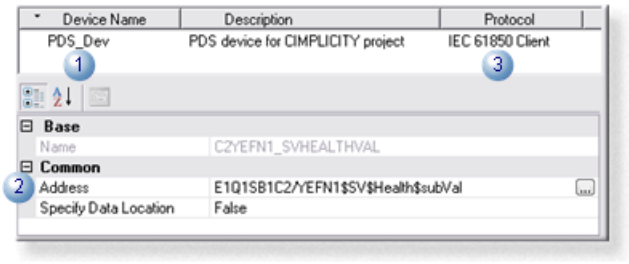
The Address field syntax is as follows.

```
[Device Name];Address
```

Where

Device name	Is the name of the Driver Server device.
;	Separates the device name from the entered address.
Address (content/syntax)	Depends on the selected protocol.

The three common items that directly affect the address are as follows.



1	Device Name	The selected device was assigned to the associated CIMPLICITY project/device in the Driver Configuration window.
2	Address	The content/syntax in the **Address** field depend on the protocol assigned to the selected device. **Note:** Click the button to the right of the **Address** field to open a Point Browser.
3	Protocol	The protocol: • Was selected when the device was created in the Driver Configuration window. • Can cause additional fields to display in the Configuration dialog box.

Review the protocol documentation for detailed information about protocol configuration.

Following are addressing examples.

1	SRTP Protocol Addressing.
2	IEC 61850 protocol.
3	BACnet Protocol Addressing

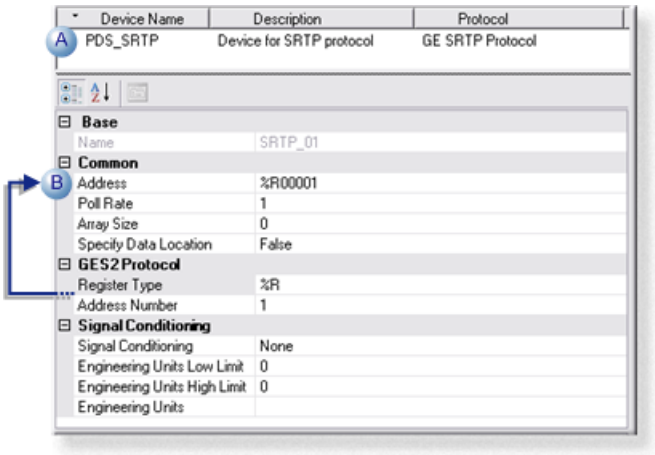
Example 1. SRTP Protocol Addressing

A Driver Server device uses a `GE SRTP` protocol.

The GE SRTP entry in the Configuration dialog box **Address** field notes the following.

Register Type	%R
Address Number	1

Fields in the Configuration dialog box that define the address are as follows.

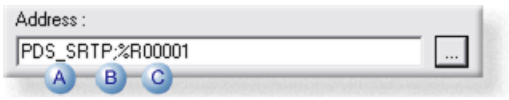


A	Device Name	A device with a GE SRTP protocol is selected in the Configuration dialog box.
B	Address	The SRTP address includes the: • Register type • Address number.

When the Configuration dialog box is closed, the **Address** field in the Point Properties dialog box displays the following string.

PDS_SRTP;%R00001

The elements in the string are:



	Entry	Description
A	PDS_SRTP	Device name
B	;	Syntax separator
C	%R00001	The address for the PDS_SRTP device

Review Driver Server and SRTP protocol documentation for detailed information about all of the fields in the Configuration dialog box.

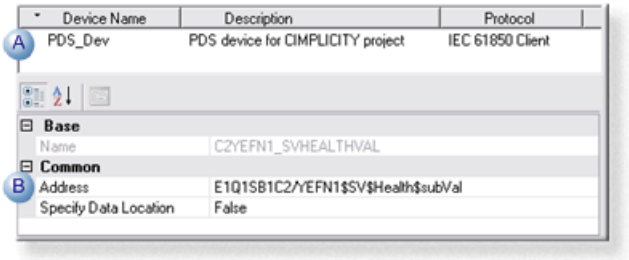
Example 2. IEC 61850 protocol

A Driver Server device uses an `IEC 61850` protocol.

The IEC 61850 entry in the Configuration dialog box **Address** field defines the levels on the device down to the level where the value will be found, including a:

1. Logical device
2. Device node
3. Structured attribute down to the bottom (leaf) attribute that contains the point value.

   Fields in the Configuration dialog box that define the address are as follows.

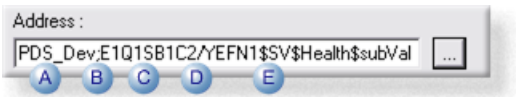


A	Device Name	A device with an IEC 61850 Client protocol is selected in the Configuration dialog box.
B	Address	The address corresponds to a data attribute that is entered in the Driver Configuration window. Note: If the attribute is not included in an IEC 61850 predefined data block, CIMPLICITY will include it in a dynamic data block that exist when the CIMPLICITY project is running.

When the Configuration dialog box is closed, the **Address** field displays the following string.

PDS_Dev;EQQ1SB1C2/YEFN1SVHealth$subVal

The elements in the string are:



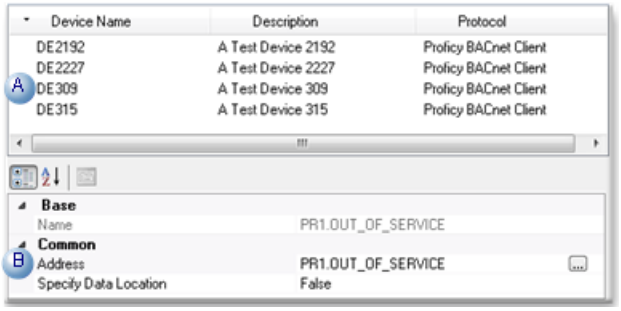
	Entry	Description
A	PDS_Dev	Device name
B	;	Syntax separator
C	EQQ1SB1C2	Logical device
D	/	Syntax separator
E	YEFN1	Logical node
F	SVHealth$subVal	A structured attribute whose last (leaf) attribute contains the point data.

Review Driver Server and IEC 61850 protocol documentation for detailed information about all of the fields in the Configuration dialog box.

Example 3: BACnet Protocol Addressing

A Driver Server device uses a `BACnet` protocol.

The BACnet entry in the Configuration dialog box **Address** field defines the levels on the device down to the level where the value will be



A	Device Name	The actual BACnet device name, configured on the device itself.	
B	Address	`ObjectIdXXXX.PropertyID` Where `OBJECTIDXXX` includes Object ID entries, which include the Object Type+Object Instance.	
		Object type:	A particular object type.
		Object instance:	Which instance of the specified type is being referenced.
		`PropertyID` includes the PLC Variable+an Associated Property.	

When the Configuration dialog box is closed, the **Address** field displays the following string.

`DE309;PR1.OUT_OF_SERVICE`

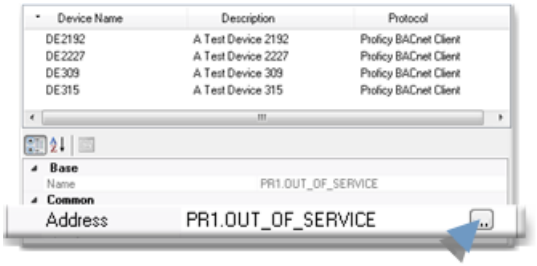
	Entry	Description

A	DE309	Device name
B	;	Syntax separator
C	PR1	PLC variable
D	OUT_OF_SERVICE	Property associated with the selected PLC variable.

Review Driver Server and BACnet protocol documentation for detailed information about all of the fields in the Configuration dialog box.

Point Browser

Click a button to the right of a selected **Address** field in the Configuration dialog box.



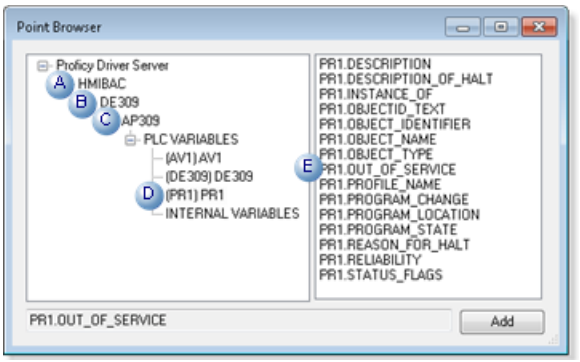
A Point Browser opens.

Important: The **Scan Variables** property for selected device's access path must be set to True to enable the Point Browser to scan for variables.

Example

A Point Browser for a BACnet device opens.

The hierarchy to make the property selection is as follows.



	Entry	Description
A	HMIBAC	CIMPLICITY project name.
B	DE309	Selected BACnet device.

C	AP309	Access path for selected BACnet device.
D	(PR1)PR1	Selected PLC variable.
E	PR1.OUT_OF_SERVICE	Property that will be used in the point's **Address** field.

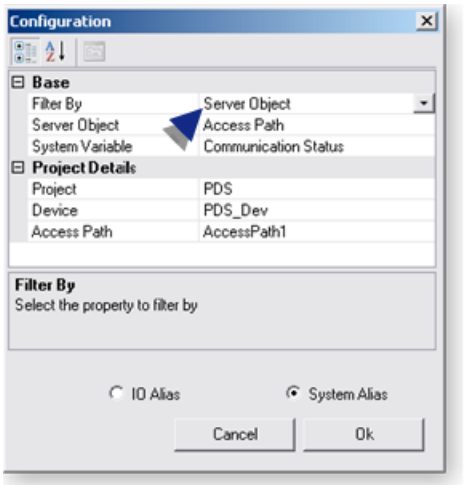
System Alias: Diagnostic Point Data

- The System Alias view of the Configuration dialog box provides the same options as the Configuration dialog box for System Aliases that can be added to the Driver Server in the Driver Configuration window.
- Variable values are also available on the Statistics tab for a Driver Server project that is opened in the Driver Configuration window.
- The Statistics tab displays when the Driver Configuration window is opened separately from the CIMPLICITY project.

- An example of an address that CIMPLICITY creates based on selections in the Configuration dialog box is, as follows.

<SystemItem Project="PDS" Device="PDS_Dev" AccessPath="AccessPath1" id="CommunicationState" />

- Diagnostic data can be collected for either server objects or system variables.
- The available options depend on the selected filter and your system configuration.
- Filter options are as follows.



Filter options are:

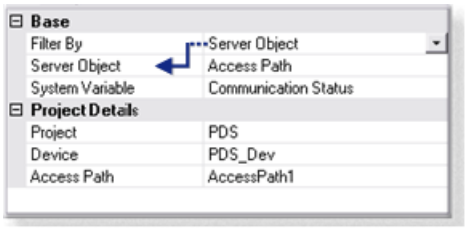
- Server object
- System variable

The option that is:

- Selected provides a.dropdown list in the dialog box second row

Example

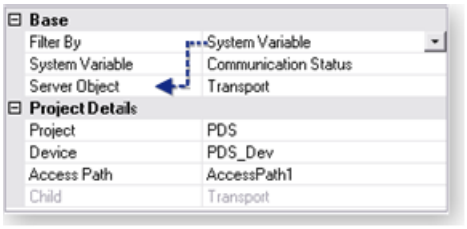
Server object is the filter



- Not selected provides dropdown list in the dialog box third row.

Example

Server object is not the selected filter



D	Diagnostic

Check Diagnostic if the point should be a diagnostic data point.

Important: If Diagnostic is checked, the address must match one of the following addresses.

Important: If you use Poll after Set with the Driver Server device communications, you may decrease performance. Using **Poll after Set** with OPC device communications is not recommended.

The following three point addresses can be used in conjunction with the **Diagnostic** check box on the Device tab of the Point Properties dialog box (in addition to the ones selected in the Enable Tracing section such as **!OPC_TRACE_ALL**). The point ID can be anything that makes sense. The point address should be the values provided below, and the **Diagnostic** checkbox must be checked. These points may be configured for each CIMPLICITY device configured.

$OPC_SERVER_STATUS	SINT or USINT	
	Read Only	
	Will be one of the following values:	
	OPC_STATUS_RUNNING OPC_STATUS_FAILED OPC_STATUS_NOCONFIG OPC_STATUS_SUSPENDED OPC_STATUS_TEST	= 1 = 2 = 3 = 4 = 5
$OPC_FORCE_REFRESH_NOW	BOOL	
	Read / Write	
	Writing a non-zero value to this diagnostic point instructs CIMPLICITY to call the Refresh method for all groups in the Driver Server. This instructs the server to send all point values (changed or not). The value displayed will: • Not reflect the written value • Always display zero. An entry is written to the status log indicating that the command was received.	
$OPC_FORCE_DISCONNECT_NOW	BOOL	
	Read / Write	
	Writing a non-zero value to this diagnostic point instructs CIMPLICITY to abort the connection with the Driver Server then re-establish it. **Important:** Use of the $OPC_FORCE_DISCONNECT_NOW diagnostic is not recommended! Normally the Driver Server does not need it. Using this diagnostic can leave the Driver Server in a bad state. After using this diagnostic, it is likely that the Driver Server will not shut down properly once the project is stopped.	

Guidelines for Configuring Group Addresses

guide: Guidelines for configuring group addresses:

1. You can specify a group from the **OPC group** list.

   If you do not specify a group, CIMPLICITY puts the point into one of two default groups, based on the point's **Update Criteria**.

Default Group	For points with Update Criteria that is:
DEFAULTPOLL	Polled
DEFAULTUNSO	Unsolicited

   NOTE: If all of the points in a group have polled update criteria, CIMPLICITY treats the group the same way it treats DEFAULTPOLL. As soon as one point has unsolicited update criteria, CIMPLICITY treats the group the same as it treats DEFAULTUNSO.

2. A group name can be up to 16 characters.

3. CIMPLICITY supports up to 510 groups. Two groups are defined by default, leaving 508 possible user defined groups.

   In CIMPLICITY, the maximum number of points in a group depends on the size of the points (in bytes). The total size of all points in a group cannot exceed 32 kilobytes. For example, a DINT point 4-bytes in length; an INT point is 2 bytes in length; string points are whatever length they are configured to be in bytes.

   Important: Configuration restrictions may exceed practical limits for project implementation.

Optional Debug Tracing

Optional Debug Tracing

CIMPLICITY includes the ability to enable tracing of communications between CIMPLICITY and the Driver Server in order to help diagnose communication problems between the Driver Server and OPC Server. Tracing is managed by a number of parameters and device diagnostic points. By default, if no trace parameters are specified and no diagnostic points are configured, only the default parameter values will be written to the trace file. The trace log is located in the LOG directory of your project.

Important: Enabling tracing beyond the defaults will reduce the performance of your Communication and your PC. Tracing should only be enabled for the purpose of diagnosing an issue, should one arise.

Port Parameters that manage Trace output

Parameter	Description
CircularLog	Trace output can be written to either a circular log file (1) or to a flat file (0). If CircularLog=1, trace output goes to the circular log. The circular trace log is named according to the CIMPLICITY Port it represents. MASTER_POS_0.LOG for the first port, MASTER_POS_1.LOG for the next port, etc. If CircularLog=0 (flat file), the file is also named according to the CIMPLICITY Port, but the convention is slightly different: POS_0.OUT for the first port, POS_1.OUT for the next port and so on. The advantage of the flat file is that, since it never wraps around, no trace data will be overwritten (nothing is lost). This is only an issue when a large amount of trace data is written over an extended period of time. As you will see, output to a flat file is seldom necessary to diagnose an issue. Flat files can grow very large, very quickly. You must monitor their size closely. If the flat file is used, the LogFileSize parameter is ignored.
LogFileSize	This property determines the maximum size the circular log can grow to before it wraps around and begins writing to the top of the file again. The default size is one megabyte (1000000). If you find that the circular log wraps too soon and begins overwriting old trace records you feel may be important to the diagnosis of a problem, increase this property to prolong the wrap.

Enable Tracing

CIMPLICITY performs various activities while it is running. It may be polling the Driver Server or writing a value to a point in the Driver Server. It pings the Server. At startup, CIMPLICITY creates Groups and Items in the Driver Server. It handles point update events being fired by the Driver Server.

You decide which of these activities you want to see trace data for, and you decide that on a device-by-device basis. You can set any of the trace properties for every CIMPLICITY device you configure. These are TRUE or FALSE (1 or 0) properties.

These properties can be manipulated at run time by defining device diagnostic points of type BOOL using the point addresses specified in the following table. Remember to check the Diagnostic option next to the **Point Address** when configuring these points in the Workbench. This allows you to modify the activities that are traced, while the project is running.

With some creative engineering, these device diagnostic points can be very powerful. You could start and stop tracing based on a CIMPLICITY event manager script, or with a button on a CimView screen.

Trace Activity properties and corresponding diagnostic point addresses:

Device Parameter	Diagnostic Address	Type	Trace Activity Related to:
TraceAll	**!OPC_TRACE_ALL**	BOOLEAN	All activities.
TraceConnection	**!OPC_TRACE_CONNECTION**	BOOLEAN	'Connecting to' and 'disconnecting from' the Driver Server.
TraceGroupActivity	**! OPC_TRACE_GROUP_ACTIVITY**	BOOLEAN	Groups; adding them to the server, calling Refresh and so on.
TraceItemActivity	**!OPC_TRACE_ITEM_ACTIVITY**	BOOLEAN	Driver Server Items; adding them to the group, removing them from the group, adding them to the Driver Server and so on.
TracePinging	**!OPC_TRACE_PINGING**	BOOLEAN	Pinging the Driver Server. Did it succeed, did it fail, how long did it take and so on.
TracePolling	**!OPC_TRACE_POLLING**	BOOLEAN	Polling the Driver Server for point values (calling the SyncIO.Read method). Successfully retrieved values are added to the point update queue.
TraceEvent	**!OPC_TRACE_EVENTS**	BOOLEAN	Point update 'events' 'fired' by the Driver Server. When these events occur, CIMPLICITY puts the new value into the point update queue, which is then emptied as the new values are passed into CIMPLICITY.
TraceWriting	**!OPC_TRACE_WRITING**	BOOLEAN	Writing a new value to a point in the Driver Server from CIMPLICITY.

TraceDequeue	**!OPC_TRACE_DEQUEUE**	BOOLEAN	Processing the point update queue.

Every CIMPLICITY Device you configure can be set to output its trace records to its own log file with the TraceSeparate property.

Trace Separate Property

Value	Description
TraceSeparate	TraceSeparate is TRUE or FALSE (1 or 0). The default value is FALSE. If one of your devices contains TraceSeparate=0, this device will output its trace records to the CIMPLICITY Port's circular log file (MASTER_POS_x.LOG). Otherwise, this device will output its trace records to a file of its own, named with the device ID as follows: MASTER_POS_x_<YourDeviceID>.LOG In this way, trace data for each configured CIMPLICITY Device is logged separately. This will aid in isolating an issue. Turning TraceSeparate off for one or more devices is useful when trying to diagnose a timing related issue, or some interaction between devices.

Every CIMPLICITY Device you configure contains one more, very important, trace property: TraceLevel.

TraceLevel can be an integer value between one (1) and seven (7). Think of the TraceLevel as the 'volume knob' for tracing. It determines the granularity, or how detailed the trace records will be. These seven values are defined as follows:

Trace Level granularity settings:

Value	Name	Description
1	**BIGERRORS**	This is the default value for TraceLevel. At this level, only big 'showstopper' type errors that occur during the activities you have enabled will be traced.
2	**ALLERRORS**	At this level, all errors, major and minor, are sent to the Trace file. This is perhaps the best setting to use for the diagnosis of an issue. Only error conditions will be traced, so you will not have to sift through a growing list of successful activities when searching for the cause of an issue.
3	**BIGSUCCESSES**	At this level, in addition to all error conditions, big successes like 'poll successful', 'write successful', and 'success connecting to Driver Server' are also included. This level can be very useful when you need to see an error condition within the context of what else was going on at the time. Such as, what was the last successful operation before the error, or did the next attempt succeed.
4	**ALLSUCCESSES**	At this level, all errors and all successes are written to the trace log. Trace Levels of four or higher will cause the log to fill up quickly. In addition, the log will fill with successful operations, making it harder to diagnose problems. This setting is useful when trying to isolate a timing issue or an abnormal termination.
5	**ITEMDETAILS**	This Trace Level is like `ALLSUCCESSES`. In addition, details about every point are included. You can use this level to see point updates, complete with value, as they arrive from the Driver Server. **Note:** There are other ways to get detailed point information without setting the Trace Level this high. See the Other Trace related diagnostic addresses *(page 430)* section.
6	**DEVELOPER Level 1**	Trace statements at this level are intended for the CIMPLICITY development team. They are not likely to be useful unless you have an in-depth knowledge of the CIMPLICITY source code.

7	**DEVELOPER Level 2**	**Important:** Do not use either of these 'Development' Trace Levels unless instructed to do so by CIMPLICITY support. These two Trace Levels provide an extreme amount of detail that will fill the log file very quickly.

Note:

- You should never need a Trace Level higher than three (BIGSUCCESSES) to diagnose an issue. It may be feasible to leave minimal tracing enabled, even in production, without severely affecting performance. With TraceAll=1 and TraceLevel=1 or 2, error conditions will be traced as they happen, and only error conditions will be traced. If an issue arises only after the project has been running for some time, this is the way to trap it. While everything is running fine, the log will remain empty. As errors occur, they will be written to the log.
- A device diagnostic point can be configured so that the Trace Level can be adjusted while the project is running. The address for this point is: !OPC_TRACE_LEVEL, and the point must be configured as an analog point of type SINT or USINT.
- Three more device diagnostic point addresses ($OPC_FLUSH_TRACE_NOW, $OPC_DUMP_ALL_POINTS, and $OPC_DUMP_BAD_POINTS) provide useful trace functionality. When configuring CIMPLICITY points with these addresses, the point type must be of type BOOL. Writing a value of one (1) will cause the action to occur. A status log message will indicate that the command was executed.

Other Trace related diagnostic point addresses

Point Address	Description
$OPC_FLUSH_TRACE_NOW	When trace records are written to the log, they are not expressly flushed to the file on disk. Instead, the operating system is in charge of this. If you want to view the trace log while the project is still running, you can write a one (1) to this point. This will cause all pending trace records to be written to disk.
$OPC_DUMP_ALL_POINTS	Writing a one (1) to this point will cause CIMPLICITY to immediately dump the status of every point in the device to the trace log. Each trace record contains detailed information about the point. This option will allow you to immediately know the answer to the two most popular support questions we get concerning the CIMPLICITY points communicating to the Driver Server: • "Why is that point unavailable?", and • "Why hasn't that point updated recently?" This option makes it generally unnecessary to set the Trace Level above two (2). The "ITEMDETAILS" level is just not needed since you can dump point information on demand (as you need).
$OPC_DUMP_BAD_POINTS	This diagnostic point performs the same operation as the one above, but it only dumps points marked as unavailable. It will allow you to answer the first of the two big questions above without having to sift through points that are updating fine.

Viewing Trace Information

Whether using the circular log or the flat file, the format of the trace record is the same. Each trace record contains several fields of data, separated by the vertical bar character (|). This design was adopted because the trace log can be easily imported into most database and spreadsheet programs. This allows you to employ the powerful features of these programs to help view the trace data, and diagnose any issue more effectively.

Depending upon the context of the trace record, some fields may be empty. If the message does not pertain to a certain point, then Point ID, value, address, etc., will be empty. In general, the first four fields, and the last field will always contain data. The **Result Code** field will often contain zero (0x00000000), which means either success, or there was no **Result Code** to report.

Trace Record Fields

The fields contained in every trace record are as follows (in the order they appear in the trace record, from left to right):

Field name		Description
Time Stamp	The time stamp when the record was logged to the file in the following format: yyyymmdd.hhmmssttt **Note:** This format represents times to a precision of one ten thousandth of a second.	
Device ID	If the trace record was written by the Port, this will contain PortLevel, otherwise it will contain the CIMPLICITY Device ID of the device that generated the trace record.	
Thread ID	This contains the name of the thread that generated the trace record. CIMPLICITY communications with the Driver Server have four threads of execution.	
	TOOLKIT	CIMPLICITY devcom toolkit thread.
	PING	Ping thread owned by device object.
	WATCHER	Watcher thread owned by device object.
	SERVER	OPC Server's thread (when events are fired).
Message	The message to you. Many of these have additional data embedded in them. Some are very verbose. This message can be up to 512 bytes in length.	
Result (error) Code	If the trace record represents an error, and an error code was returned to CIMPLICITY from the Driver Server, this field will hold that error code. If the OS has a text description of the error code, that will display as well. If there is no error code, this field will be zero (0x00000000), and the description will indicate success.	
Group ID	Each CIMPLICITY Device contains at least two groups (DEFAULTPOLL and DEFAULTUNSO). There will be more if they were defined in the project. If the trace record is associated with a group, this field will contain the group's name.	
Point ID	If the trace record is associated with a certain point, this field will contain the point's ID (name).	
Point Address	If the trace record is associated with a certain point, this field will contain the point's address.	
Point Value	If the trace record is associated with a certain point, this field will contain the point's value and variant type.	

Point Quality	If the trace record is associated with a certain point, this field will contain the point's quality.
Trace Message ID	Each trace record in the Driver Server has a unique ID. This will be used for future enhancements.

Driver Server Technical Notes

Technical Notes Overview

- Supported CIMPLICITY features.
- Supported Driver Server types.
- Driver Server Technical Notes.
- Driver Server vs. Triplex Driver.

Supported CIMPLICITY Features

The Driver Server supports the following CIMPLICITY features:

- Collection of unsolicited data.
- Poll after setpoint.
- Triggered reads.
- Analog Deadband via CIMPLICITY filtering

This enabler supports the following data types:

- Boolean.
- Signed 8, 16, and 32 bit integers.
- Unsigned 8, 16, and 32 bit integers.
- 4 and 8 byte REALs.
- Floating point.
- Text.

Supported Driver Server Types

The following types are supported by this communications interface:

Analog

3D_BCD	Positive BCD values ranging from 0 to 999.

4D_BCD	Positive BCD values ranging from 0 to 9999.
DINT	Integers ranging from -2,147,483,648 to + 2,147,483,647.
INT	Integers ranging from -32,768 to +32,767.
REAL	Floating-point numbers.
SINT	Integers ranging from -128 to +127
UDINT	Unsigned integers ranging from 0 to 4,294,967,295.
UINT	Unsigned integers ranging from 0 to 65,535.
USINT	Unsigned integers ranging from 0 to 255.

Boolean

BOOL	A one digit Boolean point with a value of 0 or 1.
BYTE	An 8-bit array of Boolean points.
WORD	A 16-bit array of Boolean points.
DWORD	A 32-bit array of Boolean points.

Text

STRING	A one character alphanumeric.
STRING_20	A 20 character alphanumeric string.
STRING_8	An 8 character alphanumeric string.
STRING_80	An 80 character alphanumeric string.

Driver Server supports all CIMPLICITY data types.

Points may be single elements or array points.

Note:

- Review Data Access 3 specification documentation to determine if Data Access 3 provides for passing arrays of data. Data Access 2.0 specification does not provide for passing arrays of data. It does not define how an OPC Server should pass array data to a Driver Server.
- Some OPC Servers provide a way of defining Visual Basic (VB) style arrays (usually by appending an element count onto the Item ID). How arrays are supported, if they are supported, is specific to the OPC server. Consult your OPC Server vendor for details.

If an OPC Server does support VB style arrays, they can be configured in the CIMPLICITY by simply setting the number of elements under Point Properties to match the number of elements defined in the Item ID (the CIMPLICITY point address).

Note that the number of elements must match.

Example

If the array is defined by the syntax MyItem_20, indicating a twenty-element array, then the **Elements** property of the Point Properties in CIMPLICITY must also be set to 20 or the array will not be successfully retrieved from the OPC Server.

OLE passes the array within a single Variant value. Therefore, it is not possible to read or write a single element. The entire array must be passed when reading from the OPC Server or writing to the OPC Server.

Driver Server Technical Notes

P roficy Driver Configuration restricted to user who installed CIMPLICITY

Driver Configuration does not appear in the Start Menu of other users.

The Server Configuration button appears on the CIMPLICITY Device properties page, but does not invoke configuration tool.

Rename/Copy a CIMPLICITY Project

You must manually rename data\[OldProjectName].xml to [NewProjectName].xml and then resolve the internal project name.

Deleting the CIMPLICITY Driver Server Points/Device/Port

If you delete the points, device, or ports, and then uncheck the protocol, the data\[ProjectName].xml remains, and will be reused if you add the Driver Server again.

Driver Configuration is Not Always Disabled

If the **CIMPLICITY Project is enabled for Change Management, t**he Driver Configuration will be disabled when following checkout rules, but the Driver Configuration can still be launched from the Start Menu.

Licensing

If Driver Server is not licensed, the CIMPLICITY points will be unavailable, but ProficyDrivers.exe remains running. There is no device down alarm or other alarm.

When a CIMPLICITY Project or the Driver Server Shuts Down

1. Normally, a CIMPLICITY project shutdown deletes the corresponding project in the Driver Server. If there are two CIMPLICITY ports for the Driver Server, and one is shutdown via

Process Control, the remaining port will have unavailable points until the first is restarted and reloads the project.

2. Abnormally, CIMPLICITY may not delete the project in the Driver Server. The Driver Server will continue to poll the target PLCs.

Shared File for User Settings

Driver Server user settings are stored in a single .xml file for each project and are shared by all projects on the computer. There are no individual settings for users.

Only one Driver Configuration per project is allowed on a single computer.

Bad Address Configuration

CIMPLICITY does not detect the configuration of a bad address, so no error will be reported.

Diagnostic Points

The following three point addresses can be used in conjunction with the **Diagnostic** check box on the Device tab of the Point Properties dialog box (in addition to the ones selected in the Enable Tracing section such as **!OPC_TRACE_ALL**). The point ID can be anything that makes sense. The point address should be the values provided below, and the **Diagnostic** checkbox must be checked. These points may be configured for each CIMPLICITY device configured.

$OPC_SERVER_STATUS	**Read Only** Must be configured as SINT or USINT	Will be one of the following values:	
		OPC_STATUS_RUNNING OPC_STATUS_FAILED OPC_STATUS_NOCONFIG OPC_STATUS_SUSPENDED OPC_STATUS_TEST	= 1 = 2 = 3 **=** 4 = 5
$OPC_FORCE_REFRESH_NOW	**Read / Write** Must be configured as BOOL	Writing a non-zero value to this diagnostic point instructs CIMPLICITY to call the Refresh method for all groups in the Driver Server. This instructs the server to send all point values (changed or not). The value displayed will: • Not reflect the written value • Always display zero. An entry is written to the status log indicating that the command was received.	

$OPC_FORCE_DISCONNECT_NOW	**Read / Write** Must be configured as BOOL	Writing a non-zero value to this diagnostic point instructs CIMPLICITY to abort the connection with the Driver Server then re-establish it. **Important:** Use of the $OPC_FORCE_DISCONNECT_NOW diagnostic is not recommended! Normally the Driver Server does not need it. Using this diagnostic can leave the Driver Server in a bad state. After using this diagnostic, it is likely that the Driver Server will not shut down properly once the project is stopped.

Driver Server vs. Triplex Driver

1. No alarm is generated if a target PLC is down. You will only get an alarm if the Driver Server itself shuts down.
2. With Driver Server, CimView and other viewer applications cannot perform a setpoint on an array element. The Point Control Panel can be used to set the entire array.
3. CIMPLICITY can configure Driver Server “system aliases” as points to get information similar to the standard CIMPLICITY “diagnostic” points. System aliases can also be used to reset statistics and toggle Driver Server logging.

Chapter 22. DDE Client Communications

About DDE Client Communications

⚠ **Important:** Net DDE Communications is obsolete, starting CIMPLICITY 7.5. DDE client Communications (Local only) is an As-Is product.

The CIMPLICITY DDE Client Communication enabler acts as a DDE Client and is able to communicate with standard DDE Servers. A DDE Server uses a three-level hierarchy:

- Application (or Service) Name
- Topic Name
- Item Name

to uniquely identify a unit of data the Server can exchange during a conversation with a Client. Conforming to this hierarchy:

- A CIMPLICITY device is associated with the Application and the Topic Name.
- A CIMPLICITY point is associated with an Item Name.

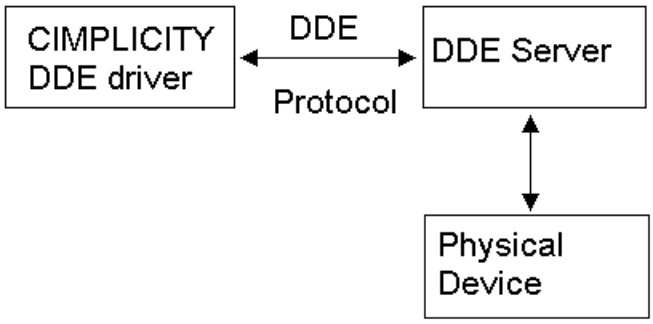
Typically, you must first configure a Topic through a configuration tool that is specific to the type of Server being used (usually the Topic identifies the physical address of the device). You then use that Topic Name, along with the Application Name, to configure the CIMPLCITY Device (see the DDE Device Properties screen below). The CIMPLICITY Device Point Address is the Item Name whose format is dictated by the specific Server being used.

The DDE Client Communications enabler is able to communicate with multiple DDE Servers at one time. Each server is defined as a separate CIMPLICITY device.

The DDE Client Communications enabler provides the following features:

- Communication with multiple DDE Servers at the same time.
- Access to multiple Topics under a single Server at the same time.
- Communication with DDE or NetDDE Servers.
- Alarm generation when a communication failure occurs.
- Read and write accessible Items.
- A utility program, **dde_diag.exe**, which can be used to validate a DDE Server's Service, Topic and Item names.

Sample DDE Network



CIMPLICITY DDE Client Supported Protocols

The DDE Client Communications enabler supports the following protocols:

- DDE
- NetDDE
- AdvancedDDE™

DDE Required Hardware and Software

There is no additional hardware or software required for the DDE Client Communications enabler other than what might be required for the specific DDE Server you are using.

DDE Setup

1. From the Start menu, select Settings.
2. From the Settings menu, select Control Panel.
3. In the Control Panel program group, click **Services**.
4. In the Services dialog box, select the CIMPLICITY service.

5. Select **Startup...**

6. In the Service dialog box:
    a. Set the **Log On As** field to **system account**.
    b. Enable the **Allow Service to Interact with Desktop** option.
    c. Click **OK** to close the Service dialog box and save your changes.

7. Click **Close** to close the Services dialog box.

8. Exit the Control Panel program group.

Note: When using a Visual Basic server to gather data for a CIMPLICITY project, it must have the ability to support more than one topic. It has been observed that Visual Basic has the capacity to support 128 items per topic.

NetDDE Setup

1. From the Start menu, select Settings.

2. From the Settings menu, select Control Panel.

3. In the Control Panel program group, click Services.

4. In the Services dialog box, select the CIMPLICITY service.

5. Select **Startup...**

6. In the Service dialog box:
    a. Set the **Log On As** field to **system account**.
    b. Enable the **Allow Service to Interact with Desktop** option.
    c. Click **OK** to close the Service dialog box and save your changes.

7. On the computer where the DDE Server is running, run **DDESHARE.EXE** (usually found in **\winnt\system32\ddeshare.exe**).
    a. Select DDE Shares from the Shares menu.
    b. Add a new Share by creating a share name.
    c. Enter the DDE Server's Application Name and Topic into the **Old style**, **New style** and **Static** fields.
    d. Enable the **Allow start application** and **Is Service** option.
    e. Select **Permissions….**
    f. Select **Everyone.**
    g. Set **Type of Access** to **Full Control**.
    h. Click **OK** to exit the permission and DDE Share Properties dialog box.

Important: When configuring NETDDE, the CIMPLICITY Service should be configured to log on as a user with a non-null password rather than the system account. The computer must be configured so that the user selected has full access to the DDE share.

DDE Diagnostic Utility

The **dde_diag.exe** program, located in the CIMPLICITY **\exe** directory, is provided to assist you in validating communication to a specific DDE Service, Topic and Item name. The Service and Topic name are used in the CIMPLICITY device configuration and the Item name is used in the CIMPLICITY point configuration.

This program is invoked by using one of the following commands:

- For read tests

dde_diag.exe < service> <topic> <item>

- For write tests

dde_diag.exe < service> <topic> <item> <type> <value>

- For HotLink tests

dde_diag.exe < service> <topic> **<item> OnChange**

Where :

< service> is the Application (or Service) name of the DDE Server.

< topic> is a valid Topic for the DDE Server.

< item> is a valid Item for the DDE Server (format is Server specific).

< type> is one of the following data types:

C-Character

S-Short

L-Long

F-Floating

T-Text

< value> is a valid value to be written to the item.

Note: HotLink means a Hot Link will be created for the given item. Once the link is established, the value of the item is updated automatically by the DDE Server.

Upon successful connection to the DDE Server for the given Topic, this test utility reads and prints the value of the given Item. An appropriate error message is printed if the utility fails to connect to the DDE server or the Item is invalid.

When validating communication over NetDDE, the Service is:

```
\\<nodename>\ndde$
```

Where:

< nodename> is the computer name where the DDE Server is running.

The Topic is the Share Name you created using the DDE share utility.

CIMPLICITY Configuration for DDE

CIMPLICITY Configuration for DDE

As with other CIMPLICITY Communication enablers, you must complete the Port, Device, and Device Point configuration to set up the DDE Client Communications enabler.

DDE Port Configuration

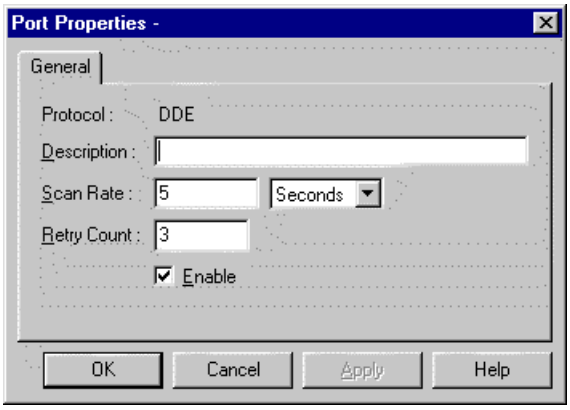
DDE Port Configuration

When you configure a new port for DDE Client communications, enter the following information in the New Port dialog box:

1. In the **Protocol** field, select **DDE** from the list of protocols.
2. In the **Port** field, select the communication port that will be used for DDE Client communications.

   When you click **OK** to create the port, the Port Properties dialog box for the protocol opens.

General Port Properties



Use the General tab in the Port Properties dialog box to enter general information for the port. You can define the following:

Description	Enter an optional description to help you identify the port.
Scan Rate	Enter the base scan rate for the port. Point scan rates will be multiples of the base rate. You can specify a scan rate in Ticks (100 ticks = 1 second), Seconds, Minutes, or Hours.
Retry Count	If communications cannot be established for a device on the port, the device is considered to be down, and a $DEVICE_DOWN alarm is generated. Once a device is down, periodic attempts are made to resume connections to it.
	Enter the number of scans to wait before attempting to reconnect to a device after a communications error has been detected.
Enable	Set this check box if you want the port to be enabled when the project starts. If you clear this check box, the port will not be enabled when the project starts, and points will not be available for devices on the port.

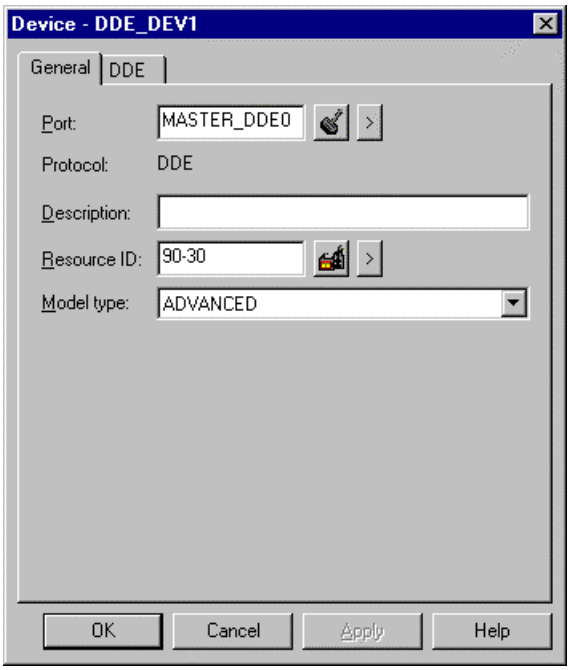
DDE Device Configuration

DDE Device Configuration

1. In the **Device** field, enter the name of the device you are configuring.
2. In the **Port** field, select the DDE port to be used by the device.

   When you click **OK** to create the port, the Device Properties dialog box opens.

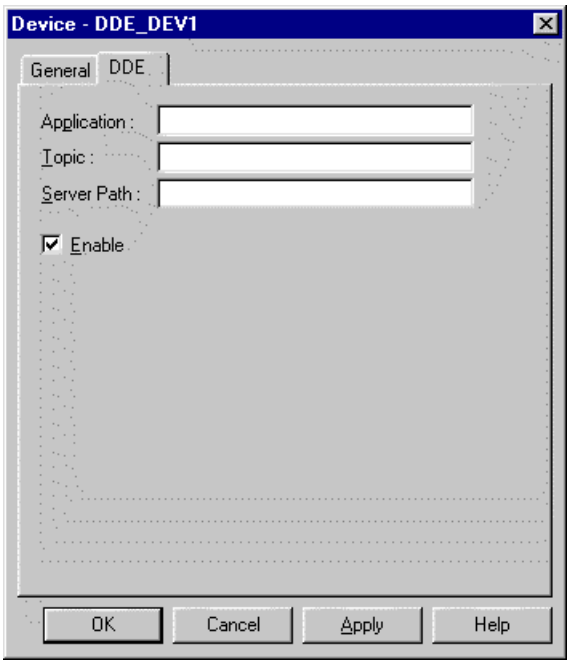
General Device Properties



Use the General tab in the Device dialog box to enter general information for the device. You can define the following:

Port	Select the port for this device. • You can click the Browser button to the right of the field to display the list of ports and select one. • You can click the Pop-up Menu button to create a new port, edit the current port, or select a port from the list of ports.
Description	Enter an optional description to help you identify the device.
Resource	Enter a resource that can be associated with this device for alarm generation. • You can click the Browser button to the right of the field to display the list of resources and select one. • You can click the Pop-up Menu button to create a new resource, edit the current resource, or select a resource from the list of resources.
Model Type	Select the type of device. Click the drop-down button to the right of the input field to display your choices, then make a selection. The choices are: ADVANCED DDE Client communications only communicates with the DDE Server using the AdvancedDDE format. **DEFAULT** DDE Client communications negotiates with the DDE Server to decide the format of the communication. **TEXT** DDE Client communications only communicates with the DDE Server using the TEXT format.

DDE Device Properties



When you configure a physical device for the DDE Client Communications enabler, use the DDE properties to enter information about DDE communications for the device. You can define the following:

Application	Enter the Application name: • For DDE communications, enter the Service Name of the DDE server. • For NetDDE communications, enter **\\Nodename\ndde$**, where **Nodename** is the computer name where the DDE Server is running.
Topic	Enter the Topic name: • For DDE communications, enter the Topic Name that is configured for the specific DDE Server you are using. • For NetDDE communication, enter the **DDE share name** created within the **ddeshare** utility.
Server Path	Enter the full server executable name if the server should be started by the CIMPLICITY DDE Client Communications enabler. • If you provide the executable name of the Server, the Enabler will start the server before attempting to connect to the Server. • If you do not provide the Server's executable path, you must start the Server independently and ensure the CIMPLICITY project and the Server are running under the same USER_ID.
Enable	Set this check box if you want the device to be enabled when the project starts. If you clear this check box, the device will not be enabled when the project starts, and points will not be available for this device.

DDE Point Configuration

DDE Point Configuration

You can configure points that correspond to values in a physical device. The DDE Client communications enabler establishes a Hot Link to the DDE Server for all DDE points in your project configuration. For Hot Link Items, the Server provides an initial value, and then provides an update whenever the Item changes. However, the data is passed back to the Point Manager based on the update criteria you select for your DDE points.

When you define a point, the fields in the Point Properties dialog box have values that are unique to or have special meaning for DDE communications.

General Point Properties

On the General tab in the Point Properties dialog box, you may configure points for **Read** or **Read/ Write** access.

Device Point Properties

On the Device tab in the Point Properties dialog box:

Update Criteria:	The update criteria determines how the data will be requested. Click the drop-down button to the right of the input field to display your choices, then make a selection.
	For device points–Select **Unsolicited** for the DDE Client communications to send data it receives to Point Management whether or not the point's value has changed..
	For device points–Select **Unsolicited On Change** for the DDE Client communications to send data it receives to Point Management only if the point's value has changed.
	For diagnostic points–Select **On Change** to have the DDE Client Communications poll the values at regular intervals and update only changed values.
	For diagnostic points–Select **On Scan** to have the DDE Client communications poll and update the values at regular intervals, whether or not the values have changed.
	For diagnostic points–Select **On Demand On Change** for the DDE Client Communications to poll values whenever they are needed by an application and to update only changed values.
	For diagnostic points–Select On Demand On **Scan** for the DDE Client Communications to poll and update values when they are needed by an application whether or not the values have changed.
	For device points–Select **On Demand** Unsolicited.
	For device points–Select On Demand **Unsolicited On Change**. **Important:** Device points that are configured with an update criteria other than Unsolicited or Unsolicited On Change will be found to be invalid. When using point by address, the point must be configured for an unsolicited update criteria.
Address:	The general point address format is the format of the data item defined by the specific DDE server. See the DDE server documentation for the syntax of the data item.

Diagnostic Point Configuration

Diagnostic Point Configuration

For some DDE Servers, there is no mechanism or event to determine when a device is disconnected (down). For this type of DDE Server, you can configure a diagnostic (heartbeat) point for each physical device with which the DDE Server is communicating. The DDE Client Communications enabler can then read the point and set the device down if the read fails. The diagnostic point can be configured as any point type with the following address format:

```
$HP <DDE_item_address>
```

Where <DDE_item_address> is the point address to read from the DDE Server. The scan rate of the point will be used as the heartbeat interval for the device.

In addition, Device heartbeat points can be configured to specify that a device is available based on the value of a specified point. Previously, a diagnostic point with the address **$HP<**addr> would cause the device to be marked unavailable whenever the address **<addr>** could not be polled.

Using the new hearbeat point syntax, the device can be marked unavailable whenever the address <addr> does not have a specific value. For example, if the point **$HP_0** <addr> were configured, the device would only be available when **<addr>** could be read and contained the value **0**.

Procedure for Enabling the Connection Reset Feature

1. Create the following **glb_parm** entry.

   DDE_RESET_ON_WRITE_FAILURE|1|Yes

   When the devcom forces the connection to be closed (instead of the server), a default mechanism is needed to determine when to reconnect.

2. Configure a heartbeat point to enable the default mechanism capability.

   Configuring the heartbeat point will cause the connection to be automatically re-established.

3. Change the CPU_ID in the **device.idt** file as follows.

   bit 0 --> accept NULL value back from the DDE server

   bit 1 --> wait for 3 consecutive failure of heartbeat before marking device dead

   bit 3 --> For text heartbeat point we want to make sure that any Unsol. data coming after the Device Dead that has device dead string should not mark the device up.

bit 4 --> Use "advised heartbeat point"

DDE Communications Limitations

You need to be aware of the following limitations for the DDE Client Communications enabler:

The DDE Client Communications enabler does not support CIMPLICITY Array points (formats of array points vary between Servers), except for Text points.

DDE Global Parameter

DDE_UNAVAIL_ON_ILLEGAL_VAL

Purpose	To make a point go to an unavailable state if the data received from the DDE Server contains an invalid character static to the point's Data Type
Description	When communicating with a DDE server whose model type is TEXT, the DDE enabler will attempt to convert the values received based on the configured point's data type. This global parameter will make the point unavailable to the DDE enable.
Value	Y or N
Default	Y
Example	The global is defined and set to Y A point is an INT data type. The DDE Server sends a value of ABC. The point transitions to an unavailable state.

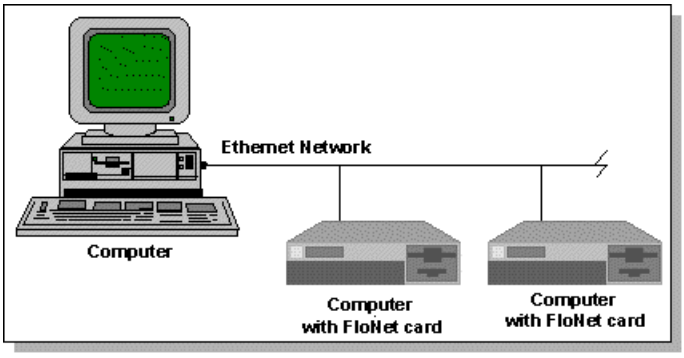
Chapter 23. FloPro/FloNet Ethernet Device Communications

About FloPro/FloNet Ethernet Device Communications

FloPro is a program that runs on IBM PC compatible computers and lets them perform functions similar to PLCs. FloPro can communicate with a host computer via a special interface card across 10/Base-T Ethernet physical interconnections. This card is referred to by its model number (IPN-200) or as a FloNet card. The physical network it communicates across may be referred to as the Ethernet or as FloNet.

The FloPro/FloNet Communications option and the FloNet cards communicate across a 10/Base-T Ethernet standard physical layer in loose compliance with the UDP/IP protocol standard. As noted in the FloNet Specification for IPN-200 FloNet Interface Board document, the FloNet card's flash ROM area holds all the firmware involved with network communications. FloPro itself merely polls the FloNet card to see if a command packet has arrived from a project on the CIMPLICITY host computer.

The physical communications link between a CIMPLICITY computer and the processors running FloPro is a 10 MHz Ethernet network with a twisted-pair connection to each FloNet card.



FloPro/FloNet Communications Features and Restrictions

The FloPro/FloNet Communications option provides the following features:

- Configuration functions for defining the devices accessed via the communication port and the data to be exchanged.
- Synchronous exchange of data between CIMPLICITY software and a variety of devices.

- A utility program, **flopro_diag**, which is used for validating the operation and communication ability of a FloNet device residing at one particular IP address.
- Since the FloPro/FloNet system accepts commands from any IP address and replies on a message-by-message basis, a CIMPLICITY computer may converse with many FloPro devices over one FloPro dedicated network.

You need to be aware of the following restrictions:

- Because FloNet cards occupy and converse only with sockets bound to privileged IP port number 165, only one FloPro/FloNet Communications process can run on a CIMPLICITY computer.
- Because FloNet cards cannot initiate communications with a remote host, the FloPro/FloNet device communications does not support unsolicited messages.
- Because FloPro/FloNet communications supports a subset of UDP, the FloPro card does not support the **ping** command.
- A dedicated network is required for communication between FloNet devices and their remote hosts.

FloPro/FloNet Communications Supported Devices

PCs equipped with FloNet cards running version 1.14 firmware, and FloPro version 2.774, including model/capacity variants FloPro/64, FloPro/256, FloPro/512, FloPro/1024, FloPro/4096 and FloPro/6144 are supported locally via an Ethernet link.

Note: The motion-control disabled variants have not been tested, but should operate identically, since the device communications module does not deal with the options of FloPro such as motion-control and RF scanner.

FloPro/FloNet Communications Supported Memory Types

For all FloPro versions, data can be read and written to the following data types:

Domain Name	Data Type	Direction (as seen by CIMPLICITY software)
Input	Digital	Read/Write
Output	Digital	Read/Write
Flag	Digital	Read/Write
Input Force	Digital	Read only
Output Force	Digital	Read only
Flag Force	Digital	Read only

Counter	16 bit signed	Read/Write
Register	4 digit unsigned BCD	Read/Write
Number	32 bit signed	Read/Write
ASCII	8 bit signed	Read/Write
Timer		
Limit	32 bit unsigned (10 ms increments)	Write only
Current value	32 bit unsigned (10 ms increments)	Read only

FloPro/FloNet Communications Related Documents

You should have the following documents on hand:

IPN-200 configuration Guide, Revision 1.0 - Diversified Technology, Inc.

FloNet Specification for IPN-200 FloNet Interface Board, Version 2.0 - Universal Automation, Inc.

Running the FloPro/FloNet Communications Test Program

1. Open the project's Workbench.
2. From the Tools menu, select Command Prompt....
3. In the Command Prompt window, type the following command:

   flopro_diag [<address>]

   Where **<address>** is an optional argument and is the IP address of the target PC. If you do not supply this argument, the IP address defaults to 3.26.5.159.

   When you run **flopro_diag** and select a nonexistent IP address or an IP address for a computer where FloPro is not configured, the output will show domain sizes of zero.

   When you run **flopro_diag** for a target computer where FloPro is working correctly, it produces output like the following:

```
Target is IP address 3.26.5.159
Test to run 1 times
The Sockets DLL version is 1.1
Examining domain limits database...
For FloPro server at IP addr 9f051a03,
```

```
limit[ 0]=4095    Input
limit[ 1]=4095    Input Force
limit[ 2]=4095    Flag
limit[ 3]=4095    Flag Force
limit[ 4]=1023    Timer
limit[ 5]=2047    Counter
limit[ 6]=1023    Register
limit[ 7]=4095    Output
limit[ 8]=4095    Output Force
limit[ 9]=2047    Number
limit[10]=2047    Ascii
```

CIMPLICITY Configuration for FloPro/FloNet

CIMPLICITY Configuration for FloPro/FloNet

When you are configuring ports, devices, and device points, you will be asked for some information that is specific to FloPro/FloNet communications.

FloPro/FloNet Port Configuration

FloPro/FloNet Port Configuration

1. Select the following in the New Port dialog box.

Field	Description
Protocol	Select FLOPRO from the list of available protocols.
Port	Select the communication port that will be used for FloPro/FloNet Ethernet device Communications.

2. Click OK.

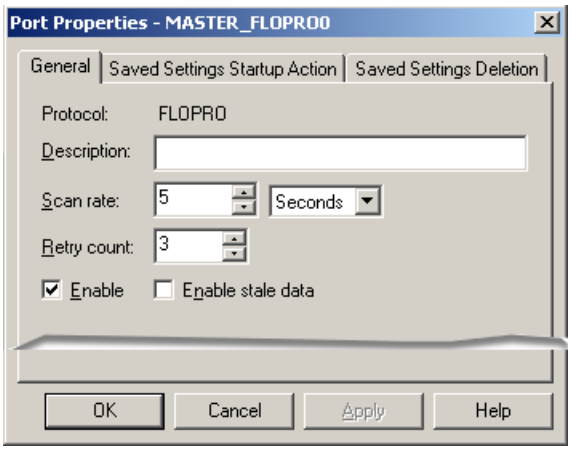
The Port Properties dialog box for FloPro/FloNet Communications opens.

Note: Only one FloPro port is supported on a CIMPLICITY computer.

General Port Properties

Select the General tab in the Port Properties dialog box.

Configuration includes the following.



Feature	Description
Description	(Optional) description to help you identify the port.
Scan Rate	Base scan rate for the port. Point scan rates will be multiples of the base rate. The rate can be the number of: • Ticks (100 ticks = 1 second) • Seconds • Minutes • Hours
Retry Count	Number of scans to wait before attempting to reconnect to a device on this port after a communications error is detected.
	If communications cannot be established to a device on this port: • The device is considered to be down • A $DEVICE_DOWN alarm is generated. Once a device is down, periodic attempts are made, after the **RetryCount**, to resume connection.
Enable	Check
	Clear
Enable stale data	Check
	Clear

Saved Settings Startup Action

FLOPRO Communications provides you with the option to reduce normal and/or recovery start up time by saving device characteristics for subsequent re-use.

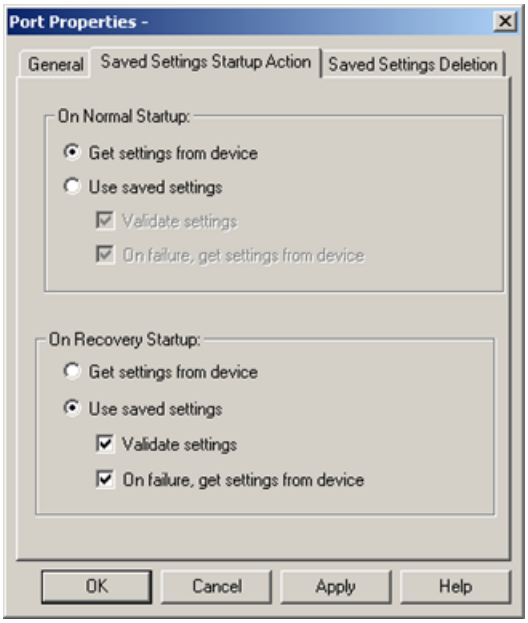
Note: Saved Settings Startup Action is intended for devices whose characteristics, such as the memory size, do not change. If the settings change, make sure they are deleted *(page 455)* .

Saved settings startup is supported only for devices where the following memory types have an element count that is evenly divisible by 8: Input, InputForce, Flag, FlagForce, Output, and OutputForce.

Select the Saved Settings Startup Action tab in the FloPro Port Properties dialog box.

Selections can be made for the following.

- Normal startup
- On Recovery Startup



rect 24, 61, 137, 77 *(page 453)*
rect 36, 84, 171, 100 *(page 454)*
rect 30, 221, 165, 237 *(page 454)*
rect 29, 106, 235, 172 *(page 454)*
rect 27, 240, 233, 306 *(page 454)*
rect 26, 201, 161, 217 *(page 455)*

Normal Startup

- Normal Startup occurs when CIMPLICITY starts.
- CIMPLICITY is started either when:

- The computer is booted up or
- A CIMPLICITY user starts the project.

Options for each of the startups are as follows.

Get settings from device

Defines the actions that the device communication interface takes to determine the supported memory types and ranges for a specific device.

The methods vary by device communication interface.

Use saved settings

Checked: The device communication interface will use the stored settings to define the device-specific memory types and ranges.

- The device configuration data is recorded and stored for later use.
- Options if Use saved settings is checked are:

Option	Description
Validate settings	Checked
	Clear
On failure, get settings from device	When using saved settings, failures may occur. Two typical failures are: • An integrity error is detected in the saved information. • There is a failure to verify a memory range when Validate settings is selected.
	Checked
	Clear

On Recovery Startup

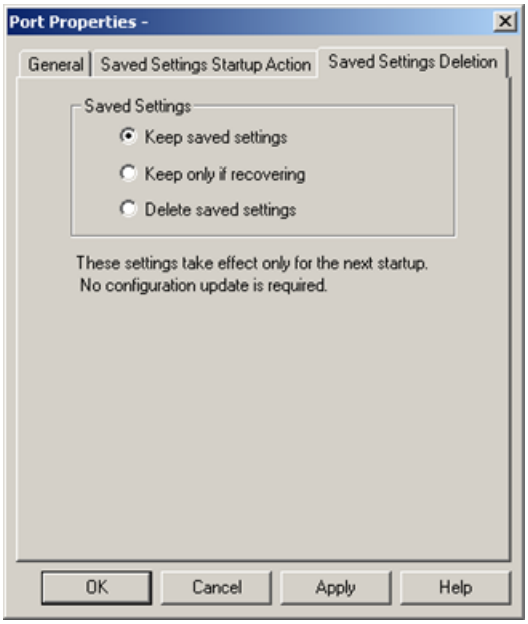
Recovery startup occurs after:

- A cluster fails.
- Process health kills a project as a result of a process failure.

Options *(page 454)* for On Recovery Startup are the same as they are for On Normal Startup.

Saved Settings Deletion

Check one of the following to specify what FloPro/FloNet Communications should do with saved settings.



Option	Description
Keep saved settings	Do not automatically delete the saved settings on the next startup
Keep only if recovering	Delete the current saved settings unless the next startup is in recovery mode.
Delete saved settings	Deletes: • The current saved settings at the next startup. • Settings for all the devices that are configured for the port. **Note:** If the configuration of the device is changed, make sure to check and apply Delete saved settings.

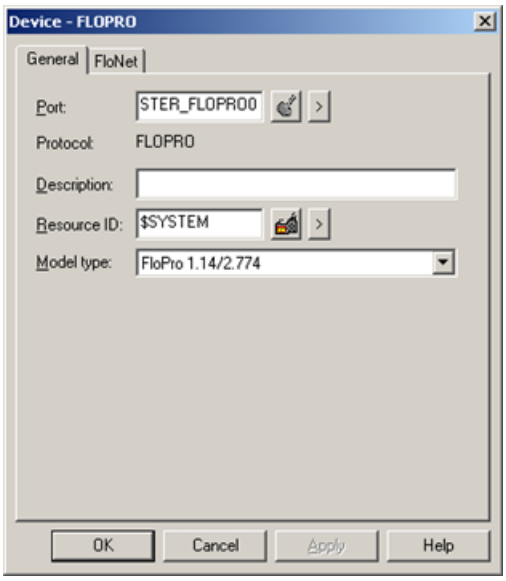
FloPro/FloNet Device Configuration

FloPro/FloNet Device Configuration

1. In the **Device** field, enter the name of the device you are configuring.
2. In the **Port** field, select the FloPro/FloNet port to be used by the device.

   When you click **OK** to create the device, the Device Properties dialog box opens.

General Device Properties

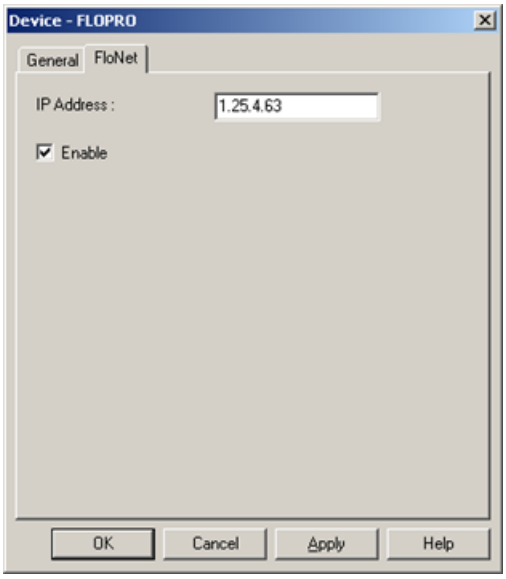


Use the General tab in the Device dialog box to enter general information for the device. You can define the following:

Port	Select the port for this device. Click buttons to the Port field to select the port as follows.
	
	
Description	Enter an optional description to help you identify the device.
Resource	Enter a resource that can be associated with this device for alarm generation. Click buttons to the right of the **Resource** field to select the resource, as follows.
	

	>
Model Type	Select the type of device. Click the drop-down button to the right of the input field to display your choices, then make a selection. For this protocol, the choices are:
	FloPro 1.44/2.774
	This model type supports PCs equipped with FloNet cards running version 1.14 firmware, and FloPro version 2.774. Opening an older CIMPLICITY project in CIMPLICITY version 6.0 will upgrade the FloPro model type to FloPro 1.44/2.774.

Device Properties



Use the FloNet tab in the Device dialog box to enter information about communications for the device. You can define the following:

IP Address	IP address of the target FloNet card in the FloPro PC.
Enable	Set this check box to enable the device when the project starts. If you clear this check box blank, the device will not be enabled and points associated with the device will be unavailable.

FloPro/FloNet Point Configuration

FloPro/FloNet Point Configuration

Once your devices are configured, you may configure points for them. The fields described below have configuration values that are unique to or have special meaning to FloPro/FloNet Ethernet device communications.

General Point Properties

On the General tab in the Point Properties dialog box, the type of

Access you choose depends on the point's data type.

- Select **Read** for Input Force, Output Force, Flag Force and Current Value points.
- Select **Read/Write** for Input, Output, Flag, Counter, Register, Number, ASCII and Timer Limit points.

Device Point Properties

On the Device tab in the Point Properties dialog box:

Update Criteria	The update criteria field determines how the data will be requested.
	Select **On Poll** or **On Scan** for points whose values should be polled by FloPro/FloNet communications at regular intervals. Select **On Demand On Scan** or **On DemandOn Poll** for points whose values should be polled by FloPro/FloNet communications at regular intervals while users are viewing them.
Address	A FloPro/FloNet device point address consists of a one or two character alphabetic field followed by an n-digit decimal-numeric field. The numeric field reflects the offset within an addressing domain, of the particular point. The alphabetic fields map to domains according to the following table:
	Enter the point address as follows:

	Data Type	Address format
	Input	**In**
	Input Force	**Ifn**
	Output	**On**
	Output Force	**Ofn**
	Flag	**Fn**
	Flag Force	**FFn**
	Timer	**Tn**
	Counter	**Cn**
	Register	**Rn**
	Number	Nn
	ASCII	**An**

Advanced Configuration Requirements

Advanced Configuration Requirements

In most cases, the standard startup procedure and response timeout are acceptable. For those cases where they are not, you can use the following global parameters to modify them.

- FLOPRO_STATIC_MODEL
- FLOPRO_RESPONSE_TIMEOUT
- DC_RETRY_ONE_DEVICE

FLOPRO_STATIC_MODEL

By default, when the FloPro/FloNet communications process starts, it queries all the FloPro/FloNet devices to find the maximum file size for each file type. If you have many FloPro/FloNet devices configured in the project, this can take considerable time, and slows project startup.

Purpose	To disable the queries.
Value	TRUE
	FALSE or Delete
Default value	FALSE

FLOPRO_RESPONSE_TIMEOUT

Purpose	To increase or decrease the response time out when the FloPro/FloNet communications process reads the UDP socket for a response from the PLC.
Value	Milliseconds between 1 and 1000.
Default value	100 milliseconds

DC_RETRY_ONE_DEVICE

Purpose	To implement an alternate retry method for how the FloPro/FloNet communications process deals with Down Devices
Value	TRUE

	No parameter
Default value	No parameter

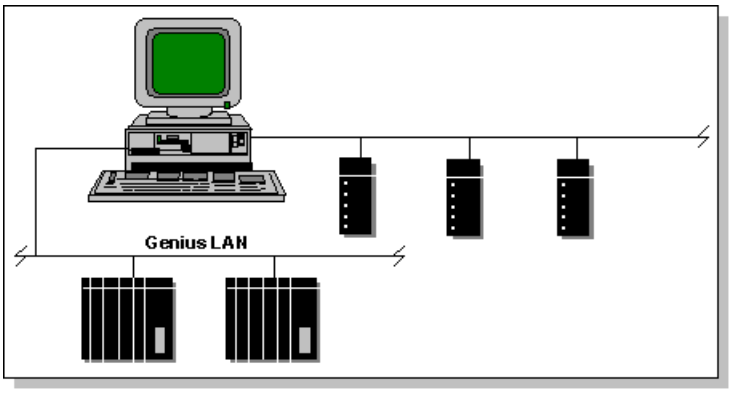
Chapter 24. Genius PCI Communications

About Genius PCI Communications

The Genius® Communications enabler supports communications over a high-speed Genius LAN to Series 90™-70, Series 90-30, PACSystems™ RX7i and PACSystems RX3i programmable controllers, Genius Blocks, Genius Bus Interface Units (also called BIU or Field Control), and other computers with PCIM cards.

The driver is available for use on Intel-based PCs running Windows® XP Professional or Windows 2003 Server with the IC660ELB931 PCIM cards.

You may have up to four Genius ports (called PCIM0, PCIM1, PCIM2 and PCIM3) on your computer.



This enabler supports the following CIMPLICITY features:

- Poll after setpoint.
- Triggered reads.
- Unsolicited data.
- Analog deadband.
- Alarm on communications failure.
- Server Redundancy configurations.

This enabler supports the following CIMPLICITY data types:

- Boolean
- Signed 8, 16, and 32 bit integer
- Unsigned 8, 16, and 32 bit integer
- Floating point

- Arrays

Supported Genius Devices

The Genius Communications enabler currently supports the following devices:

For datagram communications

- Series 90-70 programmable controllers with Genius Bus Controller
- Series 90-30 programmable controllers with Genius Bus Controller
- PACSystems RX7i programmable controllers with Genius Bus Controller
- PACSystems RX3i programmable controllers with Genius Bus Controller

For global data communications

- Series 90-70 programmable controllers with Genius Bus Controller
- Series 90-30 programmable controllers with Genius Bus Controller
- PACSystems RX7i programmable controllers with Genius Bus Controller
- PACSystems RX3i programmable controllers with Genius Bus Controller
- Genius Analog and Digital Blocks
- Genius Bus Interface Units (BIUs)
- Other PCIM cards on the network that are broadcasting global data

For fault data

- 115 VAC 4 In/2 Out Analog Blocks (IC660BBA100)
- 24/48 VDC 16 Circuit Sink I/O Digital Blocks (IC660BBD020)

Supported Genius Memory Types

Supported Genius Memory Types

The Genius Communications Enabler supports:

- Reading and writing point data to Series 90-70 , Series 90-30, PACSystems RX7i and PACSystems RX3i programmable controllers via Datagram and Global communications

- Reading and writing point data to Series 90-70, Series 90-30, PACSystems RX7i, PACSystems RX3i, Analog and Digital Genius blocks, Genius Bus Interface Units and other PCIM cards via Global communications.

Communications include:

- Series 90 Datagram communications.
- Block datagram Communications.
- Global communications.

Note: If you add a block to a programmable controller, reconfigure a Bus Interface Unit's bus map, or resize domains, you must stop and restart the Genius Communications Enabler in order for it to see the changes.

Series 90 Datagram Communications

Data may be read from or written to the following memory types using Datagram communications:

Memory Type	Description
%AI	Analog Input Table
%AQ	Analog Output Table
%I	Discrete Input Table
%Q	Discrete Output Table
%R	Register Memory
%M	Discrete Momentary Internals
%S	System Fault Table
%SA	Special Contacts A
%SB	Special Contacts B
%SC	Special Contacts C

CAUTION: Do not write data to the system registers (%S, %SA, %SB, and %SC). Doing so may interfere with the normal operation of the programmable controller

Block Datagram Communications

Fault data may be read from the following memory type on 24/48 VDC 16 Circuit Sink I/O Blocks (IC660BBD020) and 115 VAC 4 In/2 Out Analog Blocks (IC660BBA100):

Memory Type	Description

%GD	Read Block Level diagnostics

Block inputs and directed outputs are referenced as Global data. See the following section.

Global Communications

Global Communications

Data may be read from or written to the following memory types using Global data communications:

Memory Type	Description
%I	Read discrete inputs from Genius BIU
%AI	Read analog inputs from Genius BIU *
%Q	Write discrete outputs to the Genius BIU Discrete Output table
%AQ	Write analog outputs to the Genius BIU Discrete Output table *
%GL	Read Global Data from Analog and Digital blocks Read Global Data from programmable controllers Write Outputs to Digital Blocks Read Global Data from other Genius devices (such as PCIM cards)
%EO	Enable Outputs to Blocks Enable Outputs to Genius BIU
%DO	Directed Outputs to Analog Blocks

* Analog points configured in Discrete memory on a BIU must be configured to start on even-byte boundaries.

Special Note for Directed Outputs to Analog Blocks

For **%DO** values, the actual output values will match the values displayed by your CIMPLICITY project if and only if the Analog Block's outputs are controlled by the CIMPLICITY Genius Communications driver and the outputs are not otherwise forced.

Genius Hardware Configuration Requirements

Target Series 90-70, Series 90-30, PACSystems RX7i and PACSystems RX3i programmable controllers require the following:a Genius Bus Controller

Personal computers running CIMPLICITY Software and the Genius Communications Enabler require one or more IC660ELB931 PCIM cards.

You can read fault data from:

24/48 VDC 16 Circuit Sink I/O Block-Phase B	IC660BBD020
115 VAC 4 In/2 Out Analog Block-Phase B	IC660BBA100

Genius Related Documents

PCI Genius Card IC660ELB931 Quick Install Guide (GFK-2342)

Series 90-70 PLC Genius Bus Controller User's Manual (GFK-0398)

Series 90-30 Genius Communications Module User's Manual (GFK-0412)

Genius I/O System and Communications User's Manual - Volume 1 (GEK-90486-1)

Genius I/O Discrete and Analog Blocks User's Manual - Volume 2 (GEK-90486-2)

PCIM Card Installation Procedures

You can use the Genius Communications enabler with IC660ELB931.

Refer to PCIM Genius Card Quick Install Guide for details about installing the PCI Genius card in your computer.

The Genius product option includes a kernel level driver for the PCIM (Genius PCI) card(s), a utility to install the Genius kernel level driver and configure the card(s), a console application to test the card operation on the network, and the Genius Device Communications Enabler for CIMPLICITY. To install and configure the Genius PCI card, please follow these steps:

Installation of the driver is similar on each of the supported operating systems.

Installation on Windows XP for an administrative user is as follows:

1 *(page 466)*	Install the PCIM cards.
2 *(page 466)*	Configure the PCIM cards.
3 *(page 467)*	Execute the console application.

Install the PCIM cards.

Install the PCIM cards (IC660ELB931) into the PC.

Configure the PCIM cards.

A utility, `CfgGeniusCard` is used to install the Genius device driver and configure the Genius card for use with CIMPLICITY.

1. The utility may be run from the command prompt or it may be selected from the start menu as follows:

```
Start->All Programs->HMI SCADA - CIMPLICITY 8.2->PCIM Card Configuration
```

Notes

- On Windows XP and Windows 2003 Server, you must be a user with administrative privileges successfully use the utility.
- On Vista and Windows 2008, the program requires elevation to run.
- When the utility is initiated, if required, the User Account Control will prompt the user for permission to continue.
- The program will identify itself as CfgGeniusCard.exe with GE Intelligent Platforms, Inc. as its publisher.

1. Once initiated, the utility will check to see if the current version of the driver is installed and associated with the PCIM hardware.

If the current version of the driver is not installed, you will be asked if you want to install the driver. If you indicate that the driver is to be installed, a wizard will walk you through the driver installation.

For each card, you will identify the following.

- Serial Bus Address
- Serial Bus Baud Rate
- Whether Outputs should be enabled.

Note: The watchdog timer will be disabled.

When running the utility:

1. Select the card to be configured.
2. Click OK.
3. Enter the following.

- Serial Bus Address,

- Serial Bus Baud Rate
- Whether Outputs should be enabled.

1. Click one of the following.

- OK to save your settings.
- Cancel to ignore your settings.

Important:

- The PCIM cards do not need to be installed in your PC in order to configure the card settings.
- You cannot use the PCIM card in CIMPLICITY or with CIMPLICITY provided utilities until after you have configured the card with the above utility.

1. If you installed the Genius device driver or want to retest the interface's operation, you may use the utility included with the device driver.

Important: If the following settings are incompatible with your Genius network configuration (e.g. there is already a device at SBA 0 or a different Serial Bus Baud Rate is used, do not use this utility with the card connected to your Genius network.

At the command prompt:

1. Type devmgmt.msc.
2. Select Genius Devices.
3. Select your Genius card.
4. Select the Test tab under Device Properties.
5. Click the Test button to execute the test.

Result. The test will test the card by bringing it up at Serial Bus Address (SBA) 0 using a Serial Bus Baud Rate of 153.6K. STD The utility reports on the state of the card.

Execute the console application

Once a PCIM card has been configured and connected to the network, the console application (run from the Command Prompt) GeniusDevices, can be executed.

This utility displays all of the devices visible to the PCIM card on the genius network.

The PCIM card will start up using the SBA and Baud Rate configured in the CfgGeniusCard utility.

The card will remain active only while it determines the devices on the network. By default, the utility will use the first card (card 0).

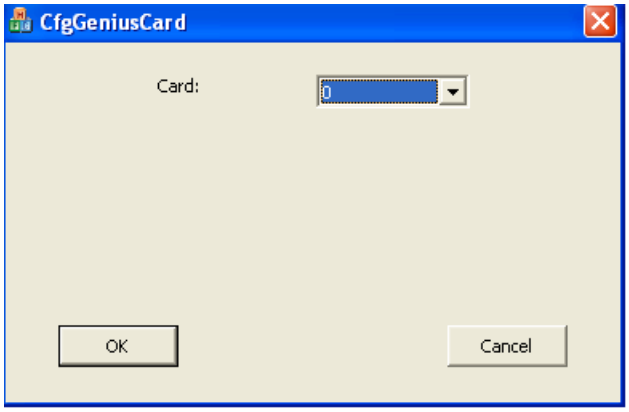
To select a different card, specify the card number to be used as the second argument to the command. The first card is 0, the second 1, etc.

PCIM Configuration Application

1. Click the PCIM Configuration icon 

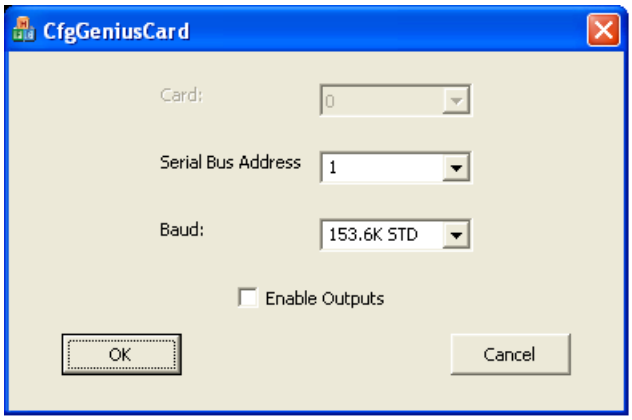
in the CIMPLICITY program group.

   The PCIM Configuration dialog box opens.



   You can use the dialog box to configure a PCIM card.

2. Select the card to be configured and press OK. The following dialog opens thereafter:



3. Enter the following for each Genius card that will be used in your system:

Serial Bus Address	The Serial Bus Address used to identify the PCIM card on the Genius network.

Baud	The Baud Rate used by the select PCIM card to communicate with devices on the Genius network.
Enable Outputs	Check to enable outputs.

Note: The watchdog timer will be disabled.

Genius Application Configuration

Genius Application Configuration

When you configure ports, devices, and device points, you will need to enter some data that is specific to Genius communications.

The following sections review the configuration requirements.

- Genius port configuration.
- Genius device configuration.
- Genius device point configuration.

Genius Port Configuration

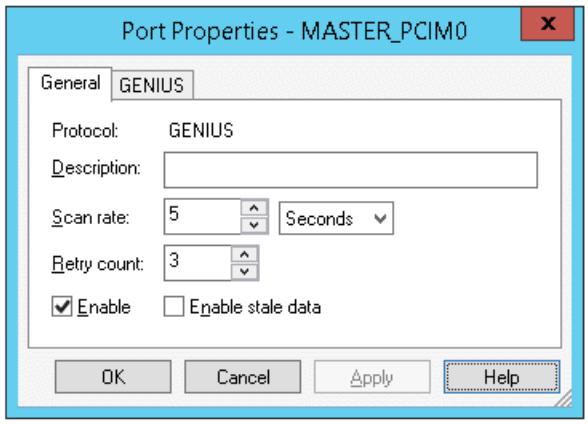
Genius Port Configuration

When you configure a port for Genius Communications, enter the following information in the New Port dialog box:

1. In the **Protocol** field, select **GENIUS** from the list of available protocols.
2. In the **Port** field, select the PCIM port that will be used for Genius communications.

   When you click **OK** to create the port, the Port Properties dialog box for the protocol opens.

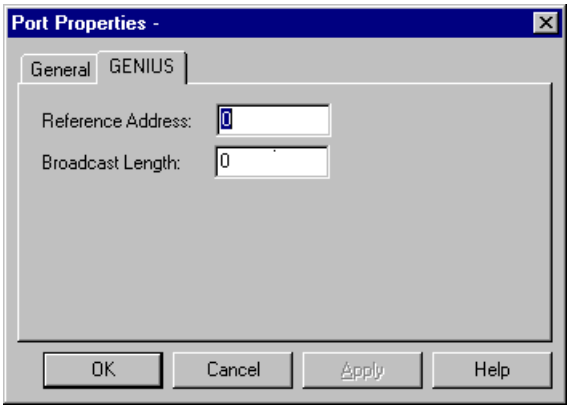
Genius General Port Properties



Use the General tab in the Port Properties dialog box to enter general information for the port. You can define the following:

Description	Enter an optional description to help you identify the port.
Scan Rate	Enter the base scan rate for the port. Point scan rates will be multiples of the base rate. You can specify a scan rate in Ticks (100 ticks = 1 second), Seconds, Minutes, or Hours.
Retry Count	If communications cannot be established, the device is considered to be down and a $DEVICE_DOWN alarm is generated. Once a device is down, periodic attempts are made to resume connections to it.
	Enter the number of scans to wait before attempting to reconnect to a device on this port after a communications error is detected.
Enable	Set this check box if you want the port to be enabled. If you clear this check box, the port will not be enabled when the project starts, and points will not be available for devices on the port.
Enable stale data	Set this check box if you want the point to remain available in most circumstances that would have made it unavailable. However, this attribute will report that the point value is stale. It is the last known good value and may or may not have changed.

Genius Port Properties



You only need to access the Genius tab in the Port Properties dialog box if you intend to broadcast Global Data. For more information on broadcasting Global Data, see "PCIM Port Global Data Broadcast".

Use the Genius property page to define the following:

Reference Address	The starting address of Global data for Series Six PLCs in the Genius network.
Broadcast Length	The length, in bytes, for outgoing Global Data from this computer to devices in the Genius network.

Genius Device Configuration

Genius Device Configuration

1. In the **Device** field, enter the name of the device you are configuring.
2. In the **Port** field, select the Genius port to be used by the device.

   When you click **OK** to create the device, the Device Properties dialog box opens.

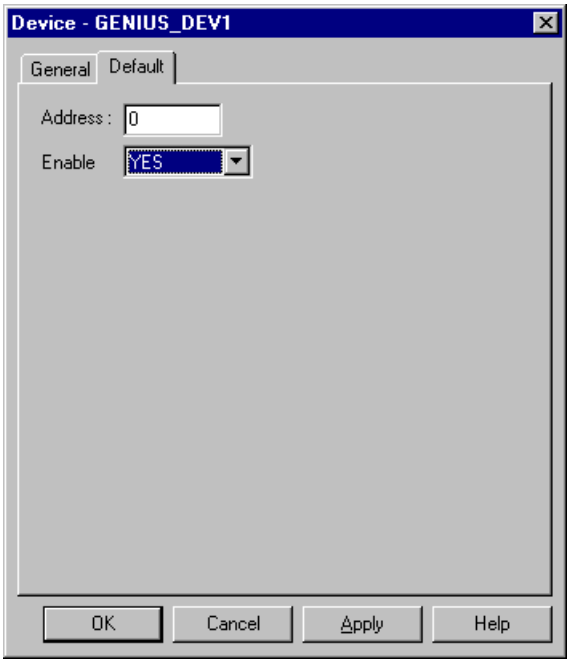
Genius General Device Properties

Use the General properties tab in the Device dialog box to enter general information for the port. You can define the following:

Port	Select the port for this device. Buttons to the right of the Port field can help, as follows.
	Button
	
	
Description	Enter an optional description to help you identify the device.
Resource	Enter the resource that can be associated with this device for alarm generation. Buttons to the right of the Resource field can help, as follows.
	Button
	
	

Model Type	Enter the type of device. Click the drop-down button to the right of the input field to display your choices, then make a selection. For this protocol the choices are: Bus Interface Unit Genius Analog Block Genius Digital Block RX3i RX7i Series 90-30 or RX3i Series 90-70 or RX7i Generic Global Analog Blocks are 4 input/2 output Genius Analog Blocks. Digital Blocks include 8, 16, and 32 channel digital blocks.

Genius Default Device Properties



Use the Default tab in the Device dialog box to enter the following information about the device:

Address	Enter the Serial Bus Address (SBA) of the device that you are communicating with in this field.
Enable	Enter YES to enable the device. If you enter NO, the device will not be enabled, and points associated with the device will be unavailable.

Genius Device Point Configuration

Genius Device Point Configuration

When you define a point, the fields in the Point Properties dialog box have values that are unique to or have special meanings for Genius Communications.

Genius General Point Properties

On the General tab in the Point Properties dialog box, you may configure points for **Read** or **Read/ Write** access.

Genius Device Point Properties

Specify the following on the Device tab in the Point Properties dialog box.

Field	Description
Address	For all programmable controller memory types, enter the memory type and offset in the Address field on the Device page of the Point Properties dialog box. The address format is: <memory type><address> Example To specify Register 1: • The address is `%R1`. • All offsets have a minimum value of 1.

Field	Description
Update Criteria	The update criteria determines how the data will be requested. Select On Change or On Scan for points whose values should be polled by Genius communications at regular intervals. Select On Demand On Scan or On Demand On Change for points whose values should be polled by Genius communications at regular intervals while users are viewing them.

When you are configuring Boolean points for memory types that are not bit addressable, you must also enter data in the following field:

Field	Description
Address Offset	Enter the bit offset that corresponds to the bit position within the word. An address offset of.
	Offset
	0
	15

If you are configuring output points for Genius blocks, don't forget to implement a method for enabling outputs to the blocks.

To write to the outputs of an analog block, use (Directed Output) for the first output circuit and %DO2 for the second output circuit. Prior to writing data to the block, you must enable outputs to it.

Enabling Directed Outputs to Analog Blocks

Before you can set directed outputs **(%DO**) on an Analog Block or receive unsolicited fault reports, you must enable outputs to that block. To enable outputs to a block, configure a digital point on that block with the address **%EO1**. When the CIMPLICITY project starts, the point at **%EO1** will be set to FALSE (0) and outputs to the block are disabled. When you set **%EO1** to TRUE (1), outputs are enabled to the block and can be used to control the block's outputs.

You can use any of the following CIMPLICITY functions to set **%EO1**:

- A Setpoint action on a CimView screen.
- A script that performs a setpoint in the Basic Control Engine.
- A Point Management API procedure that performs a setpoint.

Once the **%EO1** point is set to TRUE, directed outputs may be set.

Enabling Outputs to BIUs

Before you can set outputs **(%Q** and **%AQ**) on a BIU, you must enable outputs to it. To enable outputs to a BIU, configure a digital point on the BIU with the address **%EO**. When the CIMPLICITY project starts, the point at **%EO** will be set to FALSE (0) and outputs to the BIU are disabled. When you set **%EO** to TRUE (1), outputs are enabled to the BIU and can be used to control the block's outputs.

You can use any of the following CIMPLICITY functions to set **%EO**:

- A Setpoint action on a **CimView** screen.
- A script that performs a setpoint in the Basic Control Engine.
- A Point Management API procedure that performs a setpoint.

Once the **%EO** point is set to TRUE, outputs to the BIU may be set.

Configuring Global Data Points on Programmable Controllers

Configuring Global Data Points on Programmable Controllers

To read Global Data that is broadcast from a programmable controller, use the memory type %GL. The address is in bytes and ranges from 0 to 127. For example, if you have configured a Series 90-70, Series 90-30, PACSystems RX7i or PACSystems RX3i programmable controller to broadcast 64 registers of global data starting at register 100, then to read the data that is in register 100, configure an analog point with the address %GL0. Because each register uses two bytes of data, data from register 101 is at address %GL2, and data from register 110 is at address %GL20.

Review:

- Configuring global data from digital blocks.
- Configuring global data from analog blocks.

Note: Global data can only be read from programmable controller memory. You cannot use global memory to write to programmable controller memory.

Configuring Global Data from Digital Blocks

For all digital blocks, use **%GL** to read the global data or to write data to the block. The first circuit on a digital block is addressed as **%GL1**. Prior to writing data to the block, you must enable outputs to it.

Configuring Global Data from Analog Blocks

For 4 input/2 output analog blocks, use **%GL** to read the global data from the block. The first circuit on an analog block is addressed as **%GL1**; the second is **%GL2**, and so on.

Genius Fault Data

Genius Fault Data

You can collect fault data from the following Genius blocks:

- 115 VAC 4 In/2 Out Analog Block - Phase B (IC660BBA100)
- 24/48 VDC 16 Circuit Sink I/O Block - Phase B (IC660BBD020)

Review:

- Configuring fault points for IC660BBD020 blocks.
- Configuring fault points for IC660BBA100 blocks.

- Decoding fault data.

Configuring Fault Points for IC660BBD020 Blocks

For the 24/48 VDC 16 Circuit Sink I/O Block - Phase B (IC660BBD020), configure points **%GD0** through **%GD16**, where **%GD0** is the Block-Level diagnostic point, and **%GD1** through **%GD16** are the Circuit-Level diagnostic points.

Note: If you do not configure **%GD0**, no fault data is collected. If you do configure **%GD0**, but do not configure points between **%GD1** and **%GD16**, messages for the missing points are generated in the Status Log at startup and whenever a fault occurs.

All diagnostic points are **Analog** device points with a Point Type of **USINT**. The **Update Criteria** is **On Change** for the Block-Level diagnostic point, and **Unsolicited** for the Circuit diagnostic points.

Prior to receiving fault data from the block, you must enable outputs to it.

Configuring Fault Points for IC660BBA100 Blocks

For the 115 VAC 4 In/2 Out Analog Block - Phase B (IC660BBA100), configure points **%GD0** through **%GD6**, where:

- **%GD0** is the Block-Level diagnostic point.
- **%GD1** through **%GD4** are the Circuit-Level diagnostic points for inputs.
- **%GD5** through **%GD6** are the Circuit-Level diagnostic points for outputs.

Note: If you do not configure **%GD0**, no fault data is collected. If you do configure **%GD0**, but do not configure points between **%GD1** and **%GD6**, messages for the missing points are generated in the Status Log at startup and whenever a fault occurs.

All diagnostic points are Analog device points with a Point Type of **USINT**. The **UpdateCriteria** is **On Change** for the Block-Level diagnostic point and **Unsolicited** for the Circuit-Level diagnostic points.

Prior to receiving fault data from the block, you must enable outputs to it.

Decoding Fault Data

Decoding Fault Data

You can use the information contained in the Block-Level and Circuit-Level diagnostic points on 24/48 VDC 16 Circuit Sink I/O Block - Phase B (IC660BBD020) and 115 VAC 4 In/2 Out Analog

Block - Phase B (IC660BBA100) blocks to display detailed information or generate alarms when faults occur.

Review:

- Decoding IC660BBD020 Block Fault Data
- Decoding IC660BBA100 Block Fault Data

Decoding IC660BBD020 Block Fault Data

Use the following table to decode the reported values for the Block-Level diagnostic points for 24/48 VDC 16 Circuit Sink I/O Block - Phase B Blocks:

Bit	Diagnostic Fault
`0-2,4-7`	Not used
`3`	Terminal Assembly EEPROM fault

Use the following table to decode the reported values for Circuit-Level diagnostic points for 24/48 VDC 16 Circuit Sink I/O Block - Phase B Blocks:

Bit	Diagnostic Fault
`0`	Loss if I/O power
`1`	Short circuit
`2`	Overload
`3`	No load, or input open wire
`4`	Over temperature
`5`	Failed Switch
`6`	Not used
`7`	Not used

Decoding IC660BBA100 Block Fault Data

Use the following table to decode the reported values for the Block-Level diagnostic points for 115 VAC 4 In/2 Out Analog Block - Phase B Blocks:

Bit	Diagnostic Fault
`0-2,4-7`	Not used
`3`	Terminal Assembly EEPROM fault
`5`	Electronics Assembly EEPROM fault
`7`	Internal circuit fault

Use the following table to decode the reported values for Circuit-Level diagnostic points for 115 VAC 4 In/2 Out Analog Block - Phase B Blocks:

Bit	Diagnostic Fault
0	Input low alarm
1	Input high alarm
2	Input under range
3	Input over range
4	Input open wire
5	Output under range
6	Output over range
7	Feedback error

PCIM Port Global Data Broadcast

PCIM Port Global Data Broadcast

You can configure a PCIM card to broadcast up to 128 bytes of global data.

- PCIM Port configuration.
- PCIM Device configuration.
- PCIM Memory addressing.
- PCIM Special cautions.

PCIM Port Configuration

PCIM cards can be setup to broadcast up to 128 bytes of global data. The CIMPLICITY Genius communications interface now supports the ability to configure PCIM cards used for communication in CIMPLICITY so that they can broadcast data. For any PCIM card to broadcast global data, it is necessary to go to the port selection of the project and reconfigure the Port of interest.

The **Reference address** is used by some Genius devices to specify the logical address that may correspond to where the device's I/O data is stored in host memory.

- For many devices such as the Series 90, this information is not used.
- For other devices, such as a Series Five or Series Six, you will need to enter a value.

See the Bus Controller (or Device) User's Manual for each device on your Genius network to determine what the configuration requirements are. You may only configure one reference address for each port of a PCIM card and this must be compatible with all devices on the Genius network. The value entered must fit within 16 bits. The high bit is always ON, regardless of the value entered for the reference address.

The **Broadcast length** specifies the number of bytes a port on a PCIM card will broadcast. A value of zero, the default value, specifies NO BROADCAST data. A value between 1 and 128 defines the number of bytes of data to be broadcast as well as the size of the directed control input table on the port of the PCIM card.

PCIM Device Configuration

To configure your PCIM card to broadcast data, create a device that corresponds to your PCIM card. To do this:

- Select GENERIC GLOBAL for the Model Type.
- Enter the Serial Bus Address (SBA) of the PCIM port in the CPU ID.

PCIM Memory Addressing

Broadcast Global data is referenced by using the **%GL** memory type. The offset entered corresponds to the byte offset of the PCIM card. **%GL0** is the first offset and refers to BYTE 1, **%GL1** is BYTE 2, etc.

Broadcast data may be modified by performing setpoint operations on the points configured for the device. You must ensure that the number of byte being broadcast is compatible with other devices on the network. For example, if the broadcast data is being mapped to the Register memory of a Series 90 Bus Controller, the number of bytes broadcast needs to be on an even-byte boundary.

PCIM Special Cautions

You need to know that:

- Each time the Genius Communication enabler starts, the data broadcast by the PCIM card is initialized to zero. Previous values are not maintained.
- For applications where PCIM cards are broadcasting data and Series 90 processors are present on the bus, you must map the System Configuration Mismatch to a Diagnostic Category rather than the default classification of FATAL on each Series 90 processor.
- If you do not modify this classification, the Series 90 will go to a FAULT/STOP category each time the PCIM is initiated.

Monitoring Broadcast Data from Generic Devices

To monitor data broadcast from devices not otherwise configurable from a CIMPLICITY project, configure a device on the PCIM port with:

- Model type of GENERIC GLOBAL
- CPU ID corresponding to the Serial Bus Address (SBA) of the device.

To access the broadcast data, use the **%GL** memory type with an offset between 0 and the broadcast length of the device. Broadcast data is byte addressable. **%GL0** is used to reference the first byte of the data, **%GL1** the second byte, etc.

Broadcast data from devices other than your own PCIM port may not be modified via CIMPLICITY software.

Broadcast data for a device is assumed to have a static length. If the broadcast data length changes, you need to disable and re-enable the device so that the change will be properly processed by the Genius Communications enabler.

Chapter 25. Honeywell IPC 620 Communications

About Honeywell IPC 620 Communications

The Honeywell IPC 620 Protocol Device Communication driver lets you exchange data between CIMPLICITY software and Honeywell's IPC 620 family PLCs using the Asynchronous Byte Count (ABC) protocol.

Supported Devices

The Honeywell ABC Protocol device communication option supports the following Data Collection Modules:

- 620-0048 DCM Module (revision VR 3.1 or later)
- 620-0042 DCM Module (prom revision 21024L or later)

and the following PLCs:

- IPC 620-12
- IPC 620-15
- IPC 620-16
- IPC 620-30

Up to 32 programmable controllers may be connected to CIMPLICITY over on a single serial connection.

If your DCM module is at an earlier revision than listed above, you may encounter interoperability issues. For additional information on this subject and upgrades, please contact your Honeywell representative.

Additional Documentation

You should have the following documentation, as it applies to your hardware configuration, available when configuring this interface:

- 620-0048 & 620-0052 Data Collection Modules User Manual, 620-8980
- IPC 620 Communications Interface Module User Manual, Model IPC 620-0042

- 620 Logic Controller Modular Automation System
- IPC 620 Programmable Controller Module IPC 620-10 and 620-15, 620-8999
- IPC 620 Programmable Controller Module IPC 620-20, 620-8998
- IPC 620 Programmable Controller Module IPC 620-30

Supported Memory Types

Memory Types	Access
Digital	Read/Write
Register *	Read/Write
Extended Register *	Read/Write
System Status Register	Read

***** On 620-15, 620-20 and 620-30 programmable controllers, register data is represented in 17 bits. When accessing these registers, using the register memory type, values between 0 and 32767 will display consistently between CIMPLICITY and the programming software. For the aforementioned processors, for values outside of this range, the extended register memory type should be used. When the extended memory type is used, 17 bit values are converted to 32 bits. This 32 bit representation can be mapped to a standard CIMPLICITY point type. (Analog_32 signed, for example). For 620 programmable controllers which represent the register data using 16 bit registers.

When floating point data values are configured, they must be mapped over the register type only. Extended register will not support floating point data. One floating point value consumes two 16-bit registers of storage, so arrays of floating point values consume twice as many registers as they contain elements.

Hardware Configuration Requirements

Hardware Configuration Requirements

The Honeywell ABC Protocol allows up to 32 programmable controllers to be connected to CIMPLICITY system over on a single serial connection.

If using an RS232 serial port and connecting to a DCM configured for RS485 operation, an RS232 to RS485 converter box will be required. When conversion is required, the use of an isolated converter box is recommended.

Custom Cable

The following connections are required in the cable that connects DCM and CIMPLICITY system:

There are two types of configurations for interconnecting DCM(s) to a host:

- Point-to-point connection is between only one DCM and the host
- A multipoint/multidrop configuration connects two or more DCMs to the host on a common line.

CIMPLICITY System Configuration

In your NT system's Control Panel, select Ports. The various ports that can be used for device communications will be listed. The serial communication ports are listed as COM1, COM2, etc. Select one port for each physical device communication process required. Hardware Installation External to the Computer.

Hardware Checklist

1. Honeywell IPC 620 family PLC and Data Collection Module as described in section **Supported Devices.**
2. Verify that green indicators in PLC and DCM are on for power and self-test.
3. Custom Cable wired as shown in the diagram in **Custom Cable**.
4. RS232 Port on NT Host Computer.
5. Connect DCM and CIMPLICITY system using the custom cable.
6. Use **hwabc_diag** program to verify communication.

Installation Verification Procedures

Installation Verification Procedures

- Honeywell ABC diagnostics.
- About CIMPLICITY Honeywell IPC 620 Communications.

Honeywell ABC Diagnostics

The Honeywell ABC Diagnostics program, **hwabc_diag.exe**, may be used to verify hardware installation and communication operation.

To run the N2 diagnostics, execute the following command:

```
hwabc_diag.exe <port> <network drop #> <Baud > <Parity>
```

Where

<port> is the name of the port connected to the connection cable.

<network drop #> is the network nodal address of DCM which can be from 0 to 31. The default value is 0.

<Baud Rate> has a default value of 9600.

<Parity> is none by default.

For example, if your connection cable is attached to COM2, network drop number is 2 for the DCM, baud rate is 9600 and parity is none, execute the following command:

```
hwabc_diag.exe COM2 2
```

The Honeywell ABC Diagnostics program will open the port and read values of a few points:

Comm parameters: drop 2, baud = 9600, parity = no

CIMPLICITY Configuration for Honeywell IPC 620

CIMPLICITY Configuration for Honeywell IPC 620

When you are configuring ports, devices, and device points, you will be asked for some information that is specific to Honeywell ABC Device Communications.

Honeywell IPC Communications 620 Port Configuration

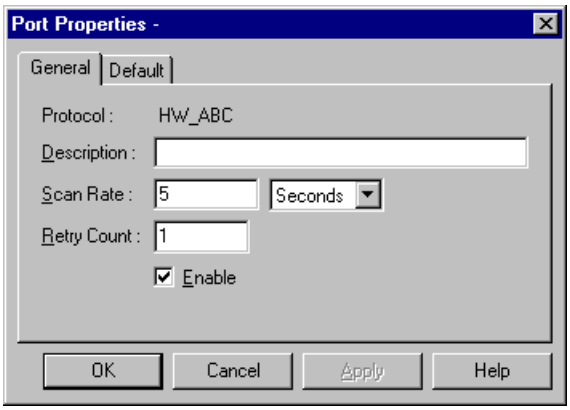
Honeywell IPC Communications 620 Port Configuration

When you create a new port for Honeywell ABC Device Communications, enter the following information in the New Port dialog box:

1. In the **Protocol** field, select **hw-abc** from the list of available protocols
2. In the **Port** field, select that communication port that will be used for Honeywell ABC Device communications.

   When you click **OK** to create the port, the Port Properties dialog box for the protocol opens.

Honeywell IPC 620 General Port Properties

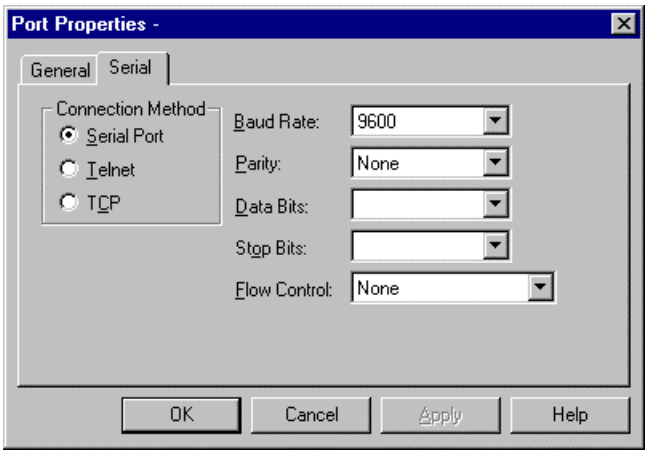


Use the General tab in the Port Properties dialog box to enter general information for the port. You can define the following:

Description	Enter an optional description to help you identify the port.
Scan Rate	Enter the base scan rate for the port. Point scan rates will be multiples of the base rates. You can specify a scan rate in Ticks (100 ticks = 1 second), Seconds, Minutes, or Hours.
Retry Count	If communications cannot be established to a device on this port, the device is considered to be down and a $DEVICE_DOWN alarm is generated. Once a device is down, periodic attempts are made to resume connection to it.
	Enter the number of scans to wait before attempting to reconnect to a device on this port after a communications error is detected.
Enable	Set this check box if you want the port to be enabled when the project starts. If you clear this check box, the port will not be enabled when the project starts, and points will not be available for devices on the port.

Honeywell IPC 620 Serial Port Properties

Use the Default tab in the Port Properties dialog box to enter communication information for the port.



If you select the serial port connection method, you need to define the following:

Baud Rate	Enter the baud rate for communications. Click the drop-down list to the right of the input field and select a rate from the list.
Data Bits	Select the number of data bits per word to be used for communications.
Parity	Select the parity to be used for communications.
Stop Bits	Select the number of stop bits to be used for communications
Flow Control	Select the type of flow control to be used for communications. If you change the Flow Control type, you must reboot the PC for the changes to take affect.

Remember that you must configure the same baud rate, data bits, parity, stop bits and flow control for all PLCs using the serial port.

Honeywell IPC Communications 620 Device Configuration

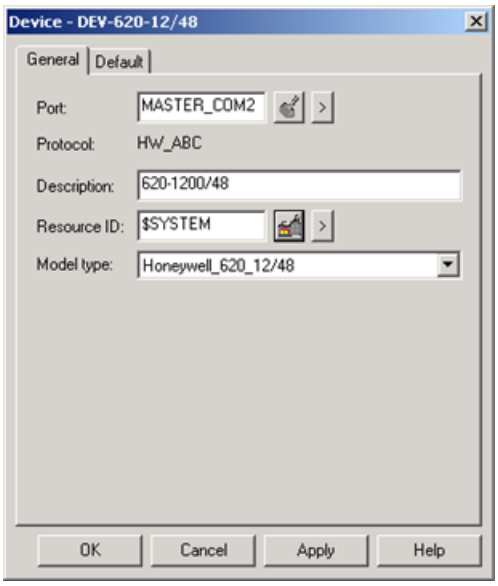
Honeywell IPC Communications 620 Device Configuration

1. In the **Device** field, enter the name of the device you are configuring.
2. In the **Port** field, select the Honeywell ABC Device communications port to be used by the device.

   When you click **OK** to create the device, the Device Properties dialog box opens.

Honeywell IPC 620 General Device Properties

Use the General tab in the Device dialog box to enter general information for the device.

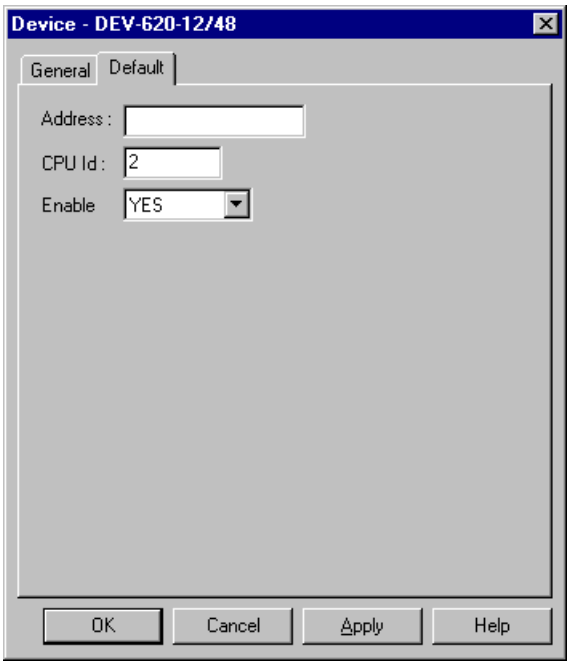


You can define the following:

Port	Select the port for this device. Click buttons to the **Port** field to select the port as follows.	
	Port button	Display the list of ports and select one.
	Popup Menu button	Create a new port, edit the current port, or select a port from the list of ports.
Description	Enter an optional description to help you identify the device.	
Resource	Enter a resource that can be associated with this device for alarm generation. Click buttons to the right of the **Resource** field to select the resource, as follows.	
	Resource button	Display the list of resources and select one.
	Popup Menu button	Create a new resource, edit the current resource, or select a resource from the list of resources.
Model Type	Select the type of device. Click the drop-down button to the right of the input field to display your choices, then make a selection. For this protocol, the choices are:	
	• Honeywell IPC 620_12/48 • Honeywell IPC 620_15/48 • Honeywell IPC 620_16/48 • Honeywell IPC 620_20/48 • Honeywell IPC 620_30/48 • Honeywell IPC 620_12/52 • Honeywell IPC 620_15/52 • Honeywell IPC 620_16/52 • Honeywell IPC 620_20/52 • Honeywell IPC 620_30/52	

Honeywell IPC 620 Default Device Properties

Use the Default tab in the Device dialog box to enter information about communications for the device.



You can define the following:

Address	Enter the network node number here when the type of configuration for interconnecting DCMs to the host is multipoint/multidrop. This number is set up by DIP switch on the actual device. The node numbers may run from zero through 31.
CPU Id	Not used.
Enable	Set this check box to enable the device when the project starts. If you clear this check box blank, the device will not be enabled and points associated with the device will be unavailable.

Honeywell IPC 620 Communications Point Configuration

Honeywell IPC 620 Communications Point Configuration

Once your devices are configured, you may configure points for them. The fields in the Point Properties dialog box have configuration values that are unique to or have special meaning for Honeywell ABC device communications.

General Point Memory Types

On the General tab in the Point Properties dialog box, select **Read/Write** for all the memory types except System Status Register.

Honeywell IPC 620 Device Point Properties

On the Device tab in the Point Properties dialog box:

Update Criteria	The update criteria field determines how the data will be requested.
	Select **On Change** or **On Scan** for points whose values should be polled by the Honeywell driver at regular intervals.
Address	A Honeywell device point address consists of a one or two character alphabetic field followed by an n-digit decimal-numeric field. The numeric field reflects the offset within an addressing domain, of the particular point. The alphabetic fields map to domains according to the following table:
	Enter the point address as follows:

	Type of Memory	Address Format	Numeric Range
	Digital	**D**n	0 – system-specific (up to 4095) 4096 – system-specific (17 bit models)
	Register *	**R**n	4096 – system-specific
	Extended Register *	**ER**n	4096 – system-specific
	System Status Register	**S**n	2048 – 4095

Note: Points may be configured to use any CIMPLICITY point type, however, it is advisable to maintain consistency between the CIMPLICITY point type and the data representations supported by the underlying programmable controller. For example, only 620-12 and 620-16 processors support floating point numbers. When using other processor types, it is recommended that you do not configure REAL CIMPLICITY point type.

Warning: Extreme care should be taken when referencing Intelligent Module I/O. Honeywell indicates that references to invalid Intelligent Module I/O may cause inter-operability issues with programmable control operation.

Honeywell IPC 620 Global Parameters

Diagnostic messages may now be selectively enabled or disabled in the Honeywell device communication interfaces.

When enabled, the diagnostics are directed to the .err file located in the project's log directory.

The following global parameters are now available to enable of disable diagnostics.

`HWABC_DEBUG`

For	Project	
Purpose	To enable or disable diagnostics for all device communication interfaces that use the Honeywell IPC620 protocol.	
Value	Y	Enables the diagnostics.
	N	Disables the diagnostics.
Default	N	

`<PORT>_DEBUG`

For	Project	
Purpose	To enable or disable diagnostics for a selected port that uses the Honeywell IPC620 protocol.	
Value	Y	Enables the diagnostics.
	N	Disables the diagnostics.
Default	N	

Note: If both `<PORT>_DEBUG` and `HWABC_DEBUG` are defined. `<PORT>_DEBUG` has precedence.

Chapter 26. Johnson Controls N2 Bus Communications

About Johnson Controls N2 Bus Communications

The Johnson Controls N2 Bus Communications option lets you exchange data between CIMPLICITY software and Johnson Controls devices.

The Johnson Controls N2 Bus Communication option supports Read and Write values. This communication option does not support array points.

Supported Devices

The N2 System Protocol device communication option supports Johnson Controls N2 Open and DX9100 protocols.

Supported device types are:

- AHU Companion
- AHU Facilitator
- DC9100R1 Controller
- DC9100R2 Controller
- DX9100R1 Companion
- DX9100R1 Facilitator
- DX9100R2 Companion
- DX9100R2 Facilitator
- N2 Base Model
- N2 VND Vendor
- Unitary Companion
- Unitary Facilitator
- VAV Companion
- VAV Facilitator

While not explicitly supported, other Johnson Controls and thirdparty devices may work to the extent that they implement the available protocols.

Additional Documentation

The Johnson Controls document Metasys DX9100 Protocol Specification lists all the data items available in the DX9100.

Supported Memory Types

Supported Memory Types

The Johnson Controls N2 Bus Communications option supports Unitary Controller and DX9100 memory types.

Unitary Controller Memory Types

Data may be read from the following regions in the Unitary Controller:

- Analog Inputs
- Analog Outputs
- Binary Outputs
- Internal Floating-Point data
- Internal Integer data
- Internal Byte data
- Device Memory

Data may be written to the following regions in the Unitary Controller:

- Analog Inputs
- Analog Outputs
- Binary Outputs
- Internal Floating Point data
- Internal Integer data
- Internal Byte data

DX9100 Memory Types

The DX9100 supports reading from all the memory types supported by the Unitary Controller, except Device Memory. In addition, the following memory types may be read:

- Logic Results
- Programmable Module Constants
- Programmable Module Outputs
- Programmable Module Logic
- Programmable Module Accumulator

The DX9100 supports writing to all the memory types supported by the Unitary Controller except Analog Inputs. In addition, the following memory types may be written:

- Logic Results
- Programmable Module Constants
- Programmable Module Outputs
- Programmable Module Logic
- Programmable Module Accumulator

Hardware Configuration Requirements

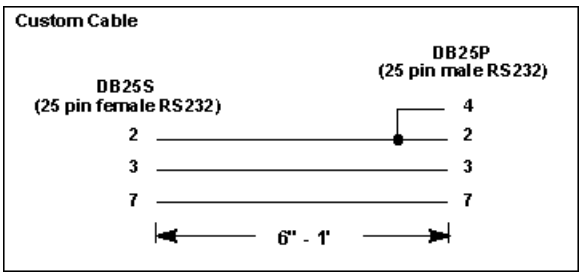
Hardware Configuration Requirements

The N2 Protocol allows one server and up to 255 devices on a common line. In the CIMPLICITY computer, the Johnson Controls N2 Bus Device Communications Enabler is the server.

The N2 bus is RS485 based but uses special biasing to allow 50 nodes without a repeater. The Johnson Controls MMCVT1010 converter is required for adapting the CIMPLICITY computer's RS-232 serial port to the N2 bus.

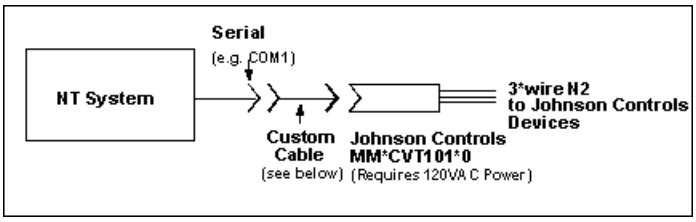
Custom Cable

The following connections are required in the custom cable.



N2 Bus Cabling

The N2 bus cabling looks like this.



Hardware Installation External to the Computer

1. Acquire MMCVT1010 from Johnson Control.
2. Acquire or fabricate Custom Cable as shown in Figure 11.
3. Select RS232 Port on NT Host Computer.
4. Assemble as shown in Figure 12.
5. Verify that green power indicator in MMCVT1010 is on.
6. Use **n2_diag** probe command to verify communication.

Installation Verification Procedures

Installation Verification Procedures

You can use the N2 Diagnostics program, **n2_diag.exe**, to verify hardware installation and network operation.

To run the N2 diagnostics, execute the following command:

```
n2_diag.exe <port>
```

where **<port>** is the name of the port connected to the N2 bus.

For example, if your N2 bus is attached to **COM2**, execute:

```
n2_diag.exe COM2
```

The N2 Diagnostics menu displays:

```
==================
```

```
N2 Diagnostics menu
-------------------
1) Probe N2 network
2) Select a device
3) Parse an address
4) Dump XT config.
5) Read a value
6) Write a value
7) Turn on tracing
8) Turn off tracing
9) Dump statistics
0) Quit diagnostics
```

Use the numbers at the left to select options. You are prompted for other parameters as needed. The options are briefly described in the following sections.

Probe N2 Network

When you select **Probe N2 network**, you are prompted for a starting and ending address. The diagnostic program scans from the starting address to the ending address trying to identify devices. For each address where a device is found, the device type ID is reported in hexadecimal (base 16). For each address where no device is found, two question marks are printed.

In this example, the starting address is 1 and ending address is 8. The only devices configured are a DX9100 at N2 bus address 1 and a UNT at N2 bus address 7.

```
First address to probe [1]:1
Last address to probe [255]:8
.15.??.??.??.??.??.71.??
```

Select a Device

Many of the diagnostic functions are dependent on the type of the device you select. For these operations, you must select a device to work with. When you choose **Select a device**, you can change devices.

```
Device [0]:7
====================
N2 diagnostics menu
-------------------
Type 0x71 device at address 7 selected
```

Parse an Address

Use **Parse an address** to test address strings for validity on the currently selected device.

For example, for a unitary controller, AI[300] is invalid because there are only 256 records in each region.

```
Address<region[record].attr>:AI[2]
Valid
 domain_index = 1, domain_offset = 54
Address<region[record].attr>:AI[300]
Invalid
```

Dump XT Configuration

A DX9100 may have one or more extension (XT) modules installed. Use the **Dump XT configuration** option to query the currently selected device for its XT configuration and report it to the screen.

In the report:

- **X** corresponds to an installed input/output point.
- **-** corresponds to an input/output point which is not installed.

For example, there are two XT modules.

- The first module has 8 analog points: 6 inputs and 2 outputs. For this module, AI[9] through AI[14] are valid input addresses and AI[15] through AI[16] are valid output addresses.
- The second module has 8 binary points: four inputs and four outputs. For this module, BI[17] through BO[20] are valid input addresses, and BO[21] through BO[24] are valid output addresses.

The report to the screen looks like this:

```
XT 1            Analog                      Binary
        09 10 11 12 13 14 15 16     09 10 11 12 13 14 15 16
  In    X  X  X  X  X  X  -  -      -  -  -  -  -  -  -  -
  Out   -  -  -  -  -  -  X  X      -  -  -  -  -  -  -  -
XT 2            Analog                      Binary
       17 18 19 20 21 22 23 24     17 18 19 20 21 22 23 24
  In    -  -  -  -  -  -  -  -      X  X  X  X  -  -  -  -
  Out   -  -  -  -  -  -  -  -      -  -  -  -   X  X  X  X
```

Read a Value

Read a value prompts for an address string, and then attempts to read a value from that address in the currently selected device. Values are reported in floating point, decimal, and hexadecimal format. You must know what type of data to expect to properly interpret the read results.

For example, reading an Analog Input might return:

```
Read 4 bytes, Value is 260.000000 1132593152 (0x43820000)
```

Because this is a floating-point value, only the first value (260.0) is meaningful.

Reading an integer value such as an Internal Integer (ADI) might return:

```
Read 2 bytes, Value is 0.000000 12346 (0x0000303a)
```

In this case, because the ADI is an integer value, the 0.0 is meaningless and the correct value is the decimal integer 12346.

Write a Value

Write a value prompts for an address string, data type, and value, then attempts to write the value to the specified address in the currently selected device. You must know the data type for the point for the write to succeed.

For example, to write a value to Analog Output 1:

```
Address(region[record].attr): ao[1]
Data type?
1) Bit
2) Byte
3) Integer
4) Long
5) Float
5
Value:3.141593
```

Because Analog Outputs are floating point numbers, data type 5 (float) was picked from the Data Type list in the above example.

NT System Configuration

You must specify the number of N2 System Protocol interfaces to be configured for your CIMPLICITY system. In your NT system's Control Panel, select Ports. The various ports that can be

used for device communications will be listed. The serial communication ports are listed as COM1, COM2, etc. Select one port for each physical device communication process required.

CIMPLICITY Configuration for Johnson Controls

CIMPLICITY Configuration for Johnson Controls

When configuring ports, devices, and points that use the Johnson Controls N2 Bus Communications enabler, some fields must contain unique values for the communications to work successfully. These are detailed below.

Johnson Controls Port Configuration

Johnson Controls Port Configuration

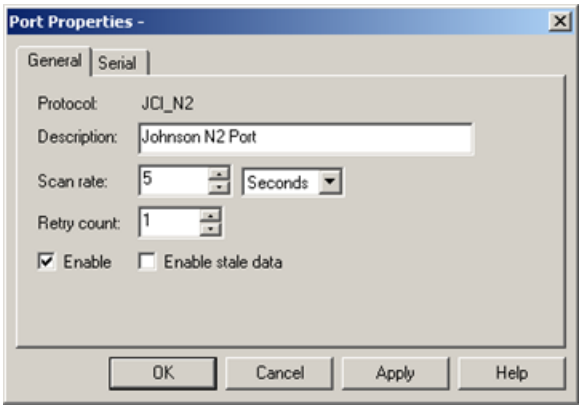
When you configure a port for Johnson Controls N2 Bus Communications, enter the following in the New Port dialog box:

1. In the **Protocol** field, select **JCI_N2** in the list of available protocols.
2. In the **Port** field, select the communication port that will be used for Johnson Controls N2 Bus Controls Communications.
3. Click OK.

The Port Properties dialog box for the protocol opens.

General Port Properties

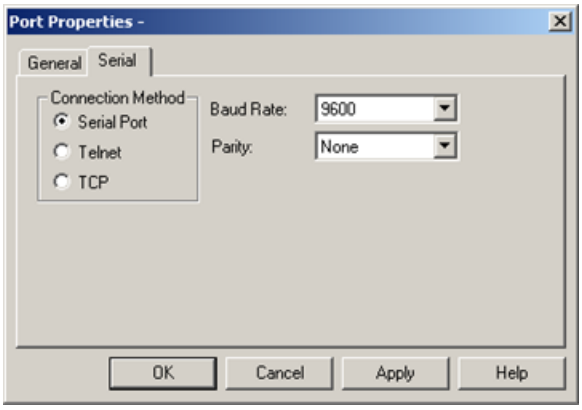
Use the General properties to enter general information for the port. You can define the following:



Description	(Optional) description to help you identify the port.
Scan Rate	Base scan rate for the port. Point scan rates are multiples of the base rate. You can specify a scan rate in **Ticks** (100 ticks = 1 second), **Seconds**, **Minutes**, or **Hours**.
Retry Count	If communications cannot be established for a device on the port, the device is considered to be down, and a $DEVICE_DOWN alarm is generated. Once a device is down, periodic attempts are made to resume connections to it.
	Enter the number of scans to wait before attempting to reconnect to a device after a communications error has been detected.
Enable	Check
	Clear

N2 Serial Connection Port

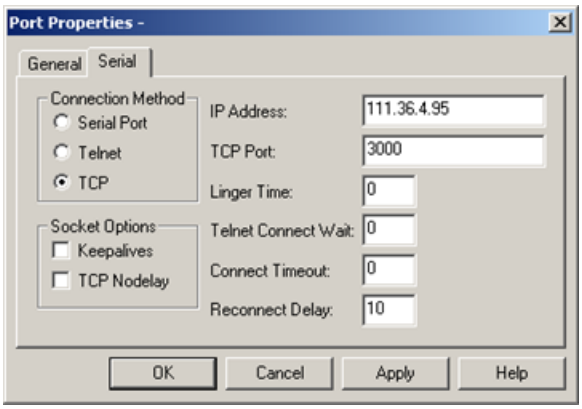
If you check the Serial Port connection method, define the following.



Baud Rate	Baud rate for communications. Click the drop-down list to the right of the field and select a rate from the list.
Parity	Parity to be used for communications.

N2 Telnet or TCP Connection Port

If you check the Telnet or TCP connection method, define the following.



Socket Options	Keepalives	Check	Keepalives will be used to detect the loss of the terminal server.
		Clear	Keepalives will not be used.
	TCP Nodelay	Check	Nodelay flag is set on the socket.
		Clear	Nodelay flag is not set.
IP Address	IP address of the terminal server.		
TCP Port	Port number of the TCP port on the terminal server.		
Linger Time	Seconds to wait after closing the socket before aborting the socket.		
Telnet Connect Wait	Seconds to wait for the Telnet protocol to initialize.		
Connect Timeout	Seconds to wait for the TCP connection to form.		
Reconnect Delay	Seconds to wait before attempting to reconnect to a device. **Note:** If you set this value to zero and the terminal server is not available, then no attempts will be made to reconnect to the terminal server.		

Johnson Controls Device Configuration

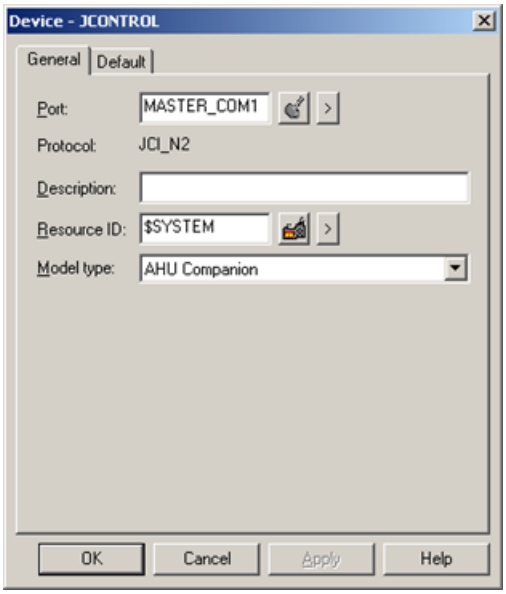
Johnson Controls Device Configuration

When you configure a device for Johnson Controls N2 Bus Communications, enter the following information in the New Device dialog box:

1. In the **Device** field, enter the name of the device you are configuring.
2. In the **Port** field, select the Johnson Controls N2 Bus Communications port to be used for the device.
3. Click OK.

The Device Properties dialog box for devices using this protocol opens.

General Device Properties

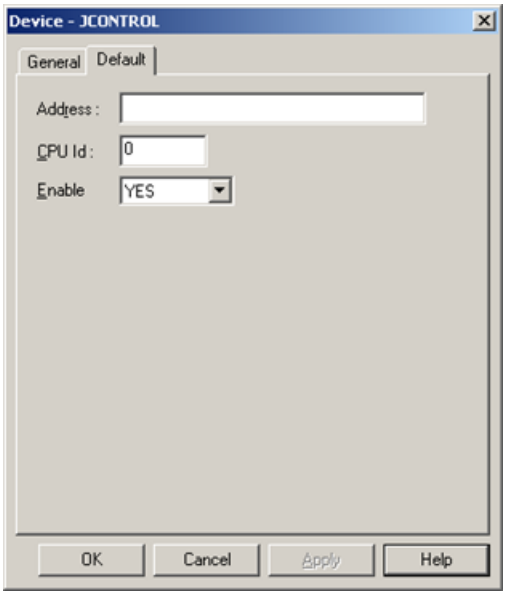


Use the General tab in the Device dialog box to enter general information for the device.

You can define the following:

Port	Select the port for this device. Click buttons to the Port field to select the port as follows: Browser or Popup Menu.
Description	Enter an optional description to help you identify the device.
Resource	Enter a resource that can be associated with this device for alarm generation. Click buttons to the right of the **Resource** field to select the resource, as follows: Browser or Popup Menu.
Model Type	Select the type of device. Click the drop-down button to the right of the input field to display your choices, and make a selection. For this protocol, the choices are: • AHU Companion • AHU Facilitator • DC9100R1 Controller • DC9100R2 Controller • DX9100R1 Companion • DX9100R1 Facilitator • DX9100R2 Companion • DX9100R2 Facilitator • N2 Base Model • N2 VND Vendor • Unitary Companion • Unitary Facilitator • VAV Companion • VAV Facilitator

Default Device Properties



Use the Default properties to enter information about the Johnson Controls N2 Bus Communications for the device. You can define the following:

Address	Not used.
CPU ID	Enter the decimal address of the device on the N2 bus.
Enable	Set this check box to enable the device when the project starts. If you clear this check box, the device will not be enabled and points associated with the device will be unavailable.

Johnson Controls Point Configuration

Johnson Controls Point Configuration

When you define a point, fields in the Point Properties dialog box have values that are unique to or have special meanings for Johnson Controls N2 Bus Communications.

General Point Properties

On the General tab in the Point Properties dialog box, you can configure points for Read or Read/ Write access.

Array points are not supported.

Set **Elements** to **1**.

Device Point Properties

On the Device tab:

Address	Enter the point address of the data as follows: `region[record].attribute` Where: **region** specifies the point type from the table. **record** is the index within the region. **attribute** specifies what information about that record is to be processed.
	For example, to access the value of the Analog Input 3, configure a point with the address: `AI[3].VALUE`
	If you do not specify an attribute, the current value is assumed. Thus, the previous example is equivalent to: `AI[3]`
	For Unitary Controllers, the record range is 1–256. See Point Types for the record number ranges in each DX9100 region.
Update Criteria	The update criteria determines how the data will be requested.
	Select **On Poll** or **On Scan** for points whose values should be polled by Johnson Controls N2 Bus Communications at regular intervals. Select **On Demand On Scan** or **On Demand On Poll** for points whose values should be polled by Johnson Controls N2 Bus Communications at regular intervals while users are viewing them.

When you are configuring Boolean points for memory types that are not bit addressable, you must also enter data in the following field:

Address Offset	Enter the bit offset that corresponds to the bit position within the word.
	The valid range for bit offset is 0-15, where 0 is the least significant bit. The bit offset must be specified as a decimal value.
	For binary I/O on Unitary Controllers, the bit offset for a point is always 6.
	For binary inputs on a DX9100, the bit offset is one less than the remainder from dividing the record number by 8. For example, BI[1] is at bit offset 0 and BO[45] is at bit offset 4 (45 % 8 = 5. 5 – 1 = 4).
	For the first 8 binary outputs on a DX9100, consult the table on page A3. For binary outputs 9 and above, the bit offset is one less than the remainder from dividing the record number by 8.

Point Types

Point Types

The UNT and DX9100 point types are organized in regions, which correspond to various types of input/output and internal value. The following table summarizes region names and corresponding CIMPLICITY point types.

Long Name	Short Name	Point Type
Analog Input	AI	FLOAT
Analog Output	AO	FLOAT
Binary Input	BI	DIGITAL
Binary Output	BO	DIGITAL
Internal Floating Point data	ADF	FLOAT
Internal Integer data	ADI	ANALOG_16
Internal Byte data	BD	ANALOG_U8
Logic Results	LRS	DIGITAL
Programmable Module Constants	PMK	ANALOG_U16
Programmable Module Outputs	PMO	FLOAT
Programmable Module Logic	PML	DIGITAL
Programmable Module Accumulator	PMA	ANALOG_U32
System Data	SD	Various
Memory	MD	Application dependent

Each region has a number of records. The exact number of records in a region depends on hardware and software configuration of the device.

Each record has a number of attributes such as current value and alarm limits. Within each region, all records have the same attributes.

Memory

UNT device memory may be read by addressing the MD region name. The device memory is treated as a continuous block of bytes from 0 to 65535. The interpretation of any byte is device dependent. No data conversion is done on MD data.

System Data

The System Data region is a pseudoregion maintained within the device communication process that contains information on the device. There is only one record in the SD region; the device information is accessed as attributes of this record.

For the UNT, the SD region provides the following data:

Description	Address	Point Type
Model ID	SD[1].DEVID	ANALOG_U8
Model name	SD[1].MODEL	TEXT[16]
Days in service	SD[1].SERVICE	ANALOG_U32
Device status	SD[1].STS	ANALOG_U32

For the DX9100, the SD region provides the following data:

Description	Address	Point Type
Model ID	SD[1].DEVID	ANALOG_U8
Firmware Revision	SD[1].FWREV	ANALOG_U8
Supervisory Control Word	SD[1].SUP	ANALOG_U16

Releasing DX9100 Output Holds

Releasing DX9100 Output Holds

When a user setpoint request for a DX9100 output point is processed, the device communication process must set the point value and a control bit. The control bit causes the output value to be held at the setpoint value and prevents the DX9100 internal programming from changing it. The hold may be manually released by clearing the control bit via a setpoint.

There are three types of control bits.

- A supervisory control bit controls the first 8 binary outputs.
- A control and status bit controls the first 8 analog outputs.
- All other outputs are controlled by a hold control bit.

Supervisory Control

The supervisory control bits are in a one-byte word. Bits 0 to 5 enable supervisory control of binary outputs 3 through 8. (Binary outputs 1 and 2 are reserved and may not be controlled.)Bits

For example, to monitor and control the supervisory status of BO[5]:

- Configure a binary point with address BO[5].HOLD and bit offset 2.
- After CIMPLICITY software sets a value in BO[5], BO[5].HOLD has a value of 1.

- To release control of BO[5] back to the DX9100 programming, set BO[5].HOLD to 0.

Analog Control and Status

Each analog outputs from 1 to 8 has its own control and status byte. The hold bit is bit 1.

For example, to monitor and control the hold status of AO[2]:

- Configure a point with address AO[2].HOLD, bit offset 0.
- After CIMPLICITY software sets a value in AO[2], AO[2].HOLD has a value of 1.
- To release control of AO[2] back to the DX9100 programming, set AO[2].HOLD to 0.

Hold Control

After BO[8] and AO[8], each group of 8 outputs is controlled by a single hold control byte. The HOLD attribute of several different records map to the same byte and control of individual outputs is accomplished via bit offsets.

The bit offset is one less than the remainder from dividing the record number by 8. For example, the hold control bit for BO[21] is at bit offset 4 and the hold control bit for AO[9] is at bit offset 0.

For example, to monitor and control the hold status of BO[23]:

- Configure a point with address BO[23].HOLD, bit offset 6.
- After CIMPLICITY software sets a value in BO[23], BO[23].HOLD has a value of 1.
- To release control of BO[23] back to the DX9100 programming, set BO[23].HOLD to 0.

DX9100 I/O Map

DX9100 I/O Map

The following table shows the valid range of record numbers in each of the DX9100 regions. Depending on the configuration of extension modules installed in the DX9100, not all records may be available in any particular device.

Region	Valid Range	Notes
AI	**1–72**	Read only
BI	**1–72**	Read only
AO	**1–72**	Setpoint sets hold bit.

BO	**1–72**	Setpoint sets hold bit.
ADF	**1–8**	
ADI	**1–72**	
BD	**1–32**	
LRS	**1–64**	Read only
PMK	**1–240**	These values are stored in EEPROM. They are writable but setpoint should be limited in frequency.
PMO	**1–96**	
PML	**1–96**	
PMA	**1–96**	

The following tables show the mapping between the regions and record used in CIMPLICITY point addresses, and the hardware address of points in the DX9100. These mappings are the same as that of the Johnson Controls Metasys Companion product.

Region Analog Input (AI)

The Region Analog Inputs are:)

Record	R/W	Item Description
1–8	**R**	Analog Input #1–8
9–16	**R**	Analog Input #1–8 – Expander #1
17–24	**R**	Analog Input #1–8 – Expander #2
25–32	**R**	Analog Input #1–8 – Expander #3
33–40	**R**	Analog Input #1–8 – Expander #4
41–48	**R**	Analog Input #1–8 – Expander #5
49–56	**R**	Analog Input #1–8 – Expander #6
57–64	**R**	Analog Input #1–8 – Expander #7
65–72	**R**	Analog Input #1–8 – Expander #8

Region Binary Input (BI)

The Region Binary Inputs are:)

Record	Bit	R/W	Item Description
1–8	**0–7**	**R**	Binary (Digital) Input #1–8

9–16	**0–7**	**R**	Binary (Digital) Input #1–8 – Expander #1
17–24	**0–7**	**R**	Binary (Digital) Input #1–8 – Expander #2
25–32	**0–7**	**R**	Binary (Digital) Input #1–8 – Expander #3
33–40	**0–7**	**R**	Binary (Digital) Input #1–8 – Expander #4
41–48	**0–7**	**R**	Binary (Digital) Input #1–8 – Expander #5
49–56	**0–7**	**R**	Binary (Digital) Input #1–8 – Expander #6
57–64	**0–7**	**R**	Binary (Digital) Input #1–8 – Expander #7
65–72	**0–7**	**R**	Binary (Digital) Input #1–8 – Expander #8

Region Analog Output (AO)

Setting AO valves sets an output hold that must be manually reset to allow algorithm control of the point. See Releasing DX9100 Output Holds for details.)

Record	R/W	Item Description
1	**W**	Analog Output #1
2	**W**	Analog Output #2
3	**W**	Analog Output #9
4	**W**	Analog Output #10
5	**W**	Analog Output #11
6	**W**	Analog Output #12
7	**W**	Analog Output #13
8	**W**	Analog Output #14
9–16	**W**	Analog Output #1–8 – Expander #1
17–24	**W**	Analog Output #1–8 – Expander #2
25–32	**W**	Analog Output #1–8 – Expander #3
33–40	**W**	Analog Output #1–8 – Expander #4
41–48	**W**	Analog Output #1–8 – Expander #5
49–56	**W**	Analog Output #1–8 – Expander #6
57–64	**W**	Analog Output #1–8 – Expander #7
65–72	**W**	Analog Output #1–8 – Expander #8

Region Binary Output (BO)

Setting BO valves sets an output hold that must be manually reset to allow algorithm control of the point. See Releasing DX9100 Output Holds for details.)

Record	Bit	R/W	Item Description
1	**6**	**W**	Shut Off Mode
2	**7**	**W**	Start Up Mode
3	**0**	**W**	Binary (Digital) Output #3
4	**1**	**W**	Binary (Digital) Output #4
5	**2**	**W**	Binary (Digital) Output #5
6	**3**	**W**	Binary (Digital) Output #6
7	**4**	**W**	Binary (Digital) Output #7
8	**5**	**W**	Binary (Digital) Output #8
9–16	**0–7**	**W**	Binary (Digital) Output #1–8 – Expander #1
17–24	**0–7**	**W**	Binary (Digital) Output #1–8 – Expander #2
25–32	**0–7**	**W**	Binary (Digital) Output #1–8 – Expander #3
33–40	**0–7**	**W**	Binary (Digital) Output #1–8 – Expander #4
41–48	**0–7**	**W**	Binary (Digital) Output #1–8 – Expander #5
49–56	**0–7**	**W**	Binary (Digital) Output #1–8 – Expander #6
57–64	**0–7**	**W**	Binary (Digital) Output #1–8 – Expander #7
65–72	**0–7**	**W**	Binary (Digital) Output #1–8 – Expander #8

Region Internal Floating Point Data (ADF)

The Region Internal Floating-Point Data is:

Record	R/W	Item Description
1–8	W	Analog Constant #1–8

Region Internal Integer Data (ADI)

The Region Internal Integer Data are:

Record	R/W	Item Description
1–8	**W**	DI1–DI8 Pulse Count
9–16	**W**	DI1–DI8 Pulse Count – Expander #1

17–24	**W**	DI1–DI8 Pulse Count – Expander #2
25–32	**W**	DI1–DI8 Pulse Count – Expander #3
33–40	**W**	DI1–DI8 Pulse Count – Expander #4
41–48	**W**	DI1–DI8 Pulse Count – Expander #5
49–56	**W**	DI1–DI8 Pulse Count – Expander #6
57–64	**W**	DI1–DI8 Pulse Count – Expander #7
65–72	**W**	DI1–DI8 Pulse Count – Expander #8

Region Internal Byte Data (BD)

The Region Internal Byte Data are:

Record	Bit	R/W	Item Description
1–16	**0–15**	**W**	Logic Constant #1–16
17–32	**0–15**	**W**	Logic Constant #17–32

Region Logic Results (LRS)

The Region Logic Results are:

Record	Bit	R/W	Item Description
1–16	**0–15**	**R**	Logic Result #1–16
17–32	**0–15**	**R**	Logic Result #17–32
33–48	**0–15**	**R**	Logic Result #33–48
49–64	**0–15**	**R**	Logic Result #49–64

Region Programmable Module Constants (PMK)

The Region Programmable Module Constants are:

Record	R/W	Item Description
1–20	**W**	Module #1 – Constant #K1–K20
21–40	**W**	Module #2 – Constant #K1–K20
41–60	**W**	Module #3 – Constant #K1–K20
61–80	**W**	Module #4 – Constant #K1–K20

81–100	**W**	Module #5 – Constant #K1–K20
101–120	**W**	Module #6 – Constant #K1–K20
121–140	**W**	Module #7 – Constant #K1–K20
141–160	**W**	Module #8 – Constant #K1–K20
161–180	**W**	Module #9 – Constant #K1–K20
181–200	**W**	Module #20 – Constant #K1–K20
201–220	**W**	Module #11 – Constant #K1–K20
221–240	**W**	Module #12 – Constant #K1–K20

Region Programmable Module Outputs (PMO)

Setting PMO valves sets an output hold that must be manually reset to allow algorithm control of the point. See Releasing DX9100 Output Holds for details.

Record	R/W	Item Description
1–8	**W**	Module #1 Output – Channel #1–8
9–16	**W**	Module #2 Output – Channel #1–8
17–24	**W**	Module #3 Output – Channel #1–8
25–32	**W**	Module #4 Output – Channel #1–8
33–40	**W**	Module #5 Output – Channel #1–8
41–48	**W**	Module #6 Output – Channel #1–8
49–56	**W**	Module #7 Output – Channel #1–8
57–64	**W**	Module #8 Output – Channel #1–8
65–72	**W**	Module #9 Output – Channel #1–8
73–80	**W**	Module #10 Output – Channel #1–8
81–88	**W**	Module #11 Output – Channel #1–8
89–96	**W**	Module #12 Output – Channel #1–8

Region Programmable Module Logic (PML)

Setting PML valves sets an output hold that must be manually reset to allow algorithm control of the point. See Releasing DX9100 Output Holds for details.

Record	R/W	Item Description
1–8	**W**	Module #1 DO – Channel #1–8

9–16	**W**	Module #2 DO – Channel #1–8
17–24	**W**	Module #3 DO – Channel #1–8
25–32	**W**	Module #4 DO – Channel #1–8
33–40	**W**	Module #5 DO – Channel #1–8
41–48	**W**	Module #6 DO – Channel #1–8
49–56	**W**	Module #7 DO – Channel #1–8
57–64	**W**	Module #8 DO – Channel #1–8
65–72	**W**	Module #9 DO – Channel #1–8
73–80	**W**	Module #10 DO – Channel #1–8
81–88	**W**	Module #11 DO – Channel #1–8
89–96	**W**	Module #12 DO – Channel #1–8

Region Programmable Module Accumulator (PMA)

The Region Programmable Module Accumulators are:

Record	R/W	Item Description
1–8	**W**	Module #1 Accumulator – Channel #1–8
9–16	**W**	Module #2 Accumulator – Channel #1–8
17–24	**W**	Module #3 Accumulator – Channel #1–8
25–32	**W**	Module #4 Accumulator – Channel #1–8
33–40	**W**	Module #5 Accumulator – Channel #1–8
41–48	**W**	Module #6 Accumulator – Channel #1–8
49–56	**W**	Module #7 Accumulator – Channel #1–8
57–64	**W**	Module #8 Accumulator – Channel #1–8
65–72	**W**	Module #9 Accumulator – Channel #1–8
73–80	**W**	Module #10 Accumulator – Channel #1–8
81–88	**W**	Module #11 Accumulator – Channel #1–8
89–96	**W**	Module #12 Accumulator – Channel #1–8

Chapter 27. Marquee Introduction

About the Marquee Driver

The CIMPLICITY Base System functionality – Point Management, Alarm Management, Data Logging and Reporting, and a fully functioned User Interface – enables CIMPLICITY users to collect data, then visualize it via lists, graphic status displays, alarms, and reports. Standard data communications capabilities make CIMPLICITY software a factory floor tool that can provide services such as the following:

- Downtime reporting
- Production reporting
- Records of production counts at workstations
- Graphic Monitoring of automatic data point values
- Fault reporting via direct point values and alarms

The Marquee Driver provides the ability to configure ports for and messages to be displayed on marquees. This optional driver enhances the existing port and printer configuration.

While GE Intelligent Platforms does not endorse or claim compatibility with all marquees, the CIMPLICITY Marquee Driver works with marquees produced by UTICOR Technology, Inc., Total Control Products, Inc., Static Controls Corporation, and American Ledgible, Inc.

You configure the Marquee Driver in the same way you configure CIMPLICITY devices, ports, points and alarms. You will need to configure the following marquee information for your project:

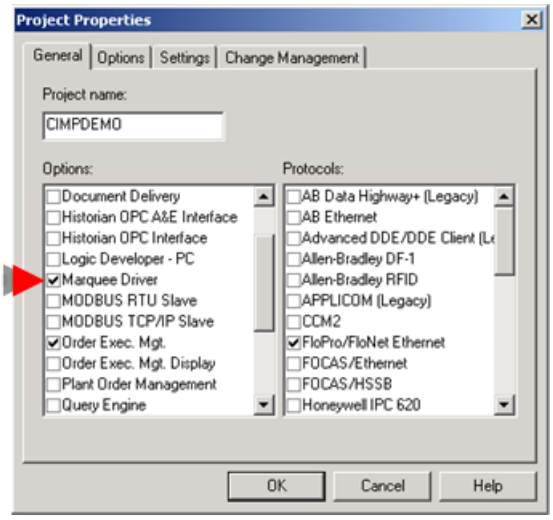
	Marquee Types	All marquee devices that use the same header, footer, message wrap, empty message, and attributes belong to the same Marquee Type. Define these settings for each Marquee Type used by the project.
	Marquee Ports	Configure the PC communication port or the network communication port to which one or more serial marquee devices are connected. Generally, if more than one device is on a port, the header and/or footer information (from the Marquee Type) indicate on which device the message is to be displayed.
	Marquee Devices	Assign a Marquee ID, Marquee Type, Marquee Port, header, footer, empty message, and display time for each marquee device within a project. (Optional) associate sets of marquees into Marquee Groups.
	Marquee Messages	(For each Marquee message) assign a message ID, Alarm ID, alarm state, message header, message footer, message text, the marquees on which to display the message, the attributes associated with the message and any CIMPLICITY point values to be displayed with the message.

Add the Marquee Driver to a Project

1. Open the CIMPLICITY Workbench for your project.
2. Click the Project>Properties on the Workbench menu bar.

   The Project Properties dialog box opens.

3. Select the General tab.



4. Check Marquee Driver.
5. Click OK.

The Marquee Driver is now available for configuration.

Chapter 28. Marquee Devices

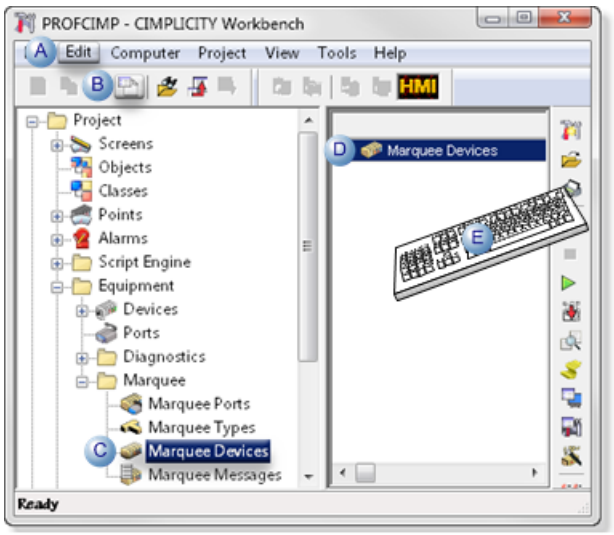
Marquee Device Configuration

Each physical marquee device controlled by a given CIMPLICITY system is assigned a unique ID along with the port to which it is attached and the type of device it is. Additionally, you may group together one or more marquee devices.

1 *(page 517)*	Open the Marquee Groups - Configuration window.
2 *(page 518)*	Change Marquee device configuration display attributes.
3 *(page 519)*	Add a Marquee device.
4 *(page 520)*	Modify a Marquee device.
5 *(page 521)*	Cut a Marquee device.
6 *(page 521)*	Copy a Marquee device.
7 *(page 522)*	Search the Tree View.
8 *(page 522)*	Add a Marquee device group..
9 *(page 523)*	Rename a Marquee device group.
10 *(page 523)*	Cut a Marquee device group.
11 *(page 524)*	Copy a Marquee device group.

1. Open the Marquee Groups-Configuration Window

1. Select **Project>Equipment>Marquee>Marquee Devices** in the Workbench left pane.
2. Do one of the following.



A	Click Edit>Properties on the Workbench menu bar.	
B	Click the Properties button on the Workbench toolbar.	
C	In the Workbench left pane:	
	Either	Or
	Double click **Marquee Devices**.	a. Right-click **Marquee Devices**. b. Select Properties on the Popup menu.
D	In the Workbench right pane:	
	Either	Or
	Double click **Marquee Devices**.	a. Right-click **Marquee Devices**. b. Select Properties on the Popup menu.
E	Press Alt+Enter on the keyboard.	

3. Right-click **Marquee Devices**.
4. Select Properties on the Popup menu.
5. Right-click **Marquee Devices**.
6. Select Properties on the Popup menu.

2. Change Marquee Device Configuration Display Attributes

- Add an attribute to the display list.
- Add All Attributes to the display list.
- Remove an Attribute from the display list .
- Remove All Attributes from the display list.
- Reorder Attributes in the display attributes list

Add an Attribute to the Display List

1. Select the attribute in the **Available Attributes** box.
2. Click **Add**.

   The attribute will be placed on the **Display Attributes** list just after the currently highlighted attribute on that list.

   Add All Attributes to the Display List

   Click **Add All**.

   All attributes will then be placed in the **Display Attributes** list.

   Remove an Attribute from the Display List

3. Select the attribute in the Display Attributes box.
4. Click **Remove**.

   The attribute will be placed on the Available Attributes list.

   Remove All Attributes from the Display List

   Click **Remove All**.

   Reorder Attributes in the Display Attributes List

5. Select the attribute.
6. Click **Move Up** to move the attribute toward the front of the list.

7. Click **Move Down** to move the attribute toward the end of the list.

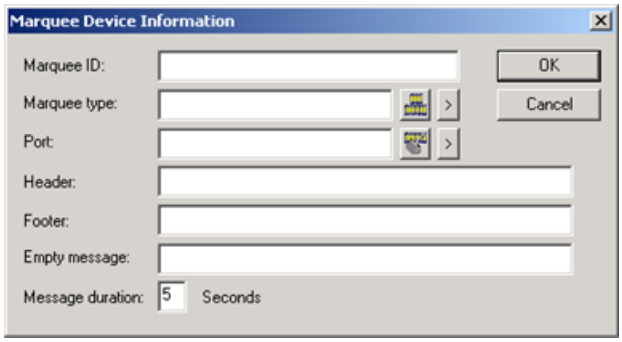
3. Add a Marquee Device

3. Add a Marquee Device

1. Select the Devices folder to add a device that is not currently associated with a device group, or select the name of the device group in which you would like to create the device.

2. Do one of the following:
   - Select New->Device from the File menu.
   - Click the **New** button on the toolbar.
   - Select New-> Device from the popup menu.

   The Marquee Device Information dialog box opens.



3. Enter the following information to define a new Marquee Device:
   a. Enter the name for this Marquee Device in the **Marquee ID** field. This ID can be up to 16 characters in length.

   The new Marquee ID must be unique. If you enter an existing Marquee ID, you will receive and error message in a popup window.

   a. Enter the name of an existing Marquee Type in the **Marquee type** field. You can also:

   - Click the Marquee Type button to the right of the input field to use the Select a Marquee Type Browser to display the current list of Marquee Types and select one.
   - Click the **Popup Menu** button to create a new Marquee Type or browse for a Marquee Type.

   a. Enter the name of an existing Marquee Port in the **Port** field. You can also:

   - Click the Port button to the right of the input field to use the Select a Marquee Port Browser to display the current list of Marquee Ports and select one.

- Click the Popup Menu button  to create a new Marquee Port or browse for a Marquee Port.

a. Enter the device-specific header in the **Header** field.
b. Enter the device-specific footer in the **Footer** field.
c. Enter the device-specific empty message in the **Empty message** field.
d. Enter the device-specific duration for messages displayed on this device in the **Message duration** field. By default, this field is set to 5 seconds, and may be set to any integer value between 1 and 99 seconds.

Marquee Control Characters

The **Header**, **Footer**, and **Empty message** fields may include the following control characters. Note that not all marquees support all these control characters.

%\n	newline
%\r	carriage return
%\b	backspace
%\v	vertical tab
%\t	horizontal tab
%\f	formfeed
%^[	generates an escape
%^ <character>	generates a control character for any alphabetic character
%x <xx>	generates any hex character between 00-FF. <xx> should be the 2 hex characters.
	To generate the null character (00), you must configure the MARQ_RESERVED_NULL_CHAR global parameter .
% <xx>	generates the control characters you specified for message attribute <xx> in the Message Attributes box, where <xx> is a two-digit number between 01 and 25. For example, to generate the control characters you specified for fifth attribute in the list, enter %05.

4. Modify a Marquee Device

To modify information for an existing marquee device, you can double-click on the device in the tree or attribute view, or select the device in the tree view, and then do one of the following:

- Select Properties from the Edit menu
- Click the **Modify** button on the toolbar.

Alternatively, you can:

- Move the cursor to the device name and click the right mouse button to open the popup menu and select Properties.

The Marquee Device Information *(page 519)* dialog box for the selected port opens.

You can change any of the information in this dialog box.

5. Cut a Marquee Device

1. Select the device in the tree view.
2. Do one of the following.
   - Select Cut from the Edit menu.
   - Click the **Cut** button on the toolbar.
   - Select Cut from the popup menu.

   Note: If you cut a marquee device from a device group, this has no effect on the other device groups that contain this device, or on the existence of the device within the **Devices** folder.

   You can only cut a marquee device from the Devices folder if it is not contained in any device group and there are no marquee messages configured to be routed to this device. When you cut a marquee device from the Devices folder, you permanently delete it from the configuration.

6. Copy a Marquee Device

1. Select the device in the tree view.
2. Do one of the following.
   - Select Copy from the Edit menu.
   - Click the **Copy** button on the toolbar.
   - Select Copy from the popup menu.
3. Select either the Devices folder or the specific device group into which you wish to paste the device.
4. Do one of the following:
   - Select Paste from the Edit menu.

- Click the **Paste** button on the toolbar.
- Select Paste from the popup menu.

If the folder or group you are copying the device to already contains a device with the same name, the new device is named **?????**.

5. Enter the new device name. If you attempt to select anything else in the window, you will be prompted to enter a valid device name.

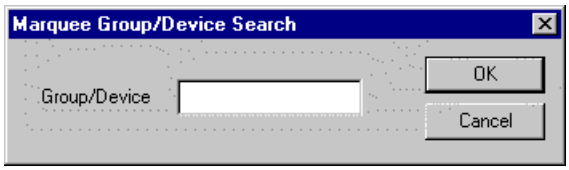
7. Search the Tree View

You can use the **Search** function to quickly find a specific Marquee Device or Device Group.

1. Select the **Devices** folder in the tree view.

2. Do one of the following
   - Select Search from the View menu.
   - Click the **Search** button on the toolbar.

   The Marquee Group/Device Search dialog box opens.



3. Enter the specific marquee device or device group name for which you wish to search.

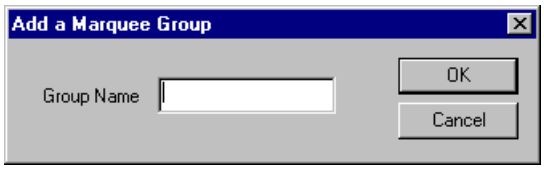
If an exact match is found, the name of the device or device group will be highlighted in the tree view.

8. Add a Marquee Device Group

1. Select the Devices folder.

2. Do one of the following:
   - Select New->Group from the File menu.
   - Select New->Group from the popup menu.

   The Add a Marquee Group dialog box opens.



3. Enter a unique name for your group.
4. Click **OK**.

The new group will be added to the tree view.

To close the dialog box without adding a group, click **Cancel**.

9. Rename a Marquee Device Group

1. Select the name of the device group.
2. Select Rename from the popup menu.

   An edit box will appear around the name of the device group.

3. Enter a new name.
4. Press **Enter** to save your changes.

10. Cut a Marquee Device Group

1. Select the name of the device group.
2. Do one of the following.
   - Select Cut from the Edit menu.
   - Click the **Cut** button on the toolbar.
   - Select Cut from the popup menu.

Note: A device group may only be cut if it currently contains no devices, and there are no marquee messages routed to it.

11. Copy a Marquee Device Group

1. Select the name of the device group.

2. Do one of the following.
   - Select Copy from the Edit menu.
   - Click the **Copy** button on the toolbar.
   - Select Copy from the popup menu.

3. Select the Devices folder.

4. Do one of the following.
   - Select Paste from the Edit menu.
   - Click the **Paste** button on the toolbar.
   - Select Paste from the popup menu.

   If the Devices folder already contains a group with the same name, the new group is named **?????**.

5. Enter the new group name. If you attempt to select anything else in the window, you will be prompted to enter a valid group name.

Chapter 29. Marquee MarqAtts.cfg File

Add Entries to the MarqAtts.cfg File

You may configure the attributes that are displayed in the Marquee Types and Marquee Messages screens. This file is created with two default attributes pre-configured for you: **Bold** and **Blink**.

Using a text editor, you may modify this file to contain a total of 25 attributes. Modify the file in your project's **master** directory. The default contents of the file are displayed below. Pay particular attention to the comments at the top of this file:

```
|-*
* This file contains project-wide attributes for Marquee Devices.
* Please only enter one attribute per line.
* The provided defaults can be modified.
* Only the first 25 attributes will be used by the Marquee processes.
Bold
Blink
```

Chapter 30. Marquee Messages

Marquee Message Configuration

Marquee messages are used to indicate what alarm conditions should trigger a display on a marquee device and what should be displayed. You can also configure continuous messages that aren't triggered by alarm conditions, but continuously communicate useful information about production conditions.

Balancing a point name length with other information that users need to access is a main consideration when planning to incorporate point IDs in a Marquee message.

Note: Beginning with CIMPLICITY v9.0 the point ID length can be a maximum of 256 characters.

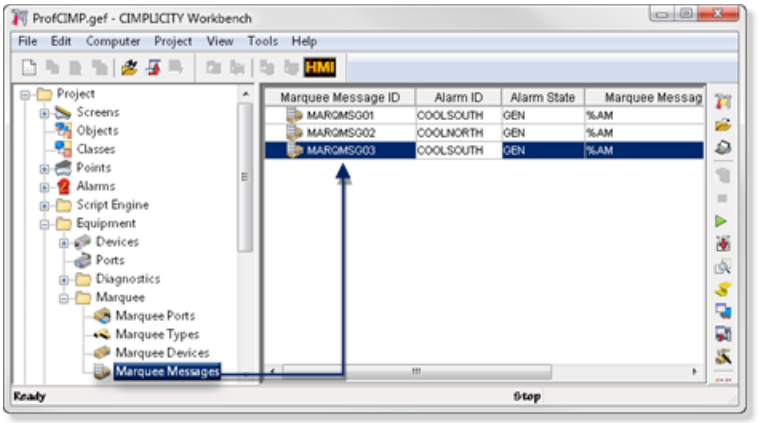
If a long point name is configured to scroll on a fixed width Marquee screen, it can possibly take too much time before the actual message to be delivered scrolls by. This, of course, is counter-productive to the Marquee purpose.

9 *(page 539)*	Change Marquee Message configuration display fields .

1. View existing Marquee Messages

Select **Project>Equipment>Marquee>Marquee Messages** in the Workbench left pane.

Result: Existing Marquee messages are listed in the Workbench right pane.



The Marquee Messages grid always displays the following:

Marquee Message ID	The identifier for a marquee device.

In addition, you can choose to display:

Alarm ID	An alarm identifier.
Alarm State	The alarm status that causes the alarm to be sent to the Marquee Driver
Alarm Footer	The footer string for the alarm message.
Alarm Header	The header string for the alarm message.
Marquee Message	The alarm message to be displayed.
Message Disabled	This flag indicates whether or not the selected message is enabled or disabled.

The list of Marquee Messages is initially sorted by **Marquee ID**, **Alarm ID** and **Alarm Value**. You may click on any of the other title buttons at the top of the grid to sort the list by that attribute.

2. Change Marquee Message Configuration Display Fields

- Add a field to the Display list .
- Remove a field from the Display list.
- Reorder fields in the Display Fields list .

Add a Field to the Display List

1. Select the attribute in the **Available Field** box.
2. Click **Add**.

   Result: The field will be placed on the Display Fields list just after the currently highlighted field on that list.

   Remove a Field from the Display List

3. Select the field in the **Display Fields** box.
4. Click **Remove**.

   Result: The field will be placed on the Available Field list.

   Note: **Marquee ID** cannot be removed from the display list.

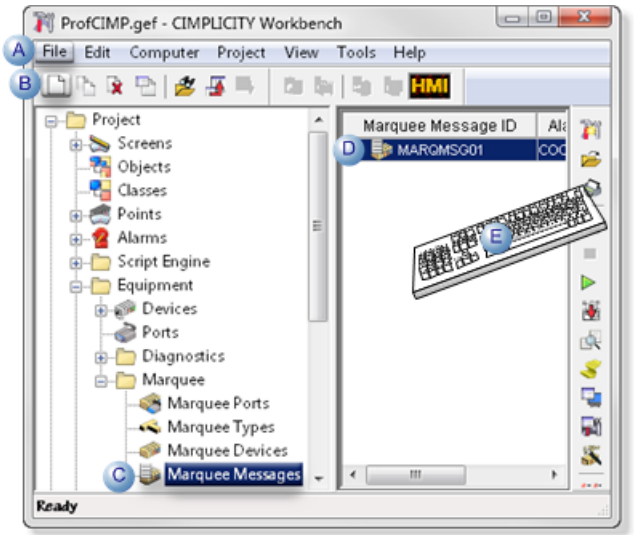
   Reorder Fields in the Display Fields List

5. Select the field name in the Display Fields list.
6. Click **Move Up** to move the field toward the front of the list.
7. Click **Mov Down** to move the field toward the end of the list.

3. Add a Marquee Message

3. Add a Marquee Message

1. Select **Project>Equipment>Marquee>Marquee Messages** in the Workbench left pane.
2. Do one of the following.

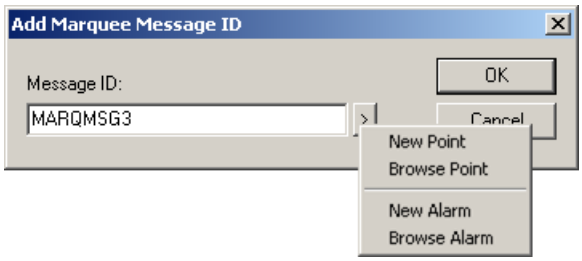


A	Click File>New on the Workbench menu bar.	
B	Click the New Object button on the Workbench toolbar.	
C	In the Workbench left pane:	
	Either	Or
	Double click **Marquee Messages**.	a. Right-click **Marquee Messages**. b. Select New on the Popup menu.
D	a. In the Workbench right pane. a. Right-click any Marquee message. b. Select New on the Popup menu.	
E	Press Ctrl+N on the keyboard.	

The Add Marquee Message ID dialog box opens.

3. Right-click **Marquee Messages**.
4. Select New on the Popup menu.
5. Right-click any Marquee message.
6. Select New on the Popup menu.
7. Enter a unique message ID in the **Message ID** field.

   The message ID can be as follows.



Message ID Options	Enter a:
Unique name	Unique name.
Point ID	Point ID or use the Popup menu to create a new point or select an existing point.
Alarm ID	Alarm ID or use the Popup menu to create a new point or select an existing point.

8. Click OK.

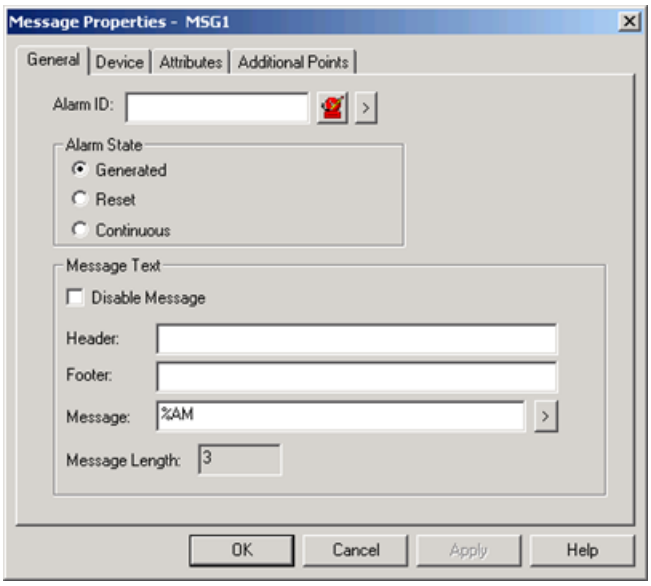
The Message Properties dialog box opens.

3.1 *(page 530)*	General message properties.
3.2 *(page 532)*	Device properties.
3.3 *(page 533)*	Attributes properties.
3.4 *(page 534)*	Additional points.

Note: Click Cancel to close the Add marquee Message ID dialog box without adding a new message.

3.1. General Message Properties

Use the General tab of the Message Properties dialog box to configure general message properties.



Note: You cannot configure two marquee messages with the same Message ID, Alarm ID, and alarm state value (Generated or Reset) combination. If you attempt to do this, you will receive an error message in a popup window.

When the alarm is in the state you select, the **Message** (or alarm message, if **Message** is blank) is displayed only if the **Message wrap** field for the Marquee Type contains the **%MG** control character. The information in the **Header** and **Footer** fields is always displayed.

Enter the following information on the General tab:

Alarm ID

Enter the CIMPLICITY Alarm ID that will trigger this marquee message. You can also:

- Click the Alarm button to the right of the input field to select from an existing set of alarms.
- Click the Popup Menu button to create a new alarm.

If you specified a Point ID for the name of the marquee message, and there is an existing Alarm ID configured for that Point ID, then this **Alarm ID** field is set to the value of the existing Alarm ID.

Alarm State

Select one of the following states:

Generated	Select this option if you want to receive the Alarm ID when the alarm is generated.
Reset	Select this option if you want to receive the Alarm ID when the alarm is reset

Continuous	Select this option if you want to display the message continuously on your system from startup. Note that when you select this option, the Alarm ID field becomes inactive.

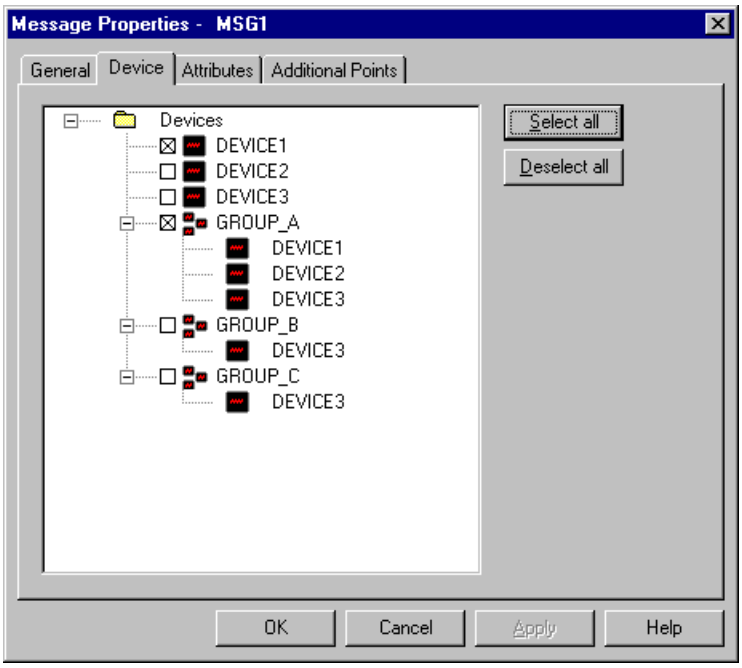
Message Text

Enter information in the following fields:

Disable Message	Select this option to disable the display of this alarm message.
Header	Enter the header message to be displayed on the marquee prior to the alarm message. The header message may be up to 80 characters long. You may include marquee control characters in the message.
Footer	Enter the footer message to be displayed on the marquee after the alarm message. The footer message may be up to 80 characters long. You may include marquee control characters in the message. If you leave this field blank, no footer is displayed.
Message	Enter the message to be displayed on the marquee when the alarm occurs. The message may be up to 80 characters long. You may include marquee control characters in the message. If you leave this field blank, the alarm message is displayed on the marquee. By default, this field will be set to **%AM**, which will display the alarm string. You can click the Popup Menu button to the right of this field  and select **Load Alarm Message** to load the raw alarm message associated with the Alarm ID you specified above, and use it as a starting point for configuring this marquee message.
Message Length	This field, which you cannot edit, displays the number of characters that you have entered in the **Message** field.

3.2. Device Properties

Use the Device tab of the Message Properties dialog box to select the marquee devices or device groups that will display this message. Each device or device group that currently displays this message has its check box set.



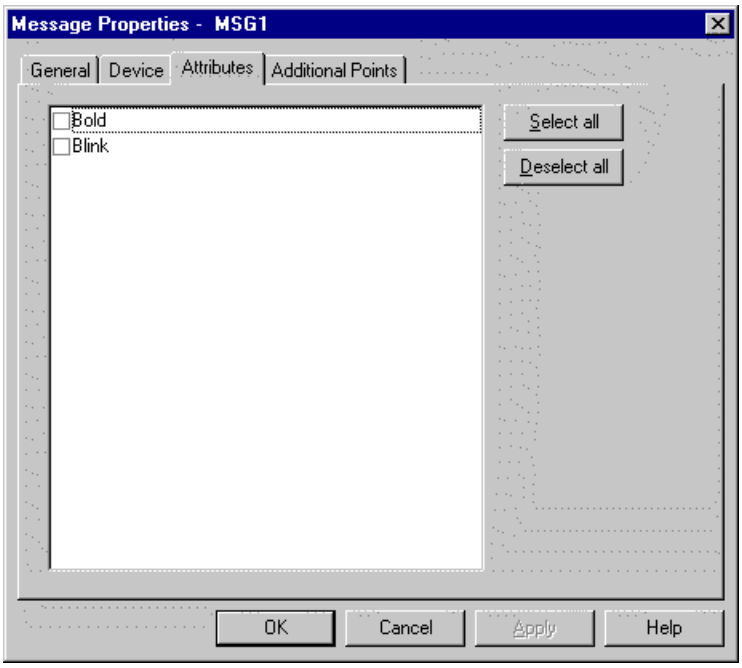
To change the state of a check box, move the cursor to the box and click once.

You can also:

- Click **Select all** to select the check boxes for all marquee devices and device group.
- Click **Deselect all** to clear the check boxes for all devices and device groups in the list.

3.3. Attributes Properties

Use the Attributes tab of the Message Properties dialog box to select the display attributes for this message.



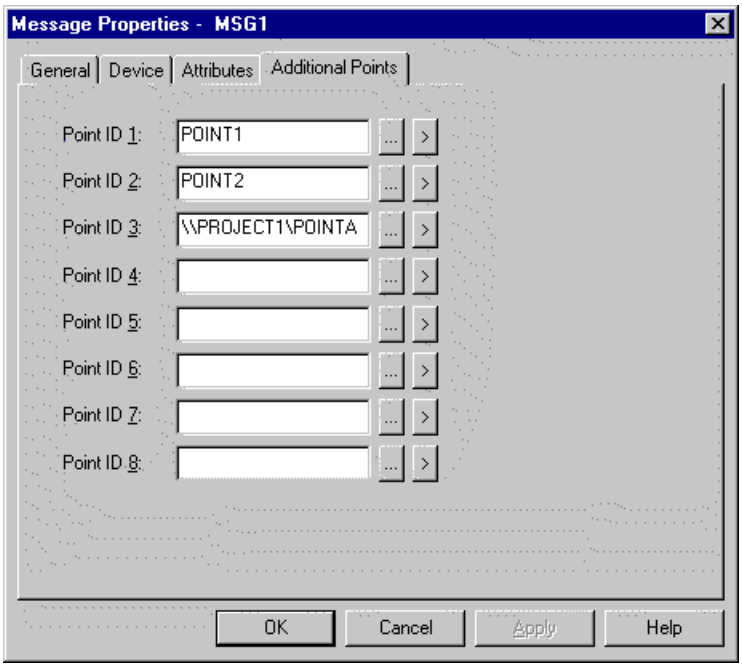
Select the check box for each attribute you want to assign to this message.

For example, if you select the **Bold** check box, then this marquee message displays in bold on each marquee.

You can also click **Select all** to select all the attribute check boxes, or click **Deselect all** to clear all the attribute check boxes.

3.4. Additional Points

Use the Additional Points tab of the Message Properties dialog box to define up to 8 additional CIMPLICITY points whose values can be displayed in this message:



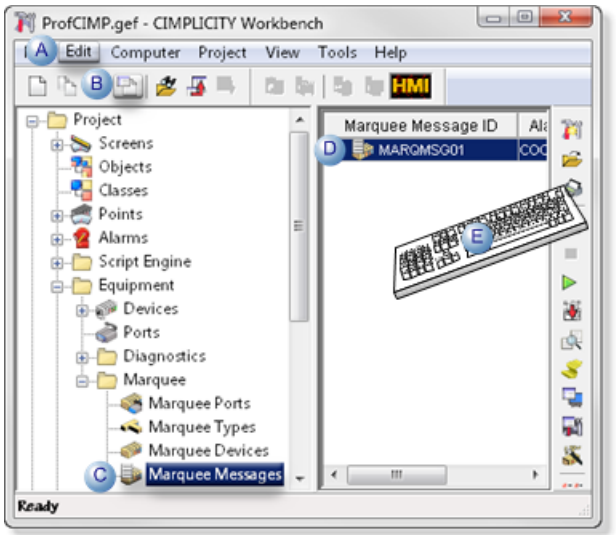
You can use tokens **%V1** through **%V8** to display the values of these additional points in the **Message** field on the General property page. For example, to display the value of **Point ID 5**, enter the token **%V5** in the **Message** field.

Additional points may have the following properties:

- They may be local, remote, or enterprise points.
- They may be Analog, Boolean, or Text points.
- They may be array points. For points of type Analog or Boolean, only the first element of the array will be displayed. For points of type Text, only the first 20 characters of the text point will be displayed.
- If the value of an additional point is not currently available within CIMPLICITY software, it will be displayed on the marquee as three asterisks (***).

4. Modify a Marquee Message

1. Select **Project>Equipment>Marquee>Marquee Messages** in the Workbench left pane.
2. Select a Marquee message.
3. Do one of the following.



A	Click Edit>Properties on the Workbench menu bar.	
B	Click the Properties button on the Workbench toolbar.	
C	In the Workbench left pane: a. Right-click **Marquee Messages**. b. Select Properties on the Popup menu.	
D	In the Workbench right pane:	
	Either	Or
	Double click a Marquee message.	a. Right-click a Marquee message.. b. Select Properties on the Popup menu.
E	Press Alt+Enter on the keyboard.	

4. Right-click **Marquee Messages**.
5. Select Properties on the Popup menu.
6. Right-click a Marquee message..
7. Select Properties on the Popup menu.

5. Delete a Marquee Message

1. Select the message in the Marquee Messages window.
2. Do one of the following.
   - Select Delete on the Edit menu.
   - Click the **Delete** button on the toolbar.

- Press the **Delete** key.

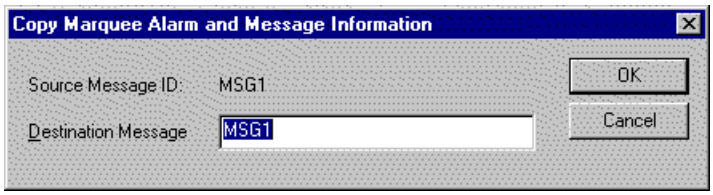
The Delete dialog box opens.

Click **Yes** to verify that you want to delete the message. Click **No** if you want to close the dialog box without deleting the message.

6. Copy a Marquee Message

1. Select the message in the Marquee Messages grid, and then do one of the following:
   - Select Duplicate on the Edit menu
   - Click the **Duplicate** button on the toolbar.
   - Press **Ctrl+D**.

   The Copy Marquee Message dialog box opens.



2. Enter the new Message ID in the **Destination Message** field.

3. Click **OK** to create the new message.

The new message contains the properties as the source message.

You can then open the Message Properties dialog box for the new message and make any further changes you require.

Note: You cannot configure two marquee messages with the same Marquee ID, Alarm ID, and alarm state value (Generated or Reset) combination. If you attempt to do this, you will receive an error message in a popup window.

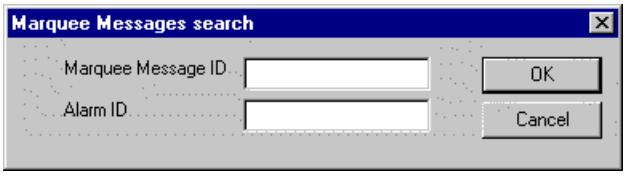
7. Filter the Marquee Message List

The Marquee Message list can contain many messages. You can use the **Search** function to quickly find a specific Marquee Message or subset of Marquee Messages.

To initiate a search, do one of the following.

- Select Search on the View menu.
- Click the **Search** button on the toolbar.

After you initiate the request, the Marquee Messages search dialog box opens.



You can filter the list by any combination of:

- Marquee Message ID
- Alarm ID

You can use the following wildcards:

*	Use this wildcard to search for any number of characters at this point in the string. For example, if you want to display a list of Marquee Messages for Marquee Devices that start with "M" and end with "X", enter M*X in the Marquee ID field.
?	Use this wildcard to search for any character in this place in the string. For example, if you want to display the list of Marquee Messages for Marquee Devices whose names are three characters long, and whose first character is "M" and third character is "X", enter M?X in the Marquee ID field.

Note: There are no implied wildcards. If you do not include or terminate your search string with an asterisk, only those items that match your request exactly will be returned.

Click **Cancel** to close the dialog box and do no further filtering.

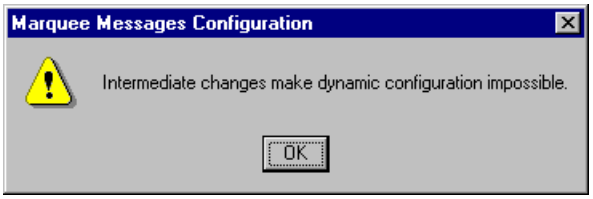
Click **OK** to display the filtered list you selected and close the dialog box.

8. Configure Marquee Messages Dynamically

When using the dynamic configuration option, note that only the following items may be modified dynamically:

- Message Header
- Message Footer
- Message
- Disabled / Enabled state of message
- Devices and device groups to which the message is routed
- Message attributes that are selected

If you make a change to a message while you are not in dynamic mode, and then attempt to change the message in dynamic mode, you will see the following dialog box:



You will need to shutdown the CIMPLICITY project, perform a configuration update, and restart the project in order to dynamically configure this message again.

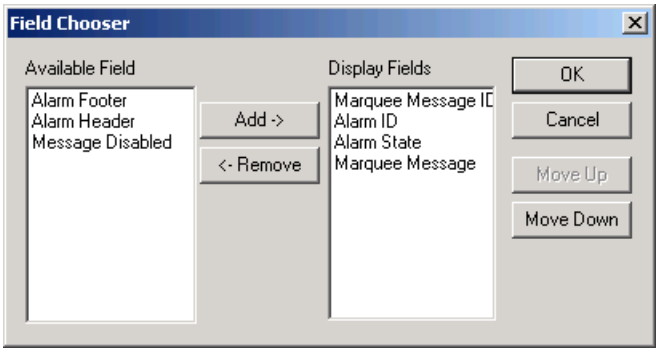
Dynamic configuration of Marquee Types, Marquee Ports, and Marquee Devices is not supported.

9. Change Marquee Message Configuration Display Fields

Do one of the following.

- Select Field Chooser on the View menu.
- Click the **Field Chooser** button on the Toolbar.

Result: The Field Chooser dialog box opens.



There are two list boxes on this dialog box:

- The Available Field box lists the marquee message fields that are not currently being displayed.
- The Display Fields box lists the marquee message fields that are currently being displayed and the order in which they are displayed.

You can:

- **Add** fields from the Available Field list to the Display Fields list.
- **Remove** fields from the Display Fields list to the Available Field list.
- Reorder the Display Fields list.

When you are through modifying the display attributes, click **OK** to close the dialog box and update the display on the Marquee Messages grid, or click **Cancel** to close the dialog box without implementing any changes.

Chapter 31. Marquee Messages - Import/Export

Import/Export of Marquee Messages

CIMPLICITY software provides you with two utilities for importing and exporting marquee messages and the associated routing information contained in the **MarqMsgs** and **MarqRout** data files. These utilities give you the ability to manipulate this data in a comma-separated file, using a tool such as Microsoft Excel.

1. Export Configuration Data to a Comma Separated File

1. In your project's CIMPLICITY Workbench, select the **Command prompt…** option on the **Tools** menu.
2. When the command prompt window appears, enter the following:

   cd master

   mqexport.exe < filename>

   Where <filename> is the name of the file into which Marquee message data should be saved.

2. Import Configuration Data from a comma Separated File

1. Do one of the following.

- Delete all existing marquee message configuration data, and replace it with the configuration specified in the import file.
- Keep the existing marquee message configuration intact and update or add definitions as needed.

For each message contained in the import file, if the marquee message ID matches an existing marquee message ID, then the definition in the import file supercedes the previous definition. If the marquee message ID does not match an existing marquee message ID, it is added to the system as a new record

2. In your project's CIMPLICITY Workbench, select the Command prompt… option on the Tools menu.

3. When the command prompt window appears, enter the following:

   cd master

   To delete all existing configuration, and replace it with the contents of the import file, enter:

   mqimport.exe <filename>

   To supercede any existing configuration, and append any new configuration to the system, enter:

   mqimport.exe –r < filename>

   Where <filename> is the name of the file from which Marquee message data should be read.

3. Format of the Comma-Separated File

1. The first line must contain only **3** characters.
   - The first character must be a comma (**,**) and is used to indicate the field separator.
   - The second character must be a dash (-) and is used to indicate the continuation character.
   - The third character must be an asterisk (*****) and is used to indicate the comment character.

2. Each additional line in the file must be in the following format:

   Field 1 is the Marquee Message ID.

   Field 2 is the Marquee Alarm ID.

   Field 3 is the Marquee Alarm state for this message, and should be set to one of **GEN**, **RES**, or **CON**.

   Field 4 is the Marquee message.

Field 5 is the Marquee message header.

Field 6 is the Marquee message footer.

Field 7 is the Marquee message attributes, which indicate which attributes are selected for this message. This data is encoded in an integer.

Field 8 is the Marquee message disabled state. A value of **0** indicates that the message is enabled, while a value of **1** indicates that the message is disabled.

Field 9 contains Additional Point 1.

Field 10 contains Additional Point 2.

Field 11 contains Additional Point 3.

Field 12 contains Additional Point 4.

Field 13 contains Additional Point 5.

Field 14 contains Additional Point 6.

Field 15 contains Additional Point 7.

Field 16 contains Additional Point 8.

You may specify up to 180 optional additional fields, where each field is the name of a valid Marquee Device or Device Group to which this message should be routed.

An example of a comma-separated file containing Marquee message data is displayed below:

```
,-*
ALARM_PT,ALARM_PT,GEN,%AM,,,0,0,,,,,,,,,GROUP1
MESSAGE1,,CON,This is a continuous
 message.,,,0,0,POINT1,POINT2,,,,,,,GROUP_A
```

Chapter 32. Marquee Messages Routed between CIMPLICITY Projects

Route Marquee Messages between CIMPLICITY Projects

You can use either of the following methods to route marquee messages between CIMPLICITY projects:

- Use extensions to the Basic Control Engine to support generating and clearing marquee messages from any CIMPLICITY project to any other CIMPLICITY project controlling a marquee
- Configure a remote marquee port, where marquee messages in the local CIMPLICITY project are routed to a remote CIMPLICITY project controlling a marquee

Before using either of these features, you need to go to the CIMPLICITY project that will be sending the marquee messages and configure access to the project that will be receiving the marquee messages. You do this by configuring a remote project.

Extensions to the Basic Control Engine

Extensions to the Basic Control Engine

Two extensions are available in the Basic Control Engine to let you generate and clear marquee messages from scripts in your CIMPLICITY project. Your CIMPLICITY project must be running in order to used these functions:

- MarqueeMessageGenerate (Function)
- MarqueeMessageClear (Function)

MarqueeMessageGenerate (Function)

Syntax

`MarqueeMessageGenerate` (Project, Process, Device, Message [, RefID])

Description

Generates a marquee messages from your CIMPLICITY project and sends it to the remote project.

Returns an Integer that represents the status of the function.

Statuses

Status	Related Integer	Message
MARQ_SUCCESS	0	The message was successfully generated.
MARQ_BAD_DEVICE	20	You have specified an invalid device.
MARQ_EMPTY_PROJECT_ID	50	You specified an empty project ID.
MARQ_EMPTY_SERVICE_ID	60	You specified an empty service ID.
MARQ_EMPTY_DEVICE_ID	70	You specified an empty device ID.
MARQ_EMPTY_REF_ID	80	You specified an empty reference ID.
MARQ_NOCOMM_PROJECT	130	A problem was experienced communicating with the remote project.
MARQ_INVALID_SERVICE_ID	150	You specified an invalid service ID.

Comments

The MarqueeMessageGenerate function requires the following parameters:

Parameter Description

Project	The name of the remote CIMPLICITY project to which the marquee message should be sent.
Process	The Service ID of the marquee resident process on the remote CIMPLICITY project to which the marquee messages should be sent.
Device	The name of the configured marquee device controlled by the marquee resident process on the remote CIMPLICITY project to which the marquee message should be sent.
Message	The text to be displayed.
RefID	An optional Reference ID to be associated with this text. If you specify a RefID, then the message you send is appended to the end of the marquee message queue of the remote process, and you can use the `MarqueeMessageClear` function to delete the message. If you do not specify a RefID, then the message you send is displayed immediately by the remote process, it is not added to its queue, and it is never displayed again. This is useful if you need to display a marquee message only once and immediately.

MarqueeMessageClear (Function)

Syntax

`MarqueeMessageClear` (Project, Process, Device, RefID)

Description

Clears a marquee message from the remote project.

This function returns an Integer that represents the status of the function.

Statuses

Status	Related Integer	Message
MARQ_SUCCESS	0	The message was successfully generated.
MARQ_BAD_DEVICE	20	You have specified an invalid device.
MARQ_CANT_DELETE	40	The remote marquee process was unable to delete the message you specified from its queue.
MARQ_EMPTY_PROJECT_ID	50	You specified an empty project ID.
MARQ_EMPTY_SERVICE_ID	60	You specified an empty service ID.
MARQ_EMPTY_DEVICE_ID	70	You specified an empty device ID.
MARQ_EMPTY_REF_ID	80	You specified an empty reference ID.
MARQ_NOCOMM_PROJECT	130	A problem was experienced communicating with the remote project.
MARQ_INVALID_SERVICE_ID	150	You specified an invalid service ID.

Comments

The MarqueeMessageClear function requires the following parameters.

Parameter Description

Project	The name of the remote CIMPLICITY project where the marquee message should be cleared.
Process	The Service ID of the marquee resident process on the remote CIMPLICITY project where the marquee messages should be cleared.
Device	The name of the configured marquee device controlled by the marquee resident process on the remote CIMPLICITY project where the marquee message should be cleared.
RefID	The Reference ID to be associated with this text. This must be the same Reference ID used by the MarqueeMessageGenerate function.

Marquee Remote Port Sample Program

A sample basic control engine script illustrating these two functions and testing for their return values is provided below. In this example, a message is generated and cleared to a project called **PRO J**, with a marquee resident process called MQRP1 controlling a marquee device called DE**V 1**.

```
Sub Main()
’
’ Sample program which demonstrates Marquee extensions to BCE.
’
Dim i As Integer
i = MarqueeMessageGenerate ("PROJ", "MQRP1", "DEV1", "MSG from BCE",
"MSG1")
CheckStatus i
i = MarqueeMessageClear ("PROJ", "MQRP1", "DEV1", "MSG1")
CheckStatus i
End Sub
Sub CheckStatus (i As Integer)
Select Case i
Case MARQ_SUCCESS
msgbox "Message was successfully generated/cleared."
Case MARQ_NOCOMM_PROJECT
msgbox "Problem communicating with the project you specified."
Case MARQ_BAD_DEVICE
msgbox "You have specified an invalid device."
Case MARQ_CANT_DELETE
msgbox "Cannot delete the message you specified."
Case MARQ_EMPTY_PROJECT_ID
msgbox "You have specified an empty Project ID."
Case MARQ_EMPTY_SERVICE_ID
msgbox "You have specified an empty Service ID."
Case MARQ_EMPTY_DEVICE_ID
msgbox "You have specified an empty Device ID."
Case MARQ_EMPTY_REF_ID
msgbox "You have specified an empty Reference ID."
Case MARQ_INVALID_SERVICE_ID
msgbox "The MQRP Service ID you specified was invalid."
Case Else
msgbox "Unexpected error:" & i
End Select
End Sub
```

Use Remote Marquee Ports

Use Remote Marquee Ports

You may configure a local marquee port to send messages to a marquee port on a remote CIMPLICITY project. This has the following advantages:

- You only need to configure a marquee message once. You do not need to have duplicate configuration in every CIMPLICITY project.
- You can send messages from your local CIMPLICITY project to any remote CIMPLICITY project that is controlling a marquee.

Configure a Local Marquee Port to Send Remote Messages

A local marquee port can send messages to a marquee port on a remote project:

1. Activate the Marquee Ports icon from your local project's Configuration Cabinet.
2. Add a new marquee port. In the New Marquee Port dialog box, specify the following information:

Port	Enter the port name for this remote port. You can enter a logical name such as `REMPOR T`, since this will not be an actual physical port on your system.
Service ID	Enter a unique local service ID that will be used to control this port.
Remote Port	Select this checkbox to indicate that this local port will be sending messages to a remote port.
Remote Project	Enter the name of the remote CIMPLICITY project to which messages will be sent.
Remote Service ID	Enter the Service ID of the marquee resident process on the remote CIMPLICITY project to which messages will be sent.
Remote Device ID	Enter the configured device ID controlled by the Service ID on the remote CIMPLICITY project to which messages will be sent.

3. Click OK to close this dialog box and save your changes. The Port Properties dialog box opens.
4. Click OK on the Port Properties dialog box. Configuration of the remote port is now complete.
5. Configure a marquee device to be associated with this port, and marquee messages to be routed to this device, just as if this port was on your local system.

Important Details when Using Remote Ports

If you are using remote ports, you need to be aware of the following:

Message Formatting

Before sending the message to the remote project, the local marquee resident process formats the message as if it were going to display it on a marquee controlled by the local project. When the message arrives at the remote project, it will be formatted subject to the settings of the remote device.

Therefore, it is recommended that you do not define any Headers, Footers, or Message Attributes for the local Marquee Type and Device associated with this port.

The remote device will take care of formatting the message to be displayed.

Determining Duration Time

The remote device determines how quickly the message you send is displayed.

For example, the duration time of the local device is 1 second, and the duration time of the remote device is 5 seconds. The local CIMPLICITY project sends the message to the remote CIMPLICITY project once a second. However, the remote project actually displays the message on the marquee every 5 seconds.

Duplicate Message ID's

The locally configured Marquee Message ID is used as a message key when sending the message to the remote project. If the remote project already has a message configured with an identical Message ID, then the local message will not be displayed on the remote project, and a warning message will be displayed in the Status Log.

Message Length

The actual marquee message that is sent from the local project to the remote project is limited to 80 characters. If the message is longer than 80 characters, the message will be truncated and a warning message will be displayed in the Status Log.

Clearing Messages on Project Shutdown

If you shut down your local CIMPLICITY project before clearing the marquee messages that have been sent to the remote CIMPLICITY project, the messages remain in the queue of the remote CIMPLICITY project.

Network Traffic

Routing messages between projects increases network traffic between the projects.

Chapter 33. Marquees Network Configuration

Network Marquees Configuration

You can configure marquee messages to be displayed on marquees that are connected to terminal servers. The Xyplex MAXserver 1620 terminal server has been qualified for this use. Your terminal server must be running version 6.0.2S7 or higher firmware.

1 *(page 550)*	Set up the Terminal Server.
2 *(page 551)*	Configure a CIMPLICITY Terminal Server port.
3 *(page 555)*	Verify the port configuration.

1. Set up the Terminal Server

1. Log into your terminal server, and set privileged mode by issuing the following command. Enter the privileged password when prompted (typically "system"):

   > set priv

   Password>

2. Enter the following command so that the permanent database and the operational database will be updated when a **define** command is issued:

   >> set server change enabled

3. For each port to which a marquee is attached, enter the following set of commands where:

   <port> is the actual port number to which the marquee is attached. Note that it is possible to specify a range of port numbers instead of a single port (for example, 1-10 signifies ports 1 through 10).

   <ip_port> is the telnet port number.

<parity> is odd, even, or none.

<stop bits> is the number of stop bits.

<speed> is the baud rate of the marquee.

```
>> define port <port> access remote
>> define port <port> telnet remote <ip_port>
>> define port <port> telnet transmit immediate
>> define port <port> parity <parity>
>> define port <port> character size 8
>> define port <port> stop bits <stop bits>
>> define port <port> autobaud disabled
>> define port <port> speed <speed>
>> define port <port> line editor disabled
>> define port <port> telnet csi escape enabled
>> define port <port> telnet binary session mode passall
>> define port <port> loss notification disabled
>> define port <port> autoprompt disabled
>> define port <port> verification disabled
>> define port <port> outboundsecurity disabled
>> define port <port> broadcast disabled
>> define port <port> default session mode transparent
>> define port <port> internet tcp keepalive 2
```

2. Configure a CIMPLICITY Terminal Server Port

2. Configure a CIMPLICITY Terminal Server Port

1. Select the Project Parameters section in the Workbench.
2. Enter the value for the appropriate project parameter.
   - TSERV_<PORT>
   - TSERV_<COM>

Note: CIMPLICITY will update the glb_parms.idt file with your entries.

At startup, the Marquee resident process will check the glb_parms.idt file for each Marquee port that is configured. If an entry exists, it will use the corresponding Terminal Server port for its communication.

You make an entry for each port to which a marquee is attached.

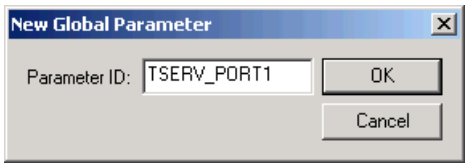
Important: When configuring ports in the Marquee Ports configuration window, a given marquee resident process can control at most one port on a terminal server.

In addition, a given port on a terminal server can be controlled by at most one Marquee process. Thus, when configuring ports in CIMPLICITY:

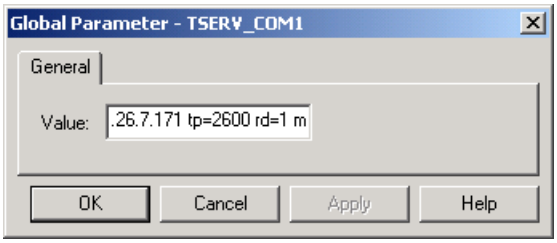
- Be sure to specify a unique Service ID for each port to be controlled on the terminal server.
- Make sure that two CIMPLICITY projects do not attempt to control the same terminal server port.

TSERV_<PORT>

TSERV_<PORT> Configuration

For	Marquee network Project		
Purpose	To add a Marquee network port with specifications.		
Parameter ID	TSERV_<PORT> Where **<**PORT**>** is replaced by the Port ID that you specify in the Marquee Ports configuration window in CIMPLICITY. 		
Value	Format	`2 ipa=<ip_address> tp=<ip_port> rd=1 mt=5`	
	Where		
	<PORT>	Is replaced by the Port ID that you specify in the Marquee Ports configuration window in CIMPLICITY.	
	<ip_address**>**	Is replaced by the IP address of your terminal server.	
	<ip_port**>**	Is replaced by the telnet port number.	
	`rd`	(Reconnect delay) defines the number of seconds to wait before attempting a reconnection on this communication port. A reconnection delay value of 0 will treat a reconnection failure as a fatal error condition, and the Marquee resident process will terminate. Thus, it is recommended that you use an **rd** of 1 or 2.	
	`mt`	(Minimum time) defines the minimum number of seconds to wait after opening a connection and before attempting a write on this communication port. Where	
		0	Means no delay.
		5 Seconds	Will incur a delay of 5 seconds to allow the connection to be fully established.
Default Value	NA		

TSERV_<PORT> Example



Note: This entry indicates that the marquee resident process should communicate with the port on the terminal server:

IP address	3.26.7.171
Telnet port number	2600
Reconnect delay	1 second
Minimum time	5 seconds

You would specify **PORT1** as the Port ID in the Marquee Ports configuration window.

Note: If you open the glb_parms.idt file you will see that the global parameter is listed in the following format.

TSERV_<PORT>|1|2 ipa=<ip_address> tp=<ip_port> rd=1 mt=5

Where

|1| defines the parameter type as a string.

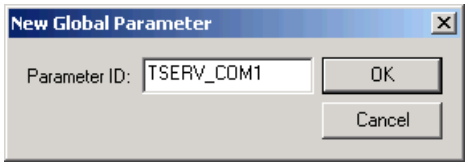
Other segments are defined above .

TSERV_<COM>

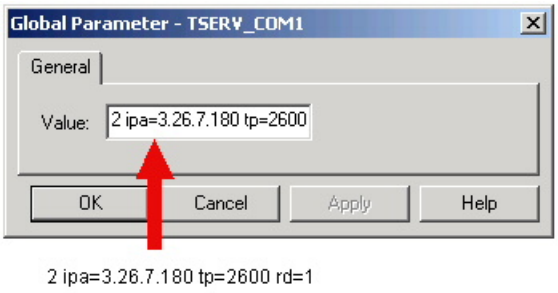
TSERV_<COM> configuration
TSERV_<COM> example

TSERV_<COM> Configuration

For	Marquee network Project
Purpose	To add a Marquee network COM with specifications.

Parameter ID	TSERV_<COM> Where **<COM>** is replaced by the COM ID that you specify in the Marquee Ports configuration window in CIMPLICITY. 	
Value	Format	`2 ipa=<ip_address> tp=<ip_port> rd=1`
	Where	
	<PORT>	Is replaced by the Port ID that you specify in the Marquee Ports configuration window in CIMPLICITY.
	<ip_address**>**	Is replaced by the IP address of your terminal server.
	<ip_port**>**	Is replaced by the telnet port number.
	`rd`	(Reconnect delay) defines the number of seconds to wait before attempting a reconnection on this communication port. A reconnection delay value of 0 will treat a reconnection failure as a fatal error condition, and the Marquee resident process will terminate. Thus, it is recommended that you use an **rd** of 1 or 2.
Default	NA	

TSERV_<COM> Example



Note: This entry indicates that the marquee resident process should communicate with the port on the terminal server:

IP address	3.26.7.180
Telnet port number	2600
Reconnect delay	1 second

You would specify **COM1** as the Port ID in the Marquee Ports configuration window.

Note: If you open the glb_parms.idt file you will see that the global parameter is listed in the following format.

TSERV_<COM>|1|2 ipa=<ip_address> tp=<ip_port> rd=1 mt=5

Where

|1| defines the parameter type as a string.

3. Verify the Port Configuration

After you have configured your ports and started your CIMPLICITY project, you can check the Status Log to verify that each marquee resident process was able to connect to its configured port on the terminal server.

- Successful connections.
- Write errors.
- Communication failures.

Successful Connections

For each port, an entry will appear in the Status Log similar to the following:

```
8/14/97 6:00:00PM Success PORT1 ReaderThread COR_DCRP_ERROR 20016 0
 connection established: 1600@3.26.7.171, port=PORT1
```

This entry indicates that for process **PORT1**, a successful connection was made to telnet port **1600** on the terminal server whose IP address is **3.26.7.171**.

Write Errors

Occasionally, you may see the following warning message in the Status Log:

```
8/14/97 6:00:00PM Failure PORT1 dc_telnet_write_windows_tty COR_DCRP_ERR
 20016 0 write failure,port=PORT1
```

This message indicates that a write error occurred. The Marquee Resident process will automatically attempt to resend the message to the marquee.

Communication Failures

You may see the following failure message in the Status Log:

```
8/14/97 6:00:00PM Failure PORT1 dc_telnet_write_windows_tty COR_DCRP_ERR
 20016 0 connection reset, 1600 @ 3.26.7.171, port=PORT1
```

This Status Log entry indicates that the Marquee Resident process is unable to communicate with the port on the terminal server. It resets the IP connection and attempts to re-establish communications

with the port. Check your network state, network connections and your terminal server configuration if you receive this message.

Chapter 34. Marquee Ports

Marquee Port Configuration

Marquee ports are used to identify which communication port(s) will be used for marquee devices. Some of the things you can control about a port are:

- The process that controls it
- The page width for all devices on that port
- The amount of time after which pending messages will be re-sent to the device.
- Alarm class and event filtering.

Review how to:

1 *(page 557)*	Open a Marquee Port dialog box.
2 *(page 560)*	Enter Marquee port details.
3 *(page 562)*	Delete a Marquee port.
4 *(page 562)*	Copy a Marquee port.
5 *(page 563)*	Filter the Marquee port list.

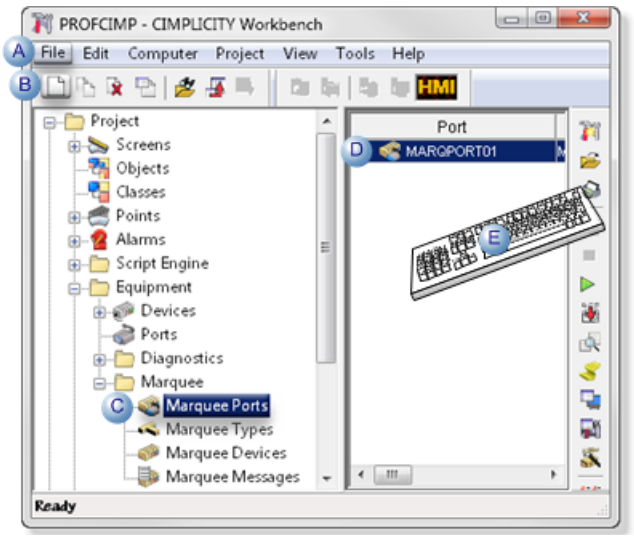
1. Open a Marquee Port Dialog Box

1. Open a Marquee Port Dialog Box

1.1 *(page 558)*	Create a new Marquee port.
1.2 *(page 559)*	Open an existing Marquee Port dialog box.

1.1. Create a new Marquee Port

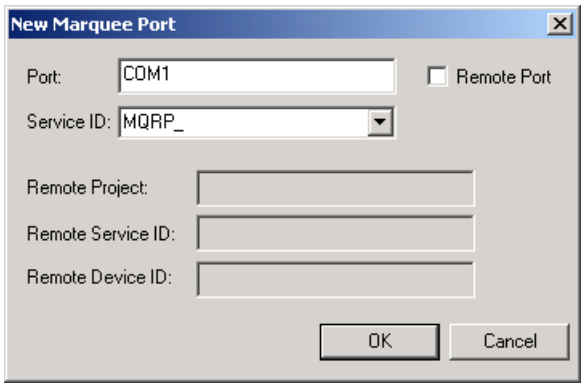
1. Select **Project>Equipment>Marquee>Marquee Ports** in the Workbench left pane.
2. Do one of the following.



A	Click File>New>Object on the Workbench menu bar.	
B	Click the New Object button on the Workbench toolbar.	
C	In the Workbench left pane:	
	Either	Or
	Double click **Marquee Ports**.	a. Right-click **Marquee Ports**. b. Select New on the Popup menu.
D	a. In the Workbench right pane. a. Right-click any marquee port. b. Select New on the Popup menu.	
E	Press Ctrl+N on the keyboard.	

A New Marquee Port dialog box opens when you use any method.

3. Right-click **Marquee Ports**.
4. Select New on the Popup menu.
5. Right-click any marquee port.
6. Select New on the Popup menu.
7. Fill in the fields as follows.

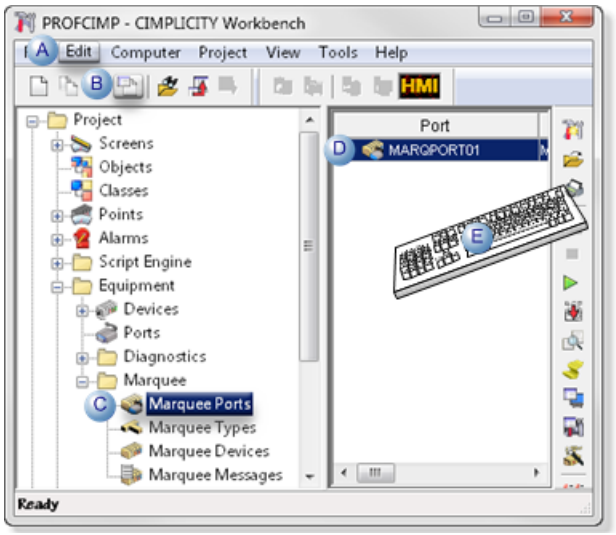


Field	Description
Port	Port to be used by the marquee driver. The port must be an existing communications port on your computer (for example, COM1 or COM2), or it can be a logical name to be associated with a terminal server port (for example, PORT1 or PORT2). **Note:** See Configuring Network Marquees for detailed information on how to do this. You cannot configure two marquee ports with the same port ID. If you attempt to do this, you will receive an error message in a popup window.
Service ID	Service ID for the marquee driver process. For physical ports on your computer, one process may control more than one port, but you should usually configure one process per port. For terminal server ports, one process may only control one port. It is suggested that you maintain the default prefix of **MQRP**_ for the service ID of all marquee driver ports.
Remote Port	Check if this port is to be associated with a marquee port in another CIMPLICITY project, Result: The **Remote Project**, **Remote Service ID**, and **Remote Device ID** fields become active.
Remote Project	Name of the CIMPLICITY project to which messages associated with this local port should be sent.
Remote Service ID	Name of the CIMPLICITY Marquee process to which messages associated with this local port should be sent.
Remote Device ID	Name of the marquee device to which messages associated with this local port should be sent.

8. Click **OK**.

1.2. Open an existing Marquee Port Dialog Box

1. Select Project>Equipment>Marquee>Marquee Ports in the Workbench left pane.
2. Select a Marquee port in the Workbench right pane.
3. Do one of the following.

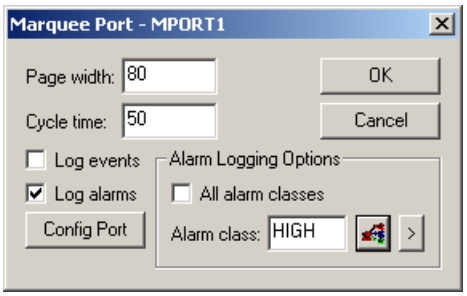


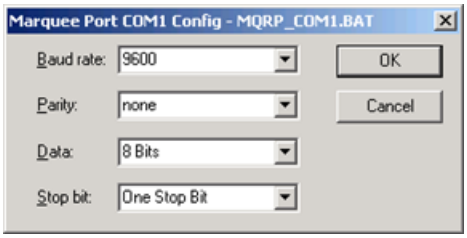
A	Click Edit>Properties on the Workbench menu bar.	
B	Click the Properties button on the Workbench toolbar.	
C	In the Workbench left pane: a. Right-click **Marquee Ports**. b. Select Properties on the Popup menu.	
D	In the Workbench right pane:	
	Either	Or
	Double-click a Marquee port.	a. Right-click a Marquee port. b. Select Properties on the Popup menu.
E	Press Alt+Enter on the keyboard.	

4. Right-click **Marquee Ports**.
5. Select Properties on the Popup menu.
6. Right-click a Marquee port.
7. Select Properties on the Popup menu.

2. Enter Marquee Port Details

Use the Marquee Port details dialog box to set the page with and cycle time for messages. You can also select the types of messages to display.



Option	Description	
Page width	Width to be used when calculating word wrapping for the devices on the port..	
Cycle time	Maximum number of seconds to be spent displaying messages from the queue before the displaying of messages will re-start from the beginning of the queue	
Log events	Checked	Log events to be processed.
Log alarms	Checked	Log alarms to be processed.
Config Port	Opens the Marquee Port <Port> Config dialog box, which provides options to configure the Marquee port. Options are:  	
	Baud rate	Selections are: 75 110 150 300 600 1200 2400 4800 9600 19200
	Parity	Selections are: none odd even mark space
	Data	Selections are: 4 Bits 8 Bits
	Stop bit	Selections are: One & Half Stop Bits One Stop Bit Two Stop Bits
All alarm classes	Checked	Display all alarm classes. **Important:** Create an alarm class for each Marquee port and use a different alarm class to filter alarm messages to each Marquee process. Serious performance problems could result from sending all alarms to all Marquee processes if it is a normal condition for there to be a large number of outstanding alarms in your CIMPLICITY project.
	Cleared	Display the alarms for a single alarm class.
Alarm class	Display the alarms for the selected alarm class.	
		Opens the Select an Alarm Class browser
		Opens a Popup menu to create or select an alarm class.

3. Delete a Marquee Port

1. Select the port in the Marquee Ports window.

2. Do one of the following:
    - Select Delete on the Edit menu.
    - Click the **Delete** button on the toolbar.
    - Press the **Delete** key.

The Delete dialog box opens.

You will be asked to verify that you want to delete the port.

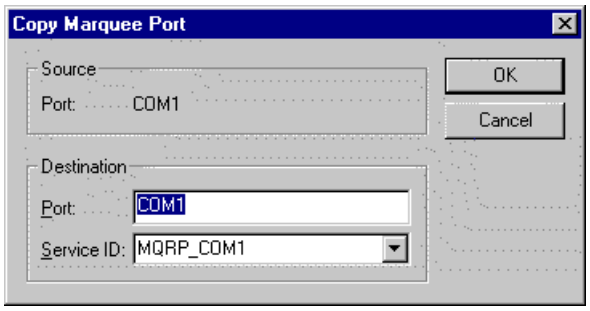
Note: If you try to delete a port that is currently in use by a Marquee Device, an error dialog box displays.

4. Copy a Marquee Port

1. Select the port you want to copy from.

2. Do one of the following.
    - Select Duplicate on the Edit menu
    - Click the **Duplicate** button on the toolbar.
    - Press **Ctrl+D**.

    The Copy Marquee Port dialog box opens.



3. Enter the name of the new Marquee Port in the **Port** field

   Remember that you cannot configure two marquee ports with the same port ID. If you attempt to do this, you will receive an error message in a popup window.

4. Enter the name of the new Service ID in the **Service ID** field.

   One process may control more than one port, but you should usually configure one process per port. It is also suggested that you maintain the default prefix of **MQRP_** for the service ID of all marquee driver ports.

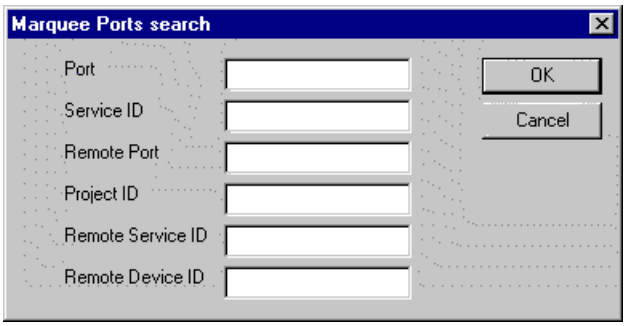
5. Filter the Marquee Port List

The Marquee Port list can contain many types. You can use the **Search** function to quickly find a specific Marquee Port or subset of Marquee Ports.

To initiate a search, do one of the following

- Select Search on the View menu.
- Click the **Search** button on the toolbar.

The Marquee Ports search dialog box opens.



You can filter the list by any combination of the following:

- Port
- Service ID
- Remote Port
- Project ID
- Remote Service ID
- Remote Device ID

You can use the following wildcards in your filter:

*	Use this wildcard to search for any number of characters at this point in the string. For example, if you want to display a list of Marquee Types that start with "M" and end with "X", enter M*X in the Marquee Type field
?	Use this wildcard to search for any character in this place in the string. For example, if you want to display the list of Marquee Types whose names are three characters long, and whose first character is "M" and third character is "X", enter **M?X** in the **Marquee Type** field.

Note: There are no implied wildcards. If you do not include or terminate your search string with an asterisk, only those items that match your request exactly will be returned.

Click **Cancel** to close the dialog box and do no further filtering.

Click **OK** to display the filtered list you selected and close the dialog box.

Chapter 35. Marquee Global Parameters and Logs

Marquee Driver Global Parameters

The following parameters in the global parameters file are used by the Marquee driver:

	MARQ_POINT_LIMIT_LEN	Defines the display limit for text points.
	MARQ_PROC_NEW_ALARMS	Determines how a new alarm will be displayed.
	MARQ_RESERVED_NULL_CHAR	Defines the ASCII character that represents the NULL character.
	MARQ_VARIABLE_NULL_CHAR	Enables the use of a variable or changing NULL termination character on a message by message basis.
	MARQ_WORD_WRAP_ON	Controls the word wrap feature
	MARQ_WRAP_HF	Controls the word wrap feature for headers and footers.

Modify log_names.cfg

The Marquee Driver uses an IPC datagram queue to receive alarm messages. The default size for this queue is 10. This default is acceptable for bursts of 50 alarms during testing, however system loading will change performance.

If you find an IPC datagram queue overflow message in the Status Log for a Marquee Driver process, you can increase the size of the IPC queue.

todo: To increase the size of the queue:

1. In your project's CIMPLICITY Workbench, select Command prompt on the Tools menu.
2. In the Command Prompt window, enter the following commands:

   ```
   cd data
   ```

   ```
   notepad log_names.cfg
   ```

3. Type the following line at the end of the log_names.cfg file in Notepad:

   MARQ_IPCQ_SIZ`|P|default|5|`<number>

   where

<number> is the new queue size.

4. Exit Notepad and save the file.
5. Exit the Command Prompt window and return to the CIMPLICITY Workbench.
6. Before you restart your project, select Configuration Update from the Project menu or click the Configuration Update button on the toolbar to update your run-time configuration.

Chapter 36. Marquee Types

Marquee Type Configuration

Use Marquee Types to identify types of devices that use the same specific sequence of characters to indicate that a message on a given port is for a device of that type. The header and footer indicate the start/stop sequences that the device might need or that you would like to use to control the display of messages.

The message wrap, preceded by the header and followed by the footer, is sent to the device with every alarm message.

1 *(page 567)*	Open a Marquee Type dialog box.
2 *(page 570)*	Enter Marquee Type details.
3 *(page 573)*	Delete a Marquee Type.
4 *(page 574)*	Copy a Marquee Type.
5 *(page 574)*	Filter the Marquee Type list.

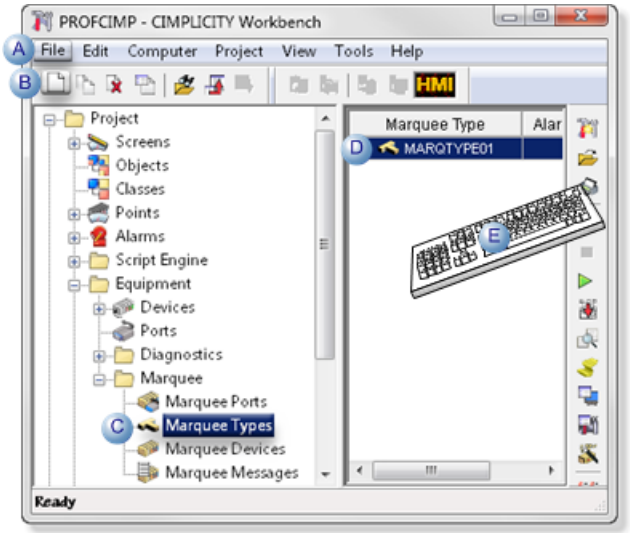
1. Open a Marquee Type Information Dialog Box

1. Open a Marquee Type Information Dialog Box

1.1 *(page 568)*	Create a new Marquee type.
1.2 *(page 568)*	Open an existing Marquee Type Information dialog box.

1.1. Create a new Marquee Type

1. Select **Project>Equipment>Marquee>Marquee Types** in the Workbench left pane.
2. Do one of the following.

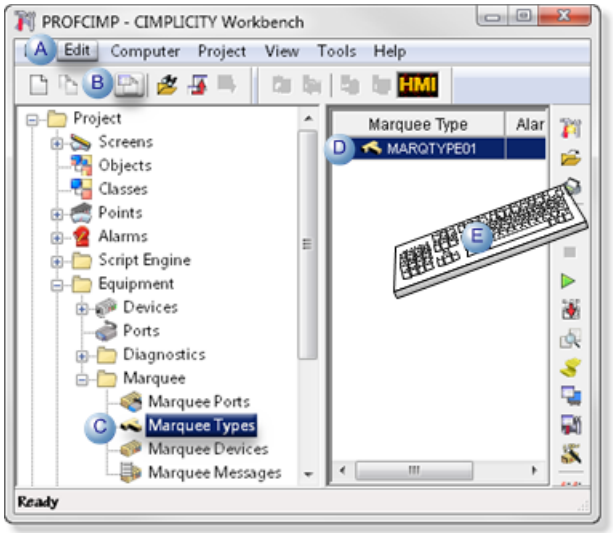


A	Click File>New>Object on the Workbench menu bar.	
B	Click the New Object button on the Workbench toolbar.	
C	In the Workbench left pane:	
	Either	Or
	Double click **Marquee Types**.	a. Right-click **Marquee Types**. b. Select New on the Popup menu.
D	a. In the Workbench right pane. a. Right-click any Marquee type. b. Select New on the Popup menu.	
E	Press Ctrl+N on the keyboard.	

3. Right-click **Marquee Types**.
4. Select New on the Popup menu.
5. Right-click any Marquee type.
6. Select New on the Popup menu.

1.2. Open an existing Marquee Type Information Dialog Box

1. Select **Project>Equipment>Marquee>Marquee Types** in the Workbench left pane.
2. Select a Marquee type in the Workbench right pane.
3. Do one of the following.



A	Click Edit>Properties on the Workbench menu bar.	
B	Click the Properties button on the Workbench toolbar.	
C	In the Workbench left pane: a. Right-click **Marquee Types**. b. Select Properties on the Popup menu.	
D	In the Workbench right pane:	
	Either	Or
	Double-click a Marquee type.	a. Right-click a Marquee type. b. Select Properties on the Popup menu.
E	Press Alt+Enter on the keyboard.	

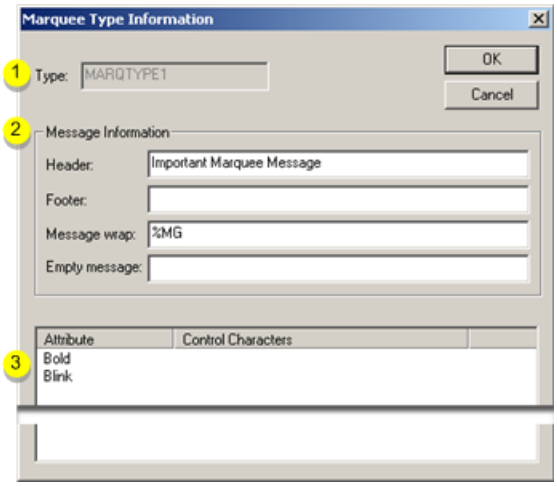
4. Right-click **Marquee Types**.
5. Select Properties on the Popup menu.
6. Right-click a Marquee type.
7. Select Properties on the Popup menu.

2. Enter Marquee Type Details

2. Enter Marquee Type Details

For every message sent to a Marquee device as a result of an alarm, the header, footer and message wrap strings that you define in the Marquee Type Information dialog box will be sent to devices of the given type.

Enter details in the Marquee Type Information dialog box as follows.



rect 4, 199, 339, 287 *(page 572)*
rect -1, 23, 186, 65 *(page 570)*
rect 0, 63, 299, 185 *(page 571)*

1 *(page 570)*	Type
2 *(page 571)*	Message Information
3 *(page 572)*	Message Attributes box

1	Type

Unique identifier for each marquee type.

The identifier can contain up to 8 characters.

2	Message Information

Fields in the Message Information box define the optional header, footer, message wrap and empty message for the selected Marquee type.

- Header
- Footer
- Message wrap
- Empty message

Header

A header sent to the Marquees of this type.

The string can contain up to 70 characters.

You may use marquee control characters *(page 573)* in the string.

Footer

A footer sent to the Marquees of this type.

The string can contain up to 70 characters.

You may use marquee control characters *(page 573)* in the string.

Message wrap

The message format used by Marquees of this type, including:

- How a message will appear on the marquee
- The number of lines in the message and the contents of each line.

This string can contain up to **70** characters.

You may use marquee control characters *(page 573)* in the string. By default, this field will be set to **%MG** .

The **Message Wrap** field may also contain the following special control characters:

%PT	Point ID (if the alarm is for a point in alarm)
%PV	Point value (if the alarm is for a point in alarm)
%FR	Resource ID
%TM	Time
%ST	Alarm State

%MG	Marquee message configured for the device and alarm.

Note: If Message field is blank:

When the alarm is in the state you select, the Message field for the Marquee message (or alarm message) displays only if the **Message wrap** field for the Marquee Type contains the `%MG` control character.

The information in the **Header** and **Footer** fields for the Marquee Type is always displayed.

Example

The following is entered in this field.

```
%TM %ST %FR%\n%PT=%PV%\n%MG%\n
```

As a result, a three line message displays for each alarm on the Marquee.

Line	Contains the:
1	time, Alarm State and Resource ID
2	Point ID and Point value
3	Alarm message.

Empty Message

Sent to Marquees of this type when the Marquee message queue is empty, i.e. when there are no current alarms to be displayed on the device.

The string can contain up to 70 characters.

You may use marquee control characters *(page 573)* in the string.

If you enter a string that contains `%TM` in the **Empty message** field, then the time of day is repeatedly sent to the Marquee device at the configured display rate for the device whenever the marquee message queue is empty.

If you do not enter a string containing **%TM** in the **Empty message** field, then the contents of this field will be sent only once to the device when the marquee message queue is empty.

3	Message Attributes box

(Optional) Configure the corresponding control characters for each attribute listed.

Example

To configure the control characters for the Bold attribute, double-click the word Bold to display the Attribute dialog box.

In the **Control** field, enter the appropriate control characters that will generate bold text on this Marquee type, and then click OK.

Note: Attributes are user-configurable in the MarqAtts.cfg file.

Marquee Control Characters

The **Header**, **Footer**, **Message wrap**, and **Empty message** fields may include the following control characters. Note that not all marquees support all these control characters.

%\n	newline
%\r	carriage return
%\b	backspace
%\v	vertical tab
%\t	horizontal tab
%\f	formfeed
%^[	generates an escape
%^<character>	generates a control character for any alphabetic character
%x<xx>	generates a hex character between 00-FF. <xx> should be the 2 hex characters.
	To generate the null character (00), you must configure the MARQ_RESERVED_NULL_CHAR global parameter.
%<xx>	generates the control characters you specified for message attribute <xx> in the Message Attributes box, where <xx> is a two-digit number between 01 and 25. For example, to generate the control characters you specified for fifth attribute in the list, enter %05.

3. Delete a Marquee Type

1. Select the marquee type you want to delete from the list in the Marquee Types window.

2. Do one of the following.
   - Select Delete on the Edit menu.
   - Click the **Delete** button on the toolbar.
   - Press the **Delete** key.

   The Delete dialog box opens.

3. Click **Yes** to confirm the delete request, or click **No** to close the dialog box without deleting the marquee type

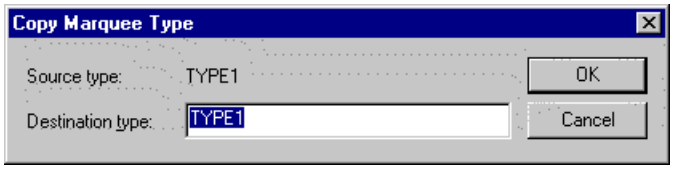
   Note: If you try to delete a marquee type that is currently in use by a marquee device, an error dialog box will open and your request will be canceled.

4. Copy a Marquee Type

1. Select the Marquee Type you want to copy from the list in the Marquee Types grid.

2. Do one of the following.
   - Select Duplicate on the Edit menu
   - Click the **Duplicate** button on the toolbar.
   - Press **Ctrl+D**.

   The Copy Marquee Type dialog box opens.



3. Enter the name of the new Marquee Type.

4. Click **OK** to confirm the copy request, or click **Cancel** to close the dialog box without copying the marquee type.

   Note: he new Marquee Type must be unique. If you enter an existing Marquee Type, you will receive an error message in a popup window.

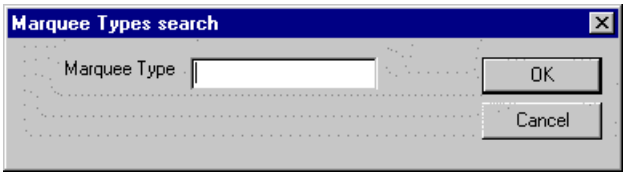
5. Filter the Marquee Type List

The Marquee Type list can contain many types. You can use the **Search** function to quickly find a specific Marquee Type or subset of Marquee Types.

Do one of the following.

- Select Search on the View menu.
- Click the **Search** button on the toolbar.

Result: The Marquee Types search dialog box opens.



You can filter the list by:

Marquee Type

You can use the following wild cards in your filter:

*	Use this wild card to search for any number of characters at this point in the string. For example, if you want to display a list of Marquee Types that start with "M" and end with "X", enter M*X in the Marquee Type field
?	Use this wild card to search for any character in this place in the string. For example, if you want to display the list of Marquee Types whose names are three characters long, and whose first character is "M" and third character is "X", enter M?X in the Marquee Type field.

Note: There are no implied wild cards. If you do not include or terminate your search string with an asterisk, only those items that match your request exactly will be returned.

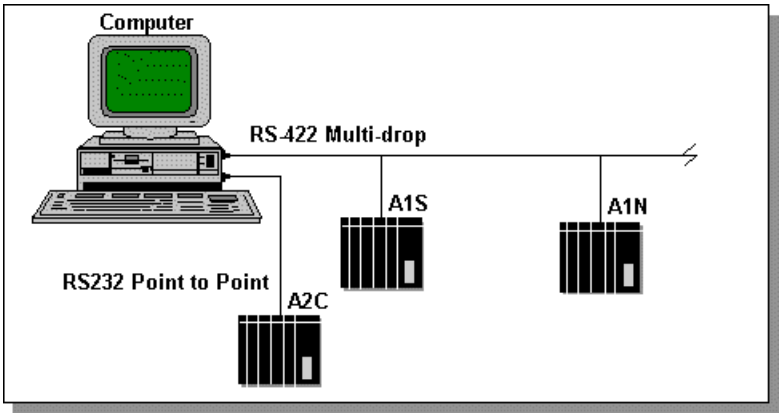
- Click **Cancel** to close the dialog box and do no further filtering.
- Click **OK** to display the filtered list you selected and close the dialog box.

Chapter 37. Mitsubishi A-Series Serial Communications

About Mitsubishi A-Series Serial Communications

The Mitsubishi ® A-Series Serial communications supports Mitsubishi A-Series programmable controllers in point-to-point and multi-drop configurations with one server device where the CIMPLICITY Mitsubishi A-Series Serial Communication enabler acts as the server. The Mitsubishi A-Series Serial Communications provides read and write access to the Mitsubishi programmable controllers via the Mitsubishi dedicated ASCII protocol using format 2 as documented in the Mitsubishi COMPUTER LINK MODULE Users Manual.

The Mitsubishi A-Series programmable controllers support Computer Link Modules designed to allow a computer serial access to data in the programmable controllers (some A-Series CPU models have built in Computer Link port interfaces). Depending on the CPU and/or the Computer Link Module, you can make either RS-232 point to point connections or RS-422 multi-drop connections. You can connect up to 32 Mitsubishi programmable controllers to a host computer via multi-drop RS-422 line. Each programmable controller connected to the host can act as a gateway to MELSECNET or MELSECNET/B and provide access to the programmable controllers connected to this proprietary Mitsubishi network. Up to 64 programmable controllers on a MELSECNET(/B) can be accessed through each gateway.



Supported Mitsubishi A-Series Devices

The Mitsubishi A-Series Serial Communications enabler supports communications to the following programmable controllers:

A0J2H	**A2**	**A2C**	**A3M**
A1	**A2 S1**	**A2N**	**A3N**
A1N	**A2A**	**A2N_S1**	
A1S	**A2A S1**	**A3**	

Supported Mitsubishi A-Series Memory (Device) Areas

The Mitsubishi programmable controllers contain multiple data areas that consist of channels of 16-bit data, discrete bit data, and completion bit data (timers and counters). In Mitsubishi terminology, these areas are referred to as device areas and the points within as devices. Device types include discrete I/O, Relays, Contacts, Coils and Registers. The type of areas and number of devices depends on the Mitsubishi programmable controller model. The amount of data read and returned to CIMPLICITY software is determined by the Point Type configured for the device type you select.

All communication commands are implemented as active/standby communications where the CIMPLICITY Mitsubishi A-Series Serial Communications enabler acts as the active server responsible for requesting data from or writing data to Mitsubishi programmable controller(s). Unsolicited data sent from Mitsubishi programmable controllers is not supported.

The memory areas supported by the CIMPLICITY Mitsubishi A-Series Serial Communications enabler are:

Bit Device Area	Word Device Area
Input X	Timer (present value) T
Output Y	Counter (present value) C
Internal Relay M	Data Register D
Latch Relay L	Link Register W
Step Relay S	File Register R
Link Relay B	Special Register D
Annunciator F	Buffer Memory
Special Relay M	
Timer Contact TS	
Time Coil TC	
Counter Contact CS	
Counter Coil CC	

Mitsubishi A-Series Required

In order to properly configure Mitsubishi programmable controller(s) and their Computer Link Module(s) you must refer to the Operations and System manuals for the appropriate programmable controller. If a Computer Link Module is used, then the User's Manual for that interface is required.

These manuals contain important information on:

- Setting the Computer Link port's communication parameters such as address, communication speed and parity.
- Constructing the proper RS-232 and RS-422 cable(s) for connecting the CIMPLICITY computer to the programmable controllers.
- Interpreting response codes returned from programmable controllers and reported to the Status Log.

Mitsubishi Hardware Configuration Requirements

Mitsubishi Hardware Configuration Requirements

Mitsubishi programmable controllers are connected via RS-232 or RS-422 connections to either the Mitsubishi programmable controller's built in Computer Link interface port or through an optional Computer Link Module. Up to 32 Mitsubishi programmable controllers can be connected to a host computer via RS-422 communications on a local network (additional programmable controllers can be accessed though programmable controllers acting as gateways to MELSECNET or MELSECNET/B).

Supported Computer Link Modules

The following Computer Link modules can be used with the given programmable controller models:

- AnA and AnN Series via AJ71C24-S8 (RS-232 and RS-422), AJ71UC24 or equivalent
- A1S Series via A1SJ71C24-R2 (RS-232) or A1SJ71C24-R4 (RS-422) or equivalent
- A2CCPUC24 and A2CCPUC24-PRF via internal computer link modules supporting both RS-232 and RS-422

The choice of modules should be based on your Mitsubishi programmable controller models and system requirements (multi-drop or single, point to point). Please consult Mitsubishi documentation on the features and uses of each of the listed interface modules.

If the Mitsubishi programmable controllers are to be connected via RS-422 then you may need an RS-232 to RS-422 converter or RS-422 cards for your computer. It is recommended that you contact Mitsubishi to determine the best converter for your configuration.

Default Communication Parameters

The following communication parameters used by the CIMPLICITY Mitsubishi A-Series Serial Communication enabler cannot be changed. The A-Series Serial modules in the PLC must be configured to match these settings:

Parameter	Value	Description
Data Bits	**7**	Number of data bits used in transmissions
Stop Bits	**1**	Number of stop bits used in transmissions

The following communication parameters must match the Port configuration parameters for the CIMPLICITY Mitsubishi A-Series Serial Communication enabler:

- Baud Rate
- Parity

Mitsubishi A-Series PLC Cable Diagrams

Mitsubishi A-Series PLC Cable Diagrams

The following diagrams are intended to provide general guidance for the construction of the cable used to connect your CIMPLICITY computer to the Computer Link port on a Mitsubishi A-Series PLC.

Always refer to the appropriate A-Series programmable controller and/or Computer Link Module User's Manual that came with your system for detailed diagrams for your particular adapter model and network configuration.

- RS-232 interface cables.
- RS-422 interface cables.

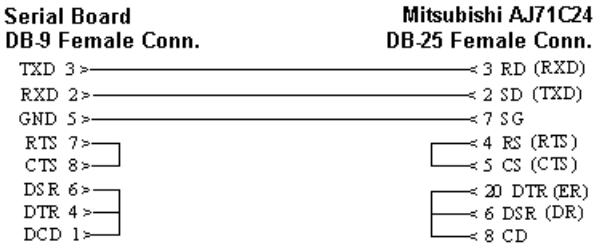
RS-232 Interface Cables

RS-232 Interface Cables

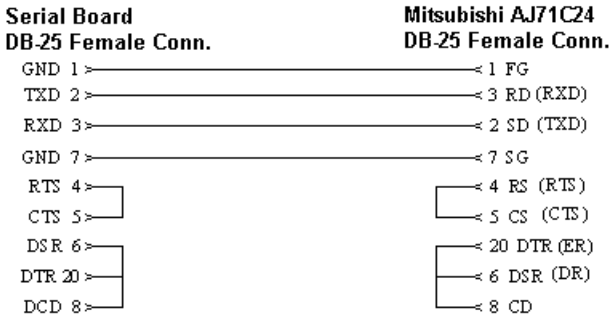
The cabling diagrams below illustrate the interfaces for:

- Serial board DB-9 connector to Mitsubishi AJ17C24 DB-25 connector.
- Serial board DB-25 connector to Mitsubishi AJ17C24 DB-25 connector.
- Serial board DB-9 connector to Mitsubishi AJ17C24 DB-9 connector.
- Serial board DB-25 connector to Mitsubishi AJ17C24 DB-9 connector.

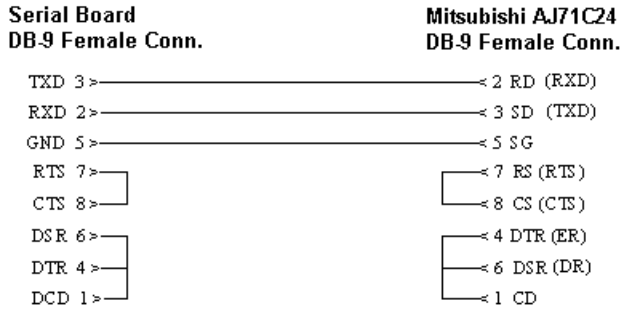
Serial Board DB-9 Connector to DB-25 Connector



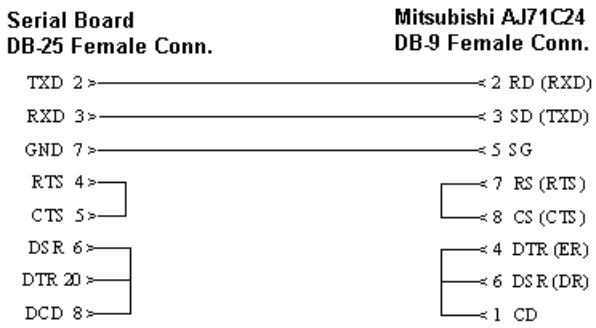
Serial Board DB-25 Connector to DB-25 Connector



Serial Board DB-9 Connector to DB-9 Connector



Serial Board DB-25 Connector to DB-9 Connector



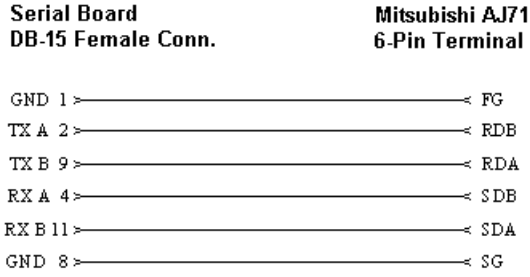
RS-422 Interface Cables

RS-422 Interface Cables

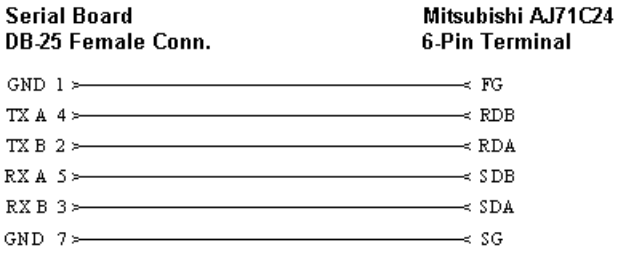
The cabling diagrams below illustrate the interfaces for:

- Serial board DB-15 connector to Mitsubishi AJ17C24 6-pin terminal.
- Serial board DB-25 connector to Mitsubishi AJ17C24 6-pin terminal.

Serial Board DB-15 Connector to 6 Pin Terminal



Serial Board DB-25 Connector to 6 Pin Terminal



CIMPLICITY Configuration for Mitsubishi A-Series Serial

CIMPLICITY Configuration for Mitsubishi A-Series Serial

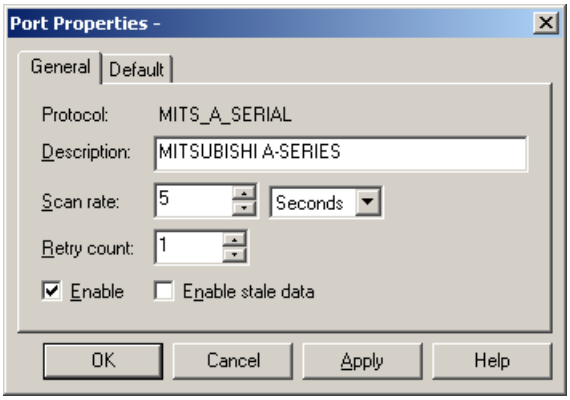
When you configure ports, devices and points that use the Mitsubishi A-Series Serial Communications enabler, some field must contain unique values for the communications to work successfully.

Mitsubishi A-Series Serial Port Configuration

Mitsubishi A-Series Serial Port Configuration

When you configure a port for Mitsubishi A-Series Serial communications, the Port Properties dialog box for the protocol opens. You can set general and default properties for the port.

General Port Properties

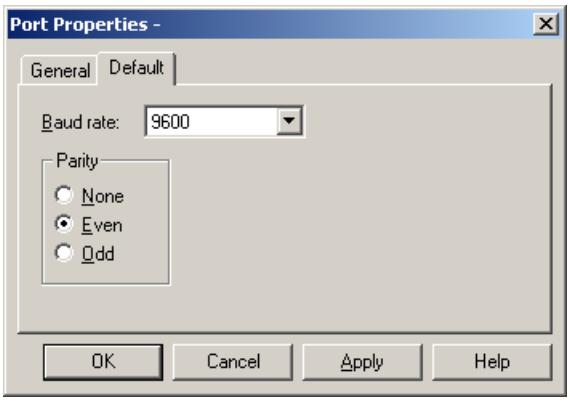


Use the General Properties tab to enter general information for the port. You can define the following:

Description	Enter an optional description to help you identify the port.
Scan Rate	Enter the base scan rate for the port. Point scan rates will be multiples of the base rate. You can specify a scan rate in **Ticks** (100 ticks = 1 second), **Seconds**, or **Hours**.
Retry Count	If communications cannot be established for a device on the port, the device is considered to be down, and a $DEVICE_DOWN alarm is generated. Once a device is down, periodic attempts are made to resume connections to it.
	Enter the number of scans to wait before attempting to reconnect to a device after a communication error has been detected. The recommended **Retry Count** is 3.

Enable	Set this check box if you want the port to be enabled when the project starts. If you clear this check box, the port will not be enabled when the project starts, and points will not be available for devices on the port.

Default Port Properties



Use the Default tab in the Port Properties dialog box to enter communications information for the port. You can define the following:

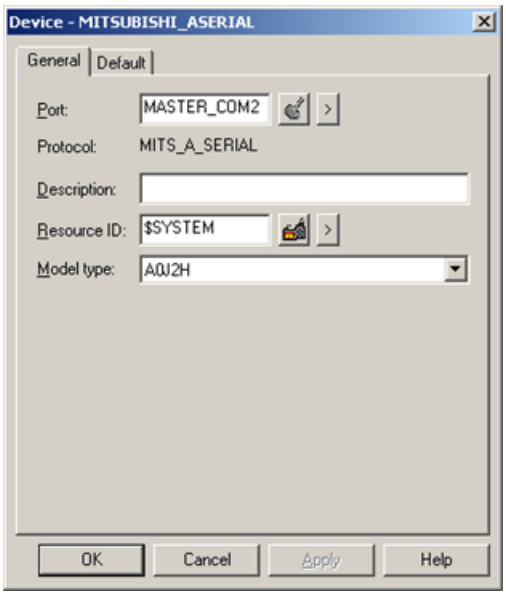
Baud Rate	Enter the baud rate for communications. Click the drop-down list to the right of the input field and select a rate from the list.
Parity	Select the parity to be used for the communications.

Mitsubishi A-Series Serial Device Configuration

Mitsubishi A-Series Serial Device Configuration

When you configure a Mitsubishi programmable controller for Mitsubishi A-Series Serial communications, the Device dialog box for the devices using this protocol opens.

General Device Properties



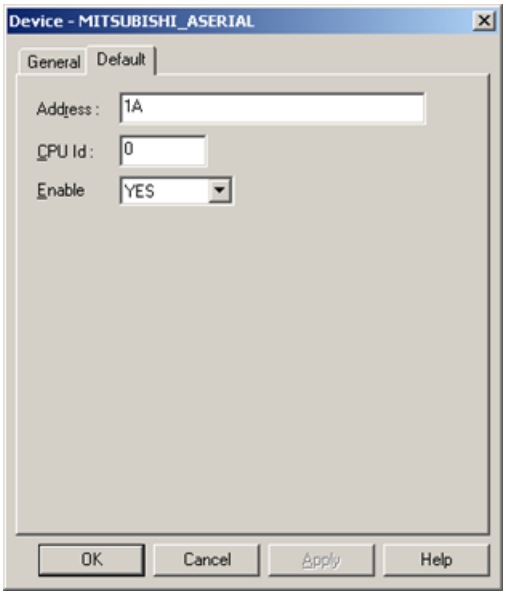
Use the General properties tab to enter general information for the device. You can define the following: Properties.

Port	Select the port for this device. Click buttons to the Port field to select the port as follows.		
	Browser		Display the list of ports and select one.
	Popup Menu		Create a new port, edit the current port, or select a port from the list of ports.
Description	Enter an optional description to help you identify the device.		
Resource	Enter a resource that can be associated with this device for alarm generation. Click buttons to the right of the **Resource** field to select the resource, as follows.		
	Browser		Display the list of resources and select one.
	Popup Menu		Create a new resource, edit the current resource, or select a resource from the list of resources.
Model Type	Select the type of device. Click the drop-down button to the right of the input field to display your choices. For Mitsubishi A-Series Serial Communications, the choices are:		

	A0J2H	**A2**	**A2C**	**A3M**
	A1	**A2 S1**	**A2N**	**A3N**
	A1N	**A2A**	**A2N_S1**	
	A1S	**A2A S1**	**A3**	

When you select the Model Type, you identify acceptable data addresses for each memory (device) type. This information is used to validate data addresses during point configuration.

Default Device Properties



Use the Default Properties tab to enter addressing information about this device.

Address	Enter the controller's address. The address is one of the following: • For target programmable controllers attached directly either point to point via RS-232 or on an RS-422 link, enter the programmable controller's Station Number **<**station number**>** where **<station number>** is in hexadecimal with a valid range of 0 to 1F. Examples of local addresses are: 1A and 2
	• For programmable controllers on a MELSECNET or MELSECNET/B subnetwork to be accessed via a locally connected programmable controller, use the following format: **<** station number**>.<**PC CPU number**>** where **<**station number**>** is the station number of the programmable controller on the local serial network acting as the MELSECNET(/B) gateway and **<**PC CPU number**>** is the address assigned to the data link module attached to MELSECNET(/B).
	Entries are in hexadecimal with a valid range of 0 to 1F for the station number and 0 to 40 for PC CPU number. Examples of MELSECNET addresses are: 3.10, A.1A, and 0.1
CPU ID	Not used.
Enable	Select YES to enable the device when the project starts. Select NO to disable the device. If you select NO, the device will not be enabled and points associated with the device will be unavailable.

Mitsubishi A-Series Serial Point Configuration

Mitsubishi A-Series Serial Point Configuration

Once your devices are configured, you may configure points for them. Through device point configuration, you may configure the following:

- Points that read or set discretes and/or bits in bit type memory (device) areas
- Points to read or set word in word type memory (device) areas
- Points to read Timer/Counter Coils and Contacts
- Points to read or set Timer and Counter Present Values

The fields described below have configuration values that are unique to, or have special meaning for Mitsubishi A-Series Serial communications.

Access	The type of access depends on the point address.
	Enter **READ** for points configured for Inputs.
	Enter **READ or WRITE** for all other points if you wish to be able to set (change) the value.
Update Criteria	The update criteria determines how the data will be requested.
	Enter **On Change** or **On Scan** for points whose values should be polled by the Mitsubishi A-Series Serial communications at regular intervals. Enter **On Demand On Scan** or **On Demand On Change** for points whose value should be polled by the Mitsubishi A-Series Serial communications at regular intervals while users are viewing them.
Address	Enter the point address of the data. Point addresses are device-dependent. See the Address Formats section below for more information

When you configure DIGITAL points from non-discrete memory (such as D, Data Registers), you must also enter data in the following field:

Bit Offset	Enter the bit offset in the 16-bit memory word that defines the point where 0 is the least significant bit and 15 is the most significant bit.
	Warning: Do not enter a data Bit Offset for Point Type BOOL for Bit type devices such as Input (X), Output (Y), Internal Relay (M), Timer Contact, Timer Coil, etc. Leave the field as the default 0.

Address Formats

The following table lists each of the memory (device) areas accessible by the Mitsubishi A-Serial Serial interface.

The Address Entry column gives the maximum device number that a particular device area could support. The actual allowed for the programmable controller being configured may be less and is dependent upon the model.

The Entry Type column specified how you should enter the device's number. Some devices are numbered/referenced by decimal numbers and others by hexadecimal.

The last column specifies how the interface treats the device area, either as bit (discrete) devices or as word devices.

Device Area	Address Entry	Entry Type	Device Type
Input (Read Only)	**X000000 to X0007FF**	Hexadecimal	Bit
Output	**Y000000 to Y0007FF**	Hexadecimal	Bit
Internal Relay M	**M000000 to M008191**	Decimal	Bit
Latch Relay L	**L000000 to L008191**	Decimal	Bit
Step Relay S	**S000000 to S008191**	Decimal	Bit
Link Relay B	**B000000 to B000FFF**	Hexadecimal	Bit
Annunciator F	**F000000 to F002027**	Decimal	Bit
Special Relay M	**M009000 to M009255**	Decimal	Bit
Timer (contact) T	**TS00000 to TS02047**	Decimal	Bit
Timer (coil) T	**TC00000 to TC02047**	Decimal	Bit
Counter (contact) C	**CS00000 to CS01023**	Decimal	Bit
Counter(coil) C	**CC00000 to CC01023**	Decimal	Bit
Timer (present value) T	**TN00000 to TN02047**	Decimal	Word
Counter (present value) C	**CN00000 to CN02047**	Decimal	Word
Data Register D	**D000000 to D006143**	Decimal	Word
Link Register W	**W000000 to W000FFF**	Hexadecimal	Word
File Register R	**R000000 to R008191**	Decimal	Word
Special Register D	**D009000 to D009255**	Decimal	Word
Buffer Memory	**BM00000 to BM00DFF**	Hexadecimal	Word

Notes on Addressing

Please note the following about point addresses for the Mitsubishi A-Series Serial Communications enabler:

- Letters in address entries can be upper or lower case
- Leading zeros on address numbers are optional. Example: X003 is treated the same as X3.
- Device with hexadecimal addressing may be specified in decimal by prefacing the number with D or d. Example: X00A can be entered as X0d10, XD10, X0D10, etc.
- Address ranges may be limited by the address space available in the particular controller model.
- Use BOOL Point Types for devices with Device Type Bit.

- Do not use the Offset field (leave it with the default value of 0) for devices with Device Type Bit.

Adjusting the Communications Time-out Parameter

The communications time-out parameter used by the Mitsubishi A-Series Serial Communications enabler is set to a default value of 7.5 seconds. You can change this parameter in CIMPLICITY software. To do this, add the following record to the Global Parameters file:

COM<port>_TO|1|<value>

Where <port> is the communication port number and <value> is the time out in tenths of seconds. For example,

COM1_TO|1|105

sets the time out for communications on port COM1 to 10.5 seconds.

Default Protocol Parameters

Certain protocol parameters used by the Mitsubishi A-Series Serial Communications enabler have been preset. They are:

Message Wait Time

Preset value = **0**

Unit Address - **0** indicates the programmable controller's CPU. This is not the same as the unit number (that is, node number) which is configurable.

PC CPU

Preset value = **FF**

Default PC CPU number used for addresses entered as **<station number>** only.

If <station number>**.<PC CPU number>** format is used, then the entered <PC CPU number> is used instead of **FF**.

Chapter 38. Mitsubishi A-Series Serial Communications Diagnostics

Mitsubishi A-Series Serial Communications Diagnostics

- Open the Misubishi A-Series Serial Comm Test dialog box.
- Mitsubishi A-Series Serial communications diagnostics steps.

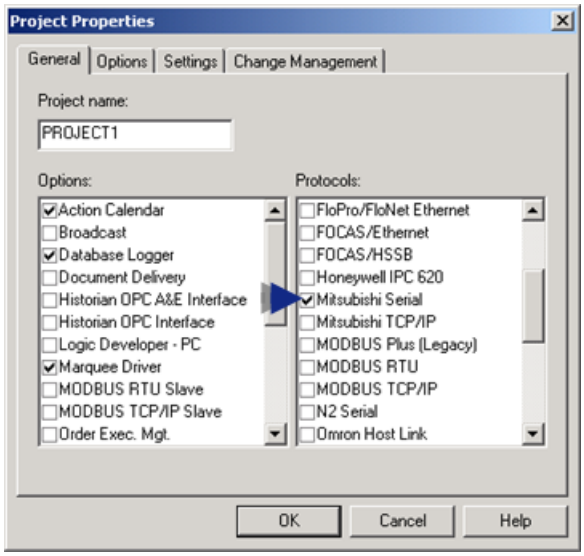
Open the Misubishi A-Series Serial Comm Test dialog box

You can use the Mitsubishi A-Series Serial Diagnostics program to check the basic configuration, cabling and operation of your network without running a CIMPLICITY project. You can use this program to read data from and write data to a specified Mitsubishi programmable controller. A single value (channel or timer/completion bit) is transferred at a time.

You can use the OMRON Host Link Diagnostics program to check the basic configuration, cabling and operation of your network without running a CIMPLICITY project. You can use this program to read and write data from and to a specified OMRON programmable controller. A single value (channel or timer/completion bit) is transferred at a time.

To open the CIMPLICITY Mitsubishi A-Series Serial Comm Test dialog box.

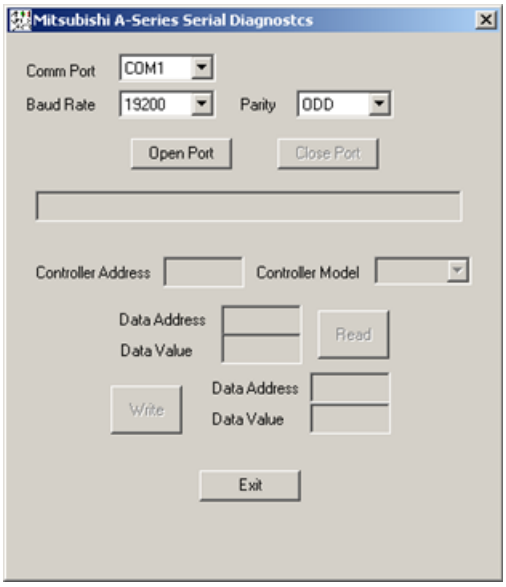
1. Make sure Mitsubishi Serial is checked in the Project Properties dialog box.



2. Select **Project>Equipment>Diagnostics** in the Workbench left-pane.

3. Double click the **Mitsubishi Serial Diag.** icon.

The Mitsubishi A-Series Serial Diagnostics dialog opens with default parameters selected.



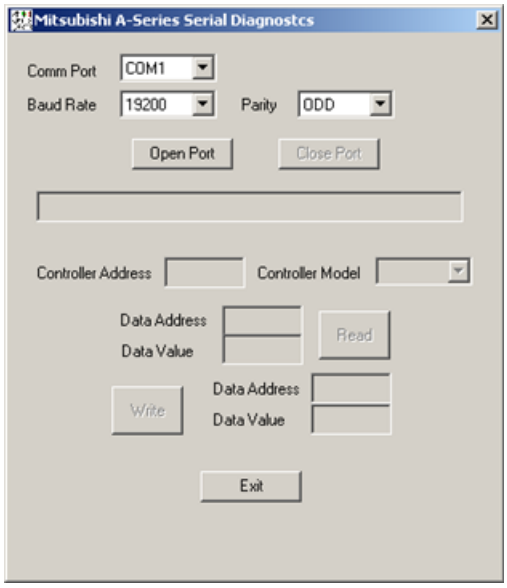
Mitsubishi A-Series Serial communications diagnostics steps

The top section of the dialog contains fields for setting the communication port parameters. A display only field in the center of the dialog displays error and status information. Below the status message field are fields you can use to specify the programmable controller Address, Data Address and Data Values for read and write testing

Step 1 *(page 591)*	Open a communications port.
Step 2 *(page 591)*	Select a target programmable controller.
Step 3 *(page 594)*	Diagnose communication problems.
Step 4 *(page 594)*	Close a communication port and/or change ports and configuration.
Step 5 *(page 594)*	Exit the diagnostic program.

Step 1. Open a Communications Port

1. Select the following.



Field	Select the:
Comm Port	Communication port for the PLC.
Baud rate	Baud rate for the programmable controller.
Parity	Parity for the programmable controller.

2. Click Open Port to initialize communications.

   The status of your request is displayed in the message field.

   If an error message is displayed, check that the selected communication port is not in use by another program and that the port is properly configured and available on the PC and operating system.

Step 2. Select a Target Programmable Controller

Step 2. Select a Target Programmable Controller

1. Enter the device's address in the **Controller Address** field.

Note: This is the same format you use when defining a device address in CIMPLICITY software.

For programmable controllers attached directly to your computer

a. Either point to point via RS-232 or on an RS-422 link,
b. Enter the programmable controller's Station Number, using the following format:

<station number>

Where

< station number > is a hexadecimal number with a valid range of 0 to 1F.

Examples of local addresses

`1A` and `2`

For programmable controllers on a MELSECNET or MELSECNET/B subnetwork

To be accessed via a programmable controller connected to your computer

a. Use the following format:

< station number >.< PC CPU number >

Where

< station number > is the station number of the programmable controller on the local serial network acting as the MELSECNET(/B) gateway and **<PC CPU number>** is the address assigned to the data link module attached to MELSECNET(/B).

a. Make entries in hexadecimal with a valid range of:
- 0 to 1F for the station number
- 0 to 40 for PC CPU number.

Examples of MELSECNET addresses

`3.10`, `A.1A`, and `0.1`

If you enter an invalid programmable controller address, an error message displays on the status line. You must enter a valid address to continue.

2. (After you enter a valid programmable controller address) make a selection in the **Controller Model** field.

Tip: You can click on the down arrow beside the field to view the list of valid selections.

Once you select a controller model, the Read and Write **Data address** fields are enabled.

Option 2.1 *(page 593)*	Read data.
Option 2.2 *(page 593)*	Write data.

Option 2.1. Read Data

You can read data from valid data areas *(page 577)* . Valid data areas are determined by the programmable controller type. The format for entering a data address is the same one you use when you configure points for a CIMPLICITY project.

To read data from the targeted programmable controller:

1. Enter the address of the data in the **Data address** field to the left of the Read button. This activates the Read button
2. Click Read.

   If the read request completes successfully, the data appears in the **Data value** field and the status message field says the read was successful. All read data is displayed in hexadecimal. Bit type data such as discrete I/O, Timer Contact, Time Coil, Counter Contact and Counter Coil are displayed as 1 or 0.

   If the read request fails, an error message is displayed in the status message field.

Option 2.2. Write Data

You can write data to valid data areas *(page 577)* . Valid data areas are determined by the programmable controller type. The format for entering a data address is the same one you use when you configure points for a CIMPLICITY project.

To write data to the targeted programmable controller:

1. Enter a valid data address in the **Data address** field to the right of the Write button.

   This activates the **Data value** field.

2. Enter the data you want to write in the **Data value** field. Enter data values for word devices in hexadecimal notation and enter 1 or 0 for bit device data.

   This activates the Write button.

3. Click Write.

   The status message field displays a message indicating the success or failure of the write request.

   The status line will display a message indicating the result of the write operation, either successful or, in the case of a failure, an error message.

Step 3. Diagnose Communication Problems

If the read or write request fails, a message is displayed in the status message field giving an indication of the failure. Use this information to help diagnose communication problems. Error messages will include the Error Code returned from the target programmable controller where appropriate. Refer to the Mitsubishi A-Series programmable controller's or Computer Link Module's User Manuals for complete error descriptions. Errors will indicate possible problems with data addresses, data ranges, etc.

Time-out errors

Time-out errors may indicate an invalid programmable controller address, incorrect communication parameters settings, or communication port problems. The most common cause of time-out errors is an improperly constructed communication cable.

Step 4. Close a Communication Port and/or Changing Ports and Configuration

You need to close communications when:

- You are finished with diagnostic testing
- You want to select a different port
- You want to change the port's baud rate or parity.

To close communications, click Close port.

Step 5. Exit the Diagnostic Program

1. Click Close port to close and release the communication port.

2. Click Exit to close the dialog and terminate the program.

Chapter 39. Mitsubishi TCP/IP Communications

About Mitsubishi TCP/IP Communications

The Mitsubishi TCP/IP Communications enabler lets you exchange data between Mitsubishi Melsec programmable controllers using TCP/IP protocol and CIMPLICITY projects.

This communications module requires that an Ethernet card be installed and configured in your CIMPLICITY project. In addition, TCP/IP communications must be configured on your computer.

Mitsubishi TCP/IP Supported Devices

The Mitsubishi TCP/IP communications enabler supports the following devices:

Mitsubishi Melsec-A0J2H	Mitsubishi Melsec-A2N_S1
Mitsubishi Melsec-A1	Mitsubishi Melsec-A2S
Mitsubishi Melsec-A1N	Mitsubishi Melsec-A2S_S1
Mitsubishi Melsec-A1S	Mitsubishi Melsec-A2U
Mitsubishi Melsec-A1SJ	Mitsubishi Melsec-A2U_S1
Mitsubishi Melsec-A1S_S1	Mitsubishi Melsec-A3
Mitsubishi Melsec-A2	Mitsubishi Melsec-A3A
Mitsubishi Melsec-A2_S1	Mitsubishi Melsec-A3AU
Mitsubishi Melsec-A2A	Mitsubishi Melsec-A3M
Mitsubishi Melsec-A2A_S1	Mitsubishi Melsec-A3N
Mitsubishi Melsec-A2AS	Mitsubishi Melsec-A3U
Mitsubishi Melsec-A2AS_S1	Mitsubishi Melsec-A3X
Mitsubishi Melsec-A2C	Mitsubishi Melsec-A4U
Mitsubishi Melsec-A2N	

An Ethernet Interface Module type AJ71E71 or A1SJ71E71-B2 must be installed in the programmable controller.

Communications to these devices is only available when the devices are in **RUN** mode.

Mitsubishi TCP/IP Supported Memory Types

The Mitsubishi TCP/IP communications enabler supports reading and writing to the following memory types:

- Data Register
- Link Register
- Files Register
- Input
- Output
- Internals
- Link Relay
- Annunciator
- Random Access

Mitsubishi TCP/IP Hardware Configuration Requirements

Mitsubishi TCP/IP Hardware Configuration Requirements

An Ethernet Interface Module type AJ71E71 or A1SJ71E71-B2 must be installed in the programmable controller. These modules have switches that you may want to change from the factory settings.

A1SJ71E71-B2 Settings

For the A1SJ71E71-B2 module, the switches of interest are:

Switch	Setting	Description
SW1	**OFF**	Line is closed when TCP time-out occurs
	ON	Line is not closed when TCP time-out occurs
SW2	**OFF**	Use binary communications
	ON	Use ASCII communications
SW3	**OFF**	Writes to the programmable controller are disabled when CPU is in RUN state
	ON	Writes to the programmable controller are enabled when CPU is in RUN state
SW4	**OFF**	Start initial processing immediately when a message is received (quick start)

	ON	Start initial processing 20 seconds after delay time when a message is received (normal start)

Note: All switches are set to the OFF position in the factory.

AJ71E71 Settings

For the AJ71E71 module, the switches of interest are:

Switch	Setting	Description
SW1	**OFF**	Line is closed when TCP time-out occurs
	ON	Line is not closed when TCP time-out occurs
SW2	**OFF**	Use binary communications
	ON	Use ASCII communications
SW7	**OFF**	Writes to the programmable controller are disabled when CPU is in RUN state
	ON	Writes to the programmable controller are enabled when CPU is in RUN state
SW8	**OFF**	Start initial processing immediately when a message is received (quick start)
	ON	Start initial processing 20 seconds after delay time when a message is received (normal start)

Note: Switches 3-6 are not currently used. All switches are set to the OFF position in the factory.

Mitsubishi TCP/IP Related Documents

You should have the following documentation available when configuring this interface:

Mitsubishi Melsec Programmable Controller A User's Manual Ethernet Interface Module type AJ71E71 with a Revision of July, 1993 or later.

The Mitsubishi User's Guide for your model of programmable controller.

In addition, the following Mitsubishi manual are recommended:

ACPU Programming Manual (Fundamentals)IB(NA)?66249

ACPU Programming Manual (Common Instructions)IB(NA)?66250

The following application note is recommended:

Developer's Guide For Mitsubishi Ethernet Module Using the AJ71E71 by Albert G. Baier.

To obtain a copy of this application note, please contact your Mitsubishi distributor and request that they obtain this document for you from Mitsubishi Electronics America, Industrial Automation Division

Mitsubishi TCP/IP Communications Configuration Checklist

Mitsubishi TCP/IP Communications Configuration Checklist

Prior to using this communications interface, you must perform the following steps:

1. Install an Ethernet card in your computer.
2. Configure the minimum TCP/IP Communications for your computer, including assignment of a name and TCP/IP address for the computer.
3. Assign a TCP/IP address for each Mitsubishi Melsec-A programmable controller. If you are on a company-wide or standardized network, you should obtain your TCP/IP addresses from your network administrator
4. For each programmable controller communicating with CIMPLICITY software via Ethernet, assign a Socket port number for the connection. If you are on a company-wide or standardized network, you should obtain your Ethernet port number from your network or system administrator.
5. Configure the TCP/IP address for each Mitsubishi Melsec-A programmable controller on your computer.

   In the CIMPLICITY supported Windows operating system, use the

   \Windows\System32\drivers\etc\hosts

   file for the configuration.

   In the **hosts** file, add one line for each Mitsubishi Melsec-A programmable controller in the following format:

   < IP_address> <PLC_name> <PLC_alias>

   Note that each field is separated by at least one space. For example, if the IP address configured for the Mitsubishi Melsec-A programmable controller is 1.2.3.4 and the name for the PLC is "MELSEC", the line in the hosts file for this controller will be:

1.2.3.4 MELSEC melsec

6. On each programmable controller, configure the host address for the programmable controller as well as assign the Internet address and socket port number to a connection. Please refer to the required documents above for assistance in developing the needed ladder logic

7. Use the Mitsubishi TCP/IP test program to check the configuration of your network without starting a CIMPLICITY project.

Using the Mitsubishi TCP/IP Test Program

You can use the Mitsubishi TCP/IP test program (**melsec_diag.exe**) to check the configuration of your network without starting CIMPLICITY software.

To invoke the test program:

1. Open your project's Workbench.

2. Select Command Prompt... from the Tools menu to open an MS-DOS window.

3. In the MS-DOS window, type the following at the MS-DOS prompt:

   melsec_diag <host_name> <socket_port_number>

   where <host_name> is the name of the Mitsubishi PLC in the **hosts** file and <socket_port_number> is the socket port number you assigned to the PLC.

   If the test succeeds, you will see output with the following format:

   ```
   Socket port: <socket_port number>
   <computer_name> is connected to <host_name> PLC.
   Communication between <computer_name> and <host_name> PLC is
    disconnected (normal exit).
   ```

   For example, if the computer name is ABC123, the PLC name is MELSEC and the configured socket port number for the PLC is 1234, the MS-DOS command is:

   ```
   melsec_diag MELSEC 1234
   ```

   and the output is:

   ```
   Socket port: 1234
   ABC123 is connected to MELSEC PLC.
   Communication between ABC123 and MELSEC PLC is disconnected (normal
    exit).
   ```

If the test fails, then the error message will be displayed following the socket port number. In this case, check your Ethernet connections, ladder logic, and computer configuration to ensure that all are set correctly. Any inconsistency will prevent successful communication.

In addition, the Mitsubishi TCP/IP diagnostic program has been enhanced to validate device point addresses. Previously, only device connectivity could be validated. To use the diagnostic program, open a command prompt from the Project's Configuration Cabinet and issue the following command:

```
melsec_diag <host_name> <socket_port_number> <cpu_id> <address>
 <length> <communication mode>
```

Where:

<host_name>=name of the Mitsubishi PLC in the **hosts** file

< socket_port_number>= socket port number you assigned to the PLC

<cpu_id>=PC CPU assigned to the Mitsubishi PLC (value in decimal)

<address>=device point address (for detail on format, refer to Application Configuration section of the Mitsubishi TCP/IP Documentation)

<length>=number of bytes to be read

<communication mode>=A (for ASCII) or B (for Binary)

The following example can be used to validate reading 2 bytes from Extension File Register/ block 0 in binary communication mode. Assume the computer name is **ABC123,** the PLC name is **MELSEC**, the PC CPU value of FF(h) or **255** (decimal), the port number is **1234**. To read the Register, the MS-DOS command is:

```
melsec_diag MELSEC 1234 255 r0 2 B
```

and the output is:

```
PLC Name: MELSEC, Port Number: 1234
ABC123 is bound to the socket.
ABC123 is connected to MELSEC.
Parameters entered for reading data from MELSEC:
       Address: r0, length: 2(bytes), cpu id: 255, communication mode:
 B
Send BINARY command to PLC:
       0x17 0xffffffff 0x00 0x02 0x00 0x00 0x00 0x00 0x20 0x52 0x00
 0x00
Read 4 bytes from PLC:
       0xff97 0x00 0x01 0x00
```

```
Communication between ABC123 and MELSEC is closed (normal exit).
```

CIMPLICITY Configuration for Mitsubishi TCP/IP

CIMPLICITY Configuration for Mitsubishi TCP/IP

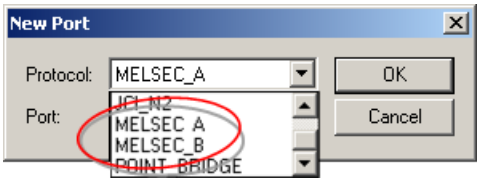
As with other CIMPLICITY software communications enablers, you must complete the Port, Resource, Device and Device Point Configuration to setup the Mitsubishi TCP/IP Communications.

Mitsubishi TCP/IP Port Configuration

Mitsubishi TCP/IP Port Configuration

When you configure a port for Mitsubishi TCP/IP communications, the New Port dialog opens.

New Port



Select the **Protocol** used by the programmable controller:

For ASCII communications, select MELSEC_A.

For Binary communications, select MELSEC _B.

Note: SW2 controls the mode of communication on the programmable controller:

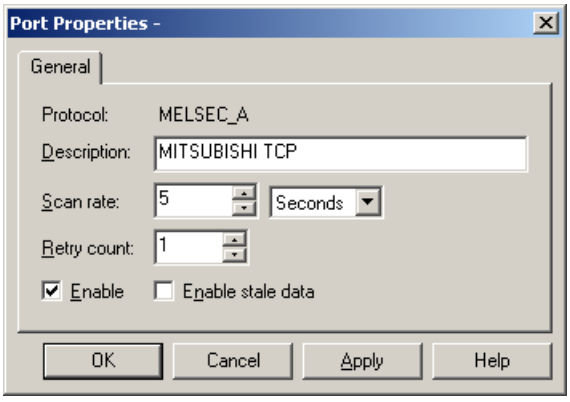
On = ASCII

Off = binary

If you change the position of the switch, the change does not take effect until the controller has been reset.

Make sure that one of the Port names is selected. Click **OK** to display the Port Properties dialog for the protocol.

Mitsubishi TCP/IP General Port Properties



Use the General tab in the Port Properties dialog box to enter general information for the port. You can define the following:

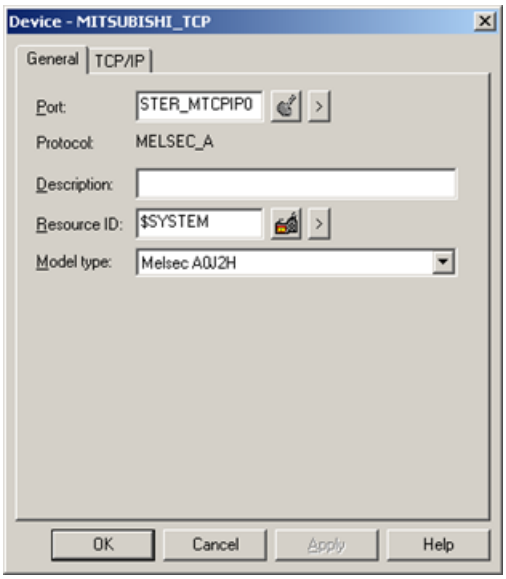
Description	Enter an optional description to help you identify the port.
Scan Rate	Enter the base scan rate for the port. Point scan rates will be multiples of the base rate. You can specify a scan rate in **Ticks** (100 ticks = 1 second), **Seconds**, **Minutes**, or **Hours**.
Retry Count	If communications cannot be established to a device on this port, the device is considered to be down and a $DEVICE_DOWN alarm is generated. Once a device is down, periodic attempts are made to resume connection to it.
	Enter the number of scans to wait before attempting to reconnect to a device on this port after a communications error is detected.
Enable	Set this check box if you want the port to be enabled when the project starts. If you clear this check box, the port will not be enabled when the project starts, and points will not be available for devices on the port.

Mitsubishi TCP/IP Device Configuration

Mitsubishi TCP/IP Device Configuration

When you configure a Mitsubishi TCP/IP device, the Device Properties dialog for devices using this protocol opens.

General Device Properties



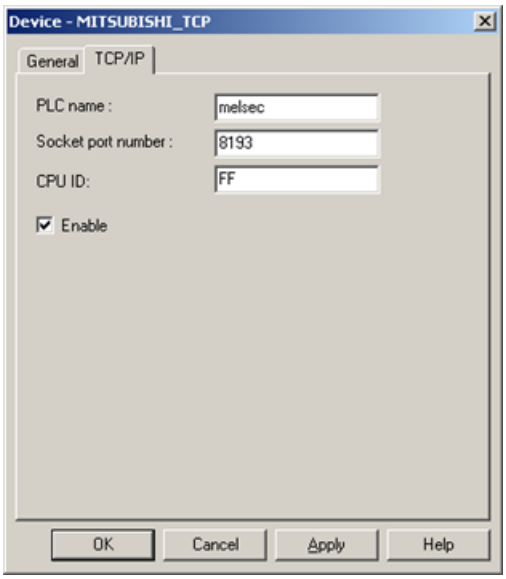
Use the General tab in the Device dialog box to enter general information for the device. You can define the following:

Port	Select the port for this device. Click buttons to the right of the **Port** field to select the port as follows.		
	Port		Display the list of ports and select one.
	Popup Menu		Create a new port, edit the current port, or select a port from the list of ports.
Description	Enter an optional description to help you identify the device.		
Resource	Enter a resource that can be associated with this device for alarm generation. Click buttons to the right of the **Resource** field to select the resource, as follows.		
	Resource		Display the list of resources and select one.
	Popup Menu		Create a new resource, edit the current resource, or select a resource from the list of resources.
Model type	Select the type of device. Click the drop-down button to the right of the input field to display your choices, then make a selection. For this protocol, the choices are:		

	A0J2H	**A2A_S1**	**A2U_S1**
	A1	**A2AS**	**A3**
	A1N	**A2AS_S1**	**A3A**
	A1S	**A2C**	**A3AU**
	A1SJ	**A2N**	**A3M**
	A1S_S1	**A2N_S1**	**A3N**
	A2	**A2S**	**A3U**
	A2_S1	**A2S_S1**	**A3X**

	A2A	**A2U**	**A4U**

Melsec Device Properties



Use the Melsec properties to enter information about communications for the device. You can define the following:

PLC name	Enter the name of the PLC. Use the same name that you configured in the **hosts** file.
Socket port name	Enter the socket port number you configured on the PLC.
CPU ID	Mitsubishi TCP/IP communications can communicate directly with a PLC with an AJ71E71 interface card or a Local Station PLC connected via MELSECNET. Enter the CPU ID as a hexadecimal number. The following values are valid:

	FF	Use when communicating directly with a PLC with AJ71E71 interface card.
	0	Use to communicate with the MELSECNET Master Station.
	1-40	Use to communicate with a MELSECNET Local Station. Value indicates station number.

Enable	Set this check box to enable the device when the project starts. If you clear this check box, the device will not be enabled and points associated with the device will be unavailable.

Mitsubishi TCP/IP Point Configuration

Mitsubishi TCP/IP Point Configuration

Once your devices are configured, you many configure points for them. Fields in the Point Properties dialog box have configuration values that are unique to or have special meaning to Mitsubishi TCP/IP communications.

Important: Point by addresses are not supported with Mitsubishi TCP/IP communications.

General Point Properties

On the General tab in the Point Properties dialog box, you may configure points for **Read** or **Read/Write** access.

Device Point Properties

Device Point Properties

On the Device tab in the Point Properties dialog box:

Update Criteria	The update criteria determines how the data will be collected and/or updated in your CIMPLICITY software application.
	Enter On Change or On Scan for points whose values should be polled by Mitsubishi TCP/IP communications at regular intervals.
	Enter On Demand On Scan or On Demand On Change for points whose values should be polled by Mitsubishi TCP/IP communications at regular intervals while users are viewing them.
	Enter Unsolicited for points that are sent by the programmable controller from the Fixed Buffer.
Address	Enter the point address of the data using the following format: **<DEVICE_TYPE><OFFSET>**
	For some device types, the offset is a hexadecimal number, while in other cases, the offset is a decimal number.
	Use the table below to determine how to specify your device and offset.

	Device Name	Device Type	Offset
	Data Register	**D**	Decimal
	Link Register	**W**	Hex
	Files Register	**R**	Decimal
	Timer (three types)		
	Present Value	**TN**	Decimal
	Contact	**TS**	Decimal
	Coil	**TC**	Decimal
	Counter (three types)		
	Present Value	**CN**	Decimal

	Contact	**CS**	Decimal
	Coil	**CC**	Decimal
	Input	**X**	Hex
	Output	**Y**	Hex
	Internals	**M**	Decimal
	Link Relay	**B**	Hex
	Annunciator	**F**	Hex
	Random Access	**Z**	Hex

Unsolicited Communication via the Fixed Buffer is also supported.

Note on Unsolicited Point Addresses

Unsolicited addresses are referenced with a U followed by a hex offset of 4 digits including leading zeros. For example, U012C is a valid address, while U12C is not. The valid range for the offset is 0000-7FFF. For example, U7FFF is a valid address, while U8000 is not.

For all device types except the Fixed Buffer, the address may be entered in any combination of upper and lower case letters. For Fixed Buffer devices, the address must be configured with all uppercase characters or digits.

Note on Addressing Formats

For Link Register, Input, Output, Link Relay, Annunciator, and Random Access addresses you can also use the following format to define the address in decimal format rather than hexadecimal format:

```
<DEVICE_TYPE>D<DECIMAL_OFFSET>
```

Unsolicited addresses can also be specified in a decimal format. Decimal unsolicited addresses are referenced with a UD followed by a decimal offset of 5 digits including leading zeros. For example, UD00800 is a valid address, while UD800 is not. The valid range for the offset is 00000-32767. For example, UD32767 is a valid address, while UD32768 is not.

The Mitsubishi TCP/IP Communications enabler allows access to Extension File Register data. Previously, for File Register data, the File Register Command was employed. The enhanced enabler now uses the Extension File Register command that provides access to both File Register data and Extension File Register data. The format used for configuring File Register/Extension File Register points remains the same: **R**x, where x = address offset.

Note on Arrays

The Mitsubishi TCP/IP protocol does not support array sizes greater than 256 bytes for the following point types:

- BOOL
- Timer Contacts
- Timer Coils
- Inputs
- Outputs
- Internals
- Internal Extended
- Link Relays
- Annunciators

Unsolicited Data

Unsolicited data is supported for this communication interface. Ladder logic must be developed to support the communications. Be sure that the unsolicited data is transmitted at a reasonable rate to the CIMPLICITY host computer. The ladder logic should also check to ensure that the connection has been made and that the requests are acknowledged.

Examples of ladder logic are provided in the Mitsubishi Melsec P.L.C. documentation.

The format of the unsolicited data message is:

```
Number of words in the message
Offset address
Data value
Offset address
Data value
   .
   .
   .
```

The communication enabler automatically assumes that the Fixed Buffer data is to be updated.

Consider the following example:

```
4
0x500
0x10
0x501
0x20
```

Any point with a point address of U0500 whose point size was less than or equal to 2 bytes is updated to a value based on 0x10. For example, an ANALOG_16 point at U0500 would be 0x10.

Any point with a point address of U0501 is also be updated as long as the point size is less than or equal to 2 bytes. The points are updated based on U0501 having a value of 0x20.

Advanced Configuration Requirements

Advanced Configuration Requirements

Advanced configuration requirements include:

- Change timing and performance characteristics.
- Match addresses with unsolicited messages.

Change Timing and Performance Characteristics

Change Timing and Performance Characteristics

You can adjust the timing and performance characteristics of the MITSUBISHI TCP/IP communications by changing or defining the logical names listed below.

These logical names are defined in the CIMPLICITY logical names file. This file is located in your project's **\data** directory and is named **log_names.cfg**.

Logical names for changing timing and performance characteristics are:

- DCQ_DEAD_TIME
- DC_TCP_POLL_MS
- DC_CONNECT_URETRY_CNT
- DC_CONNECT_MS
- MSYNC_TICKS and MMAX_SYNC_TICKS

DCQ_DEAD_TIME

When Mitsubishi TCP/IP communications writes an asynchronous request via a socket to the PLC and the request remains outstanding for more than 120 seconds, the device is declared dead.

To change the time-out:

- For all devices on a given port:

Add the following record to the log_names.cfg file for all devices on a given port:

`<PORT>_DCQ_DEAD_TIME|P|default|10|<seconds>`

- For all devices in a selected project:

Add the following record to the log_names.cfg file

`DCQ_DEAD_TIME|P|default|10|<seconds>`

Where

<seconds> is the number of seconds to wait before declaring the device dead.

You can disable this feature by using -1 for <seconds>.

⚠ **Important:** The port defined logical has precedence over the project-wide logical.

DC_TCP_POLL_MS

When Mitsubishi TCP/IP communications polls a socket for input, it uses a polling interval of 10 milliseconds.

If you see a large number of Ethernet error messages in the Status Log, consider increasing this value.

To change this interval:

- For all devices on a given port:

Add the following record to the log_names.cfg file.

`<PORT>_DC_TCP_POLL_MS|P|default|10|<milliseconds>`

- For all devices in a selected project:

Add the following record to the log_names.cfg file.

`DC_TCP_POLL_MS|P|default|10|<milliseconds>`

Where

<milliseconds> is the number of milliseconds in the polling interval.

Enter a number greater than 10.

⚠ **Important:** The port defined logical has precedence over the project-wide logical.

DC_CONNECT_URETRY_CNT

After Mitsubishi TCP/IP communications declares a device "dead", it periodically tries to re-establish communications between poll checks. The default number of tries is 10.

To change the number of retries:

Add the following record to the **log_names.cfg** file:

```
DCQ_CONNECT_URETRY_CNT|P|default|10|<retries>
```

Where

<retries> is the number of retries to re-establish communications between poll checks.

Important:

- Larger values will tend to increase the amount of time it takes to recognize that a device has recovered and re-establish the connection.
- Smaller values will increase the overhead if there are a number of devices down.

DC_CONNECT_MS

Once Mitsubishi TCP/IP communications initiates a request to a device, it waits 100 milliseconds for the connection to be established. If the connection is not established at the end of this time interval, the connect request times out.

To change this connect request interval:

- For all devices on a given port:

Add the following record to the log_names.cfg file.

`<PORT>_DC_CONNECT_MS|P|default|10|<milliseconds>`

- For all devices in a selected project:

Add the following record to the log_names.cfg file.

`DC_CONNECT_MS|P|default|10|<milliseconds>`

Where

<milliseconds> is the number of milliseconds to wait for a connection to be established. between poll checks.

Important:

- If the value is too small, the Mitsubishi TCP/IP communications will fail to establish a connection to the device.
- A larger value will tend to impact the performance of Mitsubishi TCP/IP communications in a way directly proportional to the number of down devices and the frequency of retries.
- The port defined logical has precedence over the project-wide logical.

MSYNC_TICKS and MMAX_SYNC_TICKS

Service requests such as read point, setpoint, read address and write address as well as point initialization are performed synchronously. Defaults are as follows.

Default behavior	Use the defined maximum asynchronous age request. This value is defined by the logical DCQ_DEAD_TIME (the units are automatically converted to ticks to ensure equivalence).
Default value	If no value is defined, the default value is 2 minutes.

- The logical controls how long (in ticks) Mitsubishi TCP/IP communications will wait for a synchronous response from a device, once the request has been made of the device.
- To change the time-outs, for all devices on a single port, add the following records to the log_names.cfg file: `<PORT>_MSYNC_TICKS|P|default|10|<ticks>`

- To change the time-outs, for all devices in a selected project, add the following records to the log_names.cfg file: `MSYNC_TICKS|P|default|10|<ticks>` or `MMAX_SYNC_TICKS|P|default|10|<ticks>`, where <ticks> is the time delay represented in ticks. 1 tick = 1/100 seconds.
- The order of precedence for the logicals is:

1. <PORT>_MSYNC_TICKS (port defined logical).
2. MSYNC_TICKS (project-wide defined globals).
3. MMAX_SYNC_TICKS (project-wide defined globals).
4. DCQ_DEAD_TIME (default).

Note: It may be necessary to increase the number of ticks to 50 if the Mitsubishi TCP/IP driver is timing out with errors.

Match Addresses with Unsolicited Messages

Match Addresses with Unsolicited Messages

The following logical names control how addresses are matched with unsolicited messages. By default, Mitsubishi TCP/IP communications matches each register address with either the HEX or DECIMAL addresses or both.

- If either <prcnam>**_UNSO_HEX** or <prcnam>**_UNSO_DEC** is defined to be TRUE, addresses are matched using only the defined method.
- If <prcnam>**_UNSO_ALL_TYPES** is defined to be true, Mitsubishi TCP/IP communications matches the address with both the HEX and DECIMAL addresses.
- If more than one method is defined, the default behavior will take precedence.

The value of **<prcnam>** is **MTCPIP0** in a standard configuration.

The default method is <prcnam>**_UNSO_EITHER**.

<prcnam>_UNSO_HEX

The format for this record in the **log_names.cfg** file is:

```
<prcnam>_UNSO_HEX|P|default|10|<value>
```

Where <prcnam> is the Mitsubishi TCP/IP communications process name and **<value>** is TRUE or FALSE.

When this logical name is set to TRUE, unsolicited point addresses are matched to their Hexadecimal format addresses.

<prcnam>_UNSO_DEC

The format for this record in the **log_names.cfg** file is:

```
<prcnam>_UNSO_DEC|P|default|10|<value>
```

where <prcnam> is the Mitsubishi TCP/IP communications process name and **<value>** is TRUE or FALSE.

When this logical name is set to TRUE, unsolicited point addresses are matched to their Decimal format addresses.

<prcnam>_UNSO_ALL_TYPES

The format for this record in the **log_names.cfg** file is:

```
<prcnam>_UNSO_ALL_TYPES|P|default|10|<value>
```

where <prcnam> is the Mitsubishi TCP/IP communications process name and **<value>** is TRUE or FALSE.

When this logical name is set to TRUE, unsolicited point addresses are matched to both their Hexadecimal and Decimal format addresses.

<PRCNAM>_UNSO_EITHER

The format for this record in the **log_names.cfg** file is:

```
<prcnam>_UNSO_EITHER|P|default|10|<value>
```

where <prcnam> is the Mitsubishi TCP/IP communications process name and <value> is TRUE or FALSE.

When this logical name is set to TRUE, unsolicited point addresses are matched to either their Hexadecimal or Decimal format addresses or both.

This is the default method.

Mitsubishi TCP/IP Troubleshooting

Mitsubishi TCP/IP Troubleshooting

The CIMPLICITY Status Log often provides helpful hints on communication problems that you might encounter, especially during initial configuration.

The following message indicates that the Scan Rate you selected faster than the data can be collected. This is not an error.

```
Skipping polling around <POINT ID> because request is already pending.
```

The following messages often indicate one or more configuration errors.

Other error messages can appear in the CIMPLICITY Status Log. Please check the CIMPLICITY Status Log first if you encounter problems with your device communications. If you need help

interpreting the messages in the CIMPLICITY Status Log, please contact the CIMPLICITY Technical Support Hotline.

Gethostname failed

If the PLC name and socket port number entered in the Melsec property tab are not configured in your host file, the following message will appear in the log file:

```
Gethostname for <hostname> failed  errno <number>
```

To resolve this problem, configure the host name into your computer system as described in step 5 of the Melsec PLC Communications Configuration Checklist. Remember that the names or aliases must match exactly.

Unable to get a socket

If you do not have enough sockets in your system, the following will appear:

```
Unable to get a socket  errno = <number>
```

Contact your system administrator for help in tuning the operating system.

Bind for port failed

If you have not properly configured the socket port number within your CIMPLICITY application with that configured on the PLC, you will get the following error message:

```
Bind() for <hostname> port <port number> failed  REFUSED
```

You may also get this message at startup if another application is using your port or the socket has not cleared from a previous use. You may also get this message if your CIMPLICITY configuration for the device is inconsistent between your computer and the programmable controller.

To determine if the socket is in use, type the following command:

```
netstat
```

and look for the line showing your configured <Host name>**:**<Port number> combination under the "Foreign Address". If the last column says "TIME_WAIT", this means that the connection was previously in use and is in the process of cleaning up. No further action is needed.

If the last column says "ESTABLISHED", this means that the socket port is in use. Check to make sure that you do have a connection (you can usually tell by examining the leds on your

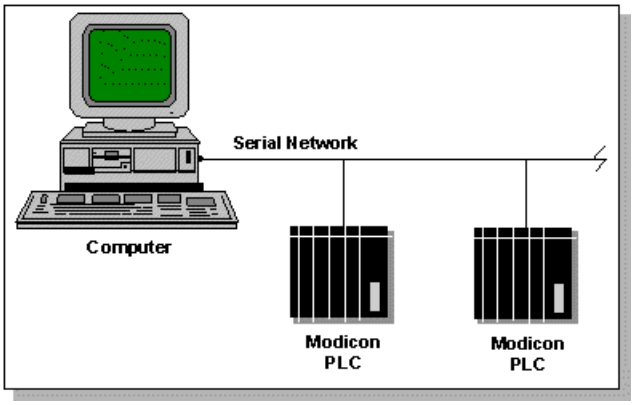
programmable controller's Ethernet card). If you do not have a connection, contact your system administrator for assistance in locating the application using the port.

If the command yields no information, this usually indicates that the IP address or socket port number configured on the computer does not match the configuration in the programmable controller. Review the configuration on both the computer and programmable controller and correct the mismatch.

Chapter 40. Modbus RTU

About Modbus RTU

The Modbus RTU Communications option supports a multi-drop configuration with the Modbus RTU Communications enabler functioning as the client. You must configure a CIMPLICITY device for each physical device (or server) from which data will be collected.



This communications option supports the following CIMPLICITY features:

- Polled reads at user defined rates
- Poll after setpoint
- Triggered reads
- Analog deadband
- Alarm on communications failure
- Server Redundancy configurations

This communications option supports the following data types:

- Signed 8, 16, and 32 bit integers
- Unsigned 8, 16, and 32 bit integers
- Floating point
- Text
- Arrays

Modbus RTU Supported Devices

The Modbus RTU Communications option supports communications to the following programmable controllers:

- Series Six with CCM3 card (Version 1.06 or later)
- 3720 ACM Electronic Power Meter (Firmware Version 1.5 or later is required if you want to read/write negative numbers)
- Modicon 484
- Modicon 584
- Modicon 884
- Modicon 984
- Modicon Micro 80
- Generic PLC - This is a special device model for generic devices where the user defines the device characteristics.
- DYN_PLC - This is a special device model for generic devices. The range of each of the memory types is determined by the device communication interface.
- STAT_PLC - This is a special device model for generic devices where the user defines the device characteristics and starting addresses for points are validated by reading them from the device.

Modbus RTU Supported Memory Types

Modbus RTU Supported Memory Types

The Modbus RTU Communications option supports the following memory types on GE Intelligent Platforms, Inc. Series Six and Modicon programmable controllers:

- Series Six Controllers
- 3720 ACM Electronic Power Meter
- Modicon Controllers
- Double precision notes.

Series Six Controllers

The Modbus RTU Communications option supports reads from the following memory types on Series Six with CCM3 in RTU mode:

Output Table	0xxxx or 0xxxxx
Input Table	1xxxx or 1xxxxx
Registers	3xxxx or 3xxxxx 4xxxx or 4xxxxx
General Reference	6yxxxx or 6yxxxxx

The Modbus RTU Communications option supports writes to the following memory types on Series Six with CCM3 in RTU mode:

Output Table	0xxxx or 0xxxxx
Registers	4xxxx or 4xxxxx
General Reference	6yxxxx or 6yxxxxx

3720 ACM Electronic Power Meter

To successfully communicate with the 3720 ACM Electronic Power Meter, you must:

- Set Register Size to 32 bits in the ACM.
- Set **Invalid objects** to **Yes** in the 3720 ACM.
- Use Firmware Version 1.5 or later if you are going to read/write negative numbers.

The Modbus RTU Communications option supports reads and writes from the following memory types on a 3720 Electronic Power Meter operating in 32 bit register size mode:

32 bit Registers	4xxxxor 4xxxxx
32 bit Registers in mod 10000 format	D4xxxx or D4xxxxx

Modicon Controllers

The Modbus RTU Communications option supports reads from the following memory types on Modicon controllers:

Coils	0xxxx or 0xxxxx
Inputs	1xxxx or 1xxxxx
Input Registers	3xxxx or 3xxxxx
Holding Registers	4xxxx or 4xxxxx
General Reference	6yxxxx or 6yxxxxx
Holding Registers as Double Precision	D4xxxx or D4xxxxx

The Modbus RTU Communications option supports writes to the following memory types on Modicon controllers:

Coils	0xxxx or 0xxxxx
Holding Registers	4xxxx or 4xxxxx
General Reference	6yxxxx or 6yxxxxx

Holding Registers as Double Precision	D4xxxx or D4xxxxx

CAUTION: When writing to Coils (0xxxx or 0xxxxx), memory protect and coil disable are overwritten.

On Modicon controllers, coils that are not programmed in the controller logic program are not automatically cleared on power up. This means that if a coil is set to 1, the output will remain HOT until explicitly set to 0.

Writing to Holding Registers overrides the controller memory protect.

Double Precision Notes

D4xxxx(x) points are Holding Register Points interpreted as Double Precision. Two registers are used for each point with both registers assumed to have a positive value.

Because of this mapping:

- Do not declare Double Precision points with overlapping addresses. For example, D40001 and D40002 overlap, while D40001 and D40003 do not.
- Do not define points that use the same Holding Register and Double Precision point address (for example, defining one point with the address D40001 and another with the address 40001).

Configure Double Precision points as **DINT** or **UDINT** point.

Arrays are not supported for Double Precision points.

Modbus RTU Required Documents

You should have the following document available when configuring a Series Six for Modbus RTU communications:

Series Six Programmable Controllers Data Communications Manual (GEK-25364)

You should have the following document available when configuring a 3720 ACM Electronic Power Meter for Modbus RTU communications:

3720 ACM / Modbus Serial Communications Protocol

For all other programmable controllers, you should have the documentation for the controller available.

Modbus RTU Hardware Configuration Requirements

Modbus RTU Hardware Configuration Requirements

The Modbus RTU protocol allows one client and up to 247 servers on a common line. In CIMPLICITY software, the Modbus RTU Communications Enabler is the client.

Many configurations limit the number of servers to a smaller number. For example, Gould's J478 modem restricts the number of servers to 32. Contact your hardware vendor for information about the number of servers that your hardware configuration can support.

Modbus RTU Hardware Installation

1. Configure the controller/communications card for Modbus RTU communications.
2. Assign each server within the configuration a unique server address between 1 and 247 (decimal).
3. Configure all servers at the same baud rate and parity as the client (the Modbus RTU Communications enabler).

   Connecting to a Series Six

   To connect a personal computer to a Series Six with a CCM3 card installed, refer to the Series Six Programmable Controllers Data Communications Manual (GFK-25364).

Modbus RTU Serial Port Configuration

Most personal computers come with two (2) serial communication ports. You may use either port for Modbus RTU device communications. No special port configuration is required.

You may also connect to Modbus RTU devices over a terminal server using Telnet or TCP protocol.

Validating Modbus RTU Communications

Validating Modbus RTU Communications

The Modbus RTU Communications option includes a diagnostic program called **mb_test.exe**. You can use **mb_test** to perform read tests for each of the supported memory types.

Before you can use this program, you must successfully install the CIMPLICITY Modbus RTU Communication option and configure the Modbus RTU ports and devices using CIMPLICITY application configuration functions.

You cannot use this program while you are running the Modbus RTU device communication enabler over the same communications port.

To start the program:

1. Open the Workbench for the project where the Modbus RTU driver is configured.
2. Stop the project.
3. From the Tools menu, select Command Prompt... to open the Command Prompt window for your project.
4. In the Command Prompt window, type the command to invoke the diagnostic test you wish to perform.

   The format for the command is:

   ```
   mb_test [-B<baud rate>] [<parity>] [-T<port_id >]
   [-S<server id>] [-R<address>] [-C<count>]
   ```

   Where

-B<baud rate>	Defines the speed of the terminal line.
	Default value is 9600.
<parity>	Defines the parity for communications. Choose one of the following:

	-E	Defines EVEN parity.
	-O	Defines ODD parity.
	-N	Defines NO parity.

	If no parity is specified, the default is EVEN parity.
-T<port_id>	Defines the port to be used for communications.
	Default value is COM1.
-S<server id>	Defines the server to be addressed.
	Default value is 1.
-R<address>	Defines where on the server controller a read should begin.

	Default is no read, just device connect.
-C<count>	Defines the number of elements to be read.
	Default value is 1 element.
	For the 3270 ACM, set **<count>** to an even number greater than or equal to 2.

For example, the command to read the first five holding registers on server 7 from COM1 at 4800 baud using ODD parity is:

```
$ mb_test -B4800 -O -S7 -TCOM1 -R40001 -C5
```

Modbus RTU Communications Problem Checklist

1. COM port is configured in the system.
2. Cable between the computer and the controller is correctly wired.
3. Baud rate and parity on the computer are consistent with those on the controller.
4. Controller's Modbus port is configured for RTU communications.
5. server address is correct.

CIMPLICITY Configuration for Modbus RTU

CIMPLICITY Configuration for Modbus RTU

When you are configuring ports, devices, and device points, you will be asked for some information that is specific to Modbus RTU communications.

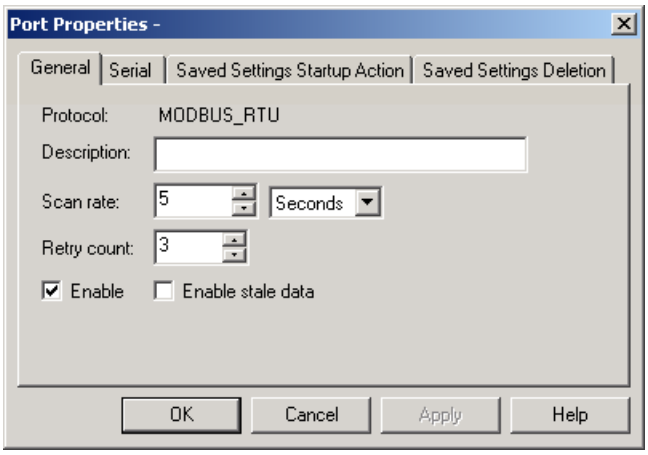
Modbus RTU Port Configuration

Modbus RTU Port Configuration

1. In the **Protocol** field, select MODBUS RTU from the list of available protocols.
2. In the **Port** field, select the communication port that will be used for Modbus RTU communications.

   When you click **OK** to create the port, the Port Properties dialog box for the protocol opens.

General Port Properties

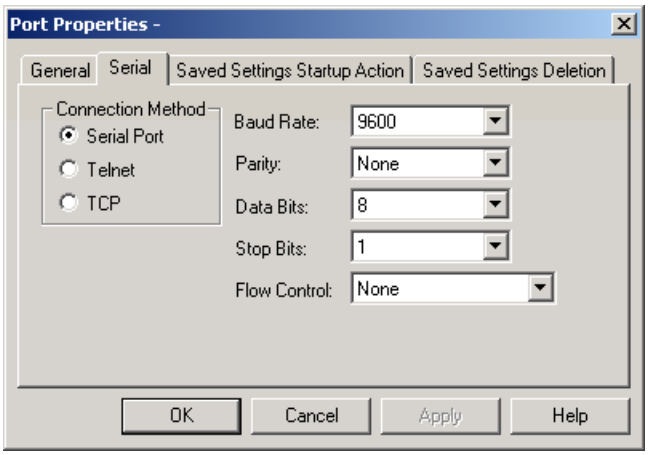


Use the General properties to enter general information for the port. You can define the following:

Description	Enter an optional description to help you identify the port.
Scan Rate	Enter the base scan rate for the port. Point scan rates will be multiples of the base rate. You can specify a scan rate in **Ticks** (100 ticks = 1 second), **Seconds**, **Minutes**, or **Hours**.
Retry Count	If communications cannot be established for a device on the port, the device is considered to be down, and a $DEVICE_DOWN alarm is generated. Once a device is down, periodic attempts are made to resume connections to it.
	Enter the number of scans to wait before attempting to reconnect to a device after a communications error has been detected.
Enable	Check if you want the port to be enabled when the project starts. If you clear this check box, the port will not be enabled when the project starts, and points will not be available for devices on the port.
Enable stale data	Check to keep the point available in most circumstances that would have made it unavailable. However the point value will be stale, meaning it is the last known value and may or may not have changed in the device.

Serial Port Properties: Serial Port Connection

If you check the Serial Port connection method, define the following.

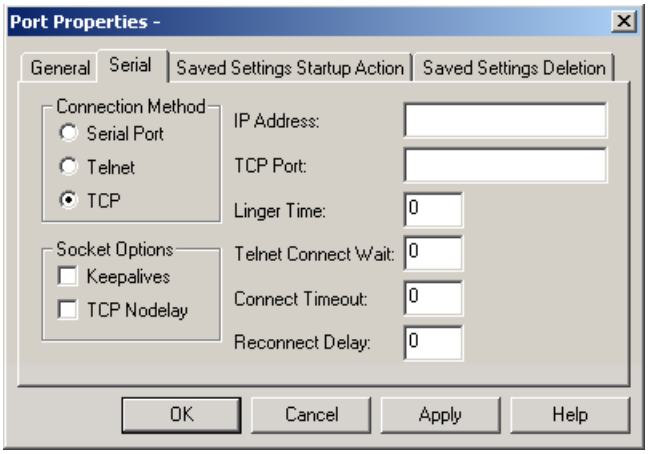


Baud Rate	Enter the baud rate for communications. Click the drop-down list to the right of the input field and select a rate from the list.
Parity	Select the parity to be used for communications.
Data Bits	Select the number of data bits per word to be used for communications.
Stop Bits	Select the number of stop bits to be used for communications
Flow Control	Select the type of flow control to be used for communications. If you change the Flow Control type, you must reboot the PC for the changes to take affect.

Remember that you must configure the same baud rate, data bits, parity, stop bits and flow control for all PLCs using the serial port.

Serial Port Properties: Telnet or TCP Connection

If you check the Telnet or TCP connection method, define the following.



Socket Options	Keepalives	Check to use keepalives to detect the loss of the terminal server.
	TCP Nodelay	Check if you want to set the Nodelay flag on the socket.
IP Address	Enter the IP address of the terminal server.	
TCP Port	Enter the port number of the TCP port on the terminal server.	
Linger Time	Enter the time in seconds to wait after closing the socket before aborting the socket.	
Telnet Connect Wait	Enter the time in seconds to wait for the Telnet protocol to initialize.	
Connect Timeout	Enter the time in seconds to wait for the TCP connection to form.	

Reconnect Delay	Enter the time in seconds to wait before attempting to reconnect to a device. If you set this value to zero and the terminal server is not available, then no attempts will be made to reconnect to the terminal server.

Saved Settings Startup Action

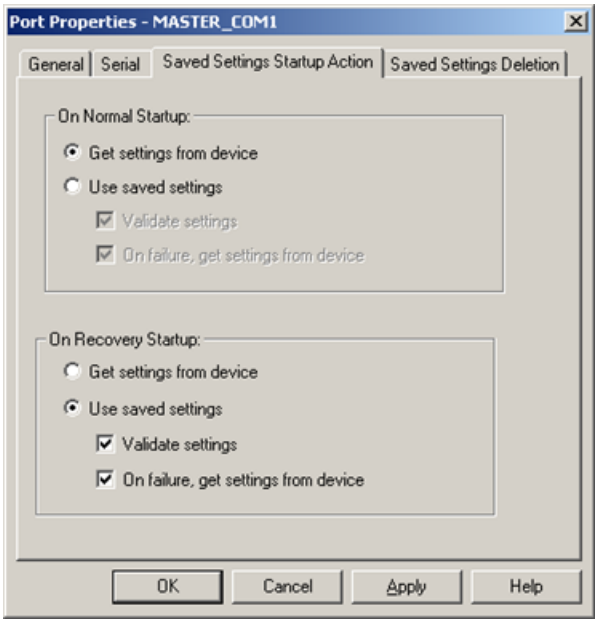
Modbus RTU Communications provides you with the option to reduce normal and/or recovery start up time by saving device characteristics for subsequent re-use.

Note: Saved Settings Startup Action is intended for devices whose characteristics, such as the memory size, do not change. If the settings change, make sure they are deleted *(page 89)*. Saved settings startup is supported only for devices where the number of coils and inputs are evenly divisible by 8.

Select the Saved Settings Startup Action tab in the Modbus RTU Communications Port Properties dialog box.

Selections can be made for the following.

- Normal startup.
- On Recovery Startup



Normal Startup

- Normal Startup occurs when CIMPLICITY starts.
- CIMPLICITY is started either when:
- The computer is booted up or

- A CIMPLICITY user starts the project.

Options for each of the startups are as follows.

Get settings from device

Defines the actions that the device communication interface takes to determine the supported memory types and ranges for a specific device.

Use saved settings

Checked: The device communication interface will use the stored settings to define the device-specific memory types and ranges.

- The device configuration data is recorded and stored for later use.
- Options if Use saved settings is checked are:

Option	Description
Validate settings	Checked
	Clear
On failure, get settings from device	When using saved settings, failures may occur. Two typical failures are: • An integrity error is detected in the saved information. • There is a failure to verify a memory range when Validate settings is selected.
	Checked
	Clear

On Recovery Startup

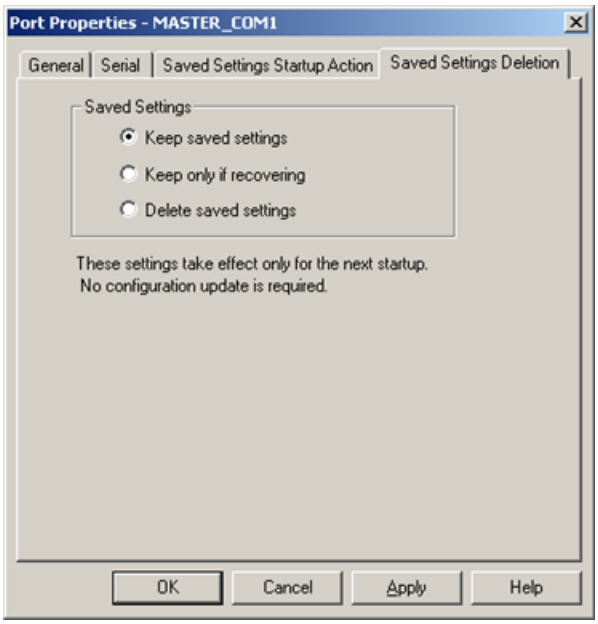
Recovery startup occurs after:

- A cluster fails.
- Process health kills a project as a result of a process failure.

Options *(page 626)* for On Recovery Startup are the same as they are for On Normal Startup.

Saved Settings Deletion

Check one of the following to specify what Modbus RTU Communications should do with saved settings.



Option	Description
Keep saved settings	Do not automatically delete the saved settings on the next startup
Keep only if recovering	Delete the current saved settings unless the next startup is in recovery mode.
Delete saved settings	Deletes: • The current saved settings at the next startup. • Settings for all the devices that are configured for the port. **Note:** If the configuration of the device is changed, make sure to check and apply Delete saved settings.

Note: These settings affect the next startup only. The Saved Settings Deletion tab will always default to the Keep saved settings option for subsequent startups.

Terminal Server Setup

1. Log into your terminal server, and set privileged mode by issuing the following command. Enter the privileged password when prompted (typically "system"):

   > set priv

   Password>

2. Enter the following command so that the permanent database and the operational database will be updated when a **define** command is issued:

>> **set server change enabled**

3. For each port to which a marquee is attached, enter the following set of commands where:

   <port> is the actual port number to which the marquee is attached. Note that it is possible to specify a range of port numbers instead of a single port (for example, 1-10 signifies ports 1 through 10).

   <ip_port> is the telnet port number.

   <parity> is odd, even, or none.

   <stop bits> is the number of stop bits.

   <speed> is the baud rate of the marquee.

```
>> define port <port> access remote
>> define port <port> telnet remote <ip_port>
>> define port <port> telnet transmit immediate
>> define port <port> parity <parity>
>> define port <port> character size 8
>> define port <port> stop bits <stop bits>
>> define port <port> autobaud disabled
>> define port <port> speed <speed>
>> define port <port> line editor disabled
>> define port <port> telnet csi escape enabled
>> define port <port> telnet binary session mode passall
>> define port <port> loss notification disabled
>> define port <port> autoprompt disabled
>> define port <port> verification disabled
>> define port <port> outboundsecurity disabled
>> define port <port> broadcast disabled
>> define port <port> default session mode transparent
>> define port <port> internet tcp keepalive 2
```

Modbus RTU Device Configuration

Modbus RTU Device Configuration

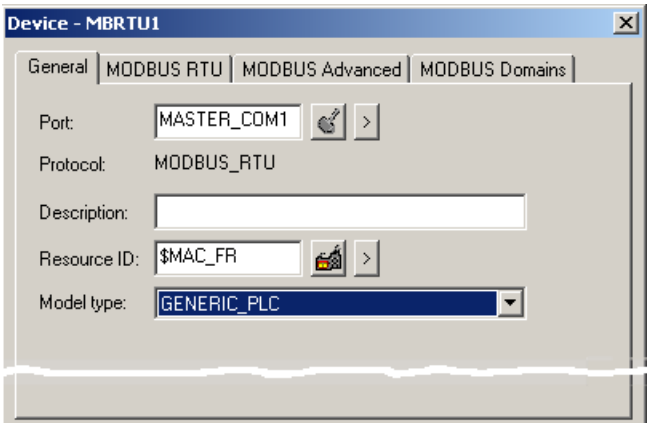
1. In the **Device** field, enter the name of the device you are configuring.
2. In the **Port** field, select the Modbus RTU port to be used by the device.

   When you click **OK** to create the device, the Device dialog box opens.

Important: The following special characters cannot be used for a Modbus RTU device name: \, /, :, *, ?, ", <. >, |, [,]. If the device name contains any of these characters, the device settings will not be preserved and saved startup will not be possible.

General Device Properties

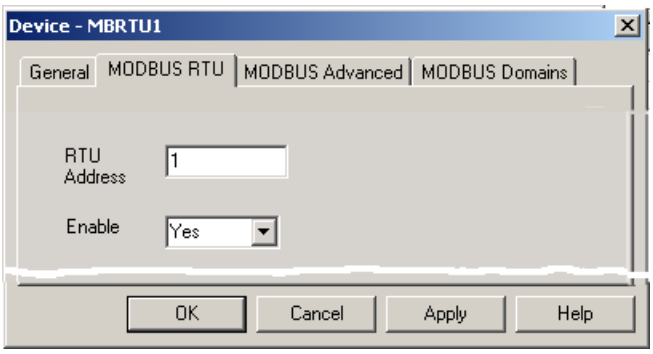
Use the General tab in the Device dialog box to enter general information for the device. You can define the following:



Port	Select the port for this device.
	
	
Description	(Optional) Enter a description to help you identify the device.
Resource	Enter a resource that can be associated with this device for alarm generation.
	
	
Model Type	Choices in the drop-down list for this protocol are:
	If your device model is not displayed in the selection list, select either:
	STAT_PLC
	DYN_PLC
	For the 3720 ACM Electronic Power Meter use
	3270ACM(32BITMODE)
	STAT_PLC or DYN_PLC
	GENERIC_PLC

MODBUS RTU Device Properties

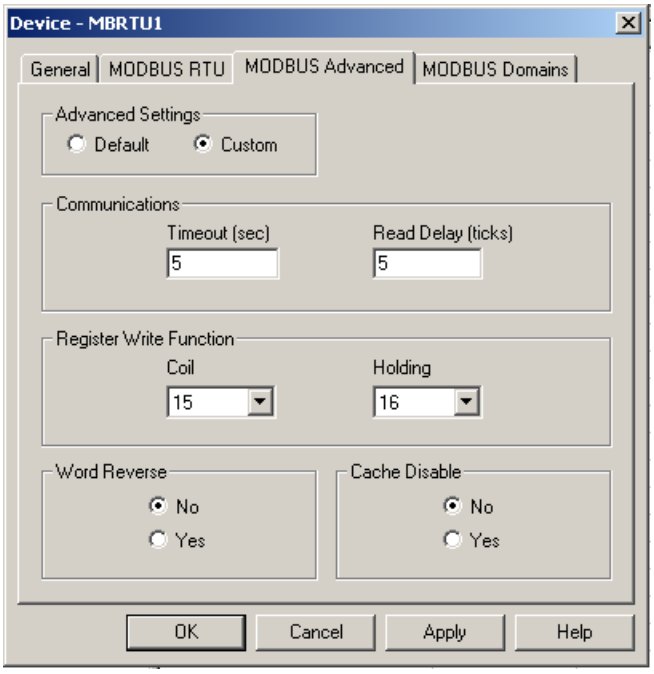
Use the MODBUS RTU tab in the Device dialog box to enter information about Modbus RTU communications for the device. You can define the following:



RTU Address	Enter the server address of the device. For the Series Six, this is the **CPU ID** of the attached Series Six. Valid server addresses range from 1 to 247 decimal. Broadcast mode (server address 0) is not supported by the Modbus RTU device communications enabler.
Enable	Select YES to enable the device when the project starts. If you select NO, the device will not be enabled and points associated with the device will be unavailable.

Modbus RTU Advanced Properties

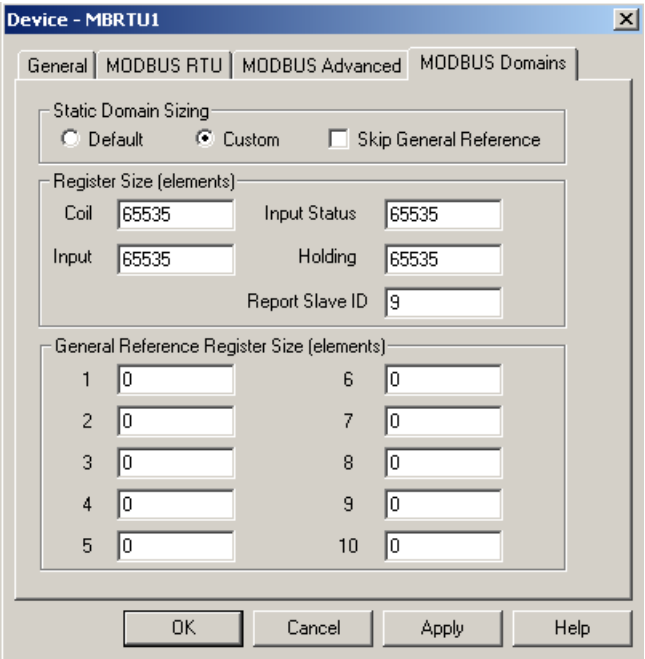
Fields in the Advanced dialog box are as follows:



Advanced Settings	Default	Click to update all fields with their default settings.
	Custom	Click to define custom values for fields on this tab.
Timeout (sec)		Number of seconds the device communication will wait for a response to a poll request before declaring a read failure.
Read Delay (ticks)		The amount of time to wait for a response after issuing a request to the device.
	Domain Type	Write Function
Register Write Function	Coil	5
		15
	Holding	6
		16
Word Reverse		This affects four byte values only (REAL, DINT, UDINT). Some devices store the LOW WORD and HIGH WORD in reverse order (from a Modicon PLC). This will reverse them so they are displayed correctly in CIMPLICITY.
Cache Disable		By default, CIMPLICITY caches continuous addresses into a cache group. It then performs a single read on the group. Some devices will not allow this kind of group read. When caching is disabled, each point is read individually from the device. Note that in projects with large point counts, this can significantly affect performance.
OK		Saves the values in the file **\\site_root:data\master_ port .ini**, which is then read by the device communication interface the next time it is started.
Cancel		Closes the device dialog box without any modification.

Modbus RTU Domains Properties

Fields in the Domains dialog box are as follows:



Domain Sizing	Select one of the following.
	Default
	Custom
Skip General Reference	Check this option to deactivate General Reference elements for this device.
	Domain Type
Register Size	Coil
(elements)	Input
	Input Status
	Holding
	Report Server ID
General Reference Register Size (elements)	(1 -10)

Modbus RTU Point Configuration

Modbus RTU Point Configuration

Once your devices are configured, you may configure points for them. Through device point configuration, you may configure the following:

- Points that read or set Coils

- Points that read Discrete Inputs
- Points that read Input Registers
- Points that read or set Holding Registers
- Points that read or set General Reference
- Points that read or set Holding Registers as Double Precision

Fields in the Point Properties dialog box have configuration values that are unique to, or have special meaning for Modbus RTU communications.

General Point Properties

On the General tab in the Point Properties dialog box, the type of **Access** you choose depends on the point address:

- Select **Read** for points from Discrete Inputs or Input Registers.
- Select **Read/Write** for points from Coils or Holding Registers.

Device Point Properties

On the Device tab in the Point Properties dialog box, the following fields have special meaning for the Modbus RTU Communications option:

Update Criteria	The update criteria you select determines how the data will be requested.
	Select **On Scan** or **On Change** for points whose values should be polled by Modbus RTU communications at regular intervals. Select **On Demand On Scan** or **On Demand OnChange** for points whose values should be polled by Modbus RTU communications at regular intervals while users are viewing them.
Address	Enter the point address of the data as follows: **0xxxx** or **0xxxxx** for Coils **1xxxx** or **1xxxxx** for Discrete Inputs **3xxxx** or **3xxxxx** for Input Registers **4xxxx** or **4xxxxx** for Holding Registers **6yxxxx** or **6yxxxxx** for General Reference **D4xxxx** or **D4xxxxx** for Holding Registers as Double Precision. For the 3270 ACM meter, this address represents a register whose value is expressed in mod 10000 format.
	where **xxxx** or **xxxxx** is the bit number or register number of the point. For example, the address of Holding Register one is 40001 and the address of Coil one is 00001.
	If you are configuring a General Reference point, the file number is **y**.
	Double Precision points in Holding Registers require two Holding Registers per point. The upper register is multiplied by 10000 and the product is added to the lower register to get the resulting value. No more than one Double Precision point may be written in a single write operation. Use **DINT** or **UDINT** types for these points.
	If you are configuring an Analog point type in Discrete Inputs or Coils, the address entered in this field must be that of the least significant bit of the point. For example, to configure a 16-bit analog point for Coils 17 through 32, enter 00017 in this field.

When you configure Boolean points in Holding Registers, Input Registers, or General Reference, you must also enter data in the following field:

Bit Offset	Enter the bit offset in the Holding Register, Input Register, or General Reference that defines the point, where 0 is the least significant bit and 15 is the most significant bit.

Modbus RTU Advanced Configuration

Modbus RTU Advanced Configuration

- MB_COMM_TIMEOUT
- MB_LOG_PROTOCOL
- MDBC
- <PORT>_LOG_PROTOCOL

MB_COMM_TIMEOUT

Purpose	To change the time out period for a read request. To reduce the time it takes for a read request to time out, and shorter time outs reduce the interruption in polling other devices which are up.
Value	Integer in seconds
Default Value	5
Comment	MB_COMM_TIMEOUT works with a generic device. Minimum retry time out is approximately 12-14 seconds.

MB_LOG_PROTOCOL

Purpose	To log the protocol level exchanges between the device communication interface and Modbus PLCs on all Modbus RTU ports in the project.
Value	Values to enable/disable the feature are as follows: Y, N, or Default.

MDBC

This global parameter applies to all devices configured on ports using the Modbus RTU protocol.

MDBC

Purpose	To disable caching for all devices in the project configured to use the Modbus RTU protocol.
Value	Y
Default Value	Y

<PORT>_LOG_PROTOCOL

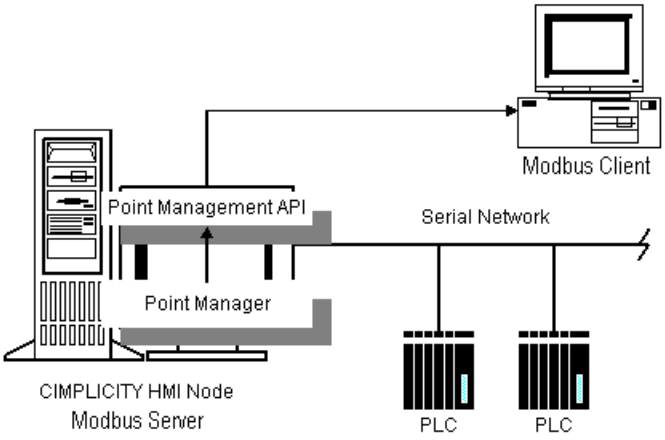
Purpose	To log the protocol level exchanges between the device communication interface and Modbus PLCs on a specific Modbus RTU port.
Value	Values to enable/disable the feature are as follows.
	Y
	N
	Default

Chapter 41. Modbus RTU Server Communications

About Modbus RTU Server Communications

The Modbus RTU Server Communications option supports a multi-drop configuration with the Modbus RTU Server Communications enabler functioning as a server. In effect, the Modbus RTU Server Communications enabler presents the CIMPLICITY Points as if they were data values from one or more Modbus RTU Server devices.

The CIMPLICITY Points are mapped to Modbus Addresses, allowing a Modbus Client to obtain Point Values as if they were coils or registers on a server device.



SystemLink uses the Modbus RTU Server feature to enable CIMPLICITY to collect point values over the Internet using the Modbus protocol. The node that is set up as the Modbus Client can collect data from the Modbus Server nodes via the Internet.

Modbus RTU Server Specifications

Modbus RTU Server Specifications

Specifications for Modbus RTU Server include:

- CIMPLICITY specifications for Modbus RTU Server
- Modbus RTU Server protocol.

- Cabling requirements.

CIMPLICITY Specifications for Modbus RTU Server

Supported Data Types

This communications option supports the following data types:

- Boolean
- Signed 8, 16, and 32 bit integers
- Unsigned 8, 16, and 32 bit integers
- Floating point
- Arrays of the above data types

Modbus RTU Server Protocol

Modbus RTU Server Protocol

See protocols that Modbus RTU Server supports for:

- Transmission mode.
- Message framing.
- Server device emulation.
- Memory types.
- Function codes.

Transmission Mode Supported by Modbus RTU Server

The Modbus RTU Server Communications option supports the RTU transmission mode.

Message Framing Supported by Modbus RTU Server

The Modbus RTU Server Communications option supports RTU framing. RTU framing implies the following communication parameters:

- Data bits: 8
- Start bits: 1
- Parity: Even, Odd, or None
- Stop bits: 1 or 2
- Error checking method: CRC

Modbus RTU Server Device Emulation

The Modbus RTU Server Communications option emulates one or more generic Modicon programmable controllers. Valid server addresses: 1 to 127.

Memory Types Supported by Modbus RTU Server

The Modbus RTU Server Communications option emulates the Modbus memory types based on Modicon programmable controllers.

The Modbus RTU Communications option supports writes to the following memory types:

Memory Type	Reference Class	Description
Coils	**0X**	Discrete Outputs
Holding Registers	**4X**	Analog Outputs

The Modbus RTU Server Communications option supports reads from the following memory types:

Memory Type	Reference Class	Description
Coils	**0X**	Discrete Outputs
Inputs	**1X**	Discrete Inputs
Input Registers	**3X**	Analog Inputs
Holding Registers	**4X**	Analog Outputs

Function Codes Supported by Modbus RTU Server

The Modbus RTU Server Communications option supports the Modbus function codes listed in the following table. As well, the maximum number of coils or registers that may be specified or included in a single Modbus message is also given.

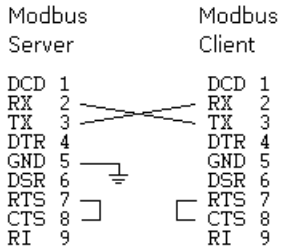
Function Code	Description	Maximum # of coils / registers
1	Read Coil Status	2000
2	Read Input Status	2000
3	Read Holding Registers	125
4	Read Input Registers	125
5	Force Single Coil	1
6	Force Single Register	1
15 (0x0F)	Force Multiple Coils	800
16 (0x10)	Preset Multiple Registers	100

Modbus RTU Server Cabling Requirements

RS-232 Interface Cable

Note: the Modbus RTU Server Interface requires RTS/CTS hardware flow control.

The cabling diagram below illustrates the interface for a serial board DB-9 connector to DB-9 connector.



RS-422 Interface Cable

Refer to documentation provided with the RS-422 serial board.

Modbus RTU Server Required Documents

The Modbus RTU Server Communications enabler adheres to the Modbus protocol specification as defined in the following document:

Modicon Modbus Protocol Reference Guide (PI-MBUS-300), Rev. J, June 1996

Modbus RTU Server Getting Started Steps

Modbus RTU Server Getting Started Steps

The CIMPLICITY Modbus RTU Server Interface provides a way to access CIMPLICITY points as Modbus coils and/or registers from Modbus Client devices. This section guides a user through the steps to become familiar with the Modbus RTU Server Interface application and to start using the application quickly.

The following getting started guide assumes that the Modbus RTU Server Interface has been successfully installed and that at least one CIMPLICITY project has been created.

Steps include:

Step 1 *(page 641)*	Identify CIMPLICITY Points.
Step 2 *(page 641)*	Map points to Modbus data addresses.
Step 3 *(page 642)*	Select the Modbus RTU Server option.
Step 4 *(page 642)*	Start the CIMPLICITY project.
Step 5 *(page 642)*	Start the Modbus Client.
Step 6 *(page 642)*	Validate communications.

Step 1. Identify CIMPLICITY Points

Identify the CIMPLICITY points you wish to access from a Modbus Client device. If necessary, create device or virtual points to receive values from a Modbus Client device. Create virtual points to re-range analog point values to prevent data coercion errors, or to minimize loss of significance. For example, real values are truncated when transferred to a Modbus analog data address.

Step 2. Map Points to Modbus Data Addresses

Map (link) CIMPLICITY points to Modbus data addresses by creating Register Mapping Records *(page 645)* definitions within the configuration file.

1. Map:
   - Boolean points to coil (0X) or input status (1X) data addresses.
   - Analog points to input register (3X) or holding register (4X) data addresses.

   Avoid gaps *(page 647)* in allocated data addresses.

2. Edit the configuration file to specify the:
   - Communications port parameters,
   - Trace log parameters, and
   - Modbus register limits.

Step 3. Select the Modbus RTU Server Option

Review the procedure to add the Modbus RTU Server interface to your CIMPLICITY project.

Step 4. Start the CIMPLICITY Project

1. Load the project from the CIMPLICITY Workbench application.
2. Either:
3. Select Run from the Workbench Project pull-down menu, or
4. Click the **Run** button on the Workbench toolbar.
5. Check the list of started resident processes for the name **MRTUSI_RP**.

Step 5. Start the Modbus Client

At this point, the Modbus RTU Server Interface is ready to process Modbus Client requests. Start the Modbus Client device according to the documentation provided with the device.

Step 6. Validate Communications

CIMPLICITY provides a number of diagnostic tools to validate communications between CIMPLICITY and the Modbus Client. For example, use the CIMPLICITY Point Control Panel to display CIMPLICITY points.

Modbus RTU Server Configuration Overview Steps

Modbus RTU Server Configuration Overview Steps

This section describes the configuration process for the Modbus RTU Server Interface. To configure the Modbus RTU Server Interface, follow these steps:

Step 1 *(page 643)*	Edit the configuration file.
Step 2 *(page 647)*	Select the Modbus RTU Server protocol.

Step 3 *(page 648)*	Implement the Modbus RTU Server configuration.

Step 1. Edit the Modbus RTU Server Configuration File

Step 1. Edit the Modbus RTU Server Configuration File

When a new CIMPLICITY project is created, a template configuration file is created for the Modbus RTU Server Interface. The file has a comma-separated value (CSV) format similar to the format used by the CIMPLICITY import/export utility.

Modify the configuration file to specify the:

- Communications port parameters.
- Trace log parameters.
- Modbus register limits
- Point / register mappings.

Information about:

- The file naming convention.
- Types of Modbus RTU Server configuration records.

File Naming Convention

The Modbus RTU Server Interface determines the file specification of the configuration file at startup by combining the file name **MRTUSI_RP.CFG** with the file path of the Project directory.

Example

If the project has been created in the directory

C:\Program Files\Proficy\Proficy CIMPLICITY\Projects\test

Then the configuration file specification is:

C:\Program Files\Proficy\Proficy CIMPLICITY\Projects\test\data\MRTUSI_RP.CFG .

Types of Modbus RTU Server Configuration Records

Types of Modbus RTU Server Configuration Records

The configuration file may contain the following types of records:

	IP address Parameters	Defines the serial communications port and settings.
	Trace Log Parameters	Specifies state (on/off) of protocol stack diagnostics logging.
	Modbus Register Limits	Defines the range of valid values for a Modbus analog register (memory type 3X or 4X).
	Register Mapping	Defines a mapping between a CIMPLICITY point and a Modbus data address.
	Comments	Any line beginning with one or more pound signs (#). Ignored by the Modbus RTU Server Interface.
	Blank lines	Ignored by the Modbus RTU Server Interface.

Communications Port Parameters Record

The configuration file must contain a Communications Port Parameters record. In addition, all fields must be present.

The format for the Communications Port Parameters record is:

CP,< port>**,**<baud rate>**,**<parity>**,**<data bits>**,**<stop bits>

Where

CP	Identifies the record as a communications port parameters record.
<port**>**	Physical serial port selection. Valid settings are 1 to 9.
<baud rate**>**	Throughput rate. Valid settings are hardware dependent, and may range from 1200 to 115000.
<parity**>**	Parity setting. Valid settings: None, Odd, Even.
<data bits**>**	Size of data field (in bits). Valid settings: 8.
<stop bits**>**	Number of stop bits. Valid settings: 1, 2.

Trace Log Parameters Record

The configuration file may contain a Trace Log Parameters record.

The format for the Trace Log Parameters record is:

TF ,<logging state>

Where

TF	Identifies the record as a trace log parameters record.
<logging state**>**	Protocol stack logging state. Valid settings: **On**, **Off**. Default value: **Off**.

Note: If the Trace Log Parameters record is not defined, the protocol stack logging state is **Off**.

Trace log messages are written to the standard error file associated with the Modbus RTU Server Interface process. .

Modbus Register Limits Record

The configuration file may contain a Modbus Register Limits record. These limits are applied only to CIMPLICITY points mapped to analog registers (register type 3X or 4X).

The format for the Modbus Register Limits record is:

RL ,<low limit>**,**<high limit>

Where

RL	Identifies the record as a Modbus register limits record.
<low limit**>**	The minimum value accepted from a CIMPLICITY point. Valid range of values: **0 to 65535**. Must be less than **<**high limit**>**. Default value: **0**.
<high limit**>**	The maximum value accepted from a CIMPLICITY point. Valid range of values: **0** to **65535**. Must be greater than <low limit>. Default value: **65535**.

Note: If the Modbus Register Limits record is not defined the low and high limits are 0 and 65535, respectively.

Register Mapping Records

The configuration file must contain one or more Register Mapping records. In each Register Mapping record, all fields must be present.

The format for the Register Mapping record is:

RM ,<server>**,**<register>**,**<point id>

Where

RM	Identifies the record as a register mapping record.
<server**>**	The emulated server device address. Valid range of values: 1 to 127.
<register**>**	One-based Modbus address including the memory type prefix. See the following table for valid range of values:

	Memory Type	Address range
	0x - Coils (Discrete)	**1** to **9999**
	1x - Inputs (Discrete)	**10001** to **19999**

	3x - Input Registers (Analog)	**30001** to **39999**
	4x - Holding Registers (Analog)	**40001** to **49999**

<point id**>**	String that uniquely references a point in a CIMPLICITY project. Must specify a non text attribute (i.e. analog or Boolean).

Sample Configuration File

The following is a sample configuration file specifying register mappings for two server devices (address 3 and 7).

```
# Modbus RTU Server Interface - sample configuration file
#
# Communication port definition:
CP,1,19200,None,8,2
# Trace log flag definition:
TF,Off
# Modbus register limits definition:
RL,0,65535
# Register mapping definitions:
# Server Address: 3
#
# - the following register mapping definitions specify coils for
# Server address 3
#
RM,3,10,TEST_POINT_DOUT_1
RM,3,11,TEST_POINT_DOUT_2
RM,3,12,TEST_POINT_DOUT_3
RM,3,13,TEST_POINT_DOUT_4
#
# - the following register mapping definitions allocate a block of holding
#   registers (memory type 4X). Note that the highest register reference is
#   40104 or data address 103 (zero-based). Thus, while only 4 registers
 are
#   mapped to CIMPLICITY points, all 104 registers are accessible to the
#   Modbus Client.
#
RM,3,40101,TEST_POINT_AOUT_2
RM,3,40102,TEST_POINT_AOUT_3
RM,3,40103,TEST_POINT_AOUT_4
RM,3,40104,TEST_POINT_AOUT_5
# Server Address: 7
#
# - the following register mapping definitions allocate the first two data
#   addresses of each memory type for Server address 7
#
RM,7,1,TEST_POINT_DOUT_A
RM,7,2,TEST_POINT_DOUT_B
RM,7,10001,TEST_POINT_DIN_A
```

```
RM,7,10002,TEST_POINT_DIN_B
RM,7,30001,TEST_POINT_AIN_A
RM,7,30002,TEST_POINT_AIN_A
RM,7,40001,TEST_POINT_AOUT_A
RM,7,40002,TEST_POINT_AOUT_B
```

Register Mappings Guidelines

This sectionprovides the user with guidelines for defining register mappings.

Guideline to Organize Records

The register mappings (and other types of records) may be specified in any order. To simplify maintenance of the configuration file, order the register mapping records by server address then register reference. This will aid in detecting unmapped data addresses.

Since the Modbus RTU Server Interface can emulate multiple Modbus RTU Server devices, one could organize mapped CIMPLICITY points into logical groupings based on a functional area.

Guideline to Optimize Throughput

The Modbus RTU Server Interface supports several Modbus requests that allow a range of data addresses (or an array of values) to be specified. Note that these requests specify a contiguous set of data addresses.

To maximize throughput, avoid gaps in allocated data addresses. This will reduce the transfer of "meaningless" values.

In addition, map Boolean points to coil (0X) or input status (1X) data addresses. Modbus requests that handle coil or input status data addresses pack sixteen (16) values into one word (two bytes). Modbus requests that handle input register or holding register data addresses require one word per value.

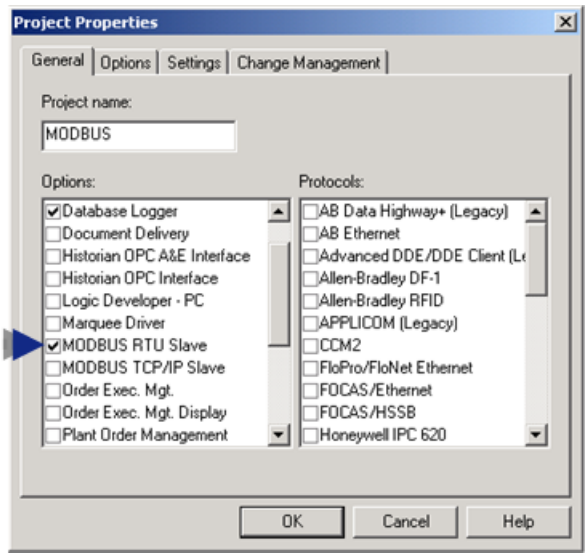
Another way to maximize throughput is to allow the Modbus Client to reduce the number of transactions by increasing the range of data addresses in each request.

Step 2. Select the Modbus RTU Server Protocol

1. Load an existing project (or create a new project) in the CIMPLICITY Workbench.
2. Click Project on the Workbench menu bar.
3. Select Properties.

The Project Properties dialog box opens.

4. Select the General tab.



5. Check Modbus RTU Server in the Options box.

6. Click OK to save the changes.

Step 3. Implement the Modbus RTU Server Configuration

Step 3. Implement the Modbus RTU Server Configuration

To apply Modbus RTU Server configuration changes, start the associated CIMPLICITY project. If the project is currently running, stop then restart the Modbus RTU Server resident process.

The start order for implementing the Modbus RTU Server configuration is:

- Start the associated CIMPLICITY project.

Note: Check the list of started resident processes for the name **MRTUSI_RP**.

Start the Associated CIMPLICITY Project

You need to start the associated CIMPLICITY project in order to start the Modbus RTU Server Interface.

todo: To start the associated CIMPLICITY project:

1. Open the associated project in the CIMPLICITY Workbench.
2. Either:

   Method 1

   a. Click Project on the Workbench menu bar.
   b. Select Run.

   Method 2

   Click the **Run** button on the Workbench toolbar.

3. Check the list of started resident processes for the name **MRTUSI_RP**.

Modbus RTU Server Troubleshooting

Modbus RTU Server Troubleshooting

There are several diagnostic tools available to the user. These tools are particularly valuable if runtime behavior deviates from expected behavior. They can help isolate and troubleshoot the erroneous or erratic behavior of a working system.

Diagnostic tools include the following:

- CIMPLICITY status log.
- Protocol stack trace log
- Modbus exception codes.
- Test program.

Expected Modbus RTU Server Runtime Behavior

Expected Modbus RTU Server Runtime Behavior

This section describes the expected behavior of the Modbus RTU Server Interface. This description provides the user with a reference point for using the diagnostic tools.

Expected behavior for:

- Data address space.
- Data types.
- Data flow.

- Data coercion.
- Write response time.
- Dynamic CIMPLICITY configuration changes.

Data Address Space

The Modbus RTU Server Interface maintains a data address space that mimics one or more PLCs and their associated coils and/or registers.

The user configuration will implicitly define the limits of valid data addresses for each unique combination of memory type and server address. The minimum data address for any given memory type is 0 (zero-based). The maximum data address for a given memory type is the highest data address specified in the configuration. For example, if the highest holding register in server address 1 is 40150 (one-based), then the maximum data address for memory type 4X is 149 (zero-based).

guide: Guidelines. All data addresses in the address space are set to binary **0** on startup. If a data address is not mapped to a CIMPLICITY point, then the following rules apply:

1. If the data address is a read-write address (memory type 0X or 4X), then a Modbus Client may update (or read) the data address value. Thus, the data address may act as a memory store.
2. If the data address is a read-only address (memory type 1X or 3X), then a Modbus Client may only read the data address. Thus, the data address value is always binary zero (0).

   Modbus Client data address references outside the configured range (i.e. greater than the highest user-specified data address for a given memory type and server address) shall generate an exception response *(page 656)*.

Data Types

The Modbus RTU Server Interface supports mapping of all non-text (i.e. analog or Boolean) CIMPLICITY points to Modbus data addresses. Non-array points are mapped to a single data address as specified in the register mapping definition. Array points and bitstring points are mapped to multiple consecutive data addresses, with the first element mapped to the register specified in the register mapping definition.

Data Flow

To maintain the current value of each mapped CIMPLICITY point within the data address space, the Modbus RTU Server Interface will accept unsolicited updates (from CIMPLICITY) for all mapped points. Thus, the Modbus Client may read values from CIMPLICITY points mapped to any supported memory type (0X, 1X, 3X, and 4X).

However, the Modbus RTU Server Interface will only attempt to write to CIMPLICITY points that are mapped to data addresses of memory type 0X or 4X. Thus, the Modbus Client may write values to CIMPLICITY points mapped to data addresses of memory type 0X or 4X.

Note that for a Modbus Client write request to succeed, all data addresses specified must be valid and all corresponding values must be written successfully to PTMAP.

Important: When a source point that is mapped to a register number (e.g. 40001) in the mrtusi_rp.cfg file becomes unavailable, the Modbus RTU Server will continue to send the last valid value to a Modbus Client that requests a poll for this point. For example: If the source point (40001) has a value of 12, and a Modbus Client polls register 40001, it will receive a value of 12. If the source point then becomes unavailable, any subsequent polls reading register 40001 will receive a value of 12 until the source point becomes available again and the value is updated.

Data Coercion

The Modbus RTU Server Interface performs runtime coercion for all data types.

Discrete data types are coerced according to the following guidelines.

Memory Type	Data Source	Source Value	Target Value
0X	CIMPLICITY	**zero (0)**	**0**
		CIMPLICITY	**non-zero (!= 0)**
		Modbus Client	**0**
		Modbus Client	**0xFF or 0x01 (depends on function code)**
1X	CIMPLICITY	**zero (0)**	**0**
		CIMPLICITY	**non-zero (!= 0)**

guide: Guidelines. Analog data types are coerced according to the following guidelines.

1. For analog points with engineering units, the Modbus RTU Server Interface will use the engineering unit value when converting to/from Modbus data values.
2. For analog points without engineering units, the Modbus RTU Server Interface will transfer the raw value.
3. CIMPLICITY point values will be coerced to two byte unsigned integer values. Thus, the fractional portion of real values will be lost.
4. The Modbus RTU Server Interface will reject any CIMPLICITY point value outside of the range *(page 645)* defined by the Modbus Register Limits. .

Write Response Time

For a given Modbus Client write request, the Modbus RTU Server Interface will wait for all set point requests to complete before returning a response to the Modbus Client. Since the set point requests

are non-deterministic, no response time can be guaranteed for a write request. The Modbus Client should be prepared to adjust its timeout parameter accordingly.

Dynamic CIMPLICITY Configuration Changes

CIMPLICITY supports reconfiguration of point properties while the project is running. The point properties that directly impact the behavior of the Modbus RTU Server Interface include the following:

- Access rights
- Data type
- Array size

The Modbus RTU Server Interface monitors the project database for changes in point properties. Once notified, point property changes are applied immediately.

Example

Changing the access rights for a point from read-write to read-only will cause Modbus write requests (that specify the corresponding data address) to fail.

In particular, changes to the array size of a point may have unexpected results. At startup, the Modbus RTU Server Interface determines the array size for every mapped point. By definition, a scalar point has an array size of 1. In combination with other tests, the Modbus RTU Server Interface uses the array size to ensure that every array element is mapped to a unique Modbus data address. The array size determined at startup becomes the maximum size allowed. Thus, while the array size may be reduced, the Modbus RTU Server Interface ignores increases.

CIMPLICITY Status Log

CIMPLICITY Status Log

If an error is detected, the Modbus RTU Server interface logs:

- Error messages on startup.
- Informational warning and error messages at runtime.

Modbus RTU Server Interface Startup Error Messages

On startup, the Modbus RTU Server Interface *(page 642)* validates the contents of the configuration file. If an error is detected, an error message is logged. These error messages include the following:

Message	Cause / Resolution

Error opening configuration file (see next message).	The configuration file could not be opened. The following message provides more details. Ensure that a valid configuration file *(page 643)* exists in the required directory.. Ensure that another application is not currently using the file.
Error reading from configuration file.	The configuration file could not be read. Check the file for invalid control characters. If necessary, replace the file with a backup copy, or rebuild the file using the provided template.
Configuration file contains more than one communication parameters record.	Ensure that the configuration file contains only one record with the prefix "CP".
Configuration file does not contain any communication parameters record.	The configuration file must contain a record with the prefix "CP".
Configuration file contains more than one trace log parameters record.	Ensure that the configuration file contains only one record with the prefix "TF".
Configuration file contains more than one Modbus register limits record.	Ensure that the configuration file contains only one record with the prefix "RL".
Configuration file does not contain any register mapping record.	The configuration file must contain at least one register mapping record (prefix "RM").
Configuration file contains the following invalid record (see next message).	Detected a record that is not a comment nor a blank line nor a valid definition. The following message contains the record. Ensure that the record starts with a valid record prefix or a valid comment character.
<record type> record is invalid.	For the given type of record, some or all of the required fields are missing, or are not properly delimited.
<record type> record contains an invalid <field name> field.	For the given type of record, the given field is not defined or contains invalid characters.

<record type> record specifies an invalid <field name>.	For the given type of record, the given field specifies an invalid parameter value (for example, the value is out of range).
Modbus register limits record specifies an invalid pair of limits.	The low register limit equals or exceeds the high register limit.
Duplicate register reference detected (see next two messages).	Detected two register mapping records that specify the same Modbus data address (server address and register reference). The next two messages give the register mapping parameters (and array size) of the current and duplicate records. If the two records do not specify the same Modbus data address, check the array size of the CIMPLICITY point specified in the Duplicate register mapping and ensure that sufficient data addresses are reserved.
Duplicate point identifier detected (see next two messages).	Detected two register-mapping records that specify the same CIMPLICITY point. The next two messages give the register mapping parameters (and array size) of the current and duplicate records.
Unsupported data type for point: <point id>.	The data type of the given CIMPLICITY point is not supported *(page 638)*. .

Modbus RTU Server Interface Runtime Error Messages

The Modbus RTU Server Interface logs informational, warning, and error messages to the CIMPLICITY project status log. These messages include the following:

Message	Cause / Resolution
Program startup completed successfully	Informational message - no action required.
Program shutdown completed successfully	Informational message - no action required.
Memory allocation error	Error creating an internal data structure. Increase available memory by stopping non-essential applications.
Error opening communication port channel: x.	Error occurred while connecting to the given COM port. Ensure that the given COM port is valid.
Error coercing Modbus value to CIMPLICITY point value (see next message).	The value of the given CIMPLICITY point cannot be converted to the mapped Modbus memory type. Ensure that the value is within the Modbus Register Limits *(page 645)*. Ensure that the data type is supported *(page 638)*
On-change response coercion error for point: <point id>.	(see the previous message)

Important: The Modbus RTU Server Interface also logs messages intended for use by GE Intelligent Platforms support personnel. If any of the following messages are logged, please contact your CIMPLICITY vendor for support.

- Error initializing protocol layer object: xxx.
- Error registering server addresses in protocol layer object: xxx
- Thread creation failed.
- Error occurred while waiting for thread to exit.
- Event object creation failed.
- Error occurred while setting event object.
- Failed to allocate an event flag from PTMAP subsystem.
- Error occurred while waiting for a response from the PIL worker thread.
- An invalid PTMAP response message was received. Response ignored.
- Invalid register count: xxx.
- Duplicate entry detected for server address xxx, register class xxx.

Protocol Stack Trace Log

The protocol stack trace log provides the user with an indication of the Modbus messages received and sent by the Modbus RTU Server Interface at the communications port level. Each logged message is time stamped.

The user may enable or disable trace logging *(page 644)* by defining a trace log parameters record within the Modbus RTU Server Interface configuration file.

When enabled, the Modbus RTU Server Interface writes protocol stack trace messages to the file **MRTUSI_RP.err** in the **log** sub-directory of the Project root directory.

Example

If the Project had been created in the directory **C:\Program Files\Proficy\Proficy CIMPLICITY\Projects\test**, then the trace message file specification is **C:\Program Files\Proficy \Proficy CIMPLICITY\Projects\test\log\MRTUSI_rp.err**.

The following is an excerpt from an actual trace log file.

```
2000-10-09 09:56:19.511 <- 0103 00000018 45c0
2000-10-09 09:56:19.611 -> 0101
 3000010002000300040005000600070008000900 0A000B000C000D000E000F00100011001200130014001500
 5EAF
2000-10-09 09:56:19.641 <- 0103 00180018 C5C7
2000-10-09 09:56:19.701 -> 0103
 300019001A001B001C001D001E001F00200021002200230024002500260027002800290 02A002B002C002E00
 8969
2000-10-09 09:56:19.741 <- 0103 00300018 45CF
```

```
2000-10-09 09:56:19.802 -> 0103
 300031003200330034003500360037003800390003A003B003C003D003E003F004000410042004300440045004
 036D
2000-10-09 09:56:19.842 <- 0103 00480018 C5D6
2000-10-09 09:56:19.922 -> 0103
 300049004A004B004C004D004E004F005000510052005300540055005600570058005900 5A005B005C005D00
 D84A
2000-10-09 09:56:20.012 <- 0103 00600004 4417
2000-10-09 09:56:20.102 -> 0103 080061006200630064 ED2C
2000-10-09 09:56:35.464 <- 8002 000000C8 678D
2000-10-09 09:56:35.554 -> 8002
 19555555555555555555555555555555555555555555555555555555 60E3
```

The format for a trace log message is:

<date> <time> <direction> <server address><function code> <data> <CRC>

Where:

<date> is \the date the message was received or sent.

<time> is the time the message was received or sent.

<direction> is an arrow indicating whether the message is a request or response message.

Request (left arrow): <-

Response (right arrow): ->

Note: the remaining fields are displayed as hex values (two characters per byte).

<server address> is the emulated server device address.

<function code> is one byte representing the requested operation. If an exception response is generated, the high bit is set.

<data> is one or more bytes containing function parameters, data values, and/or exception code.

<CRC> is trailing two bytes containing the cyclical redundancy check value.

Modbus Exception Codes

The user may diagnose errors by examining response messages returned by the Modbus RTU Server Interface to the Modbus Client. Certain conditions will cause the Modbus RTU Server Interface to return exception responses. This section documents the supported Modbus exception codes and their possible causes.

Exception Code	Cause

0x01 (Illegal Function)	Modbus request specifies an unsupported function code.
0x02 (Illegal Data Address)	Modbus request specifies an invalid data address, or specified a data address range that extends beyond the highest configured data address.
0x03 (Illegal Data Value)	Modbus write request specifies an analog data value that exceeds the data type range or the configured EU limits of the associated CIMPLICITY Point.
0x04 (Server Device Failure)	Modbus write request specifies a data address mapped to a disabled, unavailable, or read-only CIMPLICITY Point.

The format for a Modbus exception response message is:

<server address><function code><exception><CRC>

Where

<server address> is the emulated server device address.

<function code>is one byte representing the failed operation. Note that the high bit is set.

<exception> is one byte representing the exception code. See table above.

<CRC> is trailing two bytes containing the cyclical redundancy check value.

Modbus RTU Server Test Program

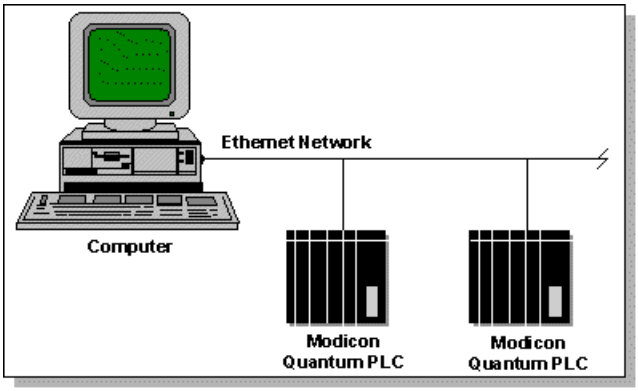
CIMPLICITY provides a sample Modbus Client test program. The test program generates Modbus requests and displays the received responses. To test communications with the Modbus RTU Server Interface, two serial ports are required - one port for the test program, and the other port for the Modbus RTU Server Interface.

For more information regarding the test program, refer to the CIMPLICITY Device Communications Manual (GFK1181G), Modbus RTU Communications, Validating Modbus RTU Communications.

Chapter 42. Modbus TCP/IP Communications

About Modbus TCP/IP Communications

The Modbus TCP/IP Communications option supports communication with Modbus Quantum programmable controllers using the TCP/IP protocol over an Ethernet network.



This enabler supports the following CIMPLICITY features:

- Polled reads at user defined rates
- Poll after setpoint
- Triggered reads
- Unsolicited data
- Asynchronous IO connection
- Analog deadband
- Alarm on communications failure
- Server Redundancy configurations

This enabler supports the following data types:

- Signed 8, 16, and 32 bit integers
- Unsigned 8, 16, and 32 bit integers
- Floating point
- Text
- Arrays

Modbus TCP/IP Required Documents

You should have the following document available when configuring devices for Modbus TCP/IP Communications:

- Modicon Quantum Ethernet TCP/IP Module User Guide (840 USE 107 00)

In addition, if you are planning to send unsolicited data from a Modbus Ethernet controller, you should have the following document available:

- Modicon Ladder Logic Block Library User Guide (840 USE 101 00)
- VersaMaxTM System Ethernet Network Interface Unit (GFK-1860)

Modbus TCP/IP Technical Notes

Refer to the technical notes below for reference information regarding Modbus TCP/IP Communications:

- Supported Modbus TCP/IP devices.
- Modbus TCP/IP unsolicited data.
- Modbus TCP/IP global parameters.

Supported Modbus TCP/IP Devices

Supported Modbus TCP/IP Devices

The CIMPLICITY Modbus TCP/IP communications software includes support for the following devices through standard configuration.

- Modicon 184/384
- Modicon 484
- Modicon 584
- Modicon 984
- VersaMax ENIU
- VersaPoint ENIU
- Quantum
- STAT_PLC (This is a special device model used to designate a generic device.)

Communication is also supported via the Modicon COBOX Modbus TCP to Modbus RTU Bridge (TSX Momentum 174-CEV-300-10) or Modicon Modbus Plus to Modbus Ethernet (TSX

Momentum 174-CEV-200 30) for Modicon 184/384, Modicon 484, Modicon 584, and Modicon 984.

Both solicited and unsolicited data collection is supported. We recommend limiting unsolicited data collection to one CIMPLICITY port to ensure reliable data collection.

Note: Unsolicited communication is received on TCP/IP socket port 502.

Functionality Differences Among Models

Not all Modicon programmable logic controllers behave/support the same function codes. The following table illustrates the differences:

Function Type	Function Code	184/384	484	584	984	Quantum	VersaPoint ENIU	VersaMax ENIU
Read Coil Status	1	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Read Input Status	2	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Read Holding Reg.	3	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Read Input Reg.	4	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Force Single Coil.	5	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Force Single Hold Reg.	6	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Force Multiple Coil Reg.	15	Yes	No	Yes	Yes	Yes	No	No
Force Multiple Holding Reg.	16	Yes	No	Yes	Yes	Yes	Yes	Yes
Program General Reg.	13	Yes	No	Yes	Yes	Yes	No	No
Poll General Reg.	14	Yes	No	Yes	Yes	Yes	No	No

Supported Modbus TCP/IP Memory Types

Supported Modbus TCP/IP Memory Types

Data may be read from the following memory types:

- Coils
- Discrete Inputs
- Input Registers
- Holding Registers
- General Reference

- Holding Registers as Double Precision

Data may be written to the following memory types:

- Coils
- Holding Registers
- General Reference
- Holding Registers as Double Precision

VersaMax ENIU Memory Map

%I1-%I2048	10001-12048	30001-30128	40001-40128
%AI1-%AI128		30129-30256	40129-40256
%Q1-%Q2048	00001-02048	30257-30384	40257-40384
%AQ1-%AQ128		30385-30512	40385-40512
FAULT TABLE			41025-41088

Writeable Memory:

Start Address	End Address
0001	02048
40257	40512
41025	41025

Note:

- Array writes for points mapped in the range of 00001-02048 are not supported.

The same I/O may be addressed via 40257-40384.

The first coil will correspond to 40257 bit 0.

- To clear the fault table, a value of 0 (zero) may be written to 41025.

Attempts to write any value other than 0 (zero) are not supported.

VersaPoint Memory Map

The VersaPoint memory map is as follows.

%I1-%I3072	10001-13072	30001-30192	40001-40192

%AI1-%AI192		30193-30384	40192-40384
%Q1-%Q3072	00001-03072	30385-30576	40385-40576
%AQ1-%AQ192		30577-30768	40577-40768
FAULT TABLE			41025-41088

Writable Memory:

Start Address	End Address
0001	03072
40385	40768
41025	41025

Note:

- Array writes for points mapped in the range of 00001-03072 are not supported.

The same I/O may be addressed via 40385-40768

- To clear the fault table, a value of **0** (zero) may be written to 41025.

Attempts to write any value other than **0** (zero) are not supported.

CAUTION:

If I/O is being controlled via CIMPLICITY , the following is very important:

- It is recommended that the I/O should be configured to HOLD LAST STATE as I/O values are only written when the application executes the write operation through a setpoint.
- If the VersaMax ENIU or VersaPoint ENIU looses power, the outputs values will revert to zero. It is essential that any application that is controlling I/O have a scheme for detecting that the outputs have reverted to zero from prior set values. If the I/O is not configured to HOLD LAST STATE, there are other conditions where the output values will revert to zero.
- In host redundant environments, please note the CIMPLICITY's default setting for devices with a model type of VersaMax ENIU to support only one connection. When left in this configuration, the device communication will terminate the connection to the device when transitioning from the secondary to the primary – leaving no connection between the secondary and the primary host. The status of the communications will not indicate a device communication failure since the interface is not designed to maintain an active connection on the secondary.
- On the VersaMax ENIU's, only one connection is permitted. If another application is using the connection, the CIMPLICITY device communication interface will not be able to form a connection.

- In host redundant environments, for the devices with a model type of VersaPoint ENIU, the connection is broken and re-made when the host states transition between primary and secondary hosts.
- The VersaPoint ENIU's display may indicate a code of `nF` if the connected client doesn't respond within the time interval defined by the Process Data Watchdog timeout. In an application, there are many factors that control the frequency with which the device communicates interface with a specific device including configuration. As a result, the device communication interface may not always maintain an active conversation within the interval dictated by the Process Data Watchdog Timer.

The Process Data Watchdog Timer can be set to 0 to disable the monitoring. Each application must be evaluated carefully to determine the ramifications of this action. If the Process Data Watchdog Timer is set to 0, the ENIU will not immediately detect the loss of a connection.

Regardless of the setting of the Process Data Watchdog Timer, for device's with a model type of VersaPoint ENIU, the CIMPLICITY device communication interface will define a device down condition as one where the interface cannot establish and communicate with the VersaPoint ENIU. As a result, there may be time periods where there is not an active connection to the VersaPoint ENIU, but also not a device down condition.

Refer to the VersaMax ENIU (GFK 1860) and VersaPoint ENIU (GFK 2087) User Manuals for more complete information about the VersaPoint and VersaMax ENIU capabilities.

Double Precision Notes

D4xxxx(x) points are Holding Register Points interpreted as Double Precision. Two registers are used for each point with both registers assumed to have a positive value.

Because of this mapping:

- Do not declare Double Precision points with overlapping addresses. For example, D40001 and D40002 overlap, while D40001 and D40003 do not.
- Do not define points that use the same Holding Register and Double Precision point address (for example, defining one point with the address D40001 and another with the address 40001).

Configure Double Precision points as DINT or UDINT point.

Arrays are not supported for Double Precision points.

Set up Modbus TCP/IP Communications

Set up Modbus TCP/IP Communications

After gathering and reviewing Modbus TCP/IP Required Documents, you are ready to set up TCP/IP communications with your Modbus programmable controllers.

Step 1 *(page 664)*	Install the Modbus TCP/IP Option
Step 2 *(page 674)*	Verify the Modbus TCP/IP Installation
Step 3 *(page 676)*	Test Communication
Step 4 *(page 679)*	Configure CIMPLICITY for Modbus TCP/IP.

Step 1. Install the Modbus TCP/IP Option

Step 1. Install the Modbus TCP/IP Option

The steps and options presented in the procedure that follows require installation of the CIMPLICITY base system software with the Modbus TCP/IP Communications option.

Configure Modbus TCP/IP communications using either:

Option 1.1 *(page 664)*	The Modbus TCP/IP Diagnostics function.
Option 1.2 *(page 665)*	Advanced Modbus TCP/IP Configurations.

Important: Some devices, such as devices with non-consecutive memory or devices installed on a slow or "noisy" network, require advanced Modbus TCP/IP configuration .

Option 1.1. Setup using Modbus TCP/IP Diagnostics

Modbus TCP/IP Installation Checklist

1. Use the Modbus TCP/IP Diagnostics function to test communications with programmable controllers on the network.
2. Use the Ports function to create port information.
3. Use the Devices function to configure one CIMPLICITY device for each programmable controller from which data will be collected.

4. Use the Device Points function to configure points for the devices on the Modbus TCP/IP network.

Option 1.2. Set up Advanced Modbus TCP/IP Configurations

Option 1.2. Set up Advanced Modbus TCP/IP Configurations

The Modbus TCP/IP device communication enabler is designed to operate with devices compatible with the Open Modbus/TCP specification release 1.0. Because the Modbus TCP/IP protocol is supported by a number of vendors on a variety of different devices and because the layout of memory varies from device to device, the range of the various defined memory types varies.

As part of its initialization, the device communication enabler determines the size of the various memory types. By default, the size of the memory is determined by probing the device to determine the range of memory.

This method may not be appropriate for all devices. For devices that have any of the following characteristics, the alternate configuration file described below should be used:

- Devices with non-consecutive memory. (All configured device points must be correctly configured to maintain reliable communication.)
- Devices that do not support the protocol for reading multiple quantities of consecutive memory locations for all of the following (independent of point configuration):
- Coils
- Discrete Inputs
- Discrete Outputs
- Holding Registers
- General Reference Registers
- Double Precision Input Registers
- Double Precision Holding Registers
- Environments that are either noisy or busy such that a retry count greater than 5 is required in order to reliably maintain communication with the device.

Step 1.2.1 *(page 665)*	Enter model information.
Step 1.2.2 *(page 666)*	Enter required configuration data.

Step 1.2.1. Entering Model Information

Important: Each device included in the project that will be using the alternate configuration must have a STAT_PLC model.

If the STAT_PLC model is not selectable from the device configuration screen, it can be added to the CIMPLICITY configuration by editing the IC646TME000.MODEL configuration file located in the BSM_DATA subdirectory of the original CIMPLICITY distribution.

Add the following line to the file using a text editor.

```
MB_TCPIP|STAT_PLC|35
```

For existing projects, these additional steps define a process for adding the model:

1. Set default directory to the master sub-directory of the project.

   ```
   Idtpop model
   ```

2. Using the text editor, add the following line to the file:

   ```
   MB_TCPIP|STAT_PLC|35
   ```

3. Save and exit from the text editor.

   ```
   Scpop model
   ```

4. Update the project.

Step 1.2.2. Enter Required Configuration Data

Step 1.2.2. Enter Required Configuration Data

For each device port in the project, a separate alternate configuration file must be created. The alternate configuration file must be located in the data sub-directory for each project.

For each alternate configuration file, the file name is based on the port configuration.

`MASTER_MBTCP0.INI`	File name for the first port. This file contains any devices configured on the first port that use the alternative configuration.
`MASTER_MBTCP1.INI`	File name for the second port. This file contains the alternate configuration information for devices. The same convention is followed for devices configured for use on the other Modbus Ethernet ports.

In the alternate configuration file, for each device using this scheme, the following information is required:

1. CIMPLICITY device name

2. Number of bytes used for each of the following memory references:
   - Coils (COILS)

- Discrete Inputs (DISC INPUTS)
- Input Registers (INPUT REG)
- Holding Registers (HOLDING REG)
- General Reference File 1 (GEN REF FILE1)
- General Reference File 2 (GEN REF FILE2
- General Reference File 3 (GEN REF FILE3
- General Reference File 4 (GEN REF FILE4
- General Reference File 5 (GEN REF FILE5)
- General Reference File 6 (GEN REF FILE6)
- General Reference File 7 (GEN REF FILE7)
- General Reference File 8 (GEN REF FILE8)
- General Reference File 9 (GEN REF FILE9)
- General Reference File 10 (GEN REF FILE10)
- Double Precision Input Registers (DP_INPUT REG)
- Double Precision Holding Registers (DP_HOLDING REG)

For example, if there are 16384 coils on the device, the coil byte count in the configuration file would be 2048 (8 coils per byte). Similarly, if there are 32767 holding registers on the device, the byte count of holding registers in the configuration file would be 65534 (2 bytes per holding register).

Sample Alternate Configuration File

The sample configuration file in this section defines the domain configuration for two devices, DEVICE1 and DEVICE2.

Note that

- Each Device ID appears as a section heading inside square brackets.
- The `UseTheseDomainSizes` variable is set to 1.

This variable must appear in each device section.

the Modbus TCP/IP devcom will auto-size the domains as it always has if any of the following is true.

- This variable is set to 0.
- There is no section heading for a device on this port.
- This .ini file does not exist.

There are 16 domains in a Modbus TCP/IP device. In the alternate configuration file example shown below, you will see each domain's name followed by the default size of that domain. If a domain name is not provided within a device section, the default size is assigned.

```
[DEVICE1]
UseTheseDomainSizes=1
```

```
COILS=65535
DISC INPUTS=65535
INPUT REG.=65535
HOLDING REG.=65535
GEN REF FILE1=0
GEN REF FILE2=0
GEN REF FILE3=0
GEN REF FILE4=0
GEN REF FILE5=0
GEN REF FILE6=0
GEN REF FILE7=0
GEN REF FILE8=0
GEN REF FILE9=0
GEN REF FILE10=0
DP_INPUT REG.=0
DP_HOLDING REG.=0
[DEVICE2]
UseTheseDomainSizes=1
COILS=65535
DISC INPUTS=65535
INPUT REG.=65535
HOLDING REG.=65535
GEN REF FILE1=0
GEN REF FILE2=0
GEN REF FILE3=0
GEN REF FILE4=0
GEN REF FILE5=0
GEN REF FILE6=0
GEN REF FILE7=0
GEN REF FILE8=0
GEN REF FILE9=0
GEN REF FILE10=0
DP_INPUT REG.=0
DP_HOLDING REG.=0
```

Step 1.2.3. Edit INI Files

Step 1.2.3. Edit INI Files

Step 1.2.3.1 *(page 668)*	Create the Generic PLC Model.
Step 1.2.3.2 *(page 670)*	Configure INI parameters

Step 1.2.3.1. Create the Generic PLC Model

The device communication interface will read the .ini parameters regardless of the configured device model.

Important: It is strongly recommended that entries in the .ini file be restricted to devices with a model type of STAT_PLC or Generic PLC.

Refer to the product documentation for instructions for creating the STAT_PLC model.

Create the Generic PLC model to use in a project using the Modbus Ethernet protocol

1. Click Tools>Command Prompt on the Workbench menu bar.
2. Type `cd master` in the Command Prompt window and press Enter.
3. Type `idtpop model` and press Enter.
4. Open model.idt in a text editor.

   For example, type notepad model.idt and press Enter.

5. Add the following line:

   ```
   MB_TCPIP|Generic PLC|180
   ```

6. Save model.idt and close the text editor.
7. Type `scpop model` at the command prompt and press Enter.
8. Close the Command Prompt window.
9. Perform a configuration update in the project's Workbench.

   Note: The STAT_PLC model sizing is different from the Generic PLC model if you do one of the following.

   - Define the parameter `UseTheseDomainSizes` to be 0.
   - Do not specify all of the domains.

   Default Sizes

   - If STAT_PLC sizes are not defined in the INI file the defaults are:

Max Coils	32768
Max Input Status	32768
Max Input Regs	16384
Max Holding Regs	16384

   All other memory types have a size of 0.

- If Generic PLC and other device models sizes are not defined in the INI file the defaults are determined by searching the memory in the PLC for the end points.

Step 1.2.3.2. Configure INI Parameters

Step 1.2.3.2. Configure INI Parameters

By default, in the .ini file, the memory sizes are defined in terms of bytes rather than elements.

To change the way the memory sizes are specified to indicate counts, define the following variable for each device affected device:

```
UseCounts=1
```

Some devices use the Force Single Coil command (function code 5) for writing single coils.

Other devices support (sometimes exclusively) the Force Multiple Coils (function code 15) when writing coils.

You can configure the Modbus Ethernet communication interface to use a specific function code when writing single coils and when interacting with a particular device.

To control the behavior, for this specific action on a specific device with a model STAT_PLC, define the following variable within the device definition of the master_port.ini file located in the project's data directory:

```
OneCoilWrite=1 will cause function code 5 to be used

OneCoilWrite=0 will cause function code 15 to be used.

Writing Single Holding Registers
```

Some devices use the Force Single Holding Registers command (function code 6) to write a single holding register.

Other devices support (sometimes exclusively) the Force Multiple Holding Registers Command (Function Code 16).

To control the behavior, for this specific function on a specific STAT_PLC device, define the following variable within the device definition of the master_port.ini file located in the project's data directory:

```
OneRegisterWrite=1will cause function code 6 to be used.

OneRegisterWrite=0will cause function code 16 to be used.
```

Some devices will terminate a connection if it there is no activity for some pre-configured interval. For devices which exhibit this behavior, a global variable may be defined to change the

device communication interface's handling of a disconnect as it relates to device down. When the device variable is enabled, the device will not be declared down unless it is unable to re-establish the connection and perform the next immediate I/O operation to request or modify data on the device. The variable is defined within the device definition of the MASTER_PORT.INI file located in the project's data directory:

ConservesConn I=1 enables this processing

ConservesConn I=0 disables this processing

Host Redundancy with Devices with One Connection

⚠ **Important:** Some devices will support only one connection. In host redundant environments, it is necessary to identify the devices with this characteristic so that the connection can be terminated on the acting secondary and initiated on the primary following a transition in the host redundancy roles.

To identify a device that only supports a single connection, define the following variable under the affected device in the master_port.ini file located in the project's data directory:

Note that a device down will not be generated as a result of not having a connection with the secondary.

```
ConnSecondary =1 enables this processing

ConnSecondary =0 disables this processing.
```

Sample .INI File

```
; Device level parameters
; UseTheseDomainSizes - When enabled (=1), use the sizes defined in the INI
;                        to defined the memory sizes for each domain
;                       When disabled (=0), for STAT_PLC, use the following
;                        defaults:
;                           Max Coils :       32768 elements
;                           Max Input Status: 32768 elements
;                           Max Input Regs:   16384 elements
;                           Max Holding Regs: 16384 elements
;                        All other memory types are 0 elements
;;                        For all other device models, the device
 communication
;                        interface will attempt to size the memory
;
;                        The default value is 0
;
; UseCounts - When enabled (=1), indicates sizes are in elements
;             When disabled (=0) indicates sizes are in bytes
;
;             Default value is 0
;
```

```
; ConservesConn - When enabled (=1), indicates that it is normal for the
 device
;                 to close the connection (typically based on inactivity).
 The
;                 device communication interface will not assume that the
;                 device is down unless it is unable to create a connection
 and
;                 get a response when it attempts the current scheduled
;                 operation to retrieve data from the device or modify
 data.
;
;                 When disabled (=0), indicates that a termination of the
;                 connection between the device communication interface and
;                 the device will cause the device communication interface
 to
;                 assume that the connection is down and terminate the
;                 connection
;
;                 Default value is 0.
;
; ConnSecondary - When Enabled (=1), in a Host Redundant environment,
;                 the device communciation interface will attempt to
;                 maintain a connection with the device on the acting
;                 secondary.
;
;                 When Disabled(=0), in a Host Redundant environment,
;                 the device communication interface will terminate its
;                 connection to the device when transitioning to the
;                 secondary.
;
;                Default value is 1.
;
; OneCoilWrite - When enabled (=1) use Function 5 to write single coils
;                When disabled (=0) use Function 15 to write single coils
;
;                VersaMax ENIU, VersaPoint ENIU and Modicon 484's ignore
 this
;                parameter.
;
;OneRegiserWrite - When enabled (=1) use Function 6 to write single holding
 registers
;                  When disabled (=0) use Function 16 to write single
 holding registers
;
;                  VersaMax ENIU, VersaPoint ENIU and Modicon 484's ignore
 this
;                  parameter.
;
[DEVICE1]
UseTheseDomainSizes=1
UseCounts=0
OneCoilWrite=0
OneRegWrite=0
```

```
ConservesConn=1
ConnSecondary=0
COILS=65535
DISC INPUTS=65535
INPUT REG.=65535
HOLDING REG.=65535
GEN REF FILE1=0
GEN REF FILE2=0
GEN REF FILE3=0
GEN REF FILE4=0
GEN REF FILE5=0
GEN REF FILE6=0
GEN REF FILE7=0
GEN REF FILE8=0
GEN REF FILE9=0
GEN REF FILE10=0
DP_INPUT REG.=0
DP_HOLDING REG.=0

[DEVICE2]
UseTheseDomainSizes=1
UseCounts=0
OneCoilWrite=0
OneRegWrite=0
ConservesConn=1
ConnSecondary=0
COILS=65535
DISC INPUTS=65535
INPUT REG.=65535
HOLDING REG.=65535
GEN REF FILE1=0
GEN REF FILE2=0
GEN REF FILE3=0
GEN REF FILE4=0
GEN REF FILE5=0
GEN REF FILE6=0
GEN REF FILE7=0
GEN REF FILE8=0
GEN REF FILE9=0
GEN REF FILE10=0
DP_INPUT REG.=0
DP_HOLDING REG.=0
```

Maintain Files

⚠ **Important:** When the alternate configuration file is used, it is essential that the information in the file be maintained so that it always accurately reflects the configuration of the actual device.

If the file defines a memory type that is too small and there are device points referencing memory locations beyond the range declared in the configuration file, but defined on the device, the point will be incorrectly declared invalid.

If the file defines a memory range that is too large, the result can be an intermittent loss of device communications where the root cause cannot be found based on environmental causes. This will occur if a device point is defined with a reference that is beyond the device memory of the device, but within the range defined in the configuration file.

Step 2. Verify the Modbus TCP/IP Installation

Step 2. Verify the Modbus TCP/IP Installation

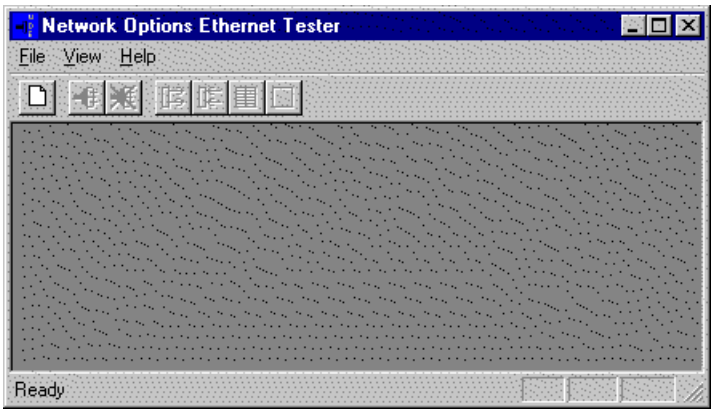
You can use the Modbus TCP/IP Diagnostics program provided with the CIMPLICITY Modbus TCP/IP Communications option to check the basic configuration and operation of your network without starting a CIMPLICITY project. You can perform the following functions:

- Read Holding Registers
- Write Holding Registers

Note: For this program to function, you must have successfully installed the CIMPLICITY Modbus TCP/IP Communications option.

To start the program, select the **Modbus TCP/IP Diagnostics** icon in the CIMPLICITY menu.

The Network Options Ethernet Tester window opens.



This menu monitors data communications for a given Modicon Quantum PLC.

Step 2.1 *(page 675)*	Connect to a PLC

Step 2.1. Connect to a PLC

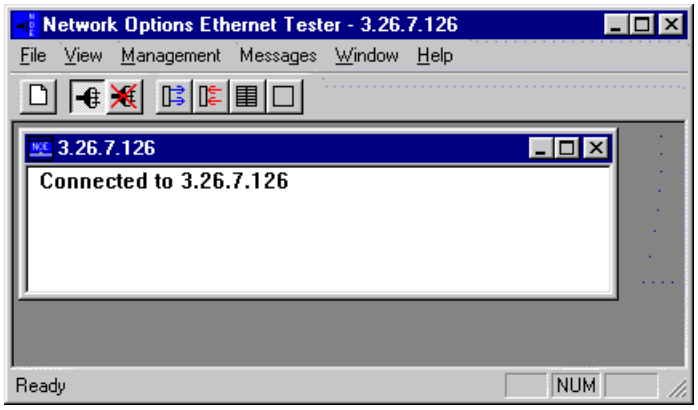
1. Select New from the File menu or click the New button on the Toolbar. The Remote IP Address dialog box opens.

2. Enter the IP address of the Modicon Quantum PLC in the **IP Address** field.

   Note: If a bridge is used, include a destination ID *(page 675)* in the device address.

3. Click OK.

   The window looks like this when the connection is made:



Configure a Destination ID

When a bridge is used, a destination ID is required in the device address.

The format of the device address is `X.X.X.X@Y`

Where

`X.X.X.X` is the Modicon Bridge IP address and

`Y` is the Destination ID of the device to route the message to.

Example

To communicate with a device mapped to Destination ID 5, through a bridge with IP address `1.2.3.4`, the device address should be `1.2.3.4@5`.

Step 3. Test Communication

Step 3. Test Communication

The following sections discuss some of the more frequently used functions of the tester utility. For more detailed information about using this utility, please refer to Modicon Quantum Ethernet TCP/IP Module User Guide (840 USE 107 00).

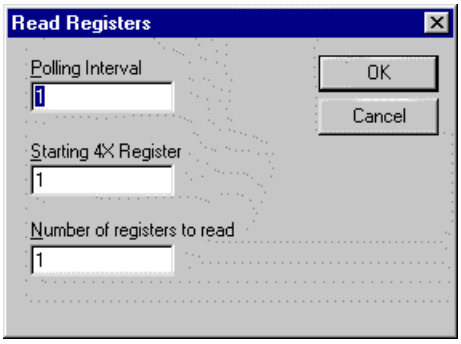
Option 3.1 *(page 676)*	Read Modbus TCP/IP Test Data
Option 3.2 *(page 677)*	Write Modbus TCP/IP Test Data
Option 3.3 *(page 678)*	Show Device Statistics
Option 3.4 *(page 678)*	Clear Device Statistics

Option 3.1.Read Modbus TCP/IP Test Data

To initiate a read request from the device, select the device's window and do one of the following:

- Select Read Registers from the Messages menu.
- Click the Read Registers button on the toolbar.

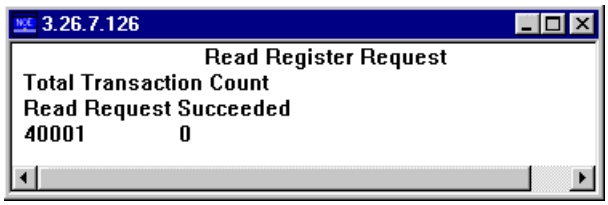
The Read Registers dialog box opens.



To specify the request:

1. Enter the polling interval in the **Polling Interval** field.
2. Enter the starting Holding Registers memory address where you want to read data in the **Starting 4X Register** field.
3. Enter the number of registers to read in the **Number of registers to read** field.
4. Click OK.

   When the request completes, the window for the device looks like this:

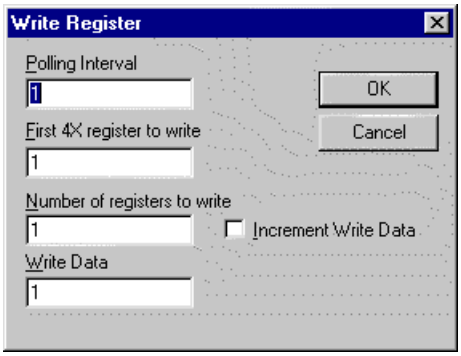


Option 3.2 WriteModbus TCP/IP Test Data

To initiate a write request from the device, select the device's window and do one of the following:

- Select Write Registers from the Messages menu.
- Click the Write Registers button on the toolbar.

The Write Register dialog box opens.



To specify the request:

1. Enter the polling interval in the **Polling Interval** field.
2. Enter the starting Holding Registers memory address where you want to write data in the **Starting 4X Register** field.
3. Enter the number of registers to write in the **Number of registers to write** field.
4. Enter the data you want written to the registers in the **Write Data** field.

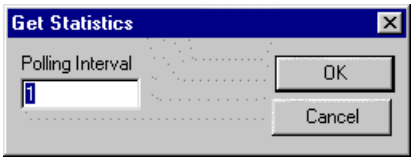
5. Click OK.

Option 3.3. Show Device Statistics

To display communication statistics for a device, select the device's window and do one of the following:

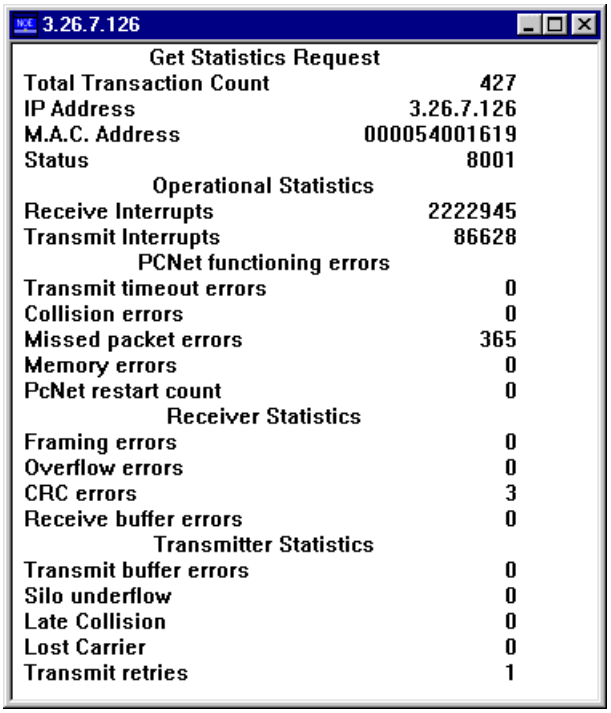
- Select Get Stats from the Messages menu.
- Click the Get Stats button on the toolbar.

The Get Statistics dialog box opens.



Enter the polling interval you want in the **Polling Interval** field, and click OK.

The device's window displays the following information:

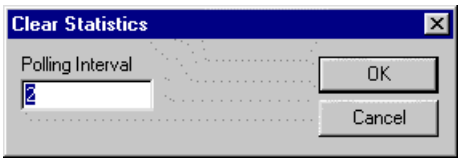


Option 3.4. Clear Device Statistics

To clear communication statistics for a device, select the device's window and do one of the following:

- Select Clear Stats from the Messages menu.
- Click the Clear Stats button on the toolbar.

The Clear Statistics dialog box opens.



Enter the polling interval you want in the **Polling Interval** field, and click OK.

Step 4. Configure CIMPLICITY for Modbus TCP/IP

Step 4. Configure CIMPLICITY for Modbus TCP/IP

When you are configuring ports, devices, and device points, you will be asked for some information that is specific to Modbus TCP/IP communications.

Step 4.1 *(page 679)*	Configure Modbus TCP/IP Ports.
Step 4.2 *(page 683)*	Configure Modbus TCP/IP Devices.
Step 4.3 *(page 686)*	Configure Modbus TCP/IP Points.

Step 4.1 Configure Modbus TCP/IP Ports

Step 4.1. Configure Modbus TCP/IP Ports

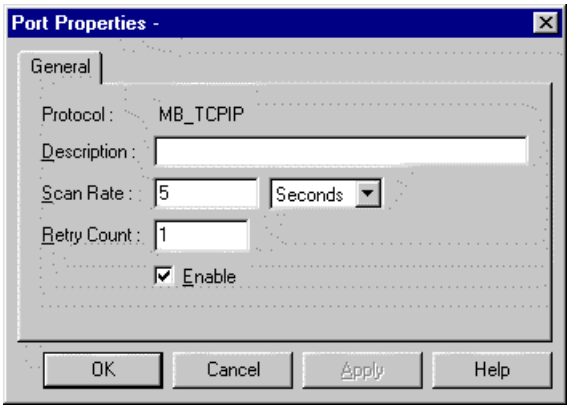
1. In the **Protocol** field, select MB_TCPIP from the list of available protocols.
2. In the **Port** field, select the communication port that will be used for Modbus TCP/IP communications.

   When you click OK to create the port, the Port Properties dialog box for the protocol opens.

Step 4.1.1 *(page 680)*	Set general port properties.

Step 4.1.2 *(page 680)*	Saved settings startup action.
Step 4.1.3 *(page 682)*	Saved settings deletion.

Step 4.1.1. Set General Port Properties



rect 21, 82, 329, 100 *(page 680)*
rect 22, 108, 237, 128 *(page 680)*
rect 22, 134, 156, 153 *(page 680)*
rect 92, 162, 148, 181 *(page 680)*

Use the General tab in the Port Properties dialog box to enter general information for the port. You can define the following:

Field	Description
Description	Enter the base scan rate for the port. Point scan rates are multiples of the base rate. You can specify a scan rate in Ticks (100 ticks = 1 second), Seconds, Minutes, or Hours.
Scan Rate	Enter an optional description to help you identify the port.
Retry Count	If communications cannot be established for a device on the port, the device is considered to be down, and a $DEVICE_DOWN alarm is generated. Once a device is down, periodic attempts are made to resume connections to it. Enter the number of scans to wait before attempting to reconnect to a device after a communications error has been detected.
Enable	Set this check box if you want the port to be enabled when the project starts. If you clear this check box, the port will not be enabled when the project starts, and points will not be available for devices on the port.

Step 4.1.2. Saved Settings Startup Action

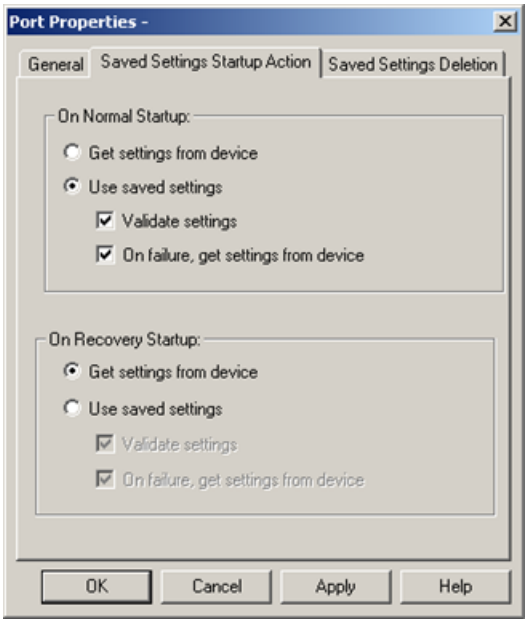
Modbus TCP/IP Communications provides you with the option to reduce normal and/or recovery start up time by saving device characteristics for subsequent re-use.

Note: Saved Settings Startup Action is intended for devices whose characteristics, such as the memory size, do not change. If the settings change, make sure they are deleted . Saved settings startup is supported only for devices where the number of coils and inputs are divisible by 8.

Select the Saved Settings Startup Action tab in the Modbus TCP/IP Communications Port Properties dialog box.

Selections can be made for the following.

- Normal startup.
- On Recovery Startup.



rect 29, 244, 229, 310 *(page 682)*
rect 25, 104, 225, 170 *(page 682)*
rect 27, 198, 155, 218 *(page 682)*
rect 31, 221, 183, 241 *(page 682)*
rect 27, 83, 178, 103 *(page 682)*
rect 24, 60, 175, 80 *(page 681)*

Normal Startup

- Normal Startup occurs when CIMPLICITY starts.
- CIMPLICITY is started either when:
- The computer is booted up or
- A CIMPLICITY user starts the project.

Options for each of the startups are as follows.

Get settings from device

Defines the actions that the device communication interface takes to determine the supported memory types and ranges for a specific device.

The methods vary by device communication interface.

Use saved settings

Checked: The device communication interface will use the stored settings to define the device-specific memory types and ranges.

- The device configuration data is recorded and stored for later use.
- Options if Use saved settings is checked are:

Option	Description	
Validate settings	Checked	The device communication interface will query the high range of each memory range as a quick check to confirm that the memory type and range represented is valid.
	Clear	The device communication interface will not do a quick check.
On failure, get settings from device	When using saved settings, failures may occur. Two typical failures are: • An integrity error is detected in the saved information. • There is a failure to verify a memory range when Validate settings is selected.	
	Checked	When a failure occurs the device communication interface will immediately attempt to determine the valid device information using the standard method for obtaining the information.
	Clear	When the failure occurs, the: • Device will immediately be marked down. • Valid device information will not be collected during initial startup. It will be determined later during retry processing.

On Recovery Startup

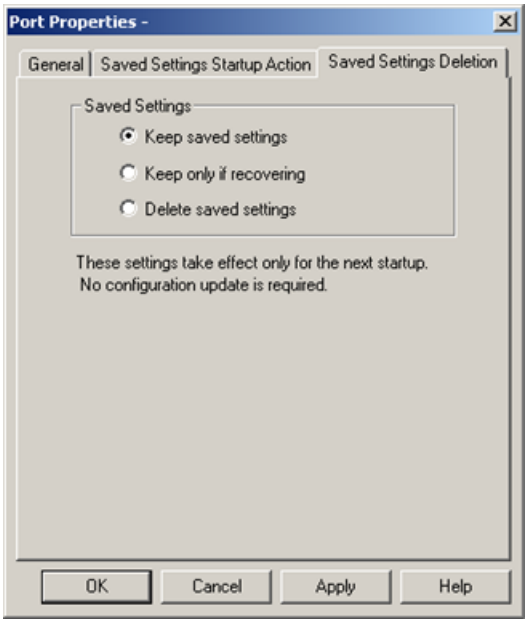
Recovery startup occurs after:

- A cluster fails.
- Process health kills a project as a result of a process failure.

Options *(page 681)* for On Recovery Startup are the same as they are for On Normal Startup.

Step 4.1.3. Saved Settings Deletion

Check one of the following to specify what Modbus TCP/IP Communications should do with saved settings.



Option	Description
Keep saved settings	Do not automatically delete the saved settings on the next startup
Keep only if recovering	Delete the current saved settings unless the next startup is in recovery mode.
Delete saved settings	Deletes: • The current saved settings at the next startup. • Settings for all the devices that are configured for the port. **Note:** If the configuration of the device is changed, make sure to check and apply Delete saved settings.

Note: These settings affect the next startup only. The Saved Settings Deletion tab will always default to the Keep saved settings option for subsequent startups.

Step 4.2. Configure Modbus TCP/IP Devices

Step 4.2. Configure Modbus TCP/IP Devices

1. In the **Device** field, enter the name of the device you are configuring.

NOTE: The following special characters cannot be used for a Modbus RTU device name: \, /, :, *, ?, ", <. >, |, [,]. If the device name contains any of these characters, the device settings will not be preserved and saved startup will not be possible.

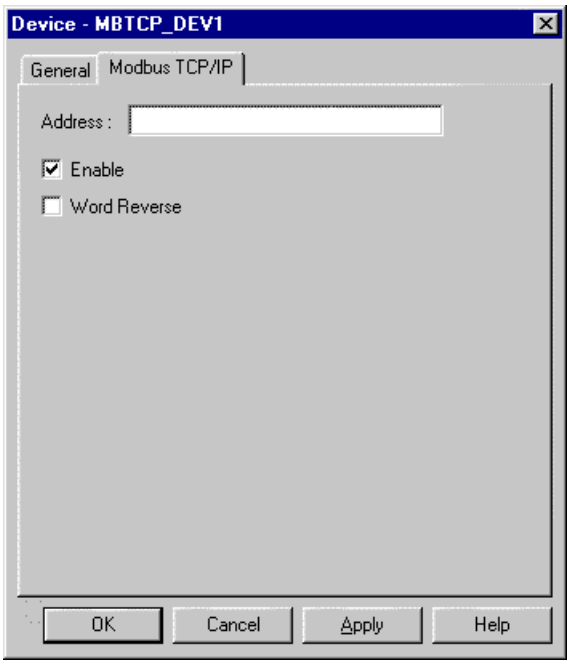
2. In the **Port** field, select the Modbus TCP/IP port to be used by the device.

   When you click OK to create the device, the Device Properties dialog box opens.

Step 4.2.1 *(page 684)*	Set general device properties.
Step 4.2.2 *(page 685)*	Set Modbus TCP/IP properties.

Important: Only one CIMPLICITY device can be configured per physical Modicon programmable controller. Modbus TCP/IP communications will not run with multiple CIMPLICITY devices configured for the same programmable controller.

Step 4.2.1. Set General Device Properties

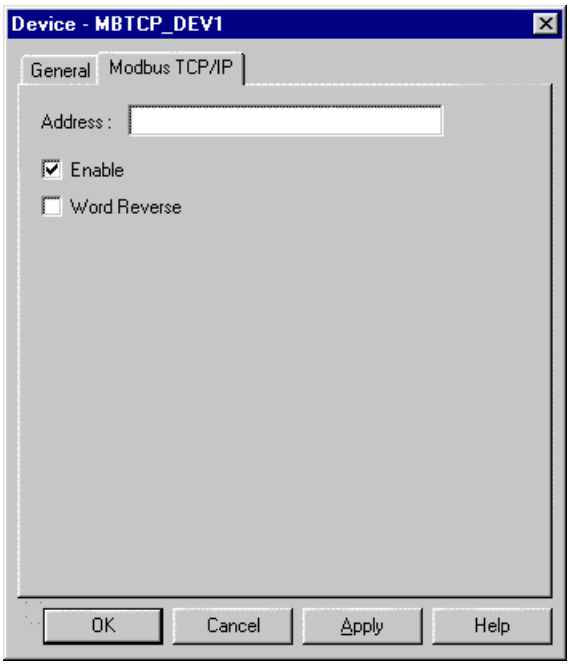


rect 12, 50, 231, 75 *(page 685)*
rect 8, 101, 297, 123 *(page 685)*
rect 17, 130, 232, 151 *(page 685)*
rect 13, 153, 294, 173 *(page 685)*

Use the General tab in the Device dialog box to enter general information for the device. You can define the following:

Port	Select the port for this device.
Description	Enter an optional description to help you identify the device.
Resource	Enter a resource that can be associated with this device for alarm generation.
Model Type	Select the type of device. Click the drop-down button to the right of the **Model type** field to display your choices, and make a selection. For this protocol, the choices are: VersaMax ENIU VersaPoint ENIU Modicon 184/384 Modicon 484 Modicon 584 Modicon 984 Quantum STAT_PLC

Step 4.2.2. Set Modbus TCP/IP Properties



rect 20, 55, 281, 86 *(page 685)*
rect 18, 90, 91, 114 *(page 685)*
rect 19, 114, 118, 137 *(page 686)*

Use the Modbus TCP/IP properties to enter information about Modbus TCP/IP communications for the device. You can define the following:

Address	Enter the unique Modbus TCP/IP IP address for the device. Address format is TCP/IP Address@Destination Id. Communications via a Modbus Ethernet Bridge are supported.
	Any CIMPLICITY devices that use the same IP address should not be enabled at the same time. This does not apply when a destination ID is used.
Enable	Set this check box to enable the device when the project starts. If you clear this check box, the device will not be enabled and points associated with the device will be unavailable.

Word Reverse	Set this check box to reverse the LOW WORD and HIGH WORD order. This affects four byte values only (REAL, DINT, UDINT). Some devices store the LOW WORD and HIGH WORD in reverse order (from a Modicon PLC). This will reverse them so they can be viewed correctly in CIMPLICITY.

Step 4.3. Configuring Modbus TCP/IP Points

Step 4.3. Configure Modbus TCP/IP Points

Once your devices are configured, you may configure points for them. Through device point configuration, you may configure the following:

- Points that read or set Coils.
- Points that read Discrete Inputs.
- Points that read Input Registers.
- Points that read or set Holding Registers.
- Points that read or set General Reference
- Points that read Holding Registers as Double Registers

Fields in the Point Properties dialog box have configuration values that are unique to or have special meaning for Modbus TCP/IP communications.

Step 4.3.1 *(page 686)*	Set general point properties.
Step 4.3.2 *(page 686)*	Set device point properties.

Step 4.3.1. Set General Point Properties

On the General tab, the type of Access you choose depends on the point address:

- Check the Read Only checkbox for points from Discrete Inputs or Input Registers.
- Clear this checkbox (i.e. Read/Write access) for points from Coils or Holding Registers.

Step 4.3.2. Set Device Point Properties

On the Device tab:

Update Criteria	The update criteria you use determines how the data will be requested.
	Select On Change or On Scan for points whose values should be polled by the Modbus TCP/IP driver at regular intervals.
	Select On Demand On Scan or On Demand On Change for points whose values should be polled by the Modbus TCP/IP driver at regular intervals while users are viewing them.

	Select Unsolicited for points whose value will be sent by the programmable controller as necessary.

Important: The address of the unsolicited data must exactly match the point address.

Address	Enter the point address of the data as follows:
	`0xxxx` or `0xxxxx` for Coils `1xxxx` or `1xxxxx` for Discrete Inputs `3xxxx` or `3xxxxx` for Input Registers `4xxxx` or `4xxxxx` for Holding Registers `6yxxxx` or `6yxxxxx` for General Reference `D4xxxx` or `D4xxxxx` for Holding Registers as Double Precision
	where `xxxx` or `xxxxx` is the bit number or register number of the point. For example, the address of Holding Register 1 is 40001, and the address of Coil 1 is 00001.
	If you are configuring a General Reference point, `y` is the file number.
	Double Precision points in Holding Registers require two Holding Registers per point. The upper register is multiplied by 10000 and the product is added to the lower register to get the resulting value. No more than one Double Precision point may be written in a single write operation. Use DINT or UDINT types for these points.
	If you are configuring an Analog point type in Discrete Inputs or Coils, the address entered in this field must be that of the least significant bit of the point. For example, if you configure a 16-bit analog point for Coils 17 through 32, enter 00017 in this field.

When you configure Boolean points in Holding Registers, Input Registers, or General Reference, you must also enter data in the following field:

Bit Offset	Enter the bit offset in the Holding Register, Input Register, or General Reference that defines the point, where 0 is the least significant bit and 15 is the most significant bit.

Modbus TCP/IP Unsolicited Data

Modbus TCP/IP Unsolicited Data

To send unsolicited point values from a Modbus TCP/IP programmable controller to the CIMPLICITY project using the Modbus TCP/IP Communication enabler, a MSTR function block must be coded in the ladder logic. The MSTR block can send a request to write to Holding Registers in another Modbus TCP/IP Server. The Modbus TCP/IP Communication enabler uses the data portion of this message to hold a command to set a CIMPLICITY point value.

Only one project per computer can process unsolicited data and only one port within that project can perform the processing. By default, all ports are enabled for processing unsolicited data. If you have multiple Modbus TCP/IP ports on your computer, you must set the Enable/Disable Unsolicited Data *(page 701)* global parameter to disable the unsolicited processing on all ports except the one that is designated to do the processing.

The Modbus TCP/IP Communications enabler can receive unsolicited values for points in all four device memory types (Coils, Discrete Inputs, Input Registers, and Holding Registers).

CIMPLICITY device points with the following criteria will be updated with values from the command:

- The starting address of the point exactly matches the address in [Word 3] of the command,
- The length of the point is less than or equal to the number of data bytes sent (as specified in [Word 4] of the command),
- The point's Modbus TCP/IP device address matches that of the device sending the command

Note: You can find information on ladder logic programming and the MSTR block function in Modicon Ladder Logic Block Library User Guide (840 USE 101 00).

MSTR Block Data Area Format

MSTR Block Data Area Format

The format of the MSTR block data area is:

Word 0	Zero
Word 1	Zero
Word 2	Zero
Word 3	Address of the point to be updated. Valid values are: 1 - 8192 for Coils 10001 - 18192 for Discrete Inputs 3001 - 39999 for Input Registers 4001 - 49999 for Holding Registers
Word 4	Number of bytes of point data being sent. Valid values are 1-190.
Word 5 through **Word 99**	Point data

Compressed Time-Stamp Format

For the compressed time-stamp, Words 5 through 7 contain the following information:

Word 5	The 16 bits of this word are formatted as follows: FFFSSSSSSTTTTTTT where: FFF = 001 SSSSSS = the number of seconds (0-59) TTTTTTT = the number of ticks after the second (0-99) (where 100 ticks = 1 second)
Word 6	The 16 bits of this word are formatted as follows: DDDDDHHHHHMMMMMM where: DDDDD = the day of the month (1-31) HHHHH = the hour of the day (0-23) MMMMMM = the minutes after the hour (0-59)
Word 7	The 16 bits of this word are formatted as follows: MMMYYYYYYYYYYYY Where MMMM = the month of the year (1-12) YYYYYYYYYYYY = the year (0-2038)

Uncompressed Time-Stamp Format

For the uncompressed time-stamp, Words 5 through 12 contain the following information:

Word 5	The 16 bits of this word are formatted as follows: FFFnnnnnnnnnnnn where: FFF = 000 nnnnnnnnnnnn = is meaningless and ignored
Word 6	The month of the year (1-12)
Word 7	The day of the month (1-31)
Word 8	The year (0-2038)
Word 9	The hour (0-23)
Word 10	The minutes after the hour (0-59)
Word 11	The seconds after the minute (0-9)
Word 12	The ticks after the second (0-99) where 100 ticks = 1 second

Compressed Time-Stamp with mSecs Format

For the compressed time-stamp, Words 5 through 8 contain the following information:

Word 5	The 16 bits of this word are formatted as follows: FFFnnnnnnnnnnnn where: FFF = 010 nnnnnnnnnnnn = is meaningless and ignored
Word 6	The seconds and ticks with msecs can be formatted as follows. SSSSSSTTTTTTTTTT where: SSSSSS = the number of seconds (0-59) TTTTTTTTTT = the number of mSecs after the second (0-999) (where 1000 mSecs = 1 second)
Word 7	The 16 bits of this word are formatted as follows: DDDDDHHHHHMMMMMM where: DDDDD = the day of the month (1-31) HHHHH = the hour of the day (0-23) MMMMMM = the minutes after the hour (0-59)
Word 8	The 16 bits of this word are formatted as follows: MMMYYYYYYYYYYY where: MMMM = the month of the year (1-12) YYYYYYYYYYY = the year (0-2038)

Uncompressed Time-Stamp with mSecs Format

For the uncompressed time-stamp, Words 5 through 12 contain the following information:

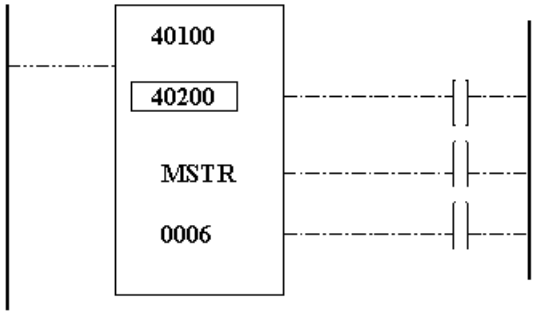
Word 5	The 16 bits of this word are formatted as follows: FFFnnnnnnnnnnnn where: FFF = 011 nnnnnnnnnnnn = is meaningless and ignored
Word 6	The month of the year (1-12)
Word 7	The day of the month (1-31)
Word 8	The year (0-2038)

Word 9	The hour (0-23)
Word 10	The minutes after the hour (0-59)
Word 11	The seconds after the minute (0-59)
Word 12	The mSecs after the second (0-999 where 1000 mSecs = 1 second

Sample MSTR Block - No Timestamping

Sample MSTR Block -No Timestamping

The diagram below summarizes the use of the MSTR block to send unsolicited data without timestamping to the CIMPLICITY Modbus TCP/IP communication enabler from a device.



In this example:

- The Control Block, found in Holding Registers 40100 through 40108, contains control information for the MSTR block. These indicate that six registers worth of data will be sent to the device at IP address 192.168.0.57 and stored starting at address 40001.
- The CIMPLICITY Modbus TCP/IP Device Communications Enabler does not use address information configured in the control block. Instead, it interprets the data area as a command.
- The Data Block, found in Holding Registers 40200 through 40205, contains a command to the CIMPLICITY Modbus TCP/IP communications enabler to set the value of a point at Input Register 17 to a value of 21035.

Control Block - No Timestamping

The Control Block used to send the sample data is stored in Holding Registers 40100 through 40108. Values stored in the Control block are:

	Register	Example

Write Command	40100	1
Status Word	40101	xxxxx
Number of registers to send	40102	6
Destination HR in Standby device	40103	40001
Quantum backplane slot address of the network adapter module (high byte)	40104	768
First byte of IP address	40105	192
Second byte of IP address	40106	168
Third byte of IP address	40107	0
Fourth byte of IP address	40108	57

These Holding Registers indicate that six registers worth of data will be sent to the device at IP address 192.168.0.57 and stored starting at address 40001. The Quantum backplane slot address of the network adapter module is 3. To put it in the high order byte, multiply the slot address by 256 (3*256 = 768).

Data Block - No Timestamping

The Data Block used to send the sample data is stored in Holding Registers 40200 through 40205. The sample data being sent is:

	Register	Exarmple
Network address of sending PLC	40200	0
	40201	0
	40202	0
Address of point receiving unsolicited data	40203	30017
Number of bytes of data	40204	2
The unsolicited value	40205	21035

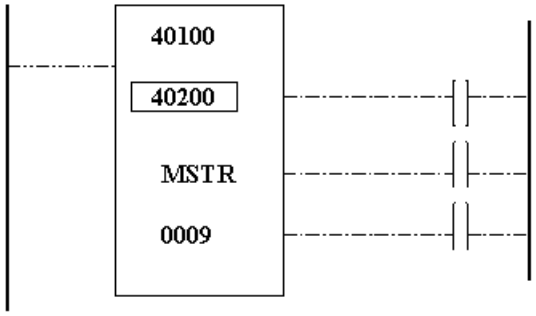
This command causes the Modbus TCP/IP communications enabler to set the point with the address of Input Register 17 to a value of 21035.

The network address of the sending PLC is not used currently.

Sample MSTR Block - Compressed Timestamping

Sample MSTR Block - Compressed Timestamping

The diagram below summarizes the use of the MSTR block to send unsolicited data with compressed timestamping to the CIMPLICITY Modbus TCP/IP communication enabler from a device.



In this example:

- The Control Block, found in Holding Registers 40100 through 40108, contains control information for the MSTR block. These indicate that nine registers worth of data will be sent to the device at IP address 192.1687.0.57 and stored starting at address 40001.
- The CIMPLICITY Modbus TCP/IP Device Communications Enabler does not use address information configured in the control block. Instead, it interprets the data area as a command.
- The Data Block, found in Holding Registers 40200 through 40208, contains a command to the CIMPLICITY Modbus TCP/IP communications enabler to set the value of a point at Input Register 17 to a value of 21035 with a compressed timestamp.

Control Block with Compressed Timestamping

Item	Register	Local Example	Bridge Example
Write Command	40100	1	1
Status Word	40101	xxxxx	xxxxx
Number of registers to send	40102	9	9
Destination HR in Standby device	40103	1	1
Quantum backplane slot address of the network adapter module (high byte)	40104	768	768
First byte of IP address	40105	192	192
Second byte of IP address	40106	168	168
Third byte of IP address	40107	0	0
Fourth byte of IP address	40108	57	57

These Holding Registers indicate that nine registers worth of data will be sent to the device at IP address 192.168.0.57 and stored starting at address 40001.

Data Area - Compressed Timestamping

The Data Area used to send the sample data with compressed timestamping is stored in Holding Registers 40200 through 40208. The sample data being sent is:

Data	Register	Local Example	Bridge Example
Network address of sending PLC	40200	0	1030(x0406)
	40201	0	0
	40202	0	0
Address of point receiving unsolicited data	40203	30017	30017
Number of bytes of data	40204	32770	32770
First timestamp word	40205	9442	9442
Second timestamp word	40206	29419	29419
Third timestamp word	40207	22477	22477
The unsolicited value	40208	21035	21035

Holding Registers 40203 through 40208 contain a command to the CIMPLICITY Modbus TCP/IP communications enabler to set the value of a point at Input Register 17 to a value of 21035 with a date of May 14, 1997 and time 11:43:09.98.

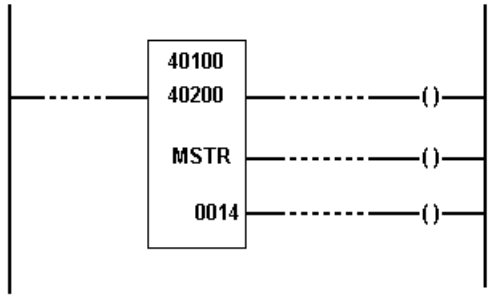
The network address of the sending programmable controller is the Modbus TCP/IP.

- For local data, the path is only 1 address long and the sending controller's IP address is put into the lower byte of the first holding register. The other two holding registers are set to zero.
- For bridged data, the path may be 5 addresses long. The example above shows the path 6.4.0.0.0.

Sample MSTR Block - Uncompressed Timestamping

Sample MSTR Block - Uncompressed Timestamping

The diagram below summarizes the use of the MSTR block to send unsolicited data with uncompressed timestamping to the CIMPLICITY Modbus TCP/IP communication enabler from a device.



In this example:

- The Control Block, found in Holding Registers 40100 through 40108, contains control information for the MSTR block. These indicate that nine registers worth of data will be sent to the device at Modbus TCP/IP 192.168.0.57 and stored starting at address 40001.

The CIMPLICITY Modbus TCP/IP Device Communications Enabler does not use address information configured in the control block. Instead, it interprets the data area as a command.

- The Data Block, found in Holding Registers 40200 through 40214, contains a command to the CIMPLICITY Modbus TCP/IP communications enabler to set the value of a point at Input Register 17 to a value of 21035 with an uncompressed timestamp.

Control Block - Uncompressed Timestamping

The Control Block used to send the sample data with uncompressed timestamping is stored in Holding Registers 40100 through 40108. Values stored in the Control block are:

	Register	Local Example	Bridge Example	
Write Command	40100	1	1	
Status Word	40101	xxxxx	xxxxx	
Number of registers to send	40102	14	14	
Destination HR in Standby device	40103	1	1	
Quantum backplane slot address of the network adapter module (high byte)	40104	768	768	
First byte of IP address	40105	192	192	
Second byte of IP address	40106	168	168	
Third byte of IP address	40107	0	0	
Fourth byte of IP address	40108	57	57	

These Holding Registers indicate that 14 registers worth of data will be sent to the device at IP address 192.168.0.57 and stored starting at address 40001.

Data Area - Uncompressed Timestamping

The Data Area used to send the sample data with uncompressed timestamping is stored in Holding Registers 40200 through 40213. The sample data being sent is:

	Register	Local Example	Bridge Example
Network address of sending PLC	40200	0	1030(x0406)
	40201	0	0
	40202	0	0
Address of point receiving unsolicited data	40203	30017	30017
Number of bytes of data.	40204	32770	32770
First timestamp word	40205	0	0
Second timestamp word	40206	5	5
Third timestamp word	40207	14	14
Fourth timestamp word	40208	1997	1997
Fifth timestamp word	40209	11	11
Sixth timestamp word	40210	43	43
Seventh timestamp word	40211	09	09
Eighth timestamp word	40212	98	98
The unsolicited value	40213	21035	21035

Holding Registers 40203 through 402058 contain a command to the CIMPLICITY Modbus TCP/IP communications enabler to set the value of a point at Input Register 17 to a value of 21035 with a date of May 14, 1997 and time 11:43:09.98.

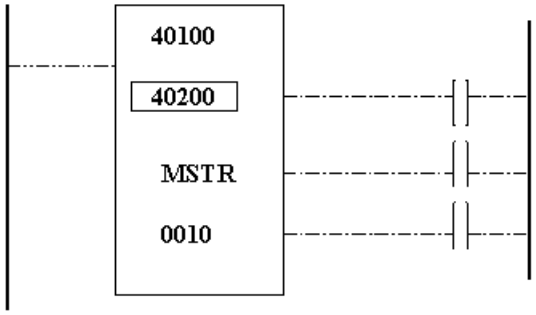
The network address of the sending programmable controller is the Modbus TCP/IP.

- For local data, the path is only 1 address long and the sending controller's IP address is put into the lower byte of the first holding register. The other two holding registers are set to zero.
- For bridged data, the path may be 5 addresses long. The example above shows the path 6.4.0.0.0.

Sample MSTR Block - Compressed Timestamping with mSecs

Sample MSTR Block - Compressed Timestamping with mSecs

The diagram below summarizes the use of the MSTR block to send unsolicited data with compressed timestamping with mSecs to the CIMPLICITY Modbus TCP/IP communication enabler from a device.



In this example:

- The Control Block, found in Holding Registers 40100 through 40108, contains control information for the MSTR block. These indicate that nine registers worth of data will be sent to the device at IP address 192.168.0.57 and stored starting at address 40001.
- The CIMPLICITY Modbus TCP/IP Device Communications Enabler does not use address information configured in the control block. Instead, it interprets the data area as a command.
- The Data Block, found in Holding Registers 40200 through 40209, contains a command to the CIMPLICITY Modbus TCP/IP communications enabler to set the value of a point at Input Register 17 to a value of 21035 with a compressed timestamp.

Control Block - Compressed Timestamping with mSecs

The Control Block used to send the sample data with compressed timestamping is stored in Holding Registers 40100 through 40108. Values stored in the Control block are:

	Register	Local Example	Bridge Example	
Write Command	40100	1	1	
Status Word	40101	xxxxx	xxxxx	
Number of registers to send	40102	10	10	
Destination HR in Standby device	40103	1	1	

Quantum backplane slot address of the network adapter module (high byte)	40104	768	768	
First byte of IP address	40105	192	192	
Second byte of IP address	40106	168	168	
Third byte of IP address	40107	0	0	
Fourth byte of IP address	40108	57	57	

These Holding Registers indicate that nine registers worth of data will be sent to the device at IP address 192.168.0.57 and stored starting at address 40001.

Data Area - Compressed Timestamping with mSecs

The Data Area used to send the sample data with compressed timestamping is stored in Holding Registers 40200 through 40208. The sample data being sent is:

	Register	Local Example	Bridge Example
Network address of sending PLC	40200	0	1030(x0406)
	40201	0	0
	40202	0	0
Address of point receiving unsolicited data	40203	30017	30017
Number of bytes of data.	40204	32770	32770
First timestamp word	40205	16384	16384
Second timestamp word	40206	10201	10201
Third timestamp word	40207	29419	29419
Fourth timestamp word	40208	22477	22477
The unsolicited value	40209	21035	21035

Holding Registers 40203 through 40209 contain a command to the CIMPLICITY Modbus TCP/IP communications enabler to set the value of a point at Input Register 17 to a value of 21035 with a date of May 14, 1997 and time 11:43:09.985.

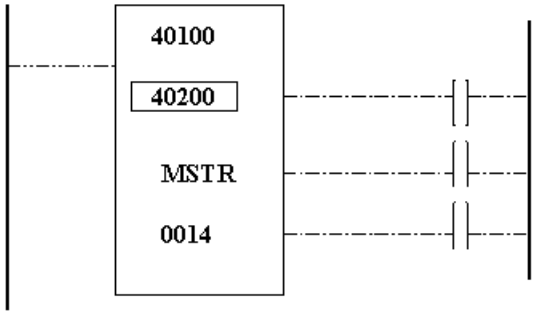
The network address of the sending programmable controller is the Modbus TCP/IP.

- For local data, the path is only 1 address long and the sending controller's IP address is put into the lower byte of the first holding register. The other two holding registers are set to zero.
- For bridged data, the path may be 5 addresses long. The example above shows the path 6.4.0.0.0.

Sample MSTR Block - Uncompressed Timestamping with mSecs

Sample MSTR Block - Uncompressed Timestamping with mSecs

The diagram below summarizes the use of the MSTR block to send unsolicited data with uncompressed timestamping to the CIMPLICITY Modbus TCP/IP communication enabler from a device.



In this example:

- The Control Block, found in Holding Registers 40100 through 40108, contains control information for the MSTR block. These indicate that nine registers worth of data will be sent to the device at IP address 192.168.0.57 and stored starting at address 40001.
- The CIMPLICITY Modbus TCP/IP Device Communications Enabler does not use address information configured in the control block. Instead, it interprets the data area as a command.
- The Data Block, found in Holding Registers 40200 through 40214, contains a command to the CIMPLICITY Modbus TCP/IP communications enabler to set the value of a point at Input Register 17 to a value of 21035 with an uncompressed timestamp.

Control Block - Uncompressed Timestamping with mSecs

The Control Block used to send the sample data with uncompressed timestamping is stored in Holding Registers 40100 through 40108. Values stored in the Control block are:

	Register	Local Example	Bridge Example	
Write Command	40100	1	1	
Status Word	40101	xxxxx	xxxxx	
Number of registers to send	40102	14	14	
Destination HR in Standby device	40103	1	1	

Quantum backplane slot address of the network adapter module (high byte)	40104	768	768	
First byte of IP address	40105	192	192	
Second byte of IP address	40106	168	168	
Third byte of IP address	40107	0	0	
Fourth byte of IP address	40108	57	57	

These Holding Registers indicate that 14 registers worth of data will be sent to the device at IP address 192.168.0.57 and stored starting at address 40001

Data Area - Uncompressed Timestamping with mSecs

The Data Area used to send the sample data with uncompressed time stamping is stored in Holding Registers 40200 through 40213. The sample data being sent is:

	Register	Local Example	Bridge Example
Network address of sending PLC	40200	0	1030(x0406)
	40201	0	0
	40202	0	0
Address of point receiving unsolicited data	40203	30017	30017
Number of bytes of data.	40204	32770	32770
First timestamp word	40205	24576	24576
Second timestamp word	40206	5	5
Third timestamp word	40207	14	14
Fourth timestamp word	40208	1997	1997
Fifth timestamp word	40209	11	11
Sixth timestamp word	40210	43	43
Seventh timestamp word	40211	09	09
Eighth timestamp word	40212	985	985
The unsolicited value	40213	21035	21035

Holding Registers 40203 through 40208 contain a command to the CIMPLICITY Modbus TCP/IP communications enabler to set the value of a point at Input Register 17 to a value of 21035 with a date of May 14, 1997 and time 11:43:09.985.

The network address of the sending programmable controller is the Modbus TCP/IP.

- For local data, the path is only 1 address long and the sending controller's IP address is put into the lower byte of the first holding register. The other two holding registers are set to zero.
- For bridged data, the path may be 5 addresses long. The example above shows the path 6.4.0.0.0.

Modbus TCP/IP Global Parameters

Modbus TCP/IP Global Parameters

If you are using the Modbus TCP/IP protocol on a computer where you will also be running the Modbus TCP/IP Server application option, you will need to add the following to the global parameters:

`MBETH_DISABLE_UNSO_DATA|1|Y`

This global parameter disables the use of the socket for the receipt of unsolicited data in the Modbus TCP/IP device communications.

Modbus Ethernet Bridge Behavior

Not all Modbus Ethernet Bridges behave the same when a request to a device that is not available through the bridge is made.

A bridge may:

- Respond and report that the device is not present.
- Not respond at all.

The following global parameter is available to define bridge response for a TCP/IP address.

`MBETH_NORESP_xxx.xxx.xxx.xxx` (where xxx.xxx.xxx.xxx is the IP address)

For	A Modbus Ethernet bridge
Purpose	Should the device communication interface request data from a device communicating through a bridge at the given TCP/IP address and no response is received, the software will declare only the device queried as down. Without the global defined as shown, all devices connected to the bridge will be declared down.
Value	Y

Important: When this global parameter is enabled, if the bridge is off, it will take longer for the device communication software to detect that all devices communicating through the bridge are

down. This is because the software cannot distinguish bridge down scenarios from single device down scenarios because of bridge functionality.

Unsolicited Communication

Unsolicited Communication

- Enable/disable unsolicited data.
- Unsolicited data queue size.
- Asynchronous connection.
- Request Millisecond Time out.
- DEVICE_TIMESTAMP_UTC
- <PORT>_DEVICE_TIMESTAMP_UTC

Enable/Disable Unsolicited Data

The Modbus TCP/IP device communication interface is designed so that only one Modbus Ethernet device communication port per computer should be processing the unsolicited data.

By default, the ability to process unsolicited data is enabled on all Modbus Ethernet ports.

The following global parameters are available to modify the default behavior.

`MBETH_DISABLE_UNSO_DATA`

For	Project
Purpose	To disable unsolicited data.
Value	Y
Default Value	N

<PORT>`_DISABLE_UNSO_DATA`

Where <PORT> is the number of the port.

For	Port
Purpose	To disable unsolicited communication on a single Modbus Ethernet port.
Value	Y
Default Value	N

Unsolicited Data Queue Size

Unsolicited data from devices is queued by each port for processing.

The default queue size is 16 messages.

The following global parameters can modify the default behavior.

`MBETH_UNSO_DATA_QUEUE_SIZE`

For	Project
Purpose	To set the value of the maximum number of messages to queue.
Value	An integer greater than 0.
Default Value	16

<PORT>`_UNSO_DATA_QUEUE_SIZE`

Where <PORT> is the number of the port.

For	Port
Purpose	To set the value of the maximum number of messages to queue for a single port.
Value	An integer greater than 0.
Default Value	16

Asynchronous Connection

The IO connection is synchronous, by default.

The following global parameter can modify the default behavior of cyclic delays for unsolicited communication and allow you to indicate that an asynchronous connection is requested.

`MBETH_ASYNC_CONNECTION`

For	Project
Purpose	To switch between synchronous and asynchronous connections.
Value	Y
Default Value	N

<PORT>`_ASYNC_CONNECTION`

Where <PORT> is the number of the port.

For	Port
Purpose	To switch between synchronous and asynchronous connections on a single port.
Value	Y
Default Value	N

Note: There are Modbus TCP/IP bridges that do not provide a response as defined by the Modbus TCP/IP protocol when the queried device is not present. If such devices are configured on a CIMPLICITY port, use a synchronous connection on the port. Ports with devices utilizing such bridges are not supported in asynchronous mode.

Request Millisecond Timeout

You can configure a 1 second or greater timeout value for the driver.

The following global parameter can modify the default behavior and allow you to enter a millisecond timeout value to reduce the waiting time for a block call in the Modbus TCP/IP device communications.

The use of synchronous communications is forced when either the MBETH_NORESP_ipaddress or <DeviceID>_CONSERVES_CONN global parameter is enabled.

Time-outs may now be specified down to the millisecond level by configuring a project and/or port level global parameters. These parameters work in conjunction with the pre-existing time-out parameters.

`MBETH_REQ_MILLISECOND_TIMEOUT`

For	Project
Purpose	To request a timeout of less than 1 second.
Value	0 - 32767 ms
Default Value	0 ms

<PORT>`_REQ_MILLISECOND_TIMEOUT`

Where <PORT> is the number of the port.

For	Port
Purpose	To request a timeout of less than 1 second for a single port.
Value	0 - 32767 ms
Default Value	0 ms

Cache Size

The following global parameters enable the cache size to be configurable or to be disabled.

<PORT>`_MAX_BUFFER_SIZE`

For	Project

Purpose	To allow the cache size to be configurable to a value smaller than defined by the default.
Comment	The <PORT>_MAX_BUFFER_SIZE value will override the default buffer size for all devices on the port.
Value	Integer greater than 2.

MBEDC_<device_id>

For	Project	
Purpose	To allow the cache size to be disabled	
Value	Values are:	
	Y	Disables caching.
	N	Enables caching
	Default	N
Important	For STAT_PLC models, the caching may be disabled in the alternate configuration file with the parameter DisableCaching. If DisableCaching is specified in the alternate configuration file, the alternate configuration file will take precedence over the global parameter. DisableCaching values:	
	1	Disables caching.
	0	Enables caching.
	Default	0

Enable/Disable Protocol Level Diagnostics

Protocol level diagnostics can be logged to the Modbus Ethernet's port file.

By default, diagnostics logging is disabled.

The following global parameters are available to specify diagnostic behavior.

MBETH_ENABLE_PROTOCOL_DEBUG

For	Project
Purpose	To enable debugging for all Modbus Ethernet interfaces.
Value	Y
Default Value	N

<PORT>_ENABLE_PROTOCOL_DEBUG

Where <PORT> is the number of the port.

For	Port
Purpose	Enable protocol level debugging on a single port.

Value	Y
Default Value	N

Important: Protocol information is logged in a HEX representation. All data is flushed to the file when the communication interface terminates. Prior to this time, all data may not have been flushed to the disk.

`MBETH_DISABLE_IO_ERRLOG`

For	Project
Purpose	To disable logging of I/O errors for all Modbus Ethernet ports
Value	Y
Default Value	**N**

<PORT>`_DISABLE_IO_ERRLOG`

Where <PORT> is the number of the port.

For	Port
Purpose	To disable logging of I/O errors for all Modbus Ethernet ports
Value	Y
Default Value	**N**

Read Request Retry

The following global parameter is used to set the number of times a read request is repeated during device initialization.

The following global parameter is available to set read requests.

`MBETH_REQ_RETRY`

For	Project
Purpose	To set the number of times a read request is repeated during device initialization (domain sizing routines).
Value	n (where n is a number greater than or equal to zero)
Default Value	1

Read Request Timeout

The following global parameter is used to override the default timeout setting of 5 seconds. This global parameter can be set on a project or port basis.

The following global parameters are available to set read requests.

MBETH_READ_REQUEST_TIMEOUT
<PORT>_READ_REQUEST_TIMEOUT

MBETH_REQ_TIMEOUT

For	Project
Purpose	To override the project's default timeout setting.
Value	n (where n is a number greater than or equal to zero)
Default Value	5

<PORT> _REQ_TIMEOUT

Where <PORT> is the name of the port.

For	Port
Purpose	To override the port's default timeout setting.
Value	n (where n is a number greater than or equal to zero)
Default Value	5

Socket Ports

Socket Ports

Users can now configure the Modbus TCP/IP protocol to use a configurable socket port instead of the default port 502. Note that all devices using the CIMPLICITY port will use the same socket port.

There are now two global parameters available that enable a system manager to specify the Modbus TCP/IP port number.

MBETH_SOCKET_PORT

For	Project
Purpose	To reset the default socket port for Modbus TCP/IP from 502 to the designated value.
Comment	PORT_SOCKET_PORT takes precedence over MBETH_SOCKET_PORT.
Value	Port number that should be the new default

For	Project
Default Value	502

<PORT>_SOCKET_PORT

For	Project
Purpose	To reset the default socket port for Modbus TCP/IP from 502 to the designated value when multiple Modbus devices are available. **Note:** <PORT> refers to the name of the Modbus TCP/IP port to be configured, such as MBTCP0, MBTCP1, etc. For example, for the MBTCP0 port, the global parameter would be MBTCP0_SOCKET_PORT.
Comment	PORT_SOCKET_PORT takes precedence over MBETH_SOCKET_PORT.
Value	Port number that should be the new default
Default Value	502

Timeout Retry Delay

The following global parameter is used to set the length of time, in ticks (i.e. 0.01 seconds per tick), to pause when a read request fails before sending the request again. This global parameter can be set on a project or port basis.

The following global parameters are available to set timeout retry delays.

MBETH_TIMEOUT_RETRY_DELAY
<PORT>_TIMEOUT_RETRY_DELAY

MBETH_TIMEOUT_RETRY_DELAY

For	Project
Purpose	To set the length of time, in ticks, to pause when a read request fails before sending the request again.
Value	n (where n is a number greater than or equal to zero)
Default Value	0

<PORT> _TIMEOUT_RETRY_DELAY

Where <PORT> is the name of the port.

For	Port

Purpose	To set the length of time, in ticks, to pause when a read request fails before sending the request again.
Value	n (where n is a number greater than or equal to zero)
Default Value	0

Additional Global Parameters

The following global variables will supersede default values and the values defined in the .ini file unless otherwise noted:

Writing Single Coils

Some devices use the Force Single Coil command (function code 5) for writing single coils.

Other devices support (sometimes exclusively) the Force Multiple Coils (function code 15) when writing coils.

You can configure the Modbus Ethernet communication interface to use a specific function code when writing single coils and when interacting with a particular device.

`DeviceId_ ONE_COIL_WRITE`

For	Project	
Purpose	To control the behavior, for writing single coils on a specific device	
Value	Y	Will cause function code 5 to be used.
	N	Will cause function code 15 to be used.
Comment	For device models Modicon 484, VersaMax ENIU and VersaPoint ENIUs, single coils writes will be selected regardless of the user-configured preferences.	

Writing Single Holding Registers

Some devices use the Force Single Holding Registers command (function code 6) to write a single holding register.

Other devices support (sometimes exclusively) the Force Multiple Holding Registers Command (Function Code 16).

`DeviceId_ ONE_REG_WRITE`

For	Project	
Purpose	To control the behavior, for writing single holding registers on a specific device	
Value	Y	Will cause function code 6 to be used.

	N	Will cause function code 16 to be used.
Comment	For device models Modicon 484, VersaMax ENIU and VersaPoint ENIUs, when performing a request to write a single register, single register writes will be selected regardless of the user configured preferences.	

Connection Conservation

Some devices will terminate a connection if there is no activity for some pre-configured interval.

`DeviceId _CONSERVES_CONN`

For	Project	
Purpose:	To re-define how the device communication interface handles a disconnect as it relates to device down for some devices that terminate a connection if there is no activity for some pre-configured interval.	
Value:	Y	The device will not be declared down unless it is unable to re-establish the connection and perform the next immediate I/O operation to request or modify data on the device.
Comment	For device model VersaPoint ENIU, the device communication interface will perform specific processing that overrides the behavior of this setting.	

Host Redundancy with Devices with One Connection

Some devices will support only one connection.

In host redundant environments, it is necessary to identify the devices with this characteristic so that the connection can be terminated on the acting secondary and initiated on the primary following a transition in the host redundancy roles.

Note that a device down will not be generated as a result of not having a connection with the secondary. To identify a device that supports only a single connection, define the following project level global parameter.

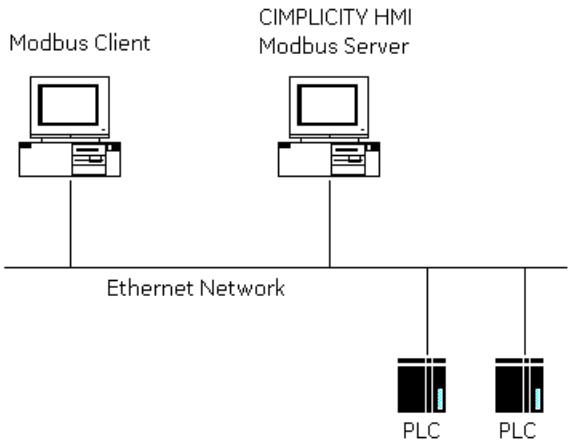
`DeviceId_CONN_SECONDARY`

For	Project	
Purpose	To identify a device that supports only a single connection.	
Value	N	

Chapter 43. Modbus TCP Server Communications

About Modbus TCP Server Communications

The Modbus TCP Server Communications option supports a multi-drop configuration with the Modbus TCP Server Communications enabler functioning as a server. In effect, the Modbus TCP Server Communications enabler presents the CIMPLICITY Points as if they were data values from one or more Modbus TCP Server devices.



The CIMPLICITY Points are mapped to Modbus Addresses, allowing a Modbus Client to obtain point values as if they were coils or registers on a server device.

SystemLink uses the Modbus TCP Server feature to enable CIMPLICITY to collect point values over the Internet using the Modbus protocol. The node that is set up as the client can collect data from the server nodes via the Internet.

Important: The Modbus TCP Server is not supported in a host redundant environment.

Modbus TCP Server Specifications

Modbus TCP Server Specifications

Specifications for Modbus/TCP Server include:

- CIMPLICITY Specifications for Modbus TCP Server.

- Protocol.

CIMPLICITY Specifications for Modbus TCP Server

Supported Data Types

This communications option supports the following data types:

- Boolean
- Signed 8, 16, and 32 bit integers
- Unsigned 8, 16, and 32 bit integers
- Floating point
- Arrays of the above data types

Modbus TCP Server Protocol

Modbus TCP Server Protocol

See protocols that Modbus/TCP Server supports for:

- Transmission mode.
- Message framing.
- Server device emulation.
- Memory types.
- Function codes.

Transmission Mode Supported by Modbus TCP Server

The Modbus TCP Server Communications option supports the RTU transmission mode.

Message Framing Supported by Modbus TCP Server

The Modbus TCP Server Communications option supports RTU framing. RTU framing implies the following communication parameters:

- Data bits: 8
- Start bits: 1
- Parity: Even, Odd, or None
- Stop bits: 1 or 2
- Error checking method: CRC

Modbus TCP Server Device Emulation

The Modbus TCP Server Communications option emulates one or more generic Modicon programmable controllers. Valid addresses: 0 to 128.

Memory Types Supported by Modbus TCP Server

The Modbus TCP Server Communications option emulates the Modbus memory types based on Modicon programmable controllers.

The Modbus TCP Communications option supports writes to the following memory types:

Memory Type	Reference Class	Description
Coils	**0X**	Discrete Outputs
Holding Registers	**4X**	Analog Outputs

The Modbus TCP Server Communications option supports reads from the following memory types:

Memory Type	Reference Class	Description
Coils	**0X**	Discrete Outputs
Inputs	**1X**	Discrete Inputs
Input Registers	**3X**	Analog Inputs
Holding Registers	**4X**	Analog Outputs

Function Codes Supported by Modbus TCP Server

The Modbus TCP Server Communications option supports the Modbus function codes listed in the following table. As well, the maximum number of coils or registers that may be specified or included in a single Modbus message is also given.

Function Code	Description	Maximum # of coils / registers
1	Read Coil Status	2000
2	Read Input Status	2000
3	Read Holding Registers	125
4	Read Input Registers	125
5	Force Single Coil	1
6	Force Single Register	1
15 (0x0F)	Force Multiple Coils	800

Function Code	Description	Maximum # of coils / registers
16 (0x10)	Preset Multiple Registers	100

Modbus TCP Server Required Documents

The Modbus TCP Server Communications enabler adheres to the Modbus protocol specification as defined in the following document:

Modicon Modbus Protocol Reference Guide (PI-MBUS-300), Rev. J, June 1996

Modbus TCP Server Getting Started Steps

Modbus TCP Server Getting Started Steps

The CIMPLICITY Modbus TCP Server Interface provides a way to access CIMPLICITY points as Modbus coils and/or registers from Modbus Client devices. This section guides a user through the steps to become familiar with the Modbus TCP Server Interface application and to start using the application quickly.

The following getting started guide assumes that the Modbus TCP Server Interface has been successfully installed and that at least one CIMPLICITY project has been created.

Steps include:

Step 1 *(page 714)*	Identify CIMPLICITY Points.
Step 2 *(page 714)*	Map points to Modbus data addresses.
Step 3 *(page 714)*	Select the Modbus TCP Server protocol.
Step 4 *(page 714)*	Start the CIMPLICITY project.
Step 5 *(page 715)*	Start the Modbus Client.

Step 6 *(page 715)*	Validate communications.

Step 1. Identify CIMPLICITY Points

Identify the CIMPLICITY points you wish to access from a Modbus Client device. If necessary, create device or virtual points to receive values from a Modbus Client device. Create virtual points to re-range analog point values to prevent data coercion errors, or to minimize loss of significance. For example, real values are truncated when transferred to a Modbus analog data address.

Step 2. Map Points to Modbus Data Addresses

Map (link) CIMPLICITY points to Modbus data addresses by creating register-mapping Register Mapping Records *(page 718)* definitions within the configuration file.

todo: To maximize throughput:

1. Map:
   - Boolean points to coil (0X) or input status (1X) data addresses.
   - Analog points to input register (3X) or holding register (4X) data addresses.

   Avoid gaps *(page 719)* in allocated data addresses.

2. Edit the configuration file to specify the
   - Communications port parameters,
   - Trace log parameters, and
   - Modbus register limits.

Step 3. Select the Modbus TCP Server Option

Review:

- Steps to get Modbus/TCP Server started.
- Review the procedure to add the Modbus/TCP Server interface to your CIMPLICITY project.

Step 4. Start the CIMPLICITY Project

You start the associated CIMPLICITY project before you start the Modbus TCP Server Interface.

todo: To start the associated CIMPLICITY project:

1. Load the project from the CIMPLICITY Workbench application.

2. Either:
   a. Select Run from the Workbench Project pull-down menu, or
   b. Click the **Run** button on the Workbench toolbar.

3. Check the list of started resident processes for the name **MTCPSI_RP**.

Step 5. Start the Modbus Client

At this point, the Modbus TCP Server Interface is ready to process Modbus Client requests. Start the Modbus Client device according to the documentation provided with the device.

Step 6. Validate Communications

CIMPLICITY provides a number of diagnostic tools to validate communications between CIMPLICITY and the Modbus Client. For example, use the CIMPLICITY Point Control Panel to display dynamic changes to CIMPLICITY points.

Modbus TCP Server Configuration Overview Steps

Modbus TCP Server Configuration Overview Steps

This section describes the configuration process for the Modbus TCP Server Interface. To configure the Modbus TCP Server Interface, follow these steps:

Step 1 *(page 716)*	Edit the configuration file.
Step 2 *(page 720)*	Select the Modbus TCP Server protocol.
Step 3 *(page 721)*	Implement the Modbus TCP Server configuration.

Step 1. Edit the Modbus TCP Server Configuration File

Step 1. Edit the Modbus TCP Server Configuration File

When a new CIMPLICITY project is created, a template configuration file is created for the Modbus TCP Server Interface. The file has a comma-separated value (CSV) format similar to the format used by the CIMPLICITY import/export utility.

Modify the configuration file to specify the:

- Communications port parameters.
- Trace log parameters.
- Modbus register limits.
- Point / register mappings.

Information about:

- File naming convention.
- Types of Modbus/TCP Server configuration records.

File Naming Convention

The Modbus TCP Server Interface determines the file specification of the configuration file at startup by combining the file name **MTCPSI_RP.CFG** with the file path of the **project\data** directory.

Example

If the project has been created in the directory

C:\Program Files\Proficy\Proficy CIMPLICITY\Projects\Data\

Then the configuration file specification is:

C:\Program Files\Proficy\Proficy CIMPLICITY\Projects\data\MTCPSI_RP.cfg.

Types of Modbus TCP Server Configuration Records

Types of Modbus TCP Server Configuration Records

The configuration file may contain the following types of records:

	IP address parameters	Defines the serial communications port and settings.

	Trace log parameters	Specifies state (on/off) of protocol stack diagnostics logging.
	Modbus register limits	Defines the range of valid values for a Modbus analog register (memory type 3X or 4X).
	Register mapping	Defines a mapping between a CIMPLICITY point and a Modbus data address.
	Comments	Any line beginning with one or more pound signs (#). Ignored by the Modbus/TCP Server Interface.
	Blank lines	Ignored by the Modbus/TCP Server Interface.

IP Address Parameters Record

The configuration file must contain a IP Address Parameters record. In addition, all fields must be present.

The format for the IP Address Parameters record is:

IP, 0.0.0.0 Picks up the first configured IP address.

IP < address>

Where

IP	Identifies the record as an IP address object definition.
<address**>**	IP address in dotted-decimal notation.

Trace Log Parameters Record

The configuration file may contain a Trace Log Parameters record.

The format for the Trace Log Parameters record is:

TF ,<logging state>

Where

TF	Identifies the record as a trace log parameters record.
<logging state**>**	Protocol stack logging state. Valid settings: **On**, **Off**. Default value: **Off**.

Note: If the Trace Log Parameters record is not defined, the protocol stack logging state is **Off**.

Trace log messages are written to the standard error file associated with the Modbus TCP Server Interface process.

- Using the Protocol Stack trace log.

Modbus Register Limits Record

The configuration file may contain a Modbus Register Limits record. These limits are applied only to CIMPLICITY points mapped to analog registers (register type 3X or 4X).

The format for the Modbus Register Limits record is:

RL ,<low limit>**,**<high limit>

Where

RL	Identifies the record as a Modbus register limits record.
<low limit**>**	The minimum value accepted from a CIMPLICITY point. Valid range of values: – 2147483648 to 2147483647. Must be less than **<**high limit**>**. Default value: – 2147483648.
<high limit**>**	The maximum value accepted from a CIMPLICITY point. Valid range of values: – 2147483648 to 2147483647. Must be greater than <low limit>. Default value: 2147483647.

Note: If the Modbus Register Limits record is not defined the low and high limits are between – 2147483648 to 2147483647, respectively.

Register Mapping Records

The configuration file must contain one or more Register Mapping records. In each Register Mapping record, all fields must be present.

The format for the Register Mapping record is:

RM ,<server>**,**<register>**,**<point id>

Where

RM	Identifies the record as a register mapping record.
<server**>**	The emulated server device address. Valid range of values: 0 to 128.
<register**>**	One-based Modbus address including the memory type prefix. See the following table for valid range of values:

	Memory Type	Address range
	0x - Coils (Discrete)	**1 to 9999**
	1x - Inputs (Discrete)	**10001** to **19999**
	3x - Input Registers (Analog)	**30001** to **39999**
	4x - Holding Registers (Analog)	**40001** to **49999**

<point id**>**	String that uniquely references a point in a CIMPLICITY project. Must specify a non text attribute (i.e. analog or Boolean).

Sample Configuration File

The following is a sample configuration file specifying register mappings for two server devices (address 3 and 7).

```
#
# MTCPSI - sample configuration file - Single Unit ID
# - note that all blank lines and comment lines (start with '#') are
 ignored by MTCPSI
#
# IP Address definition:
IP 155.177.177.102
#
# Trace log flag definition:
TF,Off
#
# Modbus register limits definition:
RL, – 2147483648, 2147483647
#
# Register mapping definitions:
#
# Unit ID: 1
# - the following register mapping definitions allocate the first two
 registers of each register type
# Discrete outputs
RM,1,1,TEST_POINT_DOUT_A
RM,1,2,TEST_POINT_DOUT_B
# Discrete inputs
RM,1,10001,TEST_POINT_DIN_A
RM,1,10002,TEST_POINT_DIN_B
# Analog inputs
RM,1,30001,TEST_POINT_AIN_A
RM,1,30002,TEST_POINT_AIN_A
# Analog outputs
RM,1,40001,TEST_POINT_AOUT_A
RM,1,40002,TEST_POINT_AOUT_B
```

Register Mappings Guidelines

This section provides the user with guidelines for defining register mappings.

Guideline to Organize Records

The register mappings (and other types of records) may be specified in any order. To simplify maintenance of the configuration file, order the register mapping records by server address then register reference. This will aid in detecting unmapped data addresses.

Since the Modbus TCP Server Interface can emulate multiple Modbus TCP Server devices, one could organize mapped CIMPLICITY points into logical groupings based on a functional area.

Guideline to Optimize Throughput

The Modbus TCP Server Interface supports several Modbus requests that allow a range of data addresses (or an array of values) to be specified. Note that these requests specify a contiguous set of data addresses.

To maximize throughput, avoid gaps in allocated data addresses. This will reduce the transfer of "meaningless" values.

In addition, map Boolean points to coil (0X) or input status (1X) data addresses. Modbus requests that handle coil or input status data addresses pack sixteen (16) values into one word (two bytes). Modbus requests that handle input register or holding register data addresses require one word per value.

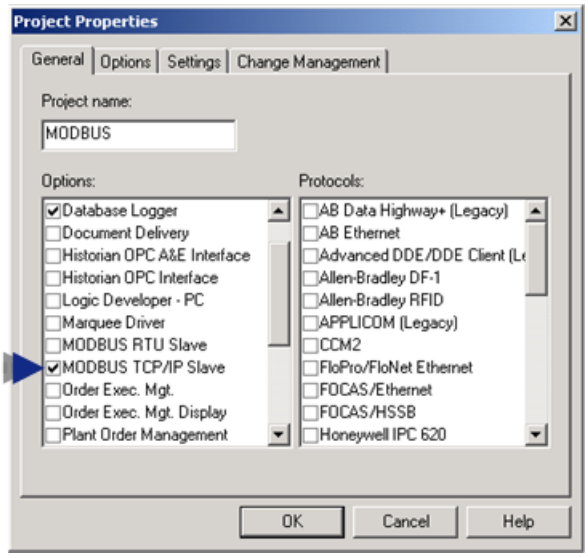
Another way to maximize throughput is to allow the Modbus client to reduce the number of transactions by increasing the range of data addresses in each request.

Step 2. Select the Modbus TCP Server Protocol

1. Load an existing project (or create a new project) in the CIMPLICITY Workbench.
2. Click Project on the Workbench menu bar.
3. Select Properties.

   The Project Properties dialog box opens.

4. Select the General tab.
5. Check Modbus TCP Server in the Options box.



6. Click OK to save the changes.

Step 3. Implement the Modbus TCP Server Configuration

Step 3. Implement the Modbus TCP Server Configuration

To apply Modbus TCP Server configuration changes, start the associated CIMPLICITY project. If the project is currently running, stop then restart the Modbus TCP Server resident process.

The start order for implementing the Modbus TCP Server configuration is:

- Start the associated CIMPLICITY project.

Start the Associated CIMPLICITY Project

You need to start the associated CIMPLICITY project in order to start the Modbus TCP Server Interface.

todo: To start the associated CIMPLICITY project:

1. Open the associated project in the CIMPLICITY Workbench.
2. Either:

   Method 1

   a. Click Project on the Workbench menu bar.
   b. Select Run.

Method 2

Click the **Run** button on the Workbench toolbar.

3. Check the list of started resident processes for the name **MTCPSI_RP**.

Modbus TCP Server Troubleshooting

Modbus TCP Server Troubleshooting

There are several diagnostic tools available to the user. These tools are particularly valuable if runtime behavior deviates from expected behavior. They can help isolate and troubleshoot the erroneous or erratic behavior of a working system.

Diagnostic tools include the following:

- CIMPLICITY status log.
- Protocol stack trace log.
- Modbus exception codes.
- Test program.

Expected Modbus TCP Server Runtime Behavior

Expected Modbus TCP Server Runtime Behavior

This section describes the expected behavior of the Modbus TCP Server Interface. This description provides the user with a reference point for using the diagnostic tools.

Expected behavior for:

- Data address space.
- Data types.
- Data flow.
- Data coercion.
- Write response time.
- Dynamic CIMPLICITY configuration changes.

Data Address Space

The Modbus TCP Server Interface maintains a data address space that mimics one or more PLCs and their associated coils and/or registers.

The user configuration will implicitly define the limits of valid data addresses for each unique combination of memory type and server address. The minimum data address for any given memory type is 0 (zero-based). The maximum data address for a given memory type is the highest data address specified in the configuration. For example, if the highest holding register in server address 1 is 40150 (one-based), then the maximum data address for memory type 4X is 149 (zero-based).

Tip: All data addresses in the address space are set to binary **0** on startup. If a data address is not mapped to a CIMPLICITY point, then the following rules apply:

1. If the data address is a read-write address (memory type 0X or 4X), then a Modbus client may update (or read) the data address value. Thus, the data address may act as a memory store.
2. If the data address is a read-only address (memory type 1X or 3X), then a Modbus client may only read the data address. Thus, the data address value is always binary zero (0).

   Modbus client data address references outside the configured range (i.e. greater than the highest user-specified data address for a given memory type and server address) shall generate an exception response *(page 729)* .

Data Types

The Modbus TCP Server Interface supports mapping of all non-text (i.e. analog or Boolean) CIMPLICITY points to Modbus data addresses. Non-array points are mapped to a single data address as specified in the register mapping definition. Array points and bitstring points are mapped to multiple consecutive data addresses, with the first element mapped to the register specified in the register mapping definition.

Data Flow

To maintain the current value of each mapped CIMPLICITY point within the data address space, the Modbus TCP Server Interface will accept unsolicited updates (from CIMPLICITY) for all mapped points. Thus, the Modbus Client may read values from CIMPLICITY points mapped to any supported memory type (0X, 1X, 3X, and 4X).

However, the Modbus TCP Server Interface will only attempt to write to CIMPLICITY points that are mapped to data addresses of memory type 0X or 4X. Thus, the Modbus Client may write values to CIMPLICITY points mapped to data addresses of memory type 0X or 4X.

Note that for a Modbus Client write request to succeed, all data addresses specified must be valid and all corresponding values must be written successfully to PTMAP.

Data Coercion

The Modbus TCP Server Interface performs run-time coercion for all data types.

Discrete data types are coerced according to the following guidelines.

Memory Type	Data Source	Source Value	Target Value
0X	CIMPLICITY	**zero (0)**	**0**
		CIMPLICITY	**non-zero (!= 0)**
		Modbus Client	**0**
		Modbus Client	**0xFF or 0x01 (depends on function code)**
1X	CIMPLICITY	**zero (0)**	**0**
		CIMPLICITY	**non-zero (!= 0)**

guide: Guidelines. Analog data types are coerced according to the following guidelines.

1. For analog points with engineering units, the Modbus TCP Server Interface will use the engineering unit value when converting to/from Modbus data values.
2. For analog points without engineering units, the Modbus TCP Server Interface will transfer the raw value.
3. The Modbus TCP Server Interface will reject any CIMPLICITY point value outside of the range *(page 718)* defined by the Modbus Register Limits.

Write Response Time

For a given Modbus Client write request, the Modbus TCP Server Interface will wait for all set point requests to complete before returning a response to the Modbus Client. Since the set point requests are non-deterministic, no response time can be guaranteed for a write request. The Modbus Client should be prepared to adjust its timeout parameter accordingly.

Dynamic CIMPLICITY Configuration Changes

CIMPLICITY supports reconfiguration of point properties while the project is running. The point properties that directly impact the behavior of the Modbus TCP Server Interface include the following:

- Access rights
- Data type
- Array size

The Modbus TCP Server Interface monitors the project database for changes in point properties. Once notified, point property changes are applied immediately.

Example

Changing the access rights for a point from read-write to read-only will cause Modbus write requests (that specify the corresponding data address) to fail.

In particular, changes to the array size of a point may have unexpected results. At startup, the Modbus TCP Server Interface determines the array size for every mapped point. By definition, a scalar point has an array size of 1. In combination with other tests, the Modbus TCP Server Interface uses the array size to ensure that every array element is mapped to a unique Modbus data address. The array size determined at startup becomes the maximum size allowed. Thus, while the array size may be reduced, the Modbus TCP Server Interface ignores increases.

CIMPLICITY Status Log

CIMPLICITY Status Log

If an error is detected, the Modbus TCP Server interface logs:

- Modbus TCP Server Interface startup error messages.
- Modbus TCP Server Interface runtime error messages.

Modbus TCP Server Interface Startup Error Messages

On startup, the Modbus TCP Server Interface Modbus TCP Server Configuration Overview *(page 715)* validates the contents of the configuration file. If an error is detected, an error message is logged. These error messages include the following:

Message	Cause / Resolution
Error opening configuration file (see next message).	The configuration file could not be opened. The following message provides more details. Ensure that a valid configuration file *(page 716)* exists in the required directory. Ensure that another application is not currently using the file.
Error reading from configuration file.	The configuration file could not be read. Check the file for invalid control characters. If necessary, replace the file with a backup copy, or rebuild the file using the provided template.
Configuration file contains more than one IP address parameters record.	Ensure that the configuration file contains only one record with the prefix "IP".

Configuration file does not contain any IP address parameters record.	The configuration file must contain a record with the prefix "IP".
Configuration file contains more than one trace log parameters record.	Ensure that the configuration file contains only one record with the prefix "TF".
Configuration file contains more than one Modbus register limits record.	Ensure that the configuration file contains only one record with the prefix "RL".
Configuration file does not contain any register mapping record.	The configuration file must contain at least one register mapping record (prefix "RM").
Configuration file contains the following invalid record (see next message).	Detected a record that is not a comment nor a blank line nor a valid definition. The following message contains the record. Ensure that the record starts with a valid record prefix or a valid comment character.
<record type> record is invalid.	For the given type of record, some or all of the required fields are missing, or are not properly delimited.
<record type> record contains an invalid <field name> field.	For the given type of record, the given field is not defined or contains invalid characters.
<record type> record specifies an invalid <field name>.	For the given type of record, the given field specifies an invalid parameter value (for example, the value is out of range).
Modbus register limits record specifies an invalid pair of limits.	The low register limit equals or exceeds the high register limit.

Duplicate register reference detected (see next two messages).	Detected two register mapping records that specify the same Modbus data address (server address and register reference). The next two messages give the register mapping parameters (and array size) of the current and duplicate records. If the two records do not specify the same Modbus data address, check the array size of the CIMPLICITY point specified in the Duplicate register mapping and ensure that sufficient data addresses are reserved.
Duplicate point identifier detected (see next two messages).	Detected two register-mapping records that specify the same CIMPLICITY point. The next two messages give the register mapping parameters (and array size) of the current and duplicate records.
Unsupported data type for point: <point id>.	The data type of the given CIMPLICITY point is not supported *(page 711)*. .

Modbus TCP Server Interface Runtime Error Messages

The Modbus TCP Server Interface logs informational, warning, and error messages to the CIMPLICITY project status log. These messages include the following:

Message	Cause / Resolution
Program startup completed successfully	Informational message - no action required.
Program shutdown completed successfully	Informational message - no action required.
Memory allocation error	Error creating an internal data structure. Increase available memory by stopping non-essential applications.
Error opening communication port channel: x.	Error occurred while connecting to the given COM port. Ensure that the given COM port is valid.
Error coercing Modbus value to CIMPLICITY point value (see next message).	The value of the given CIMPLICITY point cannot be converted to the mapped Modbus memory type. Ensure that the value is within the Modbus Register Limits *(page 718)*. Ensure that the data type is supported *(page 711)*. .
On-change response coercion error for point: <point id>.	(see the previous message)

Important: The Modbus TCP Server Interface also logs messages intended for use by GE Intelligent Platforms support personnel. If any of the following messages are logged, please contact your CIMPLICITY vendor for support.

- Error initializing protocol layer object: xxx.
- Error registering server addresses in protocol layer object: xxx
- Thread creation failed.
- Error occurred while waiting for thread to exit.
- Event object creation failed.
- Error occurred while setting event object.
- Failed to allocate an event flag from PTMAP subsystem.

- Error occurred while waiting for a response from the PIL worker thread.
- An invalid PTMAP response message was received. Response ignored.
- Invalid register count: xxx.
- Duplicate entry detected for server address xxx, register class xxx.

Protocol Stack Trace Log

The protocol stack trace log provides the user with an indication of the Modbus messages received and sent by the Modbus TCP Server Interface at the communications port level. Each logged message is time stamped.

The user may enable or disable trace logging *(page 717)* by defining a trace log parameters record within the Modbus TCP Server Interface configuration file..

When enabled, the Modbus TCP Server Interface writes protocol stack trace messages to the file **MTCPSI.err** in the **log** sub-directory of the Project root directory.

Example

If the Project had been created in the directory **C:\Program Files\Proficy\Proficy CIMPLICITY \Projects\test** , then the trace message file specification is **C:\Program Files\Proficy\Proficy CIMPLICITY \Projects\test\log\mtcpsi_rp.err** .

The following is an excerpt from an actual trace log file.

```
1999-11-09 16:35:34.876 Started Modbus TCP Server Interface (MTCPSI_RP) –
 Using IP address 0.0.0.0
1999-11-09 16:37:29.010 Accepted connection [id = 524] from IP address
 155.177.177.102 – bound to local IP address 155.177.177.102
1999-11-09 16:37:36.250 <- [id = 524] 000100000006 010300000064
1999-11-09 16:37:36.290 -> [id = 524] 0001000000CB
0103C8000100020003000400050006000700080009000A000B000C000D000E000F001000110012001300140015
.
The above will be repeated several times. It represents the Read Holding
 Registers transaction
.
1999-11-09 16:40:11.503 <- [id = 524] 0001000000CF
011000630064C8007F007F007F007F007F007F007F007F007F007F007F007F007F007F007F007F007F007F007F00
1999-11-09 16:40:11.674 <- [id = 524] 000100000006 011000630064
.
The above will be repeated several times. It represents the Read Holding
 Registers transaction
.
1999-11-09 16:40:18.373 Closing connection [id = 524] with 1p address
 155.177.177.102
```

The format for a trace log message is:

<date> <**time**> <direction> <server address><function code> <data> <CRC>

Where:

<date**>**	is \the date the message was received or sent.
<time**>**	is the time the message was received or sent.
<direction**>**	is an arrow indicating whether the message is a request or response message. Request (left arrow): **<-** Response (right arrow): **->**

Note: the remaining fields are displayed as hex values (two characters per byte).

<server address**>**	is the emulated server device address.
<function code**>**	is one byte representing the requested operation. If an exception response is generated, the high bit is set.
<data**>**	is one or more bytes containing function parameters, data values, and/or exception code.
<CRC**>**	is trailing two bytes containing the cyclical redundancy check value.

Modbus Exception Codes

The user may diagnose errors by examining response messages returned by the Modbus TCP Server Interface to the Modbus Client. Certain conditions will cause the Modbus TCP Server Interface to return exception responses. This section documents the supported Modbus exception codes and their possible causes.

Exception Code	Cause
0x01 (Illegal Function)	Modbus request specifies an unsupported function code.
0x02 (Illegal Data Address)	Modbus request specifies an invalid data address, or specified a data address range that extends beyond the highest configured data address.
0x03 (Illegal Data Value)	Modbus write request specifies an analog data value that exceeds the data type range or the configured EU limits of the associated CIMPLICITY Point.
0x04 (Server Device Failure)	Modbus write request specifies a data address mapped to a disabled, unavailable, or read-only CIMPLICITY Point.

The format for a Modbus exception response message is:

<server address><function code><exception><CRC>

Where

<server address**>**	is the emulated server device address.
<function code**>**	is one byte representing the failed operation. Note that the high bit is set.

<exception**>**	is one byte representing the exception code. See table above.
<CRC**>**	is trailing two bytes containing the cyclical redundancy check value.

Modbus TCP Server Test Program

CIMPLICITY provides a sample Modbus Client test program. The test program generates Modbus requests and displays the received responses.

For more information regarding the test program, refer to the CIMPLICITY Device Communications Manual (GFK1181G), Modbus TCP Communications, Validating Modbus TCP Communications.

Chapter 44. OMRON Host Link Communications

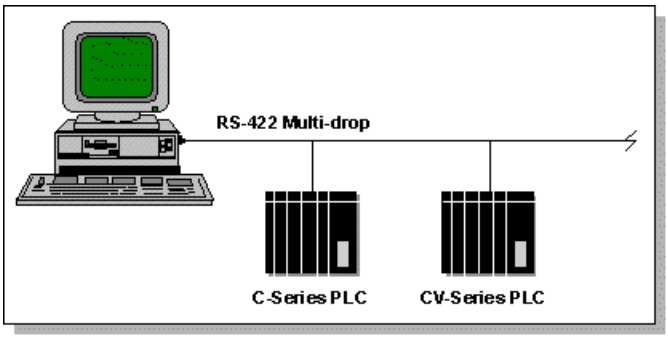
About OMRON Host Link Communications

The OMRON Host Link communications enabler supports OMRON C-Series and CV-Series programmable controllers in point-to-point and multi-drop configurations with one device where the CIMPLICITY Host Link Communications enabler acts as the client.

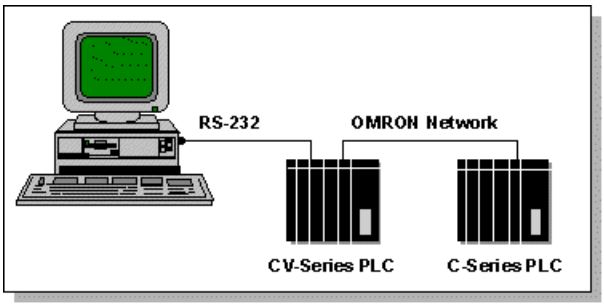
The OMRON Host Link Communications provides read and write access to the OMRON programmable controllers via a Host Link System Network. You can make RS-232 or RS-422 connections through the built in Host Link interface on the programmable controller's CPU or through an optional Host Link Unit in the OMRON PC's rack.

All communication commands are implemented as active/standby communications where the CIMPLICITY OMRON Host Link communications enabler acts as the client responsible for requesting data from or writing data to OMRON programmable controller(s). Unsolicited data sent from OMRON programmable controllers is not supported.

You can connect up to 32 OMRON programmable controllers to a host computer via multi-drop RS-422 line.



You can connect to one OMRON C-Series or CV-Series programmable controller via an RS-232 line.



This enabler supports the following CIMPLICITY features:

- Polled reads at user defined rates
- Poll after setpoint
- Triggered reads
- Analog deadband
- Alarm on communications failure
- Server Redundancy configurations

This enabler supports the following data types:

- Signed 8, 16, and 32 bit integer
- Unsigned 8, 16, and 32 bit integers
- Floating point
- Text
- Arrays

Omron Hostlink supports up to 16 COM ports.

Supported Protocols

The protocol that you use to communicate to an OMRON programmable controller is determined by the **Model type** and **Address** you configure for the programmable controller in a CIMPLICITY project. The **Address** defines whether the programmable controller is connected via the local Host Link network through an inter-network connection.

The device communication enabler supports two different protocols and command sets. The two protocols are:

- C-mode

Use C-Mode commands to communicate directly with OMRON C-Series programmable controllers.

- CV-mode or FINS

Use FINS commands to communicate directly with OMRON CV-Series programmable controllers. FINS commands also allow inter-network communications to networked OMRON programmable controllers. In this configuration, a CIMPLICITY project communicates to C- or CV-Series programmable controllers via a primary CV-Series programmable controller.

Supported OMRON Devices

The OMRON C-Series and CV-Series device communication enabler supports communications to the following programmable controllers:

Communications via C-mode Host Link			
C20	C120/C500	CxxK	C2000
C200H	C1000H	C2000H	CQM1
C200HS	CPM1	C200HX	C200HG
C200HE			
Communications via CV-mode Host Link			
CV500	CVM1-CPU01-E		
CVM1-CPU11-E	CVM1-CPU21-E	CV1000	CV2000

The following C-Series programmable controllers are supported when connected to a subnetwork, that is through a CV bridge, with FINS commands:

- C200H-CPU11-E (requires use of C200HSLK11/SLK21-V1 SYSMAC LINK Unit)
- C1000H-CPU01-EV1 (requires use of C1000H-SLK11/SLK21-V1 SYSMAC LINK Unit)
- C2000-CPU-EV1 (requires use of C1000H-SLK11/SLK21-V1 SYSMAC LINK Unit).

When configuring communications to programmable controllers on subnetworks, you must ensure that the programmable controllers attached to the Host Link is configured with the proper routing tables to reach the desired programmable controllers on the subnetworks. Refer to the OMRON documentation for details on setting up these routing tables.

Supported Addresses

Supported Addresses

The OMRON programmable controllers contain multiple data areas that consist of channels of 16-bit data and completion bit data (timers and counters). The type of areas and number of channels depends on the OMRON programmable controller model. Depending on the logic programmed, the 16-bit values may represent individual I/O bits, one 16-bit data, or multiple 16-bit data. The amount of data read and returned to CIMPLICITY software is determined by the Point Type configured for the address selected.

Read/Write Areas for C-Series on Local Host Link

The Read/Write areas and ranges supported for C-Series programmable controllers on the local Host Link are:

Memory Area	Address Entry
Core I/O (CIO)	**000 to 255**
Link Relays (LR)	**LR000to LR63**
Holding Relays (HR)	**HR00 to HR99**
Auxiliary Area (AR)	**AR00 to AR27**
Timer Present Value (PV)	**TPV000 to TPV511**
Counter Present Value (PV)	**CPV000 to CPV511**
DM Area	**DM0000 to DM9999**
Status	**ST***

Read-only Areas for C-Series on Local Host Link

The Read Only areas and ranges supported for C-Series programmable controllers on the local Host Link are:

Memory Area	Address Entry
Timer Completion Flag	**TIM000 to TIM511**
Counter Completion Flag	**CNT000 to CNT511**
Error Read w/ Clear	**EC0, EC1***
Error Read w/o Clear	**ER0, ER1***

* These areas are not accessible when communicating with the programmable controller via a CV bridge.

Read/Write Areas for CV-Series

The Read/Write areas and ranges supported for CV-Series programmable controllers are:

Memory Area	Address Entry
Core I/O (CIO)	**000 to 2555**

CPU Bus Link Area (G)	**G000 to G255**
Auxiliary Area (A)	**A000 to A511**
Timer Present Value (PV)	**TPV0000 to TPV1023**
Counter Present Value (PV)	**CPV0000 to CPV1023**
DM Area (DM)	DM00000 to DM24575
Expansion DM (E)	**E00000.b to E32765.b** where **b** is the bank number
	Available for **CV1000, CV2000, CVM1-CPU11-E** or other **OMRON** programmable controllers with Expanded **DM**

Read-only Areas for CV-Series

The Read Only areas and ranges supported for CV-Series programmable controllers are:

Memory Area	Address Entry
Timer Completion Flag	**TIM000** to **TIM1023**
Counter Completion Flag	**CNT000** to **CNT1023**
Temporary Relay Area (TR)	**TR0 - TR7**

OMRON Required Documents

In order to properly configure OMRON programmable controllers and their Host Link port(s) you must refer to the Operations and System manuals for the appropriate programmable controller. If you are using a Host Link Unit, then the User's Manual for that interface is required.

These manuals contain important information on:

- Setting the Host Link port's communication parameters such as address, communication speed and parity.
- Constructing the proper RS-232 and RS-422 cable(s) for connecting the CIMPLICITY computer to the Host Link system.
- Interpreting response codes returned from programmable controllers and reported to the Status Log.

OMRON Hardware Configuration Requirements

OMRON Hardware Configuration Requirements

You can connect OMRON programmable controllers via RS-232 or RS-422 connections to either the OMRON programmable controllers built in CPU Host Link interface or through an optional Host Link Unit. Up to 32 OMRON programmable controllers can be connected to a host computer via RS-422 communications on a local Host Link network.

Supported Host Link Units

The following Host Link Units are supported via this communication enabler:

C-Series Host Link Units:			
	C200H-LK201	**C200H-LK202**	**C500-LK201-V1**
	3G2A5-LK201-EV1	**C500-LK203**	**3G2A6-LK201-EV1**
	3G2A6-LK202-EV1		
CV-Series Host Link Units:			
	CV500-LK201		

The choice of module should be based on your OMRON programmable controller models and Host Link System requirements. Please consult OMRON documentation on the features and uses of each of the listed interface modules.

If the OMRON programmable controllers are to be connected via RS-422 then you may need an RS-232 to RS-422 converter or RS-422 cards for your computer. Contact OMRON to determine the best converter for your configuration.

Default Communication Parameters

The following communication parameters are used by the OMRON communication enabler and cannot be changed. The Host Link modules in the PLC must be configured to match these settings:

Parameter	Value	Description
Data Bits	**7**	Number of data bits used in transmissions
Stop Bits	**2**	Number of stop bits used in transmissions

The following communication parameters must match the Port configuration parameters for the OMRON communication enabler in CIMPLICITY software:

- Baud Rate
- Parity

OMRON PLC Cabling Diagrams

OMRON PLC Cabling Diagrams

The following diagrams are intended to provide general guidance for the construction of the cable used to connect the CIMPLICITY computer to the Host Link network port. You should always refer to the appropriate Host Link Unit's System Manual that comes with your Host Link Adapter for detailed diagrams for your particular adapter model and network configuration.

	C-Series Host Link adapters.
	CV Series Host Link adapters.

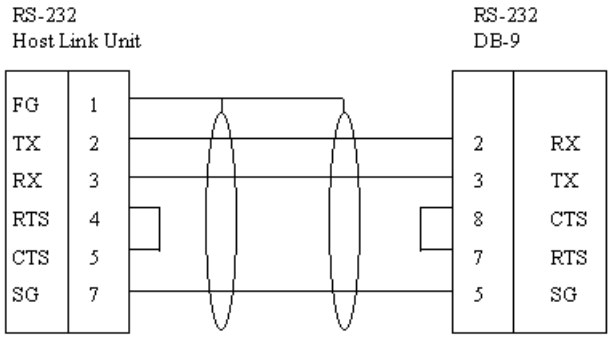
C-Series Host Link Adapters

C-Series Host Link Adapters

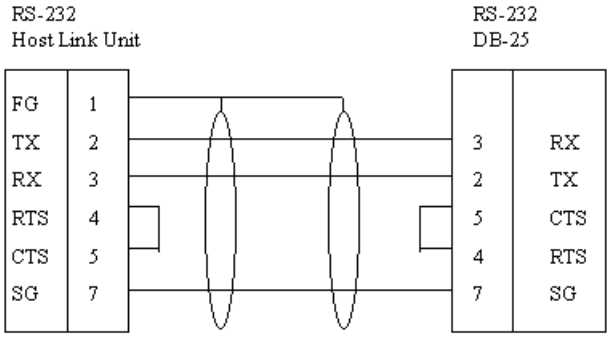
For C-Series Host Link Adapters, see the following cabling diagrams for general guidance in construction cables:

- RS-232 Host Link Unit to DB-9 connector.
- RS-232 Host Link Unit to DB-25 connector.
- RS-422 Interface for C-Series Host Link Adapters.

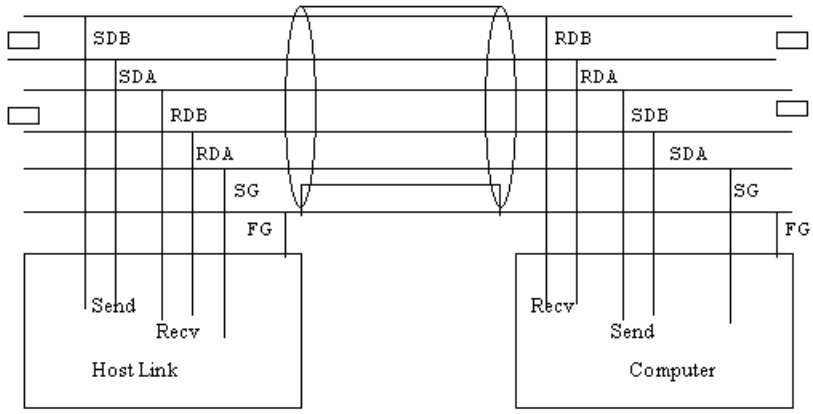
RS-232 Host Link Unit to DB-9 Connector



RS-232 Host Link Unit to DB-25 Connector



RS-422 Interface for C-Series Host Link Adapters



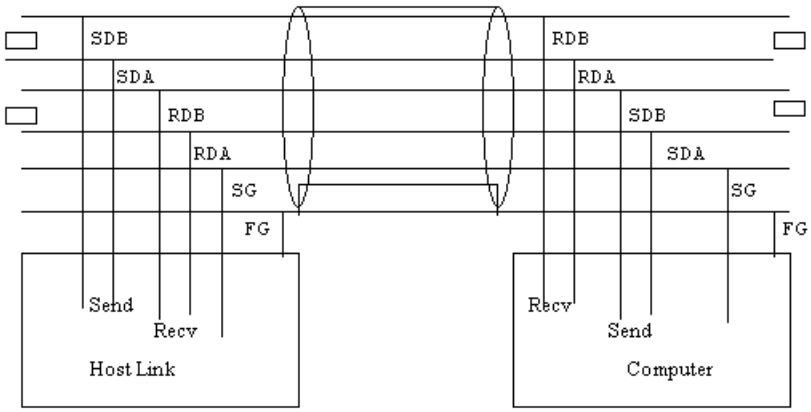
CV-Series Host Link Adapters

CV-Series Host Link Adapters

For CV Series Host Link Adapters, see the following cabling diagrams for general guidance in construction cables:

- RS-232 Host Link Unit to DB-9 connector.
- RS-232 Host Link Unit to DB-25 connector.
- RS-422 Interface for C-Series Host Link Adapters.

RS-422 Interface for CV-Series Host Link Adapters



CIMPLICITY Configuration for OMRON Host Link

CIMPLICITY Configuration for OMRON Host Link

When you are configuring ports, devices, and device points, you will be asked for some information that is specific to OMRON Host Link communications.

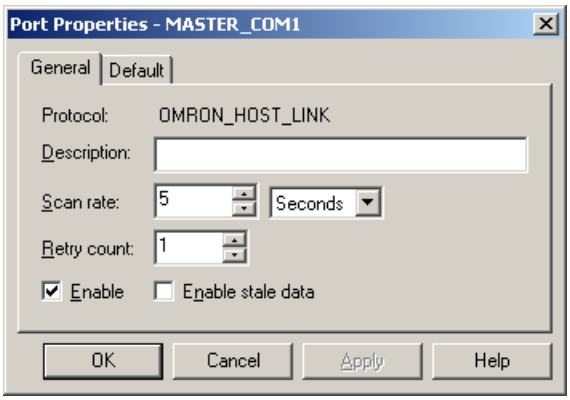
OMRON Host Link Port Configuration

OMRON Host Link Port Configuration

1. In the **Protocol** field, select **OMRON_HOST_LINK** from the list of available protocols.
2. In the **Port** field, select the communication port that will be used for OMRON Host Link communications.

   When you click **OK** to create the port, the Port Properties dialog box for the protocol opens.

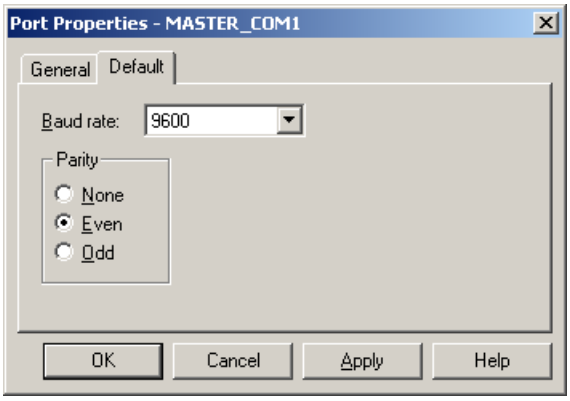
General Port Properties



Use the General properties to enter general information for the port. You can define the following:

Description	Enter an optional description to help you identify the port.
Scan Rate	Enter the base scan rate for the port. Point scan rates will be multiples of the base rate. You can specify a scan rate in **Ticks** (100 ticks = 1 second), **Seconds**, **Minutes**, or **Hours**.
Retry Count	If communications cannot be established for a device on the port, the device is considered to be down, and a $DEVICE_DOWN alarm is generated. Once a device is down, periodic attempts are made to resume connections to it.
	Enter the number of scans to wait before attempting to reconnect to a device after a communication error has been detected. The recommended **Retry Count** is **3**.
Enable	Set this check box if you want the port to be enabled when the project starts. If you clear this check box, the port will not be enabled when the project starts, and points will not be available for devices on the port.

Default Port Properties



Use the Default properties to enter communications information for the port. You can define the following:

Baud Rate	Enter the baud rate for communications. Click the drop-down list to the right of the input field and select a rate from the list.
Parity	Select the parity to be used for the communications.

OMRON Host Link Device Configuration

OMRON Host Link Device Configuration

1. In the **Device** field, enter the name of the device you are configuring.
2. In the **Port** field, select the OMRON Host Link port to be used by the device.

   When you click **OK** to create the device, the Device Properties dialog box opens.

   The information you enter in the Device Properties dialog box defines the controller's physical location, or address. The format of the address information depends on the controller type (C-Series or CV-Series) and the controller's location on the attached network. The address information and programmable controller device type you enter are used to establish which protocol will be used within the enabler to talk with the controller.

Domain Size Validation

When you select the **Model type**, you also identify how CIMPLICITY software validates the point addresses against the memory ranges of a target programmable controller.

- When you select **C-Series**, **CV-Series**, **CV-Series Ext** or **CVM1-V2**, dynamic domain sizing is used. In dynamic domain sizing, CIMPLICITY software polls the programmable controller using a binary search algorithm to establish the actual memory ranges.
- When you select **C-Series Static**, **CV-Series Static**, **CV-Series Ext Static** or **CVM1-V2 Static**, static domain sizing is used. In static domain sizing, CIMPLICITY software uses the pre-defined memory ranges based on OMRON's specifications.

If you use dynamic domain sizing, your CIMPLICITY project startup will take longer as the OMRON device communications enabler polls each OMRON programmable controller to determine its memory sizes. It is recommended that you use model types that employ dynamic domain sizing during project development, then switch to model types that employ static domain sizing when you deploy the project.

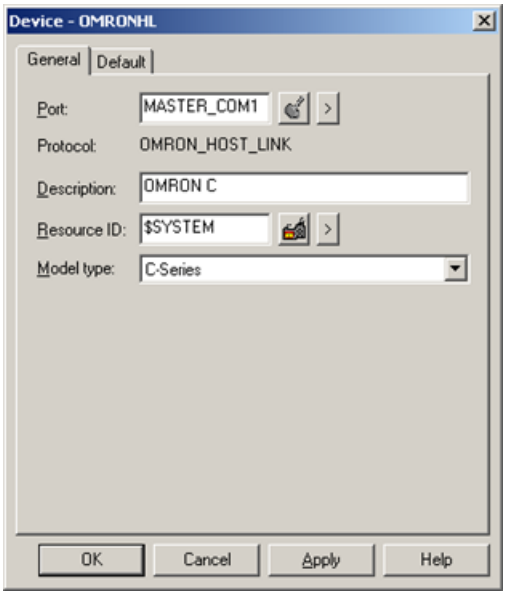
Determining the Protocol

The OMRON device communications enabler selects the protocol for a device based on the model type and address you configure for the device.

- If you select the **CV-Series**, **CV-Series Static**, **CV-Series Ext.**, **CV-Series Ext. Static**, **CVM1-V2** or **CVM1-V2 Static** model type, the OMRON communications enabler uses CV-mode (FINS) protocol.

- If you select the **C-Series** or **C-Series Static** model type, the protocol depends on the device's address.
- If the address indicates the device is on a local Host Link network, the OMRON device communications enabler uses C-mode protocol.
- If the address indicates the device is on a subnetwork, then the OMRON device communications enabler uses CV-mode protocol.

General Device Properties



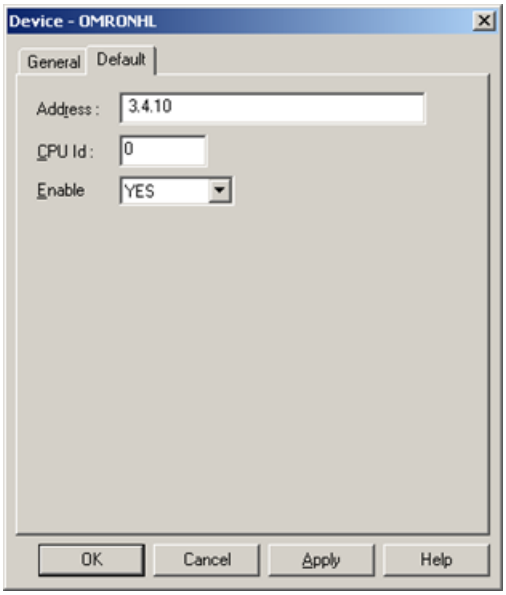
Use the General properties to enter general information for the device. You can define the following:

Port	Select the port for this device. Click buttons to the Port field to select the port as follows.	
		Display the list of ports and select one.
		Create a new port, edit the current port, or select a port from the list of ports.
Description	Enter an optional description to help you identify the device.	
Resource	Enter a resource that can be associated with this device for alarm generation. Click buttons to the right of the **Resource** field to select the resource, as follows.	
		Display the list of resources and select one.
		Create a new resource, edit the current resource, or select a resource from the list of resources.
Model Type	Select the type of device. Click the drop-down button to the right of the input field to display your choices. Use the following information to make your selection.	

	Model Type	Supported Programmable Controllers

	C-Series and C-Series Static	**C20, C120/C500, CxxK, C2000, C200H, C1000H, C2000H, CQM1, C200HS, CPM1, C200HX, C200HG** and **C200HE**
	CV-Series and CV-Series Static	**CV500, CVM1-CPU01-E**
	CV-Series Ext. and CV-Series Ext. Static	**CV1000, CV2000 and CVM1-CPU11-E** or others with Expanded **DM**
	CVM1-V2 and CVM1-V2 Static	**CVM1-CPU21-EV2** and programmable controllers with **IEEE** Floating Point capability

Default Device Properties



Use the Default properties to enter addressing information about this device.

Address	Enter one of the following: • For programmable controllers on a local Host Link, enter the programmable controllers Unit Address. Sample local address: 10 • For programmable controllers on a subnetwork, use the following format: **<**CV bridge**>.<**subnetwork number**>.<**address**>** Where **<**CV bridge**>** is the CV bridge on the local Host Link network, **<**subnetwork number**>** is the subnetwork number, and **<**address**>** is the Unit Address of the programmable controller. Sample subnetwork address: 3.4.10
	Note: Some C-Series PLC models cannot be accessed on a subnetwork via a CV Bridge. Please refer to the Supported OMRON Devices *(page 733)* section for information on which C-Series models are supported via a CV Bridge.
CPU ID	Not used.

Enable	Select YES to enable the device when the project starts. If you select NO, the device will not be enabled and points associated with the device will be unavailable.

OMRON Host Link Point Configuration

OMRON Host Link Point Configuration

Once your devices are configured, you may configure points for them. Through device point configuration, you may configure the following:

- Points that read or set discretes and/or bits in memory
- Points to read or set CIO, LR, HR, AR and DM memory areas
- Points to read Timer/Counter completion statuses
- Points to read or set Timer and Counter Present Values (PVs)
- Points to read Status and Errors
- Points to set operation mode

Fields in the Point Properties dialog box have configuration values that are unique to, or have special meaning for OMRON Host Link Serial communications.

General Point Properties

On the General tab, the type of access depends on the point address:

- Select **Read** for points configured for Error reading and Timer/Counter compilation statuses.
- Select **Read/Write** for all other points configured if you wish to be able to set (change) the value.

Device Point Properties

On the Device tab:

Update Criteria	The update criteria determines how the data will be requested.
	Select **On Change** or **On Scan** for points whose values should be polled by the OMRON Host Link Serial driver at regular intervals.
	Select **On Demand On Scan** or **On Demand OnChange** for points whose values should be polled by the OMRON Host Link Serial driver at regular intervals while users are viewing them.
Address	Enter the point address of the data. See the C-Series Point Addresses *(page 745)*, CV-Series Point Addresses *(page 746)*, and Notes on Addressing *(page 746)* sections below for more information.

When you configure Boolean points from non-discrete memory (such as CIO, HR or DM) you must also enter data in the following field:

Bit Offset	Enter the bit offset in the 16-bit memory word that defines the point where 0 is the least significant bit and 15 is the most significant bit.
	Warning: Do not enter a data Bit Offset for Point Type BOOL for Timer and Counter Completion Statuses or Temporary Relay (TR). Leave the field as the default 0.

Point Address Formats

Point Address Formats

Review:

- C-Series point addresses.
- CV Series point addresses.
- Notes on addressing.

C-Series Point Addresses

For C-Series programmable controllers connected to the local Host Link Network, the following formats are valid:

Memory Area	Address Entry
Core I/O (CIO)	**000** to **2555**
Link Relay (LR)	**LR000** to **LR63**
Holding Relay (HR)	**HR00** to **HR99**
Auxiliary Relay (AR)	**AR00** to **AR27**
Timer Completion Flag	**TIM000** to **TIM511**
Timer PV	**TPV000** to **TPV511**
Counter Completion Flag	**CNT000** to **CNT511**
Counter PV	**CPV000** to **CPV511**
DM Area	**DM0000** to **DM9999**
Status	**ST**
Error Read w/ Clear	**EC0, EC1**
Error Read w/o Clear	**ER0, ER1**

CV-Series Point Addresses

For CV-Series programmable controllers connected to the local Host Link Network, the following formats are valid:

Memory Area	Address Entry
Core I/O (CIO)	**000 to 255**
Temporary Relay (TR)	**TR0 - TR7**
CPU Bus Link Area (G)	**G000** to **G255**
Auxiliary Area (A)	**A000** to **A511**
Timer Completion Flag	**TIM000** to **TIM1023**
Timer Present Value (PV)	**TPV0000** to **TPV1023**
Counter Completion Flag	**CNT000** to **CNT1023**
Counter Present Value (PV)	**CPV0000** to **CPV1023**
DM Area (DM)	**DM00000** to **DM24575**
Expansion DM (E)	**E00000.b** to **E32765.b** where **b** is the bank number
	Available for **CV1000**, **CV2000**, **CVM1-CPU11-E** or other OMRON programmable controllers with Expanded DM

Notes on Addressing

Please note the following about point addresses for the OMRON device communications enabler:

- Letters in address entries can be upper or lower case
- Leading zeros on address numbers are optional. Example: HR003 is treated the same as HR3.
- Address ranges may be limited by the address space available in the controller device.
- The Error Read w/ Clear and Error Read w/o Clear memory areas are read only - Do not configure points with these addresses as setpoints. Refer to the C-Series Host Link Units manual for details on these memory areas.
- You can use a Status (ST) point as a setpoint to set the operating mode of the controller. When reading Status, the status data from the Status Read response is returned. This data is different from the data written. Refer to the C-Series Host Link Units manual for details on Status Read and Status Write.
- Use BOOL Point Types for Completion flag (TIM and CNT) and Temporary Relay (TR) points.
- Status (ST), Error Read w/ Clear and w/o Clear are not accessible when you communicate with the programmable controller via a CV bridge.

Advanced Configuration Topics-Adjust the Communications Time-out Parameter

Advanced Configuration Topics

- COM<PORT>_TO
- OMRON_MAX_BUFFER_SIZE

COM<PORT>_TO

Purpose	To set the time-out used by the OMRON communications on port COM1 to 10.5 seconds	
Parameter Entry	COMP<port>_TO	
	Where	
	<port**> =**	the communication port number
Value	Time out in tenths of seconds	
Default	7.5 seconds	

OMRON_MAX_BUFFER_SIZE

Purpose	To increase the size of the poll buffer.
Values	Size range of 538 to 1000

Default Protocol Parameters

Certain protocol parameters used by the OMRON communications enabler have been preset. They are:

Parameter	Value	Description
DN2	**0**	Unit Address - 0 indicates the programmable controller's CPU. This is not the same as the unit number (that is, node number) which is configurable.
SNA, SA1	**0,0**	Source of local network and node number.
SA2	**FE**	Source of communication unit

SID	**FF**	Service ID.

Chapter 45. OMRON Host Link Communications Diagnostics

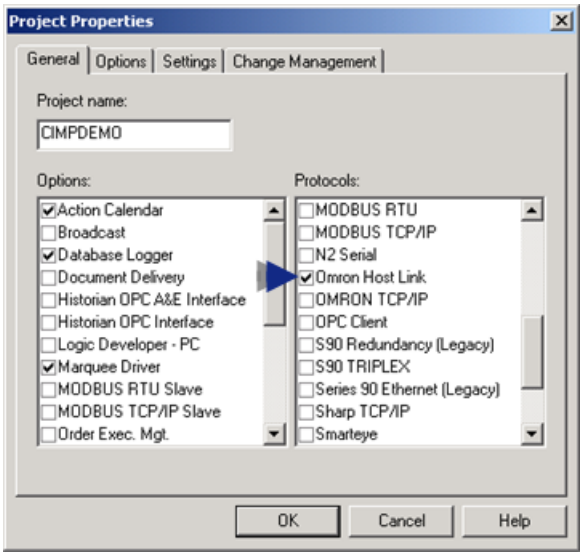
OMRON Host Link Communications Diagnostics

You can use the OMRON Host Link Diagnostics program to check the basic configuration, cabling and operation of your network without running a CIMPLICITY project. You can use this program to read and write data from and to a specified OMRON programmable controller. A single value (channel or timer/completion bit) is transferred at a time.

- Open the CIMPLICITY OMRON Host Link Comm Test dialog box
- OMRON Host Link communications diagnostics steps

Open the CIMPLICITY OMRON Host Link Comm Test dialog box

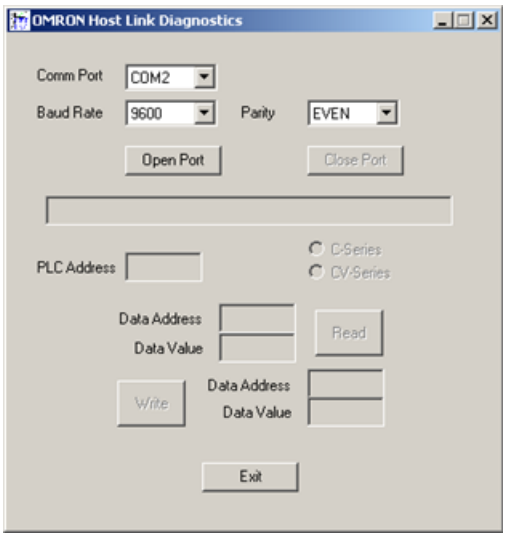
1. Make sure Omron Host Link is checked in the Project Properties dialog box.



2. Select **Project>Equipment>Diagnostics** in the Workbench left-pane.

3. Double-click **OMRON Host Link Diag**.

The OMRON Host Link Diagnostics dialog opens with default parameters selected.



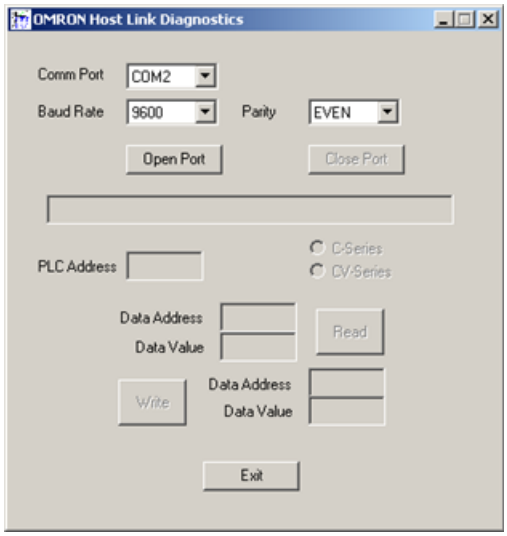
OMRON Host Link communications diagnostics steps

The top section of the dialog contains fields for setting the communication port parameters. A display only field in the center of the dialog displays error and status information. Below the status message field are fields you can use to specify PLC Address, Data Address and Data Values for read and write testing.

Step 1 *(page 750)*	Open a communication port.
Step 2 *(page 751)*	Select a target programmable controller.
Step 3 *(page 753)*	Diagnose communication problems.
Step 4 *(page 753)*	Close a communication port and/or change ports and configuration.
Step 5 *(page 754)*	Exit the diagnostic program.

Step 1. Open a Communication Port

1. Select the following.



Field	Select the:
Comm Port	Communication port for the PLC.
Baud rate	Baud rate for the programmable controller.
Parity	Parity for the programmable controller.

2. Click Open Port to initialize communications

   The status of your request is displayed in the message field.

   If an error message is displayed, check that the selected communication port is not in use by another program and that the port is properly configured and available on the PC and operating system.

Step 2. Select a Target Programmable Controller

Step 2. Select a Target Programmable Controller

1. In the **PLC address** field, enter the address of the OMRON programmable controller. Use the same format you use when configuring the controller for a CIMPLICITY project.

   To enter the address of an OMRON programmable controller on the local Host Link network, enter a single number.

2. To enter the address of an OMRON programmable controller on a subnetwork, use the following format:

< CV bridge >.< subnetwork number >.< PLC address >

Note: This is the same format you use when defining a device address in CIMPLICITY software

3. Click on the appropriate radio button to select a **C-Series** or **CV-Series** programmable controller.

   If you enter an invalid programmable controller address, an error message displays on the status line. You must enter a valid address to continue.

   After you enter a valid programmable controller address, the Read and Write Data Address fields are enabled.

Option 2.1 *(page 752)*	Read data.
Option 2.2 *(page 752)*	Write data.

Option 2.1. Read Data

You can read data from valid data areas. Valid data areas are determined by the programmable controller type. The format for entering a data address is the same one you use when you configure points for a CIMPLICITY project.

To read data from the targeted programmable controller:

1. Enter the address of the data in the **Data address** field to the left of the **Read** button. This activates the **Read** button

2. Click Read.

   If the read request completes successfully, the data appears in the **Data value** field and the status message field says the read was successful. All channel data is displayed in hexadecimal. Timer and Count completion status are displayed as 1 or 0.

   If the read request fails, an error message is displayed in the status message field.

Option 2.2. Write Data

1. Enter a valid data address in the **Data address** field to the right of the Write button.

This activates the **Data value** field.

2. Enter the data value you want to write in hexadecimal notation in the **Data value** field.

   This activates the Write button.

3. Click Write.

   The status message field displays a message indicating the success or failure of the write request.

   The status line will display a message indicating the result of the write operation, either successful or, in the case of a failure, an error message.

Step 3. Diagnose Communication Problems

If the read or write request fails, a message is displayed in the status message field giving an indication of the failure. Use this information to help diagnose communication problems. Error messages will include the Error Response Code returned from the target programmable controller where appropriate. Refer to the OMRON Host Link User's Manuals for complete error descriptions. Errors will indicate possible problems with data addresses, programmable controller routing tables, data ranges, etc.

Time-out errors

Time-out errors may indicate an invalid programmable controller address, incorrect communication parameters settings, or communication port problems. The most common cause of time-out errors is an improperly constructed communication cable.

Step 4. Close a Communication Port and/or Change Ports and Configuration

You need to close communications when:

- You are finished with diagnostic testing
- You want to select a different port
- You want to change the port's baud rate or parity.

To close communications, click **Close port**.

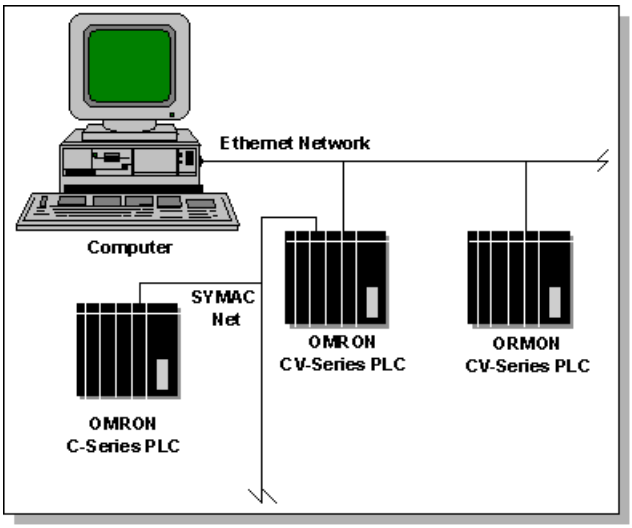
Step 5. Exit the Diagnostic Program

1. Click **Close port** to close and release the communication port.
2. Click **Exit** to close the dialog and terminate the program.

Chapter 46. OMRON TCP/IP Communications

About OMRON TCP/IP Communications

The OMRON® TCP/IP Communications enabler supports CV-Series programmable controllers in multi-drop configurations with the OMRON TCP/IP Communication Enabler acting as the active. The OMRON TCP/IP Communications provides read and write access to the OMRON programmable controllers via TCP/IP over an Ethernet network. The CV-Series controller(s) must be equipped with OMRON Ethernet Module(s).



The OMRON TCP/IP Communication enabler supports the CV-mode protocol and command set, also referred to as FINS. The FINS commands are sent in Ethernet UDP datagram transactions. The enabler uses FINS commands to communicate directly with OMRON CV-Series programmable controllers.

FINS commands also allow inter-network communications to networked OMRON programmable controllers. In this configuration, a CIMPLICITY project communicates to C- or CV-Series programmable controllers via a primary CV-Series programmable controller that acts as a gateway to other networks such as SYSMAC Net and SYSMAC LINK Net.

All communication commands are implemented as active/standby communications where the OMRON TCP/IP Communications enabler acts as the active responsible for requesting data from or writing data to OMRON programmable controller(s). Unsolicited data sent from OMRON programmable controllers is not supported.

This enabler supports the following CIMPLICITY features:

- Polled reads at user defined rates
- Poll after setpoint
- Triggered reads
- Analog deadband
- Alarm on communications failure
- Server Redundancy configurations

This enabler supports the following data types:

- Signed 8, 16, and 32 bit integers
- Unsigned 8, 16, and 32 bit integers
- Floating point
- Text
- Arrays

Supported OMRON Devices

The OMRON TCP/IP Communication enabler supports direct communications to the following programmable controllers:

	CV500	CVM1-CPU01-E	CV1000
	CVM1-CPU11-E	CVM1-CPU21-E	CV2000

The following C-Series programmable controllers are supported when connected to a subnetwork, that is through a CV bridge, with FINS commands:

- C200H-CPU11-E (requires use of C200HSLK11/SLK21-V1 SYSMAC LINK Unit)
- C1000H-CPU01-EV1 (requires use of C1000H-SLK11/SLK21-V1 SYSMAC LINK Unit)
- C2000-CPU-EV1 (requires use of C1000H-SLK11/SLK21-V1 SYSMAC LINK Unit).

When configuring communications to programmable controllers on subnetworks, you must ensure that the programmable controllers attached to the Ethernet are configured with the proper routing tables to reach the desired programmable controllers on the subnetworks. Refer to the OMRON documentation for details on setting up these routing tables.

Supported Addresses

Supported Addresses

The OMRON programmable controllers contain multiple data areas that consist of channels of 16-bit data and completion bit data (timers and counters). The type of areas and number of channels depends on the OMRON programmable controller model. Depending on the logic programmed, the 16-bit values may represent individual I/O bits, one 16-bit data, or multiple 16-bit data. The amount of data read and returned to CIMPLICITY software is determined by the Point Type configured for the address selected.

Read/Write Areas for CV-Series

The Read/Write areas and ranges supported for CV-Series programmable controllers are:

Memory Area	Address Entry
Core I/O (CIO)	000 to 2555
CPU Bus Link Area (G)	G000 to G255
Auxiliary Area (A)	A000 to A511
Timer Present Value (PV)	TPV0000 to TPV1023
Counter Present Value (PV)	CPV0000 to CPV1023
DM Area (DM)	DM00000 to DM24575
Expansion DM (E)	E00000.b to E32765.b where b is the bank number
	(CV-Series Ext. only, CV1000, CV2000, CVM1-CPU11-E,…)

Read-Only Areas for CV-Series

The Read Only areas and ranges supported for CV-Series programmable controllers are:

Memory Area	Address Entry
Timer Completion Flag	TIM000 to TIM1023
Counter Completion Flag	CNT000 to CNT1023
Temporary Relay Area (TR)	TR0 - TR7

Read/Write Areas for C-Series on Sub-network

The Read/Write areas and ranges supported for C-Series programmable controllers on a subnetwork are:

Memory Area	Address Entry
Core I/O (CIO)	000 to 255

Link Relays (LR)	LR000to LR63
Holding Relays (HR)	HR00 to HR99
Auxiliary Area (AR)	AR00 to AR27
Timer Present Value (PV)	TPV000 to TPV511
Counter Present Value (PV)	CPV000 to CPV511
DM Area	DM0000 to DM9999
Status	ST*

* These areas are not accessible when communicating with the programmable controller via a CV bridge.

Read-Only Areas for C-Series on Local Ethernet

The Read Only areas and ranges supported for C-Series programmable controllers on a subnetwork are:

Memory Area	Address Entry
Timer Completion Flag	TIM000 to TIM511
Counter Completion Flag	CNT000 to CNT511
Error Read w/ Clear	EC0, EC1*
Error Read w/o Clear	ER0, ER1*

* These areas are not accessible when communicating with the programmable controller via a CV bridge.

OMRON TCP/IP Required Documents

In order to properly configure OMRON programmable controllers and their Ethernet module(s) you must refer to the Operations Manual and System Manual for the appropriate programmable controller and to the OMRON Ethernet module's User's Manual.

These manuals contain important information on:

- Setting up the Ethernet communication parameters such as IP netmask, IP address, FINS UDP Port Number and the IP Address Table to convert between FINS node numbers and IP addresses
- Interpreting response codes returned from programmable controllers and reported to the Status Log.

Important: It is absolutely imperative that you fully understand how to configure the OMRON Ethernet Unit and how it translates between the FINS protocol values SNA, SA1, DNA and DA1 and the IP addresses assigned to the controller(s) and the CIMPLICITY computer. Furthermore, in order to access controllers on sub networks, you must understand how to properly configure routing tables in the programmable controllers.

OMRON TCP/IP Application Notes

OMRON TCP/IP Application Notes

For a complete discussion of IP address determination and configuring Port Numbers refer to the OMRON Ethernet module's User's Manual. You must have a thorough and complete understanding of this information in order to get the enabler to communicate.

Datagrams

The OMRON TCP/IP device communication enabler communicates using FINS commands over Ethernet. The FINS commands are exchanged between the computer running a CIMPLICITY project and the controllers in UDP packets. UDP packets are called datagrams. Datagram communication is termed connectionless communications in contrast to TCP, which is connection oriented communications. With datagram communications, the controllers can communicate with more devices since they do not need to maintain a separate and distinct connection for each device.

UDP datagram service does not guarantee packet delivery, as does TCP. The communication enabler always expects a response for each issued read or write command. If a response is not received within a time-out period you configure, it will retry the command for the number of retries you configure before concluding that the device is "down" or unreachable at the designated address.

UDP datagrams are sent and received over abstractions called sockets, which can be likened to serial communication ports. Multiple sockets can be opened for a given Ethernet interface; hence, the OMRON TCP/IP communication enabler can "share" the Ethernet port with other applications running on the same computer.

IP Addresses

To send commands to targeted controllers via the UDP datagrams sent through the Ethernet socket, Internet Protocol (IP) addresses are used. You define programmable controller IP addresses to this device communication enabler by entering host names. Host names are resolved to actual IP addresses by looking them up in the operating system's hosts file. You must ensure that this file contains records for resolving each controller's assigned IP address to a host name. Contact your

system administrator or the network manuals for the operating system for instructions on modifying this file. Test the entries by using the ping utility. Ping the controller by the host name in order to insure that each entry is correct and that each controller is reachable.

Source Computer ID

The controllers must send command responses back to your computer. They send these datagrams to your computer's IP address. The controllers need to determine your computer's IP address in order to do this. You can configure the OMRON Ethernet modules to use two different methods: table look-up and automatic address generation. Both use the SA1 value of your computer. SA1 is the FINS value you assign for the Node Number of your computer on the OMRON Ethernet network.

For the table look-up method, the Ethernet Unit simply looks up the SA1 value sent in the original command in a table your define in the controller to find the computer's IP address. Using OMRON programming software, you must enter both the SA1 and IP address values in the table for each OMRON Ethernet module that uses this method.

For the automatic address generation method, the Ethernet modules use the SA1 value to calculate the IP address by performing a logical AND between the Ethernet Module's IP address and the subnet mask, and then adding the FINS node number (SA1).

Port Numbers

One final piece of information is required in order for the response to find its way back to the device enabler. The IP address gets the response UDP packet to your computer but now it must find its way to the device enabler application. This is done via a Port Number. The device communication enabler connects its socket to a Port Number. This connection processed is called binding. The operating system will permit only one application per computer to bind to a given port number.

You configure the controllers to send their responses to this port number. The default FINS UPD Port Number for the OMRON Ethernet Modules is 9600. You instruct the device communication enabler to bind to this number by configuring a Service Name.

The Service Name is resolved to a number in a manner similar to the host name's resolution to an IP address. It is looked up in an operation system file called the services file when the device communication enabler is first started.

You must add an entry to the services file relating a service name to a UDP port number which is the same as the UDP Port Number programmed in your OMRON Ethernet Modules. Contact your system administrator or the network manuals for the operating system for instructions on modifying this file.

You tell the device communication enabler which name to look up (to find the UDP port number) through the CIMPLICITY project's global parameters. See Setting Required Protocol Parameters for further instructions on how to do this.

Validating OMRON TCP/IP Communications

Validate OMRON TCP/IP Communications

You can use the OMRON TCP/IP Diagnostics program to check the basic configuration, cabling and operation of your network without running a CIMPLICITY project. You can use this program to read and write data from and to a specified OMRON programmable controller. A single value (channel or timer/completion bit) is transferred at a time.

To start the OMRON TCP/IP Diagnostics program, click on the OMRON TCP/IP Diagnostics icon in the CIMPLICITY project's Configuration Cabinet.

The OMRON TCP/IP Diagnostics dialog box opens with default parameters selected.

The top section of the dialog contains fields for setting the source node information (information on your computer's location on the network) and for selecting a UPD service (port number) for communications. A display only field in the center of the dialog displays error and status information. Below the status message field are fields you can use to specify PLC Address, Data Address and Data Values for read and write testing

Opening A Communication Socket

1. In the CIMPLICITY SNA field, enter the Source Network Address (SNA) value for your computer that corresponds with the OMRON network configuration.

2. In the CIMPLICITY SA1 field, enter the Source Node number (SA1) value for your computer.

3. In the UDP service name field, enter the UDP service name listed in the TCP/IP services file that corresponds to the Port Number configured in the OMRON Ethernet module(s) that you will be corresponding with.

4. Select Open Socket to initialize communications.

   The status of your request displays in the message field.

   If an error message displays, it gives an indication of the problem that must be corrected before continuing. These problems occur during the opening and binding of the communication socket. Typical problems are:

- TCP/IP support is not loaded and/or configured for the operating system.
- The UDP Service Name entered is not in the TCP/IP service file or is listed incorrectly (as TCP verses UDP for instance).

Select a Target Programmable Controller

1. In the **Controller address** field, enter the address of the OMRON programmable controller. Use the same format you use when configuring the controller for the CIMPLICITY project.

2. Do one of the following:
   - To enter the address of an OMRON programmable controller on the local Ethernet network, simply enter the Host Name of the controller that corresponds to its IP address.
   - To enter the address of an OMRON programmable controller on a subnetwork, use the following format:

   <Host Name>.<subnetwork number>.<Controller address>

   Note: This is the same format you use when defining a device address in CIMPLICITY software

3. Select the controller model from the Controller Model drop down selection list.

   If you enter an invalid format for the programmable controller address or if you enter a Host Name that is not valid (not in the TCP/IP hosts file) an error message displays on the status line. You must enter a valid address to continue.

   After you enter a valid programmable controller address, the **Data address** and **Data value** fields for read and write requests are enabled.

Reading Data

You can read data from valid data areas *(page 756)* . Valid data areas are determined by the programmable controller type. The format for entering a data address is the same one you use when you configure points for a CIMPLICITY project.

To read data from the targeted programmable controller:

1. Enter the address of the data in the Data address field to the left of the Read button. This activates the Read button

2. Select Read.

If the read request completes successfully, the data appears in the Data value field and the status message field says the read was successful. All channel data is displayed in hexadecimal. Timer and Count completion status are displayed as 1 or 0.

If the read request fails, an error message displays in the status message field.

Writing Data

You can write data to valid data areas *(page 756)* . Valid data areas are determined by the programmable controller type. The format for entering a data address is the same one you use when you configure points for a CIMPLICITY project.

To write data to the targeted programmable controller:

1. Enter a valid data address in the Data address field to the left of the Write button. This activates the Data value field.
2. Enter the data value you want to write in hexadecimal notation in the Data value field. This activates the Write button.
3. Select Write.

   The status message field displays a message indicating the success or failure of the write request.

Diagnosing Communication Problems

Diagnosing Communication Problems

If the read or write request fails, a message is displayed in the status message field giving an indication of the failure. Use this information to help diagnose communication problems. Error messages will include the Error Response Code returned from the target programmable controller where appropriate. Refer to the OMRON Ethernet User's Manuals for complete error descriptions. Errors will indicate possible problems with data addresses, programmable controller routing tables, data ranges, etc.

Time-out Errors

Time-out errors may indicate an invalid programmable controller address, an invalid computer SNA and/or SA1 value(s), or an invalid service name.

If the controller is not at the IP address specified by the host name, the target controller will not receive the message and hence will not reply. Test the host name by using the TCP/IP utility ping (for example, ping host name). If the controller is responding, that means the network is functioning and the device at the IP address related to the host name is responding. If ping fails, then verify your network is functional and that the targeted programmable controller's Ethernet Unit has been configured for the IP address assigned to host name.

If ping is successful then the programmable controller's response may not be making it back to the diagnostic test program because of an invalid SNA/SA1 pair or an incorrect port number specified by the entered service name.

SNA should match the network number assigned to the Ethernet network in the OMRON Ethernet Module(s) setup. Either the SA1 value is the value of the IP address of your computer minus the IP netmask or it matches a record in the IP Address Table. The choice depends on whether you have configured the OMRON Ethernet module with the IP address conversion method set to automatic generation.

Finally, verify the FINS UDP Port Number you have configured in the OMRON Ethernet Unit matches the value entered in the services file and that the number is associated with the service name entered.

Closing Communication Socket and/or Changing the Service Name/ SNA/SA1 Setup

You need to close the communication socket when:

- You are finished with diagnostic testing
- You want to select a different Service Name, SNA or SA1 value

To close communications, select Close Socket.

Exiting the Diagnostic Program

1. Select Close Socket to close and release the communication socket.
2. Select Exit to close the dialog and terminate the program.

OMRON TCP/IP Application Configuration

OMRON TCP/IP Application Configuration

When you are configuring ports, devices, and device points, you will be asked for some information that is specific to OMRON TCP/IP communications.

OMRON TCP/IP Port Configuration

Port Configuration

When you create a new port for OMRON TCP/IP communications, enter the following information in the New Port dialog box:

1. In the Protocol field, select OMRON_TCPIP from the list of available protocols.
2. In the Port field, select the communication port that will be used for OMRON TCP/IP communications.

   When you select OK to create the port, the Port Properties dialog box for the protocol opens.

Port General Properties

Use the General properties to enter general information for the port. You can define the following:

Description	Enter an optional description to help you identify the port.
Scan Rate	Enter the base scan rate for the port. Point scan rates will be multiples of the base rate. You can specify a scan rate in Ticks (100 ticks = 1 second), Seconds, or Hours.
Retry Count	If communications cannot be established for a device on the port, the device is considered to be down, and a $DEVICE_DOWN alarm is generated. Once a device is down, periodic attempts are made to resume connections to it.
	Enter the number of scans to wait before attempting to reconnect to a device after a communication error has been detected. The recommended Retry Count is 3.
Enable	Set this check box if you want the port to be enabled when the project starts. If you clear this check box, the port will not be enabled when the project starts, and points will not be available for devices on the port.

Device Configuration

Device Configuration

When you create a new device for OMRON TCP/IP communications, enter the following information in the New Device dialog box:

1. In the Devices field, enter the name of the device you are configuring.
2. In the Port field, select the OMRON TCP/IP port to be used by the device.

When you select OK to create the device, the Device Properties dialog box opens.

Device General Properties

Use the General properties to enter general information for the device. You can define the following:

Port	Select the port for this device. • You can click the Browser button -  - to the right of the field to display the list of ports and select one. • You can click the Pop-up Menu button -  - to create a new port, edit the current port, or select a port from the list of ports.
Description	Enter an optional description to help you identify the device.
Resource	Enter a resource that can be associated with this device for alarm generation.
Model Type	Select the type of device. Click the drop-down button to the right of the input field to display your choices. Use the following information to make your selection.

	Model Type	Supported Programmable Controllers
	C-Series and C-Series Static	C200H-CPU11-E, C1000H-CPU01-EV1 and C2000H-CPU-EV1. Note: C-Series controllers are only accessible via a CV-Series controller acting as a gateway.
	CV-Series and CV-Series Static	CV500, CVM1-CPU01-E
	CV-Series Ext. and CV-Series Ext. Static	CV1000, CV2000 and CVM1-CPU11-E or others with Expanded DM
	CVM1-V2 and CVM1-V2 Static	CVM1--CPU21-EV2 and programmable controllers with IEEE Floating Point capability

Domain Size Validation

When you select the Model type, you also identify how CIMPLICITY software will validate the point addresses against the memory ranges of a target programmable controller.

- When you select C-Series, CV-Series, CV-Series Ext or CVM1-V2, dynamic domain sizing is used. In dynamic domain sizing, CIMPLICITY software polls the programmable controller using a binary search algorithm to establish the actual memory ranges.
- When you select C-Series Static, CV-Series Static, CV-Series Ext Static or CVM1-V2 Static, static domain sizing is used. In static domain sizing, CIMPLICITY software uses the pre-defined memory ranges based on OMRON's specifications.

If you use dynamic domain sizing, your CIMPLICITY project startup will take longer as the OMRON device communications enabler polls each OMRON programmable controller configured to determine their memory sizes. It is recommended that you use model types that employ dynamic

domain sizing during project development, then switch to model types that employ static domain sizing when you deploy the project.

OMRON TCP/IP Properties

Use the OMRON TCP/IP properties to enter addressing information about this device.

Address	Enter one of the following: • For programmable controllers connected directly to the local Ethernet network, enter the host name assigned for the IP address at which the controller's Ethernet Unit has been configured. Sample local address: etchlineB_cntrl • For programmable controllers on a subnetwork, use the following format: <host name>.<subnetwork number>.<address> where: <host name> is the host name assigned for the IP address of the CV controller acting as bridge on the local Ethernet network. <subnetwork number> is the subnetwork number <address> is the Unit Address of the programmable controller. An example of a subnetwork address is: etchlineB_cntrl.4.10
	Notes on addressing The <host name> is looked up in the operating system's hosts file when the device communication enabler is started to determine the targeted controller's IP address.
	If the <host name> includes dots, you can enclose the host name in quotations. For example, "omron_cntrl.net2".2.31
	The <subnetwork number> is the DNA and <address> is DA1 in OMRON FINS protocol terminology.
	Some C-Series controller models cannot be accessed on a subnetwork via a CV Bridge. Please refer to the Supported OMRON Devices *(page 756)* section for information on which C-Series models are supported via a CV Bridge.
Enable	Set this check box to enable the device when the project starts. If you clear the check box, the device will not be enabled and points associated with the device will be unavailable.

Point Configuration

Point Configuration

Once your devices are configured, you may configure points for them. Through device point configuration, you may configure the following:

- Points that read or set discretes and/or bits in memory
- Points to read or set CIO, LR, HR, AR and DM memory areas
- Points to read Timer/Counter completion statuses
- Points to read or set Timer and Counter Present Values (PVs)
- Points to read Status and Errors
- Points to set operation mode

The fields described below have configuration values that are unique to, or have special meaning for OMRON TCP/IP communications.

Point General Properties

On the General tab, the type of Access you choose depends on the point address:

- Select Read for points configured for Error reading and Timer/Counter compilation statuses.
- Select Read/Write for all other points configured if you wish to be able to set (change) the value.

Device Point Properties

On the Device tab:

Update Criteria	The update criteria determines how the data will be requested.
	Select On Poll or On Scan for points whose values should be polled by the OMRON TCP/IP driver at static intervals.
	Select On Demand On Scan or On Demand On Poll for points whose values should be polled by the OMRON TCP/IP driver at static intervals while users are viewing them.
Address	Point address requirements are device-dependent. Valid device types are: C-Series *(page 768)* CV-Series *(page 769)*position: static; See the sections below for detailed information on the point address requirements for each device type.

When you configure BOOL points from non-discrete memory (such as CIO, HR or DM) you must also enter data in the following field:

Bit Offset	Enter the bit offset in the 16-bit memory word that defines the point where 0 is the least significant bit and 15 is the most significant bit.
	Warning: Do not enter a data Bit Offset for Point Type BOOL for Timer and Counter Completion Statuses or Temporary Relay (TR). Leave the field as the default 0.

Point Address Formats

Point Address Formats

- C-Series Point Addresses
- CV-Series Point Addresses
- Notes on Addressing

C-Series Point Addresses

For C-Series programmable controllers connected to the local Ethernet Network, the following formats are valid:

Memory Area	Address Entry
Core I/O (CIO)	000 to 255
Link Relay (LR)	LR000 to LR63
Holding Relay (HR)	HR00 to HR99
Auxiliary Relay (AR)	AR00 to AR27
Timer Completion Flag	TIM000 to TIM511
Timer PV	TPV000 to TPV511
Counter Completion Flag	CNT000 to CNT511
Counter PV	CPV000 to CPV511
DM Area	DM0000 to DM9999
Status	ST
Error Read w/ Clear	EC0, EC1
Error Read w/o Clear	ER0, ER1

CV-Series Point Addresses

For CV-Series programmable controllers connected to the local Ethernet Network, the following formats are valid:

Memory Area	Address Entry
Core I/O (CIO)	000 to 255
Temporary Relay (TR)	TR0 - TR7
CPU Bus Link Area (G)	G000 to G255
Auxiliary Area (A)	A000 to A511
Timer Completion Flag	TIM000 to TIM1023
Timer Present Value (PV)	TPV0000 to TPV1023
Counter Completion Flag	CNT000 to CNT1023
Counter Present Value (PV)	CPV0000 to CPV1023
DM Area (DM)	DM00000 to DM24575
Expansion DM (E)	E00000.b to E32765.b where b is the bank number
	(CV-Series Ext. only, CV1000, CV2000, CVM1-CPU11-E,…)

Notes on Addressing

Please note the following about point addresses for the OMRON device communications enabler:

- Letters in address entries can be upper or lower case
- Leading zeros on address numbers are optional. Example: HR003 is treated the same as HR3.
- Address ranges may be limited by the address space available in the controller device.
- The Error Read w/ Clear and Error Read w/o Clear memory areas are read only - Do not configure points with these addresses as setpoints. Refer to the C-Series Ethernet Units manual for details on these memory areas.
- You can use a Status (ST) point as a setpoint to set the operating mode of the controller. When reading Status, the status data from the Status Read response is returned. This data is different from the data written. Refer to the C-Series Ethernet Units manual for details on Status Read and Status Write.
- Use BOOL Point Types for Completion flag (TIM and CNT) and Temporary Relay (TR) points.
- Status (ST), Error Read w/ Clear and w/o Clear are not accessible when you communicate with the programmable controller via a CV bridge.

Adjust the Communications Time-out Parameter

The communications time-out parameter used by the OMRON TCP/IP communications enabler is set to a default value of 7.5 seconds. You can change this parameter in CIMPLICITY software. To do this, add the following record to the Global Parameters file:

<port>_TO|1|<value>

where <port> is last six characters of the communication TCPIP port number assigned when you defined the port in the CIMPLICITY project and <value> is the time out in tenths of seconds. For example, if the port assigned when you defined the port in the project is MASTER_OMTCP0, then the entry would look like the following:

OMTCP0_TO|1|105

sets the time out for communications to 10.5 seconds.

For more information on configuring global parameters, see Global Parameters.

Set Required Protocol Parameters

Three communication parameters can be set via the CIMPLICITY projects global parameters that define information about the computer on which the CIMPLICITY project is running. These values

become part of the FINS structure for each transmitted command allowing the controller responses to find their way back to the enabler. One of the parameters (SNA) has a default, which if acceptable need not be changed. The others, (SA1 and Service Name) are required in order for the OMRON Ethernet device communication enabler to operate. These three parameters act as a return address and must be defined properly or you will receive time-out error messages even though the controller(s) are responding.

The three parameters are:

Parameter	Value Range and Description
SNA	This is the network number assigned to the Ethernet network in the OMRON Ethernet module setup. The SNA entry need not appear in the global parameters if the default is acceptable. Valid values are: 0 = Local network (default) 1 through 127 = Remote network
SA1	This is the Source Node Number for the computer initiating the communication. See Application Notes for information on how this is used. Valid values for the Source Node Number are: 1 through 126
Service name	This corresponds to an entry in the operating system's services file. The entry relates the name to a UDP port number for which the OMRON Ethernet Units have been configured.

To define these parameters, add the following records to the Global Parameters file:

```
<port>_OMRON_SNA|1|<value>

<port>_OMRON_SA1|1|<value>

<port>_OMRON_SERVICE|1|<value>
```

where <port> is last six characters of the communication TCP/IP port number assigned when you defined the port in the CIMPLICITY project. <value> is either the SNA value the SA1 value or the service name per the descriptions in the above table. For example, if the port assigned when you defined the port in the project is MASTER_OMTCP0, then these entries:

```
OMTCP0_OMRON_SNA|1|3

OMTCP0_OMRON_SA1|1|15

OMTCP0_OMRON_SERVICE|1|omron_service
```

set the SNA to 3, the SA1 to 15, and the Service Name to omron_service.

For more information on configuring global parameters, see Global Parameters.

Default Protocol Parameters

Certain protocol parameters used by the OMRON TCP/IP communications enabler have been preset. They are:

Parameter	Value	Description
DA2	0	Unit Address - 0 indicates the programmable controller's CPU. This is not the same as the unit number (that is, node number) which is configurable.
SA2	FE	Source of communication unit
SID	FF	Service ID.

Technical Support Worksheet

Technical Support Worksheets

Please complete and review the following information before contacting technical support if you are unable to establish communications with a controller.

CIMPLICITY Project Information

Enter the following information about the CIMPLICITY project that uses OMRON TCP/IP communications:

The IP address for the computer running the CIMPLICITY project:	_____
The netmask setting for your Ethernet network. This determines the network "class". Also referred to as the subnet mask.	_____
The SNA value defined via the global parameter file (if the default of zero is over ridden).	_____
The SA1 value defined via the global parameter file.	_____
Host name assigned to the target controller or the controller acting as a gateway if the target is on a subnetwork.	_____
The DNA value entered for the target controller. This is the <subnetwork number>.	_____
The DA1 value entered for the target controller. This is the <address>.	_____
Complete entry for the address of the controller in the Address field of the Default properties page in the Device Configuration dialog box.	_____

Make sure you have printouts of the following files on hand:

- The operating system's services file.

Locate the service name used to get the UDP Port Number for binding the communication socket and the Service Name defined for project.

- The hosts file.

Locate the Host Names and IP addresses for the controllers with which you will be communicating.

- The glb_parms.idt file.

Locate any global parameters you defined for OMRON TCP/IP communications.

OMRON Ethernet Module Configuration Information

If the targeted controller is not connected to the Ethernet directly but is on a subnetwork, then provide the following information on the CV controller connected to the Ethernet network that is acting as a gateway:

The IP address configured for the module	_____
The subnet mask configured for the module.	_____
UDP FINS Port Number configured	_____
Type of IP address conversion configured, table lookup or automatic address generation.	_____
Node Number Rotary Switch setting.	_____
The SA1 value defined via the global parameter file.	_____

Targeted OMRON Controller Information

For each targeted OMRON controller, get the following information:

The destination network address (DNA).	_____
The destination node number (DA1)	_____
Routing table configured for gateway (if target is not connected to Ethernet)	_____

Testing Results

Run ping and have information about the following on hand:

- Results from performing a ping to the controller's IP address (for example, ping 130.0.25.43).
- Results from performing a ping to the host name (for example, ping controller_1).

Run the OMRON TCP/IP Diagnostics program and have information on hand about field settings test results and any error messages displayed in the dialog box.

Status Log Messages

Check the Status Log Viewer for the following:

- Any messages displayed in the Status Log as the OMRON TCP/IP communications enabler starts or tries to start.
- Any messages displayed in the Status Log as the OMRON TCP/IP communications enabler tries to communicate to devices.

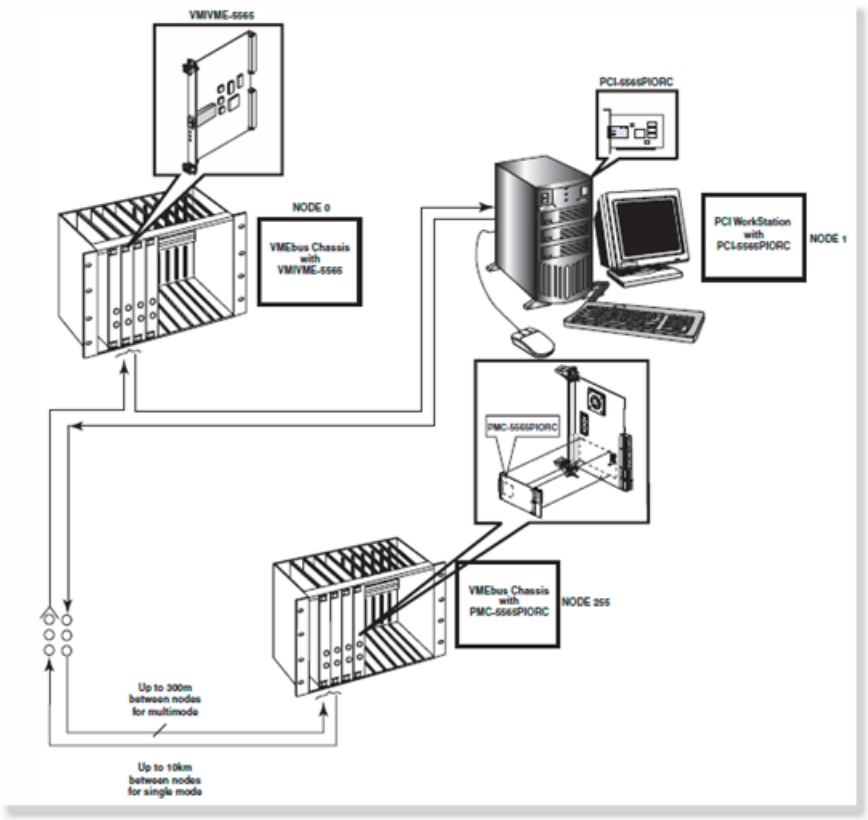
In addition, have printouts of the relevant messages on hand.

Chapter 47. Reflective Memory Device Communications

About Reflective Memory Device Communications

The Reflective Memory communications enabler can be used to support communications with other devices with Reflective Memory cards installed including GE's PACSystem RX7i and RX3i Device Controllers with RMX or CMX cards installed.

The communication network using reflective memory is based on several reflective memory (RFM) cards connected via fiber optic cables into a physical ring. Each RFM card automatically takes fiber channel data transfer's that received on the RX side of its connection, transfers that packet's data into its internal memory space, and then re-transmits that same packet on the TX side of its connection to other nodes. Once a RFM interface receives a packet with the same Network ID, it does not re-transmit that packet removing it from the network.



All reflective memory cards run autonomously; they are standalone communication processors. The built-in firmware and processor ensures that whenever data is written to the internal memory space

of the card, this written data is populated via ultra-fast fiber optic network to all other connected reflective memory cards.

The CIMPLICITY Reflective Memory device communication enabler can read and write to the internal memory space available on the reflective memory card inside the PC. Because the network creates a shared memory space between different types of applications with no explicit network control overhead, if multiple devices are specified to write to the same memory address within the reflective memory space, the applications should implement a feature or control method within the application to coordinate the writes.

Reflective Memory Hardware Requirements

To use the Reflective Memory interface, you need to have an available PCI or PCI Express slot for each card that will be installed.

The following cards are supported for use in the PC:

	Product Name	Form Factor	Memory Capacity	Bit Rate (Gb/s)s)
	PCIE-5565PIORC	Low Profile PCI Express	128 or 256 MB SDRAM	2
	PCI-5565PIORC	PCI (64 bit)	128 or 256 MB SDRAM	2
	PCIE-5565RC	PCI Express	128 or 256 MB SDRAM	2
	PCI-5565 (discontinued)	PCI (64 bit)	128 MB	2

GE software supports up to 2 cards per computer. However, not all computers can support 2 cards due to hardware or BIOS specific limitations.

Reflective Memory Related Documents

The following documents may be accessed by following the hyperlinks below. Registration at the Website is required before the documentation can be accessed and/or downloaded.

• Real-Time Networking with Reflective Memory (GF 630A)
• Reflective Memory (Brochure)
• RFM2g Network and Shared Memory Driver (Brochure)

• Hardware Reference PCI-5565PIORC Reflective Memory Node Cards Ultra High-speed, Fiber Optic Network for Distributed Processing Using Reflective Memory, 500-855565-000
• Hardware Reference PCIE-5565RC Reflective Memory Node Cards Ultra High-Speed, Fiber Optic Network for Distributed Processing Using Reflective Memory, 500-9300875565-000
• Hardware Reference VMIPCI-5565 Ultra-High Speed Fiber-Optic Reflective Memory for Distributed Processing Using Reflective Memory (discontinued), 500-855565-000
• Operator’s Reference Common RFM2g* Application Program Interface (API) and Command Line Interpreter for VMISFT/RFM2g* Drivers, 523-000447-000
• ACC-5595 2 Gb/s Reflective Memory Hub Assembly (Brochure) – Managed Reflective Memory Hub data sheet
• PACSystems RX7i Installation Manual, GFK-2223
• PACSystems Memory Xchange Modules Users' Manual, GFK-2300

Reflective Memory Card: Installation and Configuration

Reflective Memory Card: Installation and Configuration

- Reflective Memory card: Installation
- Reflective Memory card: Configuration utility.
- Reflective Memory card: Parameters.

Reflective Memory Card: Installation

1. Set the required switches on each card to be installed in the PC.

   Some of the required settings controlled by the switch settings are as follows.

   - Node ID
   - Redundant Transfer Mode

- Rogue Packet Detection and Removal

A more complete explanation of the switches and their impact on operation of the network can be found in the hardware reference manual for your card.

Node Id

Each card on the network must have a unique Node Id to identify itself. Nodes are numbered from 0 to 255. The factory settings for the reflective card set the Node Id to be 0 (zero). There are no reserved network node Id's, but there is no inherent detection of duplicate nodes on a network.

Redundant Transfer Mode

When redundant transfers are enabled, each packet is transferred twice. If the first packet is received without error by the receiving circuitry, then the second transfer is discarded. If an error occurs on the first transfer and the second transfer has no transmission errors, then the second transfer is used to update the onboard memory. If both transfers are unsuccessful, the data is not re-transmitted on the network. Using a redundant transfer mode reduces the likelihood that packets will be dropped from the network, but at a significant cost of a lower network data transfer rate.

Rogue Packet Detection and Removal

A rogue packet is a packet that does not seem to belong to any node on the network. If the packet is altered as it passes through a non-originating node or if the originating node begins to malfunction, the originating node may fail to recognize the packet as its own and will not remove the packet from the network. In such circumstances the packet passes around the network loop indefinitely.

Rogue packets are rare. Their existence indicates a malfunctioning board due to component failure or operation in an overly harsh environment. Normally, the solution is to isolate and replace the malfunctioning card or improve the environment. However, in some applications it is preferable to tolerate sporadic rogue packets rather than halt the system for maintenance, provided the rogue packets are removed from the network.

To prevent rogue packets from circulating on the network indefinitely, reflective memory cards can be configured to operate as one of two rogue masters.

A rogue master alters each packet as it passes through its node. If that packet returns to the rogue master a second time, the rogue master recognizes that it is a rogue packet and removes it from the network.

The reflective memory network supports up to two rogue masters per network, Rogue Master 0 and Rogue Master 1, so they can cross check each other.

Important: There should no more than one Rogue Master 0 configured per network and no more than one Rogue Master 1 configured per network. Otherwise, each will erroneously remove packets originated by the other.

2. Install the reflective memory card(s) in the PC.

3. Use the utility `CfgRFMCard` to run the Reflective Memory configuration utility and to confirm the hardware switch settings including the:
   a. Node Id.
   b. Redundancy Mode.
   c. Rogue Master Setting.

Reflective Memory Card: Configuration Utility

The Reflective Memory Card Configuration utility can be used to install the reflective memory device driver and to confirm hardware switch settings and card configuration as well as the PCI Bus settings. Some soft switches can also be modified through the utility.

Important

1 *(page 779)*	Run the CfgRFMCard utility.
2 *(page 780)*	Select a Configure RFM Card.
3 *(page 780)*	Detect the Configure RFM Card.
4 *(page 780)*	Configure the Configure RFM Card.

Run the CfgRFMCard utility

The following steps are required to install or confirm installation of the kernel level driver for the Reflective memory card.

1. Enter the command, `CfgRFMCard`, in the **Command Prompt** field, or select All Programs->HMI SCADA – CIMPLICITY <version>->Reflective Memory Configuration.
   - On Windows XP and Windows 2003 Server, you must be a user with administrative privileges to successfully use the `CfgRFMCard` utility.
   - On all other supported operating systems, the program requires elevation to run.
2. When the utility is initiated, if required, the User Account Control will prompt the user for permission to continue. The program will identify itself as CfgRFMCard.exe with GE

Intelligent Platforms, Inc. as its publisher. When user requirements are met, the initiated utility will check to see if the expected version of the driver is installed and associated with the reflective memory hardware.

3. Follow through with driver identification. If the driver is detected, continue configuration by selecting the card. If the driver is not detected, a message will ask if you want to install the driver. If you indicate that the driver is to be installed, a wizard will walk you through the driver installation.

Select a Reflective Memory Card

After the driver detection portion is complete, you will be prompted to select a card.

- If no cards are available for selection, then one of the following occurred.
- The reflective memory device driver did not install
- No cards are associated with the reflective memory hardware.
- One or more cards will be installed if the driver installation was successful.

(If no Cards are Detected) Troubleshoot the Problem

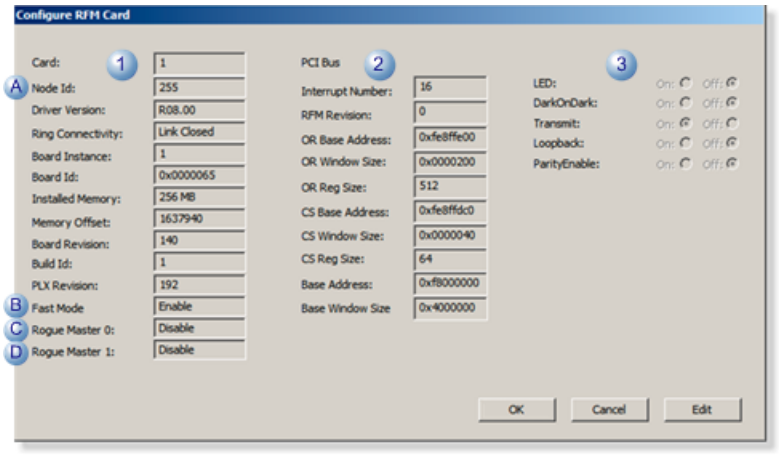
If no cards are detected do the following to troubleshoot the problem.

1. Click Cancel to exit the utility.
2. Open the Windows Device Manager. (Enter `devmgmt.msc` at the Command Prompt.)
3. Click the Scan for hardware changes button. Windows will attempt to find the cards.
4. Do one of the following depending on whether or not the card is found:

If the Cards:	Then:
Are not detected	Contact GE Intelligent Platform's GlobalCare (Embedded Hardware).
Are found.	Execute the `CfgRFMCard` utility; select the card to be tested from the drop down menu.

Configure the Configure RFM Card

After selecting the reflective memory card, a dialog box similar to the following opens.



Columns show the following:

Number	Shows
1	Some of the reflective memory card settings.
2	Some of the PCI bus specific parameters.
3	Software selectable switches.

The following should match the switch settings configured for your card:

Letter	Description
A	Node Id.
B	Fast or Redundant Mode.
C	Rogue Master 0.
D	Rogue Master 1.

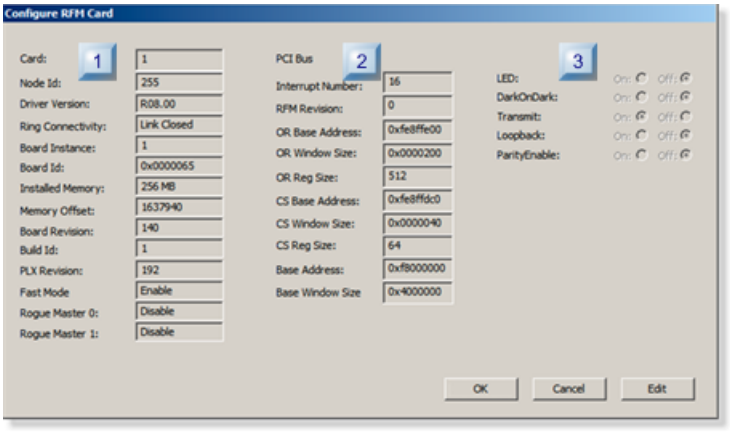
Buttons do the following, when clicked:

Button	Description
Edit	Modify one or more of the software selectable switches. If Edit is clicked, modify the software switch settings as required for your application.
OK	Save the configuration and exit the utility.
Cancel	Cancel the changes and exit the utility.

Note: If any settings are incorrect, check your switch settings again. If you have questions (e.g. you are not sure how to set the switch settings or if the values match the configured settings) contact GE Intelligent Platforms' GlobalCare (Embedded Hardware).

Reflective Memory Card: Parameters

Parameters on the Configure RFM card are as follows.



rect 347, 24, 370, 48 *(page 783)*
rect 212, 22, 235, 46 *(page 783)*
rect 46, 24, 69, 48 *(page 782)*

1 *(page 782)*	Reflective Memory card parameters.
2 *(page 783)*	PCI Bus parameters.
3 *(page 783)*	Software Selectable Switch Settings (read and writeable).

1	Reflective Memory Card Parameters

Parameter	Description
Card	Uniquely identifies the reflective memory card installed in the computer. The first card is card 1.
Node Id	Uniquely identifies the reflective memory card on the network. Valid values are 0 to 255.
Driver Version	Version of the GE device driver.
Ring Connectivity	Indicates whether there is ring continuity through all nodes in a ring.
Board ID	Board Id of the currently open reflective memory device.
Installed Memory	Total amount of board memory.
Memory Offset	Memory offset of the reflective memory board.
Board Revision	Revision of the reflective memory card.

Build Id	Build ID of the revision of the reflective memory card.
PLX Revision	Revision of the PLX chip.
Fast or Redundant Mode	Fast mode indicates non-redundant data transfer. Redundant mode indicates redundant data transfer.
Rogue Master 0	If enabled, indicates that the board is operating as the Rogue Master 0 on the network. This setting is controlled by a switch setting on the reflective memory card. There should be no more than one Rogue Master 0 on the network.
Rogue Master 1	If enabled, indicates that the board is operating as the Rogue Master 1 on the network. This setting is controlled by a switch setting on the reflective memory card. There should be no more than one Rogue Master 1 on the network.

2	PCI Bus Parameters

Parameter	Description
Interrupt Number	Indicates interrupt.
RFM Revision	Revision of the reflective memory card.
OR Base Address	Memory Base Address for access to Local Configuration registers. (Register 0).
OR Window Size	Size of PCI Window.
OR Reg Size	Actual Memory space size.
CS Base Address	Memory Base Address for access to Control and Status register info. (Register 2).
CS Window Size	Size of PCI Window.
CS Reg Size	Actual memory space size.
Base Address	Memory Base Address of Reflective Memory Board Memory.
Window Size	Size of the PCI Memory Window.

3	Software Selectable Switch Settings (read and writeable)

Parameter	Description
LED Light	State of the STATUS LED light on the reflective memory card.
DarkOnDark	State of the reflective memory card’s Dark On Dark feature. When the feature is On, the board’s transmitter will be turned Off if the receiver doesn’t detect a signal or if the receiver detects invalid data patterns. This feature is useful in hub configurations.
Transmit	State of the reflective memory card’s transmitter.
Loopback	State of the transmit loopback state. When On, the fiber optic transmitter and receiver are disabled and the transmit signal is looped back to the receiver circuit internally. This allows basic functional testing without an external cable.

ParityEnable	State of the reflective memory card's Parity Enable feature. When `ParityEnable` is enabled, parity is checked on all onboard memory accesses. While the parity is On, writes to the memory are only allowed as 32-bit Lwords or 64-bit Qwords. Write accessing data as 16-bit words or 8-bit bytes is prohibited. If you enable parity checking on one node, you should enable it on all nodes of the reflective memory network. **Note:** A node that detects an invalid memory write from the network will prevent the write to its own memory, but it will not remove the packet from the network

Reflective Memory Application: Configuration

Reflective Memory Application: Configuration

When you configure ports, domain configuration, devices and device points you will need to enter some data that is specific to your Reflective Memory communications.

The following sections review the configuration requirements:

- Reflective Memory domain: Configuration
- Reflective Memory port: Configuration.
- Reflective Memory device: Configuration.
- Reflective Memory point: Configuration.

Reflective Memory Domain: Configuration

Reflective Memory Domain: Configuration

Typically, different devices write to one or more blocks of data on the reflective memory card. The CIMPLICITY station may have its area to write to, but mostly it reads the data other devices write.

Each block of the reflective memory card is defined to be of a certain size and addressable by any one of the following.

- Byte
- Word (2 bytes)
- Quad word (4 bytes) addressable.

The CIMPLICITY Reflective Memory Devcom configuration tool does the following.

1. Provides the ability to define how the blocks of data will be used by the reflective memory devcom

2. Creates a structure for the reflective memory.

In configuration, each memory block is associated with a model. The model may reflect a particular physical device or some other logical division defined by the user.

- Reflective Memory Domain: Open the Reflective Memory DEVCOM Configuration Tool
- Reflective Memory Domain: Memory Layout

Reflective Memory Domain: Open the Reflective Memory DEVCOM Configuration Tool

In the Workbench left pane:

1. Right-click **ConfigTool**.
2. Select Properties on the Popup menu.
3. Right-click **ConfigTool**.
4. Select Properties on the Popup menu.

Reflective Memory Domain: Memory Layout

1. Select a row.
2. Click Move Up or Move Down.
3. Select a row
4. Click Delete.

A message box opens for confirmation; once confirmed the row is deleted.

Reflective Memory: Port Configuration

1. **Retry Count** Indicates the number of times communications will be attempted before the device is considered to be down.

2. A $DEVICE_DOWN alarm is generated. Once a device is down, periodic attempts are made to resume communications.
   Note: Device up/down is determined by the ability to read and write to the reflective memory card only. Down or defective nodes on the fiber network used by the reflective memory card will impact the ability for the card to receive updates, but will not affect the ability to read and write to the card.

Reflective Memory: Device Configuration

1 *(page 786)*	Create a device for reflective memory communications.

2 (page 786)	Enter general information for the device.
3 (page 787)	Enter default information for the device.

Create a Device for Reflective Memory Communications

1. Expand Project>Equipment in the Workbench left-pane.
2. Right-click Devices; select New on the Popup menu.

A New Device dialog box opens.

1. Select the following in the New Device dialog box fields.

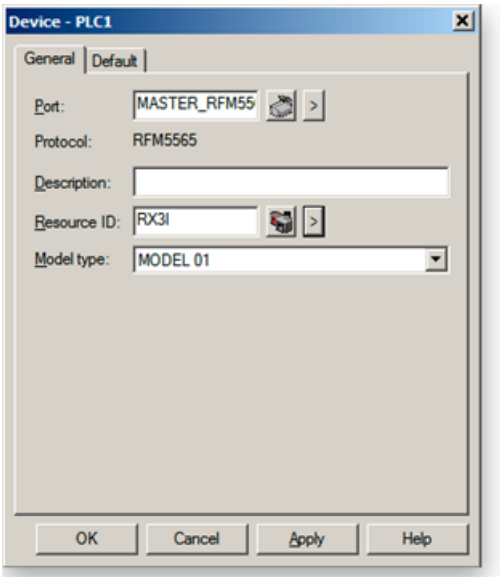
Field	Select
Protocol	RFM5565
Port	RFM5565

1. Click OK.

Result: The Device dialog box opens.

Enter General Information for the Device

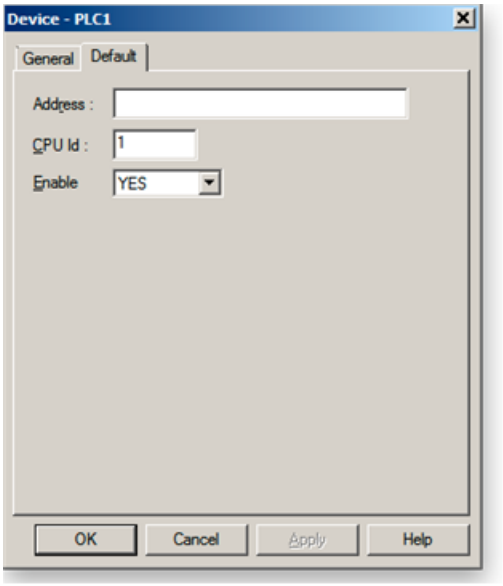
Fields on the General tab in the Device dialog box are as follows.



Field	Description		
Port	Port for the selected device. **Note:** Click the buttons to the right of the **Port** field to select the port.		
	Browse		Opens the Select a Port browser.
	Popup		Create a new port, edit the current port or select a port from the list of ports.
Description	(Optional) Description to help you identify the device.		
Resource	Resource that can be associated with this device for alarm generation. Click buttons to the right of the **Resource** field to select the resource.		
	Browse		Opens the Select a Resource browser.
	Popup		Create a new resource, edit the current resource or select a resource from the list of resources.
Model Type	Type of device. For this protocol, options in the drop down list are: MODEL 01 MODEL 02 MODEL 03 MODEL 04 MODEL 05 MODEL 06 MODEL 07 MODEL 08 MODEL 09 MODEL 10 MODEL 11 MODEL 12 MODEL 13 MODEL 14 MODEL 15 MODEL 16 The choice of models determines the defined domain information for the device. This information is configured in the Reflective Memory Devcom configuration tool.		

Enter Default Information for the Device

Fields on the Default tab in the Device dialog box are as follows.



Field	Description	
Address	The **Address** field should be blank.	
CPU Id	The Reflective memory card number from which the data is to be collected. **Important:** A maximum of 2 cards is supported per PC; however, not all PCs can support 2 cards. To address the first card, enter 1 in the **CPU Id** field.	
Enable	Enable options are, as follows.	
	YES	Enables the device when the project starts.
	NO	Disables the device, resulting in the device being marked down. **Important:** When the device is disabled, all points associated with the device will be unavailable.

Reflective Memory: Point Configuration

Once your devices are configured, you m ay configure points for them.

The fields in the Point Properties dialog box have configuration values that are unique to the Reflective Memory device communications.

1 *(page 788)*	Enter general information for a reflective memory point.
2 *(page 789)*	Enter device information for a reflective memory point.

1. Enter General Information for a Reflective Memory Point

On the General tab in the Point Properties dial box, you can select whether the points are read only or read/write.

Important: :

With the reflective memory device communication interface, it is recommended that multiple devices do not have write access to the same mapped memory in reflective memory without implementing an application level scheme to synchronize the writes and prevent concurrent or nearly concurrent updates.

1. Enter Device Information for a Reflective Memory Point

Fields on the Device tab in the Point Properties dialog box are as follows.

Field	Description		
Update criteria	This device communication interface only supports polled update criteria. When polling is performed, the frequency of data collection is based on the port's configured scan rate multiplied by the point's configured scan rate.		
	Update Criteria	Definition	
	On Change	Update the point value if the most recently read value is different than the current point value. Note that if an analog deadband is configured, the value read must differ by an amount greater than the configured analog deadband before the point value will be updated.	
	On Demand On Scan	When an application is requesting the data, and the value is read is collected, the point value is updated.	
	On Demand On Change	When an application is requesting data and the value read is different than the current point value, then the point value is updated. If an analog deadband is configured, the value read must differ by an amount greater than the configured analog deadband before the point value will be updated.	
Address	The specification of the address is based on the model selected and the domain configuration configured by the Reflective Memory DEVCOM Configuration Tool. For a project that used the default domain configuration the valid address range would be as follows:		
	Address Format	Numeric Range	Native format
	Qn	1 – 8192	Byte (Little Endian)
	Rn	1 – 4096	Word (Little Endian)
Poll After Set	Select this only if there is a need to read the value from the device at a rate greater than the actual poll rate.		

PACSystems Support for Reflective Memory

PACSystems PLC support reflective memory cards, as follows.

- Write data from reflective memory.
- Read data from reflective memory.

Write Data from Reflective Memory

On PACSystems, there are four functions that allow an application running on a PAC System CPU to communicate with a reflective memory card:

- `BUS_WRT_DWORD` or `BUS_WRT_WORD` to write data to the network.
- `BUS_RD_DWORD` or `BUS_WRT_WORD` to read data from the network.

The following examples are for use with a PAC PLC application interfacing to a RFM interface (named a `CMX` interface).

Note: Examples for the 9070 are not currently available.

Typical write `VME` functions take the following parameters, related to the location of the `CMX` module (defined in the Hardware Configuration portion of the PLC application) and the memory that is to be written:

Parameter		Description
(enable)		
data	IN	PLC Memory space that is to be written to the Memory Xchange module, *note this must match in byte size the number of items written.
Rack	R	Rack where the Memory Xchange module is located.
Slot	S	Slot where the Memory Xchange module is located.
Subslot	SS	Set to 0, unused with memory Xchange modules.
Region	RGN	Which of the 4 memory regions in the memory Xchange module.
Offset	OFF	0-based number indicating what portion of the memory region is to be accessed.
DWord		Number of DWORDs to be written.
Output	ST	Status of the operation.
Reference	In	Reference for the Data to be written.

For detailed instructions, see PACSystems Memory Xchange Modules Users' Manual, GFK-2300.

In the following PAC Ladder example, the application will:

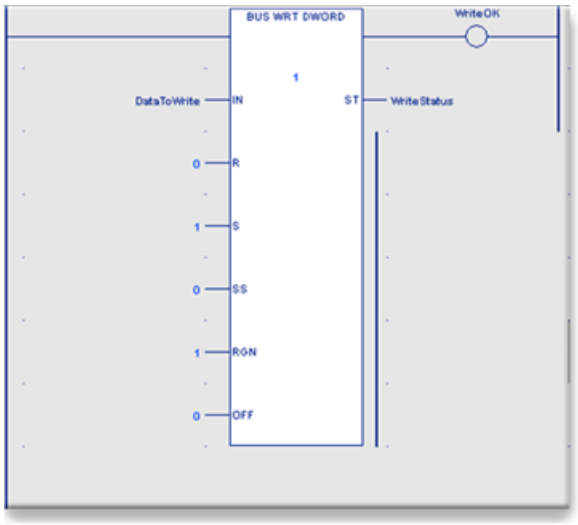
- Write every scan as a single DWord of data to an RFM interface card that is located at

1. Rack is 0

2. Slot is 1

3. Region is 1
    - Take the data in the variable `DataToWrite` and writes it.

The instruction writes 1 double word and if the bus operation is successful, the function will pass power and the ST output will have a value of 0.



Read Data from Reflective Memory

In the following example, data from the Memory Xchange module is read from Rack is 0, Slot is 7, Region is 1, Offset is 1024 bytes into the memory, and the result of the read will be 1 Double word of data placed into the PLC variable ReadData. The Bus Read is immediate, and with the ReadOk contact being ON, and the ReadStatus being a value of 0, the contents of that memory location on the CMX card will be in the variable specified.

These functions all take the following parameters:

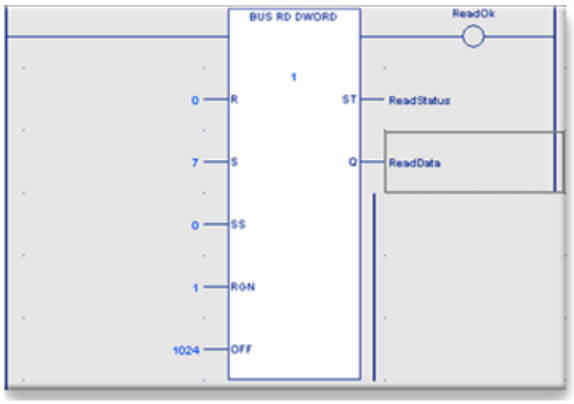
Parameter		Description
(enable)		
Rack	R	Rack where the Memory Xchange module is located.
Slot	S	Slot where the Memory Xchange module is located.
Subslot	SS	Set to 0.
Region	RGN	Which of the 4 memory regions in the memory Xchange module.
Offset	OFF	0-based number indicating what portion of the memory region is to be accessed.
DWord		Number of DWORDs to be read.
Output	ST	Status of the operation.

Reference	Q	Reference where read data is to be placed.

If the bus read operation is successful, the ST output will have a value of 0.

If the `Bus_RD` function passes power, the data requested is present and valid.

If the `Bus_RD` function does not pass power, the data will hold last state, and will not be changed, or cleared.



Chapter 48. S90 Triplex Communications Device Conversion

S90 Triplex Communications Device Conversion

The S90 Triplex communications device is a powerful, easy-upgrade for the S90 TCP/IP device communications.

Step 1 *(page 793)*	Open the Convert Project to Series 90 Triplex dialog box.
Step 2 *(page 794)*	Select Conversion to S90 Triplex project options.
Step 3 *(page 795)*	Convert the Project to use an S90 Triplex communications device.

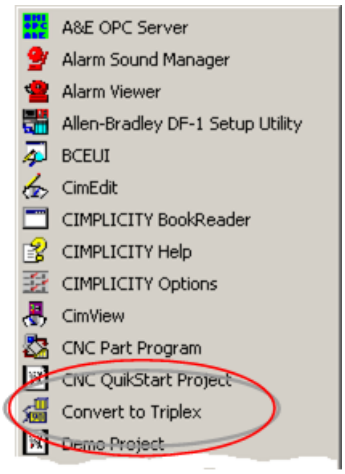
- This utility will successfully convert most projects to use Series 90 Triplex.
- The Series 90 Triplex option must be selected and loaded when CIMPLICITY is installed.
- You cannot convert an Ethernet Global device (EGD) in your configuration the project will not be converted.
- If you have a cable redundant Series 90 TCP/IP device communication interface with a cable redundant PLC each cable redundant PLC will be converted to Simplex PLC device model.

Please review the Series 90 Triplex documentation to determine additional setup requirements for your application.

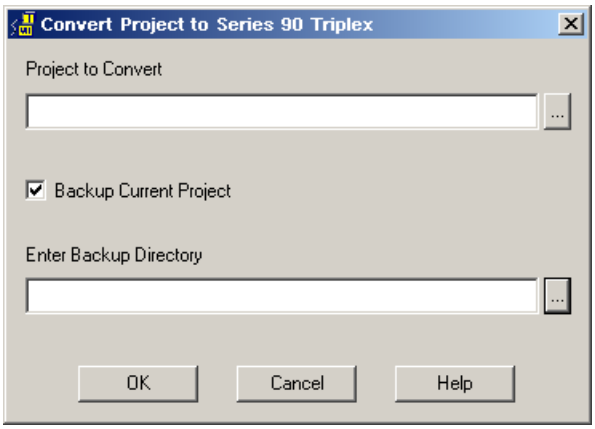
- Unused ports will be deleted as part of the conver sion.
- Diagnostic points specific to the Series 90 Communication options are not modified. They will be invalid points when the project is converted.

Step 1. Open the Convert Project to Series 90 Triplex Dialog Box

1. Click Start on the Windows task bar.
2. Select (All) Programs> HMI SCADA - CIMPLICITY version>Convert to Triplex.

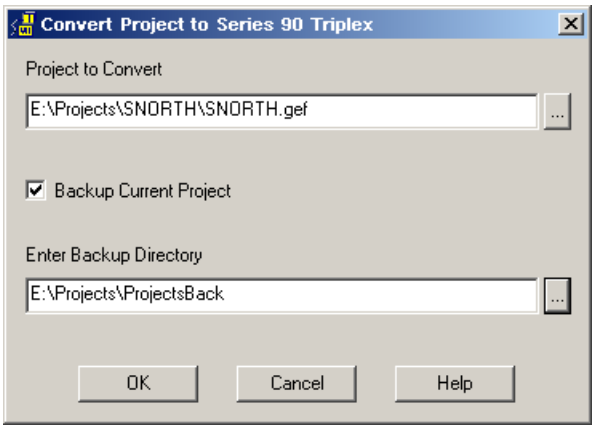


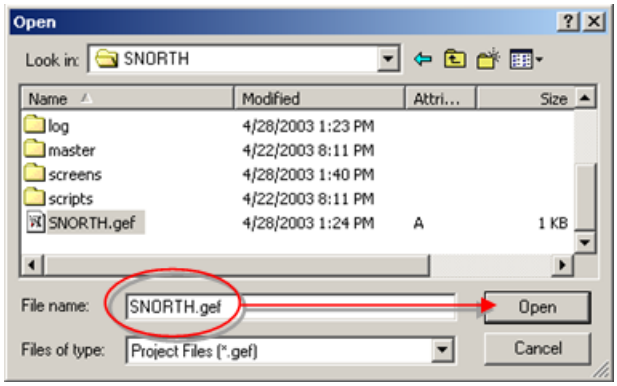
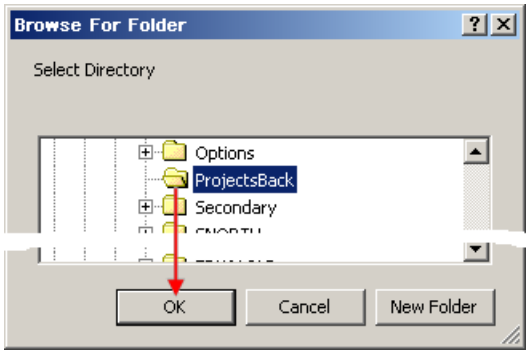
The Convert Project to Series 90 Triplex dialog box opens.



Step 2. Select Conversion to S90 Triplex Project Options

Select conversion options are as follows.

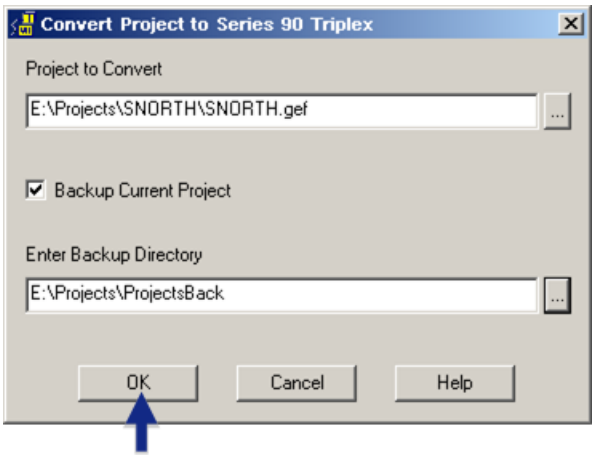


Option	Description
Project to Convert	Select a project using S90 Redundancy or Series 90 Ethernet communication options.
	... Opens an Open dialog box to find and select the project to be converted.  Click Open to select the project .gef file
Backup Current Project	(Recommended) Check to create a copy of the current project configuration.
Enter Backup Directory	Select the directory in which the backup project folder will be placed. This directory must be different from the directory in which the selected project is located. Example The SNORTH project directory is in the Projects directory. The backup project directory will also be named SNORTH. Therefore, the backup project needs to be in a directory that is different from the Projects directory.
	... Opens a Directory browser to find and select the backup directory.  Click OK to select the folder in which the backup project will be placed.

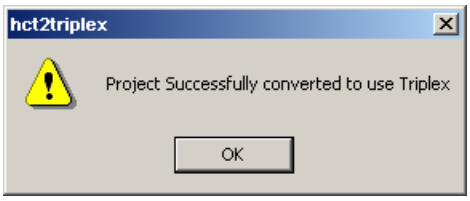
Result: The CIMPLICITY project is ready for conversion.

Step 3. Convert the Project to use an S90 Triplex Communications Device

Click OK in the Convert Project to Series 90 Triplex dialog box.



Result: CIMPLICITY converts the project to use Triplex. A message displays when the conversion is successfully completed.



Note: You can click Cancel before the conversion starts to stop and exit the conversion.

Chapter 49. Series 90 TCP/IP Triplex Communications

About Series 90 TCP/IP Triplex Communications

The Series 90 TCP/IP Triplex communications enabler supports standard (non redundant) and both cabling and PLC redundancy for Series 90 TCP/IP communications.

- Standard configuration allows CIMPLICITY to communicate with a Series 90 or VersaMax interface through a single Ethernet connection.
- PLC redundancy describes the ability to configure a group of PLCs to act as a single logical data source.
- Cabling redundancy describes the ability to configure two network paths to the same PLC.

This communication interface supports communication to the Series 90 and VersaMax products, which have an installed/configured Ethernet Interface.

For redundant communication, the enabler supports the standard Series 90-70 hot-standby, synchronized CPU and GMR system products, Trimations HBR-30, HBR-70, and TMR-30 products and custom redundancy applications based on Series 90 PLC in single and cable redundant applications.

This enabler supports the following CIMPLICITY features:

- Polled reads at user defined rates
- Poll after setpoint
- Triggered reads
- Unsolicited data
- Analog deadband
- Alarm on communications failure
- Server Redundancy configurations

This enabler supports the following data types:

- Signed 8, 16, and 32 bit integers
- Unsigned 8, 16, and 32 bit integers
- Floating point
- Text
- Arrays

Series 90TCP/IP Triplex Standard Configurations

The Series 90 TCP/IP Triplex communications enabler allows CIMPLICITY to interface to individual Series 90 and/or VersaMax PLCs. Communication is established and maintained to each PLC through a single Ethernet connection to the device. One Ethernet address is assigned to each PLC. Unlike Redundant PLC configurations, non-redundant communication is not contingent on health, mode or run bits in the PLC.

Series 90 TCP/IP Triplex Supported Devices

The Series 90 TCP/IP Communications enabler currently supports the following devices:

- Series 90-70 programmable controllers
- Series 90-30 programmable controllers
- Versamax programmable controllers
- PACSystems RX7i Firmware version 2.0 and earlier
- PACSystems RX3i Firmware version 2.0 and earlier

You can use any of the following Device Communication enablers to communicate with a Series 90 PLC over Ethernet, but you cannot use them at the same time, on the same computer:

- Series 90 Ethernet
- S90 Redundancy
- S90 TRIPLEX

For redundant communication, the enabler supports the standard Series 90-70 hot-standby, synchronized CPU and GMR system products, Trimations HBR-30, HBR-70, and TMR-30 products and custom redundancy applications based on Series 90 PLC in single and cable redundant applications.

Note: The Project wizard will only browse for devices using the Series 90 Ethernet Device Communications.

Series 90 TCP/IP Triplex Supported Memory Types

The Series 90 TCP/IP Communications enabler supports reading and writing point data to the PACSystems, Series 90-70, and Series 90-30 controllers.

Note: If you add a block to a programmable controller or resize domains, you must stop and restart the Series 90 TCP/IP Communications enabler in order for it to see the changes.

Data may be read from the following memory types:

Memory Type	Description	Memory Type	Description
%AI	Analog Input Table	%R	Register Memory
%AQ	Analog Output Table	%S	System Fault Table
%G	Genius Seamless	%SA	Special Contacts A
%I	Discrete Input Table	%SB	Special Contacts B
%L	Local Memory *	%SC	Special Contacts C
%M	Discrete Internal	%T	Discrete Temporary
%Q	Discrete Output Table	%W	Bulk Memory

* %L may only be used on Series 90-70 and PACSystems PLCs only

** %W may only be used on PACSystems RX3i and PACSystems RX7i

Data may be written to the following memory types:

Memory Type	Description	Memory Type	Description
%AI	Analog Input Table	%M	Discrete Internal
%AQ	Analog Output Table	%Q	Discrete Output Table
%G	Genius Seamless	%R	Register Table
%I	Discrete Input Table	%T	Discrete Temporary
%L	Local Memory *	%W	Bulk Memory

* %L may only be used on Series 90-70 and PACSystems PLCs only

** %W may only be used on PACSystems RX3i and PACSystems RX7i

Note: CIMPLICITY supports domain sizes of up to 65536 (64K) bytes only; however %W can be configured to a maximum size of 5242880 words (10M bytes). Therefore, the entire %W memory range cannot be accommodated within one domain.

Warning: Do not write data to the system registers (%S, %SA, %SB, and %SC). Doing so may interfere with the normal operation of the programmable controller.

Series 90 TCP/IP Triplex Hardware Configuration Requirements

Target Series 90-70 programmable controllers requires the following:

- A TCP/IP communcations module (IC697CMM741H PROM Version 1.12 or later or IC697CMM742FE Firmware 2.6 or later)
- Supporting LAN software (C651ENS042, containing GSM version 2.09 or later, TCP executive 1.28 or later, and TCP/IP Configuration editor 1.02 or later if using IC697CMM741)
- CPU firmware must be release 5.5 or later except on IC697CPU780 where the firmware must be at 4.7. When IC697CPU780 is used with a IC697CMM742, it must be setup as described in GFK-1314A.

Target Series 90-30 programmable controllers require the following:

- A TCP/IP Ethernet communications module (IC697CMM321).
- CPU firmware must be release 5.0 or later

Target Series Versamax

- CPUE05 CPU firmware must be 2.31 or later
- Versamax Micro and Versamax Micro Firmaware 1.0 or greater with IC646SET001 Versamax SE- Ethernet Bridge

If you are using Logicmaster:

- Logicmaster version 4.0 or later must be used with Series 90-70 programmable controllers.
- Logicmaster version 6.0 or later must be used with Series 90-30 programmable controllers.
- CIMPLICITY ME Version 3.0 or later
- VersaProVersion 2.0 or later

Warning: Do not use the Series 90 TCP/IP Communications enabler with a Series 90-70 programmable controller using C-blocks in the ladder unless the Series 90-70 CPU has been upgraded to version 6.0 firmware. The programmable controller may crash otherwise.

Series 90 TCP/IP Triplex Redundancy

Series 90 TCP/IP Triplex Redundancy

The Series 90 TCP/IP Triplex communications enabler allows CIMPLICITY to interface to PLC groups comprising up to 3 PLCs with up to 2 network paths per device.

Cabling redundancy operates in Hot-Standby mode; it communicates to the PLC using the first path while monitoring the second path. It switches automatically, without PLC intervention, to the second path if the first path fails. It continues to monitor the first path and will automatically switch back to the first path on recovery.

PLC redundancy also operates in Hot Standby mode, it communicates with the highest-ranking PLC and automatically switches to a lower ranking PLC in the event of the failure of the highest-ranking PLC. It continues to monitor the PLCs and will automatically switch back to a higher-ranking PLC on recovery.

CIMPLICITY software maintains connection and status information for all the PLCs in a PLC group, but only reads data from the active server. Where unsolicited communication is configured and processed. When unsolicited communication is configured, data will be accepted from any of the PLCs in the group regardless of whether the PLC is the active server. It is recommended that the PLC be setup so that it will only attempt to send unsolicited data when it is the active server. To the user, the physical devices in the same group are viewed as a single logical device. Points are configured for the logical device thus eliminating the need for duplicate configuration.

Logical devices are configured by entering multiple TCP/IP addresses, either one or two for each PLC in the group. When the enabler starts, the primary PLC is the active and the other PLCs are in Hot-Standby mode. Under normal operating conditions, the enabler reads data from and writes data to the active PLC, while checking the "health" of the standby PLC(s).

The enabler requires the redundant PLC group to communicate with each other and to indicate (via a mode flag) which programmable controller is the active or is capable of operating as active PLC. The enabler is responsible for selecting the highest-ranking PLC as the active server.

⚠ **Important:** Since the Mode Address is specified at the port level, all redundant PLCs associated with a port must use the same Mode Address. This means that all PLCs on the port must have the same model type (that is all Series 90-70s or all Series 90-30s or all VersaMax).

PLC Redundancy

The Series 90 TCP/IP Triplex Communications enabler can be configured for PLC groups of up to 3 PLCs. The PLCs are ranked in order of importance Primary (PLCA)ÞSecondary (PLCB)ÞThird (PLCC).

The enabler reads and writes data to the highest-ranking PLC. Since the CIMPLICITY data points for this PLC group are only defined once, it is a requirement that the PLCs use the same memory references.

To determine the status of each of the PLCs in the PLC group, the enabler uses:

- The network connection status (up/down),
- The PLC status (running/stopped) and
- Two mode bits per PLC.

The two mode bits designated **Health** and **Master** are controlled by the PLCs and have the following meanings:

- Health, the PLC is healthy - passes internal checks and is operating correctly.
- Master, the PLC has performed all initialization actions and data can be read from this device.

The enabler uses the highest ranked running PLC with a network path that has both the Health and Master bits set.

The write and read behavior can be modified by configuration options as follows:

- PLC writes can be modified by a configuration radio button to cause writes to be performed to all PLCs
- PLC reads can be modified by a configuration check box to ignore the Master bit in which case data will be read from the highest ranking running healthy PLC with a network connection.

Cabling Redundancy

The Series 90 TCP/IP Triplex Communications enabler can be configured for one or two network communications paths to each device in a PLC group.

CIMPLICITY normally uses the primary cable address and monitors the secondary. If the primary cable address fails, the enabler automatically switches to the secondary. The enabler continues to monitor the primary and will automatically switch back should it recover.

Cabling redundancy can be implemented with:

- One Ethernet LAN with two connections from each PLC to the Ethernet LAN.
- A redundant Ethernet LAN and one connection from each PLC to both of the Ethernet LAN's.

When multiple Ethernet cards are used in the host computer, the Host IP Address and PLC IP Address in combination with the Subnet Mask of the Ethernet card determines which card/cable combination will be used.

Series 90 TCP/IP Triplex Redundancy Failure Conditions and Action

Series 90 TCP/IP Triplex Redundancy Failure Conditions and Actions

The enabler responds to fault condition as follows:

Series 90 TCP/IP Failure conditions and actions include:

- Communication/PLC failure.
- Communication/PLC recovery.
- Mode bits: no current active
- Mode bits: multiple actives.

Communication/PLC Failure

A communication/PLC failure occurs where the enabler can no longer communicate over any network path to a running healthy PLC; for example network down, PLC powered down, PLC stopped or Health bit clear.

- Active PLC - the enabler updates the diagnostic points then performs a failover to the next highest ranking running Healthy PLC with the Health and active mode bits set.
- Standby PLC - the enabler updates the diagnostic points for the PLC.

The enabler then retries communications to the failed PLC at the configured heartbeat rate.

Communication/PLC Recovery

A communication/PLC recovery occurs where the enabler re-establishes communication over a network path to a running healthy PLC; for example network recovered, PLC powered up or Health bit becomes set.

The action taken by the enabler depends on the ranking of the PLC.

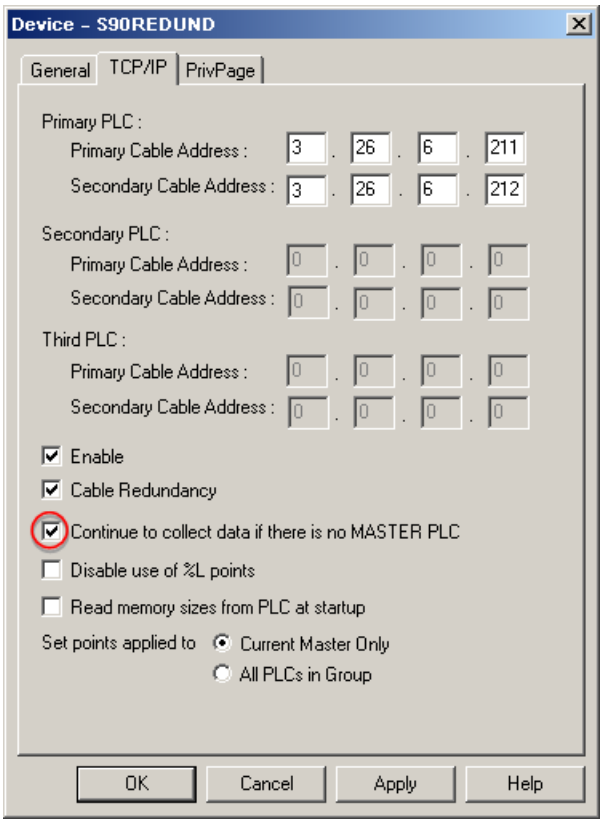
- Higher Ranking PLC - the enabler checks the Health and Master bits. It updates the diagnostic points for the PLC. If both mode bits are set, the enabler selects this PLC as the current active.
- Lower Ranking PLC - the enabler checks the Health and Master bits and, updates the diagnostic points for the PLC.

Mode bits: No Current Master

The enabler updates the diagnostic points and no further communication takes place to any PLC in the group.

The enabler continues to monitor the mode bits at the configured heartbeat timer interval.

This feature can be modified by checking **Continue to collect data if there is no MASTERPLC** on the TCP/IP tab in the Device Properties dialog box.



Mode bits: Multiple Masters

The enabler updates the diagnostic points and communicates with the highest-ranking PLC.

The enabler continues to monitor the mode bits at the configured heartbeat timer interval.

- Review a list of Series 90 TCP/IP Redundancy failure conditions and actions.

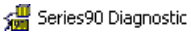
The Series 90 Diagnostic Program is a program provided with the Series 90 Triplex Communications enabler that you can use to check the basic operation and configuration of your network without starting CIMPLICITY software.

Note: It does not validate redundant PLC behavior, only basic connectivity and read & write capabilities.

You can perform the following functions:

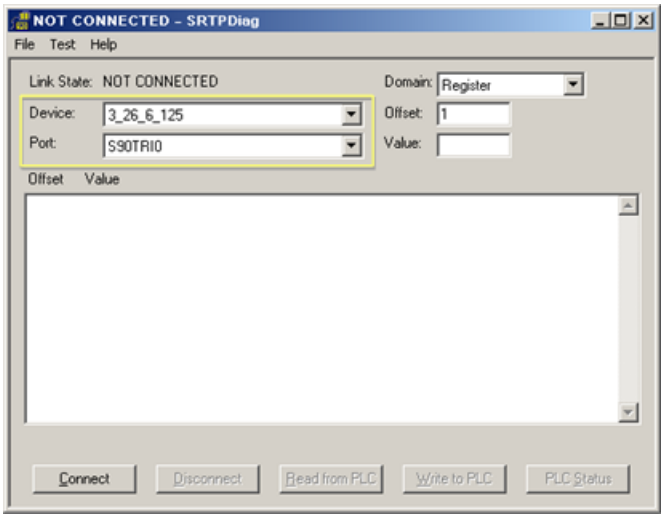
- Live update of the currently displayed domain
- Read load test
- Write load test

For this program to function, CIMPLICITY software's Series 90 Triplex Communications enabler must be successfully installed, and you must have configured the Triplex port and Series 90 devices using CIMPLICITY application configuration functions.

To start the diagnostic program, click the Series 90 Diagnostic 

icon in the CIMPLICITY Workbench.

The SRTPDiag window opens.



To connect to a Series 90 programmable controller in the TCP/IP network:

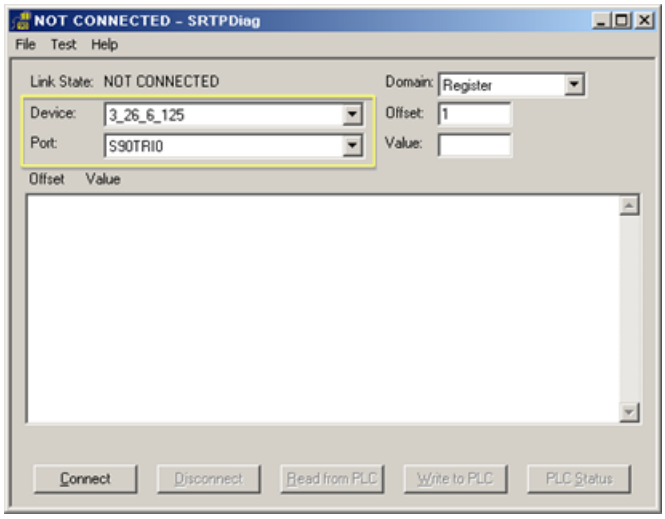
1. Click on the drop-down list button to the right of the **Device** field to display the list of currently configured devices, and select a Series 90 device from the list.

2. Click on the drop-down list button to the right of the **Port** field to display the list of currently configured ports, and select the TCP/IP port to use.

3. Click on the drop-down list button to the right of the **Domain** field to display the list of currently supported domains (memory types), and select a domain to display.

4. Click **Connect**.

   The program makes the connection with the programmable controller, and displays the values contained in the requested domain.



   To display the data for another domain, just select the domain from the drop-down list in the **Domain** field. The data for the new domain is automatically read and displayed in the data box.

   You can use the menus and buttons to do the following:

   - Toggle between manual and automatic data update modes.
   - Request manual reads of domain data
   - Request writes to the programmable controller
   - Display programmable controller status
   - Perform a read load test
   - Perform a write load test

Series 90 PLC Fault Table

Series 90 PLC Fault Table

In CIMPLICITY projects with the Triplex protocol enabled, you can obtain PLC fault information from the Triplex Fault Table. You can invoke the fault table from the Workbench by double clicking on its icon, in either the left or right pane.

todo: To find the Series 90Fault Table in the Workbench:

1. Click the Equipment folder in the left pane of the Workbench.
2. Click the Diagnostics folder.

Toggle Between Data Update Modes

Normally, domain data is only read when the domain display is initiated or when you click the **Read from PLC** or **Write to PLC** buttons. You can have data updated automatically by doing the following:

1. Click on the File menu.
2. Select Live Update from the submenu.

   When Live Update is enabled, the domain values are read and updated automatically approximately every three (3) seconds.

   todo: To return to manual mode:

3. Click the File menu. Note that Live Update has a check mark next to it.
4. Select Live Update again to turn off the Live Update mode.

Request Manual Reads of Domain Data

If you are not in Live Update mode, you can manually request reads of the domain by selecting **Read from PLC**.

Write Test Data to the Programmable

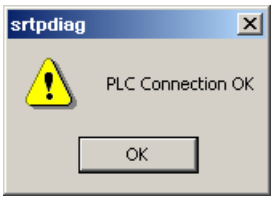
1. Select the domain to write to. Note that if you select a domain that is read-only, **Write to PLC** is disabled (grayed out).
2. Enter the memory location offset that you want to write to in the **Offset** field.
3. Enter the value you want to write in the field.

4. Click **Write to PLC**.

   The value is written to the offset location, and the data display box updates to reflect your write request.

Check Series 90 Status

Select PLC Status to display the current status of the programmable controller. An SRTP Diagnostic dialog box opens.



Click **OK** to close the dialog box.

Perform a Read Load Test

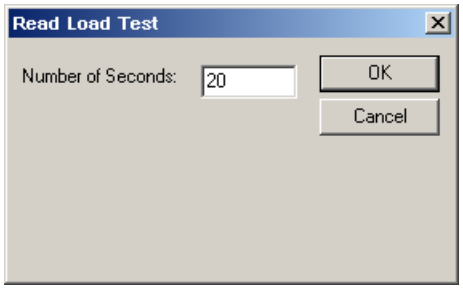
You can use **srtp diag.exe** to perform read load tests on your network. The read load test is performed in two parts:

- The first part tests the number of words per second that can be read over the network.
- The second part tests the number of read requests per second that can be made over the network.

The test reads data from all domains on the currently selected Series 90 device.

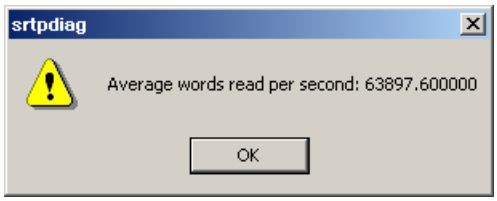
todo: To perform a read load test:

1. Click on the Test menu.
2. Select Read Load Test. from the submenu. The Read Load Test dialog box opens.



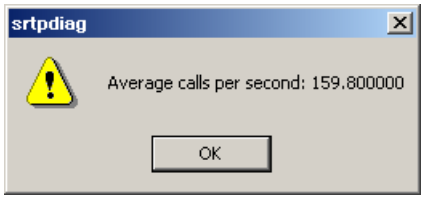
3. Enter the number of seconds you want to run the test in the **Number of Seconds** field, and click **OK** to start the test.

   The test cycles through reads of all memory types on the current device for the number of seconds you specify. At the end of the time period, an srtpdiag message box opens, telling you the average number of words read per second over the test period.



4. Click OK to proceed to the second part of the read test.

   The test again cycles through reads of all memory types on the current device for the number of seconds you specified on the Read Load Test dialog box. At the end of the time period, an srtpdiag message box opens, telling you the average number of read requests made per second over the test period.



5. Click **OK** to terminate the test.

Perform a Write Load Test

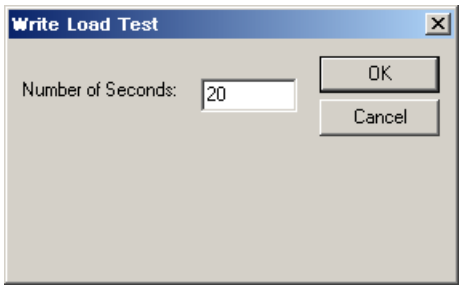
Warning: This test will overwrite the contents of the register domain (%R). Do not run this test if the register domain of the programmable controller contains important information.

You can use **srtp diag.exe** to perform write load tests on your network. The write load test tests the number of words per second that can be written over the network.

The test writes data to the register domain (%R) on the currently selected programmable controller.

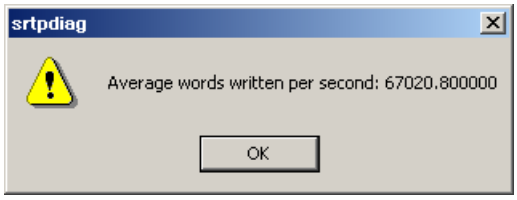
todo: To perform a write load test:

1. Click on the Test menu.
2. Select Write Load Test**.** from the submenu. The Write Load Test dialog box opens.



3. Enter the number of seconds you want to run the test in the **Number of Seconds** field, and click **OK** to start the test.

   The test cycles through writes of register memory on the current device for the number of seconds you specify. At the end of the time period, an HCT Diagnostic dialog box opens, telling you the average number of words written per second over the test period.



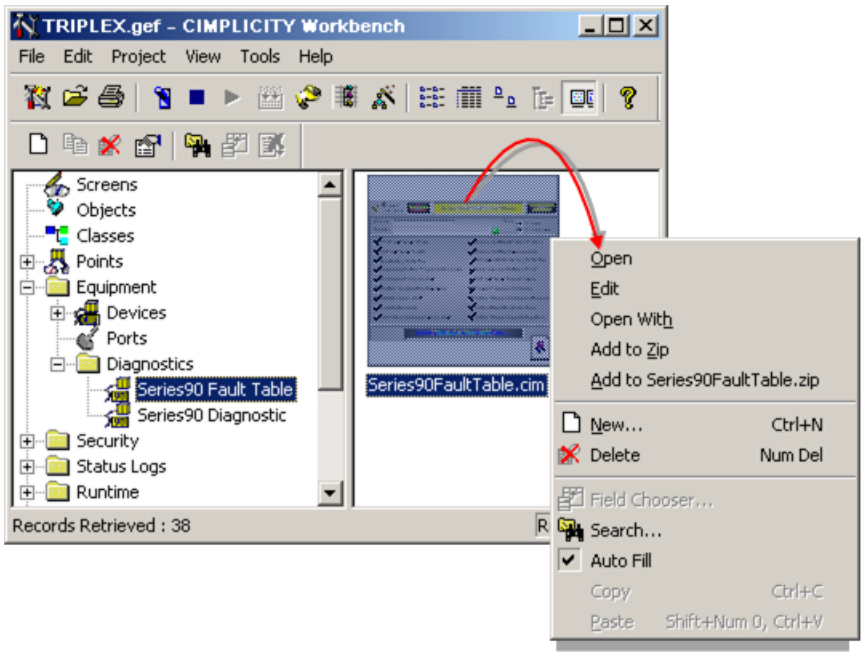
4. Click **OK** to terminate the test.

Open the Series 90 Fault Table

1. Do one of the following:
   a. Double click the **Series 90 Fault Table** icon in the left pane or
   b. Select the Series 90 Fault Table icon in the left pane and double click **Series 90FaultTable.cim** in the right pane.

2. Click the right mouse button over Series 90FaultTable.cim in the Workbench right pane.

3. Select Open from the popup menu.



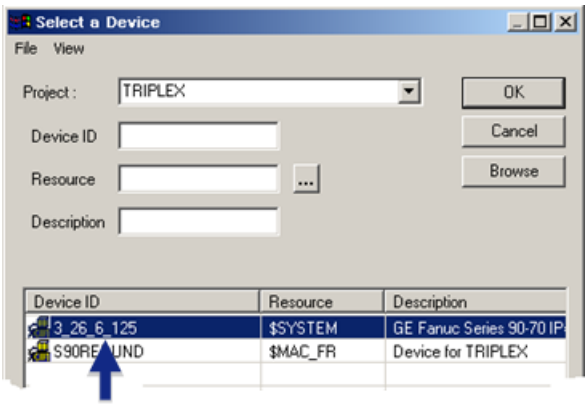
The Series 90FaultTable is empty when it opens.

When the Fault Table is opened the user can select a configured Series 90 device by using the device browser.

Obtain Series 90 PLC Fault Information

1. Click the Device browser button.

2. Select a device from the Select a Device browser.



If the device is valid (the device is up and running) and the connection is made successfully the default view will appear for that device.

Series 90 Fault Table Default View

The default view is the PLC Status, which displays the PLC System Status References (%S, %SA, and %SB registers). The System Status References used and their definitions are as follows.

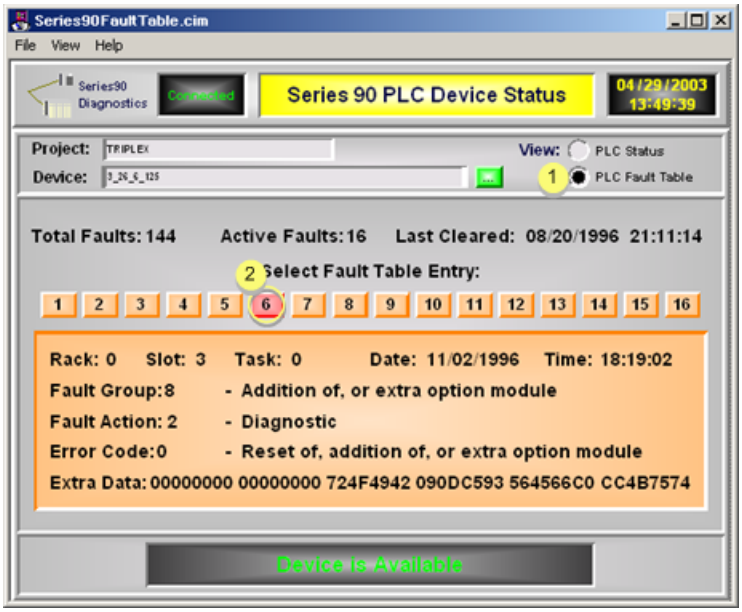
System Status Reference	Address
PLC Fault Table is Full	**%S0009**
I/O Fault Table is Full	**%S0010**
Bad Battery Detected	**%S0014**
Invalid Application Program Checksum	**%SA0001**
Application Program Fault	**%SA0003**
System Configuration Mismatch	**%SA0009**
CPU Hardware Fault	**%SA0010**

Low Battery Fault	**%SA0011**
I/O Module Communication Fault	**%SA0014**
Option Module Hardware Failure	**%SA0027**
Option Module Software Fault	**%SA0031**
Corrupted RAM Memory Detected	**%SB0010**
Unrecoverable Software Error	**%SB0013**
I/O Fault Entry Present	**%SC0013**
Password Access Violation	**%SB0011**
Option Module Communications Fault	**%SA0015**
I/O Module Added to Rack	**%SA0019**
Option Module Added to Rack	**%SA0020**

The user can select the PLC Fault Table view to display the actual fault information. The number of faults can vary from zero to sixteen. For each fault in the PLC fault table a button will appear on the screen.

Series 90 Fault Table Detailed View

Click the button that corresponds to the fault to view detailed information about the fault. The Series 90 - 70 and 90 – 30/20 Reference Manuals can be used to interpret the extra data.



1	Select PLC Fault Table.
2	Click to display detailed information about the corresponding fault.

CIMPLICITY Configuration for Series 90 TCP/IP Triplex

CIMPLICITY Configuration for Series 90 TCP/IP Triplex

When you are configuring ports, devices, and device points, most of the information required is identical whether redundant or non-redundant device communications is configured.

The difference is you must configure a **Mode Address** field and this address must be properly configured in the PLC.

Series 90 TCP/IP Triplex Port Configuration

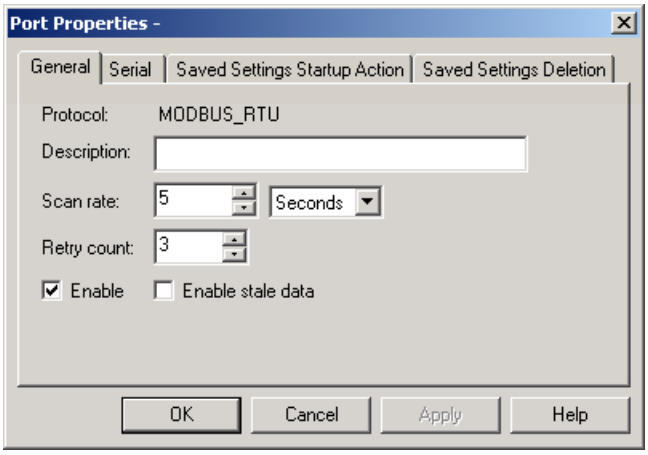
Series 90 TCP/IP Triplex Port Configuration

1. In the **Protocol** field, select S90_TRIPLEX from the list of available protocols.
2. In the **Port** field, select the communication port that will be used for Series 90 TCP/IP Triplex communications.

   When you click **OK** to create the port, the Port Properties dialog box for the protocol opens.

Port General Properties

Use the General tab in the Port Properties dialog box to enter general information for the port. You can define the following.

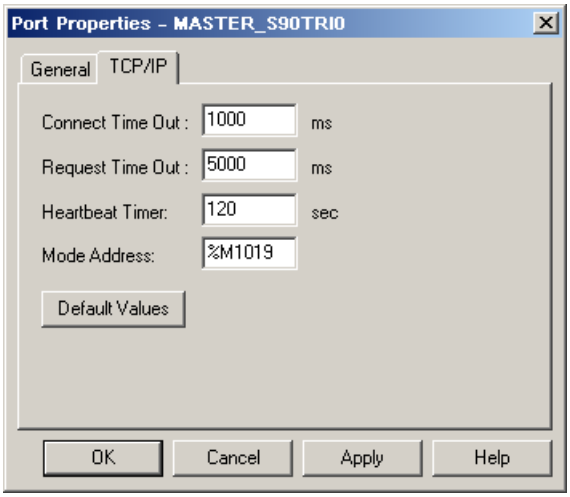


Description	Enter an optional description to help you identify the port.

Scan Rate	Enter the base scan rate for the port. Point scan rates will be multiples of the base rate. You can specify a scan rate in Ticks (100 ticks = 1second), Seconds, Minutes, or Hours.
Retry Count	If the enabler cannot complete a setpoint (write/download) to a device due to any reason, retries are performed. Enter the number of retries to perform before reporting the action as "failed".
Enable	Set this check box if you want the port to be enabled when the project starts. If you clear this check box, the port will not be enabled when the project starts, and points will not be available for devices on the port.
Enable stale data	Check to keep the point available in most circumstances that would have made it unavailable. However the point value will be stale, meaning it is the last known value and may or may not have changed in the device.

Port TCP/IP Properties

Use the TCP/IP tab in the Port Properties dialog box to enter information about TCP/IP communications for the port. You can define the following.



Connect Time Out	Enter the number of milliseconds to wait when making a connect request to a device. The maximum value is 64,000 ms. For VersaMax SE Interfaces, a value of 55,000 ms is recommended.
Request Time Out	Enter the number of milliseconds to wait when waiting for a request to complete. The maximum value is 64,000 ms. For VersaMax SE Intefaces, a value of 55,000 ms is recommended.
Heartbeat Timer	Each Series 90 or VersaMax controller in a redundant group will be probed at the `Heartbeat` interval to determine the state of the device. For standard (non-redundant) PLC configurations, this parameter controls the frequency with which the communication enabler attempts to re-establish a lost connection. Enter a value in seconds that will be the "Heartbeat" interval for probing the Series 90 controllers. If the value entered is lower than 45 seconds (Windows NT default connection retry interval) ghost TCP/IP connections could be established to a PLC when a PLC is brought back on-line. For VersaMax SE Interfaces, the following configuration suggestions are made: The Default Idle Time in the PLC's CPU configuration should be the same as the VersaMax SE's T3, Link Idle Timeout parameter Under the CH1 Serial & Protocol Settings. The Heartbeat Timer value configured should not exceed this configured value. The default value is 120,000 msec (120 seconds). Failure to properly set this parameter can result in intermittent communication between the communication enabler and the PLC and/or a failure to recover communication after a connection has been lost. Please refer to your PLC programmer, PLC CPU and VersaMax SE documentation for instructions on modifying the PLC CPU or VersaMax SE settings.
	Recommended configuration for VERSAMAX SE devices.

		Advanced configuration for ghost connections.	
Mode Address	Enter the starting address of the location of two consecutive status bits in digital memory that indicate the current state of the programmable controller. Bits indicate the following.		
	Bit	Indicates	
	First bit	The overall health of the programmable controller.	
		1=	Healthy
		The Series 90 nickname for this bit is **LOC_READY**	
	Second bit	Whether the programmable controller is the current active in the redundant group	
		1=	Active
		The Series 90 nickname for this bit is LOC_ACT.	
	Both status bits should be set by logic in the Series 90 controller, and will be read at the `Heartbeat` interval. This address must be a legal address in the Series 90 controller or the Heartbeat will fail.		
	Recommended for Series 90-70.	`%S35`	
	Recommended for Series 90-30.	`%M1019`	
	Note:		
	Redundant Systems	When configuring a Mode Address for RX7i, RX3i, Series 90-70, Series90-30, or VersaMax Simplex devices, make sure that the status bits in the PLC are set to 1.	
	Non-redundant Systems	RX7i, RX3i, Series 90-70, Series 90-30, VersaMax are the selection choices for non-redundant PLC communication. They do not use the value of the data at the locations designated for the mode bits for health or run bits for determining device or communication status. However, the configured location must be a valid address in the PLC.	

Click **Default Values** if you want to use the factory default values. The default values are:

Connect Timeout	1000 ms (1 second)
Request Timeout	5000 ms (5 seconds)
Heartbeat Timer	120 seconds (2 minutes)
Mode Address	%M1019

Series 90 TCP/IP Triplex Device Configuration

Series 90 TCP/IP Triplex Device Configuration

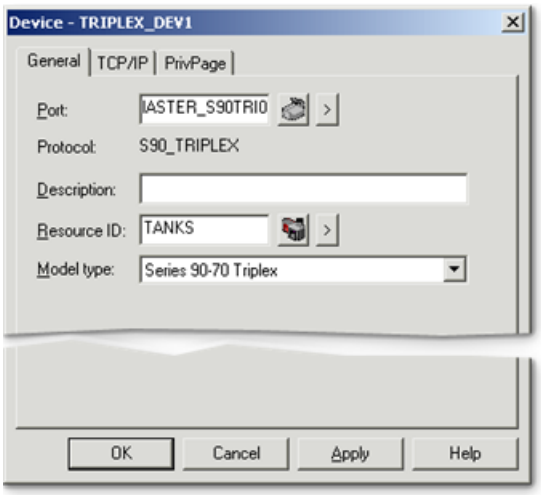
When you create a new device for TCP/IP Triplex communications, enter the following information in the New Device dialog box:

1. In the **Device** field, enter the name of the logical device you are configuring.
2. In the **Port** field, select the Series 90 TCP/IP Triplex port to be used by this device.

   When you click **OK** to create the device, the Device Properties dialog box opens.

General Device Properties

Use the General tab in the Device Properties dialog box to enter general information for the device. You can define the following.



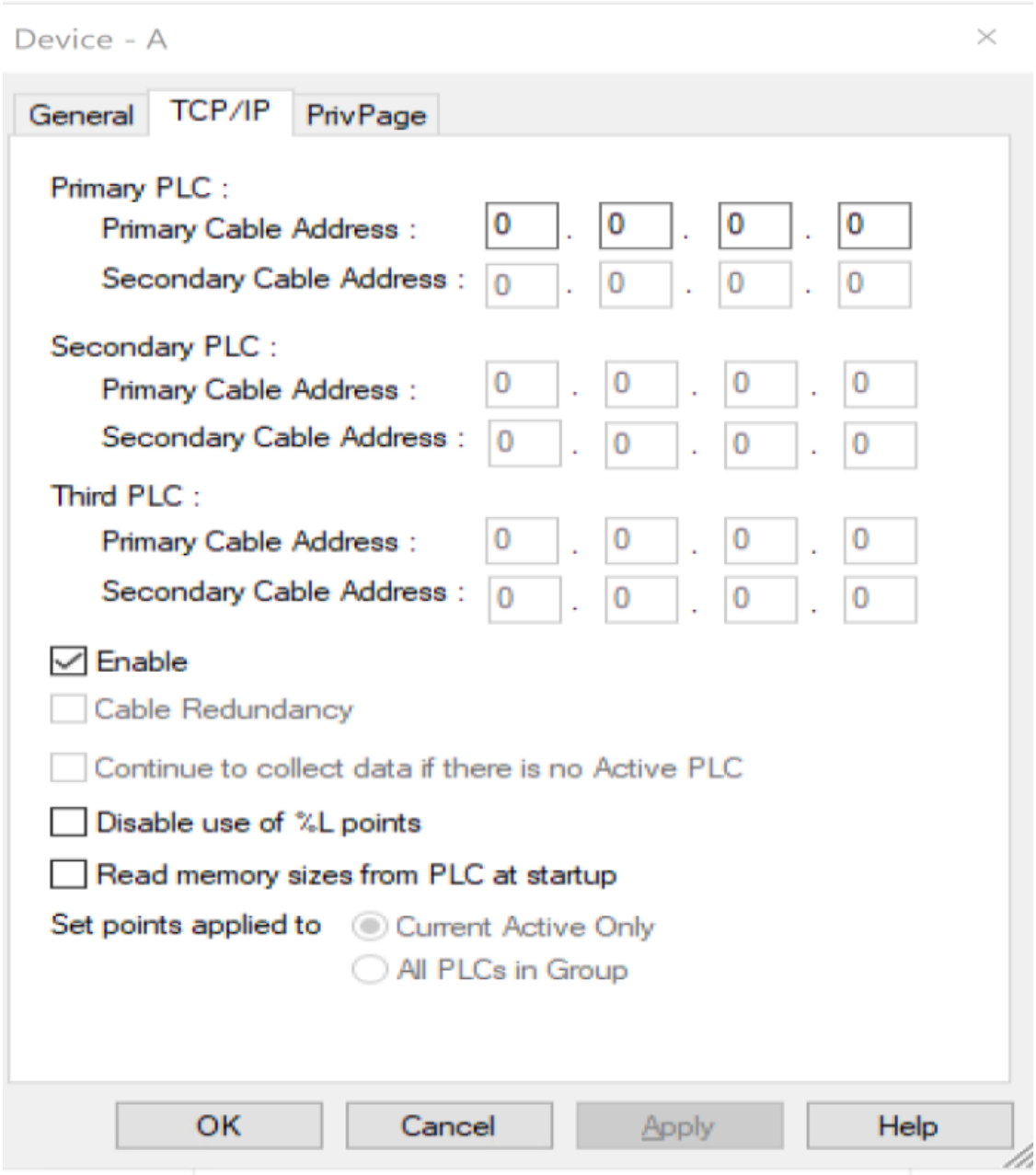
Port	Select the port for this device. Click buttons to the Port field to select the port as follows.	
		Display the list of ports and select one.
		Create a new port, edit the current port, or select a port from the list of ports.
Description	Enter an optional description to help you identify the device.	
Resource		
		Display the list of resources and select one.
		Create a new resource, edit the current resource, or select a resource from the list of resources.

Model Type	Select the type of device. Click the drop-down button to the right of the input field to display your choices, then make a selection. For this protocol, the choices are: • PACSystems RX3i • PACSystems RX7i • Series 90-70 • 90-30 • VersaMax • PACSystems RX3i Simplex • PACSystems RX3i Duplex • PACSystems RX3i Triplex • PACSystems RX7i Simplex • PACSystems RX7i Duplex • PACSystems RX7i Triplex • Series 90-70 Simplex • Series 90-70 Duplex • Series 90-70 Triplex • Series 90-30 Simplex • Series 90-30 Duplex • Series 90-30 Triplex • Series VersaMax Simplex • Series VersaMax Duplex • Series VersaMax Triplex Note that Series 90-70, Series 90-30, VersaMax are the selection choices for non-redundant PLC communication and do **not** use the mode, health or run bits for determining device or communication status. Note that a project that is currently using the Triplex driver to communicate with a Series 90 can be converted to communicate with a PACSystems device by changing the model type from the Series 90 model to the appropriate PACSystems model.

TCP/IP Device Properties

Use the TCP/IP tab in the Device Properties dialog box to enter information about TCP/IP communications for the device.

You can define the following.



Primary PLC of the redundant group:	
Primary Cable Address	Enter the IP address of the primary Ethernet card in the Primary PLC **Primary Cable Address** field.
Secondary Cable Address	This field is available only if you select the Cable Redundancy option and a redundant PLC model. Enter the IP address of the secondary Ethernet card in the Primary PLC **Secondary Cable Address** field.
Secondary PLC: Duplex or Triplex selected for the device model	

Primary Cable Address	Enter the IP address of the primary Ethernet card in the secondary PLC **Primary Cable Address** field.
Secondary Cable Address	This field is available only if you select the Cable Redundancy option. Enter the IP address of the secondary Ethernet card in the secondary PLC **Secondary Cable Address** field.
Third PLC: Triplex selected for the device model	
Primary Cable Address	Enter the IP address of the primary Ethernet card in the Third PLC **Primary Cable Address** field.
Secondary Cable Address	This field is available only if you select the Cable Redundancy option. Enter the IP address of the secondary Ethernet card in the Third PLC **Secondary Cable Address** field.
Device options include	
Enable	Select this check box to enable the device when the project starts. If you clear this check box, the device will not be enabled and points associated with the device will be unavailable.
Cable Redundancy	Select this check box if redundant cabling is being used for this device. If you clear this check box, the **Secondary Cable Address** fields in this dialog box are disabled. This option is available only for redundant PLC model types.
Continue to Collect Data if there is No Active PLC	By default, when the Series 90 TCP/IP Redundant Communications enabler loses its connection to the active PLC of a redundant group and the server PLC(s) fail to set its Master bit, the enabler marks the device down. All points go (and remain) "Unavailable" until one of the PLCs declares itself the active. You can select this check box to change this behavior so that when the active PLC of a redundant group fails, and no other PLC declares itself the active, the enabler automatically starts collecting data from the highest-ranking server PLC. This option is available only for redundant PLC model types.
Disable use of %L points	By default, the Series 90 TCP/IP Redundant Communications enabler validates **%L** device points by reading Program Blocks from the PLC. This may increase the length of time it takes to start the enabler. If you do not use **%L** points in your project, you can change the enabler, via selecting the corresponding button, to skip this validation. This option is available only for Series 90-70 PLC models (standard or redundant).
Read Memory Sizes from PLC at Startup	On start up, the Series 90 TCP/IP Redundant Communications enabler must obtain the size of each Memory type in the PLC. The enabler saves this configuration for faster initialization on subsequent starts. By default, the enabler will use this saved configuration instead of reading the configuration from the PLC. You can change the enabler, via selecting the corresponding button, to obtain the sizes from the attached PLC.
Set Points Applied to	By default, the Series 90 TCP/IP Redundant Communications enabler only reads data from and writes data to the active server. You can change the enabler, via selecting the corresponding button, to write data to all the programmable controllers in the redundant group. This option is only available for redundant PLC models.

Series 90 TCP/IP Triplex Point Configuration

Series 90 TCP/IP Triplex Point Configuration

When you define a point, the following fields have values that are unique to or have special meanings for Series 90 Triplex communications.

General Point Properties

On the General tab in the Point Properties dialog box you may configure points for **Read** or **Read/ Write** access.

Device Point Properties

On the Device tab in the Point Properties dialog box, configure the:

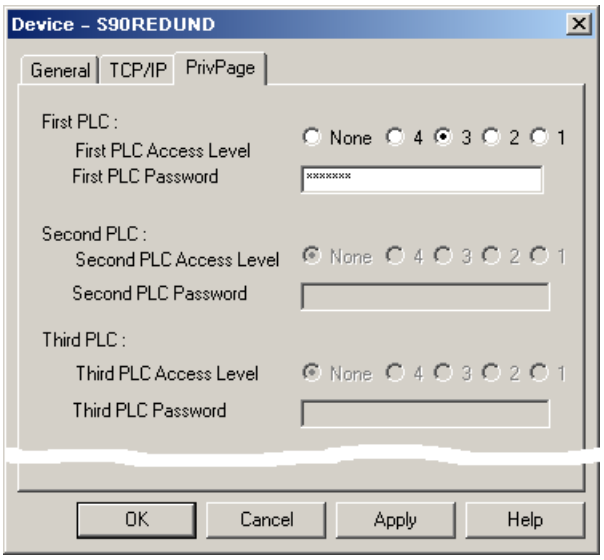
Address	For all memory types except **%L**, enter the memory type and offset in the Address field. The address format is: **<**memory type**><**address**>** For example, to specify Register 1, the address is %R1. All offsets have a minimum value of 1.
	The address of **%L** memory types requires the block name. Block names must be unique over all programs in the programmable controller. Block names cannot begin with a numeric character. The address format is: **%L<**address**><**block name**>** For example, to specify the fifth word in the %L domain in the program block ABC, the address **is %L5ABC**. **Note:** The block name must be capitalized. For example, %L5ABC is correct, but %LAbc is not correct.
	In addition to the standard diagnostic points, you can configure a set of Series 90 TCP/IP Triplex diagnostic points *(page 823)*. The data structure type of all diagnostic points is analog **UINT**.
Update Criteria	The update criteria determine how the data will be requested. Select **On Change** or **On Scan** for points whose values should be polled by Series 90 TCP/IP communications at regular intervals. Select **On Demand On Change** or **On Demand On Scan** for points whose values should be polled by Series 90 TCP/IP communications at regular intervals when users are viewing them. Select **Unsolicited** for points that are updated by devices. **Note:** By default unsolicited data is disabled; it needs to be enabled *(page 826)* for unsolicited data to be processed by this device communication interface.
Point Type	Select one of the following: • For device points, select **Normal**. • For diagnostic points, select **Diagnostic**. The Redundant TCP/IP Communications enabler uses the standard diagnostic points as well as the diagnostic points described below.

When you are configuring Boolean points in memory types that are not bit addressable, you must also enter data in the following field:

Address Offset	Enter the bit offset that corresponds to the bit position within the word.

Series 90 TCP/IP Triplex Privpage

By default, the Triplex device communication enabler will connect to a PLC at the default privilege level configured for the PLC. Use the Priv Page tab in the Device Properties to change the default access.



rect 16, 58, 175, 272 *(page 822)*

Access options for each PLC are	
None	Default level in the PLC
Level 1	Typically read-only access
Level 2	Read and Write data, Clear Fault Tables
Level 3	Read and Write Data, Clear Fault Tables, Write Logic and Configuration with the CPU stopped
Level 4	Change Passwords, Rad and Write Data, Clear Fault Tables, Write Logic and Configuration

Passwords		
May only be specified for levels 2, 3 and 4. Must be entered exactly as on the PLC. Access is controlled on a per project basis.		
PLC selections are		
Primary PLC of the redundant group or the non-redundant PLC being configured		
	First PLC Access Level	Level of access to be requested by the communication interface when establishing a connection for the first device
	First Password	Password to be used in requesting the level of access for the first device
Second PLC of the redundant group		
	Second PLC Access Level	Level of access to be requested by the communication interface when establishing a connection for the second device
	Second Password	Password to be used in requesting the level of access for the second device
Third PLC of the redundant group		
	Third PLC Access Level	Level of access to be requested by the communication interface when establishing a connection for the third device

Third PLC Password	Password to be used in requesting the level of access for the third device

Important: In dynamic mode, changes are effective the next time that the device communication interface needs to connect to the PLC.

Series 90 TCP/IP Triplex Diagnostic Points

Series 90 TCP/IP Triplex Diagnostic Points

You can configure a set of diagnostic points for each CIMPLICITY device. Each diagnostic point is automatically updated by the enabler at runtime to indicate the status of the controllers. These diagnostic points should be analog UINT device points. There are three diagnostic points that are unique for the TCP/IP Redundancy device communication enabler. They are:

- The mode point provides an overall indication of the state of the device itself. It is only available for redundant model types.
- The Status point provides an overall indication of the state of the device/cable.
- The Data Source point indicates which device/cable is supplying the current data.

There are two diagnostic points that are used to directly access the PLC and IO Fault Tables. They are:

- The %PLC_FAULT point directly accesses the PLC Fault Table.
- The %IO_FAULT point directly accesses the IO Fault Table.

Mode Diagnostic Points

You can create a mode point for each physical redundant PLC. The point contains the following information:

Description	Bit	Value
Device health	**1**	**1** = device OK
Current active CPU Status	**2 3**	**1** = current active **1** = CPU is running

The first and second bits contain the mode information from the programmable controller. The mode address is defined in the **Mode Address** field on the TCP/IP tab in the Port Properties dialog box when you create the port.

The third bit indicates the status of the CPU. The CPU is considered to be not in run mode if its state is any of the following:

- **STOP_IO_DISABLED**
- **CPU_STOP_FAULTED**
- **CPU_HALTED**
- **CPU_SUSPENDED**
- **STOP_IO_ENABLED**

The addresses for the Mode diagnostic points are:

$MODE_P1 Mode Point	Primary PLC
$MODE_P2 Mode Point	Secondary PLC
$MODE_P3 Mode Point	Third PLC

Status Diagnostic Point

You can create a Status point for each cable connection for a configured redundant PLC. If configured, the point will be set to one of the following values:

Value	Description
0	OK–Indicates the enabler can connect to the PLC over this cable and the PLC has the expected state.
1	NOT OK–Indicates there is a problem with either the cable connection or the state of the PLC.
2	This cable is not configured.

The addresses for the Status diagnostic points are:

$CONN_STATUS_P1C1 Status Point–Primary PLC over Primary Cable

$CONN_STATUS_P1C2 Status Point–Primary PLC over Secondary Cable

$CONN_STATUS_P2C1 Status Point–Secondary PLC over Primary Cable

$CONN_STATUS_P2C2 Status Point–Secondary PLC over Secondary Cable

$CONN_STATUS_P3C1 Status Point–Third PLC over Primary Cable

$CONN_STATUS_P3C2 Status Point–Third PLC over Secondary Cable

Data Source Diagnostic Point

You can create a Data Source point for each logical CIMPLICITY device in a redundant PLC configuration. If configured, the point will be set to one of the following values:

Value	Description
0	No communication on any cable
1	Current data is collected from the Primary PLC over the Primary Cable
2	Current data is collected from the Primary PLC over the Secondary cable.
3	Current data is collected from the Secondary PLC over the Primary Cable.
4	Current data is collected from the Secondary PLC over the Secondary Cable.
5	Current data is collected from the Third PLC over the Primary Cable.
6	Current data is collected from the Third PLC over the Secondary Cable.

The addresses for the Status diagnostic point is:

$DATA_SOURCE	Data Source Point

%PLC_FAULT Diagnositc Points

A %PLC_FAULT diagnostic point directly accesses the PLC Fault Table. The PLC Fault Table can contain up to 16 faults and each fault can contain up to 42 bytes of information.

The following formats can be used when accessing the PLC Fault Table:

%PLC_FAULT	Delivers a start address starting at the beginning of the fault table, including the header information, which can be up to 12 bytes of information.
%PLC_FAULT.<Fault Number>	Delivers a start address at the beginning of a specific fault (i.e. no header), where <Fault Number> is a number between 1 and 16.

%IO_FAULT Diagnostic Points

A %IO_FAULT diagnostic point directly accesses the IO Fault Table. The IO Fault Table can contain up to 32 faults and each fault can contain up to 42 bytes of information.

The following formats can be used when accessing the IO Fault Table:

%IO_FAULT	Delivers a start address starting at the beginning of the fault table, including the header information, which can be up to 12 bytes of information.
%IO_FAULT.<Fault Number>	Delivers a start address at the beginning of a specific fault (i.e. no header), where <Fault Number> is a number between 1 and 32.

Series 90 TCP/IP Triplex Unsolicited Data Support

Series 90 TCP/IP Triplex Unsolicited Data Support

The Series 90 Triplex Communications enabler supports the receipt of unsolicited data from Series 90-70 and Series 90-30 PLCs. The maximum message size is 2048 bytes.

All unsolicited points for the Series 90 Triplex Communications enabler need to be configured on the first TCP/IP port (TRIO). If multiple projects using S90 Triplex are configured, unsolicited data should be enabled for one port on one project only. (more simply - Unsolicited data should only be enabled on one active project prr computer.) Unsolicited points on other ports will not be received.

A CIMPLICITY device point can be updated by unsolicited data messages if it meets the following criteria:

- The point length in bytes is less than or equal to the data length that is transmitted in the message.
- The Series 90 TCP/IP address of the device configured for the point matches that of the sending device.
- The configured point address matches the Memory Type and Starting Address in the unsolicited data message.

If the data length of the message exceeds the point length, the point is not updated and a message is logged to the Status Log.

Enabling Unsolicited Data

By default, unsolicited data collection is disabled.

To enable unsolicited data collection, configure the project level global `S90TCP_ALLOW_UNSO` with a value of Y.

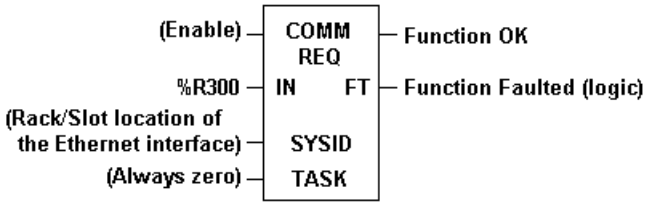
Unsolicited Messages

For detailed information about the Unsolicited Message format, see the TCP/IP Ethernet Communications for the Series 90-70 Programmable Controller (GFK-1004).

The following is an example of a COMREQ Function Block and a COMREQ Command Block that will:

- Establish a channel (channel 7 in this example) to a remote Host application server at IP address 192.168.0.4.
- Return the Communication Request Status (CRS) word to %R10.
- Send local PLC registers %R50-%R57 to the remote Host.
- Repeat the send ten (10) times once every 7 seconds with a time-out of 500 ms. for each read.

The COMREQ function block for this example is:



The COMREQ Command Block for this example is:

Address	DEC	Hex	Description
%R300	**0017**	**0011**	Length of Send Information Report Data Block
%R301	**0000**	**0000**	Always zero
%R302	**0008**	**0008**	Memory type or CRS word (%R)
%R303	**0009**	**0009**	CRS word address minus 1 (%R10)
%R304	**0000**	**0000**	Reserved
%R305	**0000**	**0000**	Reserved
%R306	**2010**	**07DA**	Send Information Report Channel Command number
%R307	**0007**	**0007**	Channel number (7)
%R308	**0010**	**000A**	Number of repetitions (10)
%R309	**0003**	**0003**	Time unit for send period (3 = seconds)
%R310	**0007**	**0007**	Minimum interval between host accesses (7 seconds)
%R311	**0050**	**0032**	Time-out on each individual host transfer response (500 ms)
%R312	**0008**	**0008**	Memory type from which to send data (%R)
%R313	**0050**	**0032**	Starting address from which to send data (%R50)
%R314	**0008**	**0008**	Number of memory units (8 registers)
%R315	**0000**	**0000**	Reserved
%R316	**0000**	**0000**	Reserved
%R317	**0001**	**0001**	Remote network address type (IP address)
%R318	**0004**	**0004**	Remote network address length in words (4)

%R319	**0192**	**00c0**	Remote Host - Register 1 of IP address (192)
%R320	**0168**	**00A8**	Remote Host - Register 2 of IP address (168)
%R321	**0000**	**0000**	Remote Host - Register 3 of IP address (0)
%R322	**0002**	**0002**	Remote Host - Register 4 of IP address (2)

Unsolicited Compound Messages with Timestamps

Unsolicited Compound Messages with Timestamps

The Series 90 TCP/IP Communications enabler supports a compound message format for unsolicited data that allows the PLC to send multiple registers in a single unsolicited message.

The enabler uses defined Transmission Areas to determine if a message is a simple or compound unsolicited message. You can configure up to eight of these areas by adding entries to the global parameters file that define the transmission area number (1-8) and starting address in the PLC.

To configure **%R500** as a transmission areas, add the project level global parameter:

UNSO_TX_AREA_1 with a value of R500

When using Compound Messages:

- You will need to define a point for each sub-packet with the correct domain, offset and length.

It is recommended that you not configure points that conflict with addresses in the unsolicited transmission areas.

Transmission Area Format in a PLC

The format for a compound message in a transmission area is:

Description	Size (in bytes)
Format	**2**
Reserved	**2**
Number of Sub-Packets	**2**
Sub-Packet 1 Domain	**2**
Sub-Packet 1 Domain Offset	**2**
Sub-Packet 1 Length	**2**
Sub-Packet 1 Timestamp	**8**

Description	Size (in bytes)
Sub-Packet 1 - first data word	
.	
.	
Sub-Packet 1 - last data word	
.	
.	
Sub-Packet n Domain	**2**
Sub-Packet n Domain Offset	**2**
Sub-Packet n Length	**2**
Sub-Packet n Timestamp	**8**
Sub-Packet n - first data word	
.	
.	
Sub-Packet n - last data word	

Where:

- **Format** must be 1.
- **Number of Sub-Packets** is the number of Simple Messages contained in the Compound Message.

Each sub-packet is essentially a Simple Message that has a common header and a variable length data area. The sub-packet header is as follows:

- **Sub-Packet Domain** - the PLC data type from which the data in the Sub-Packet originates. The contents of this field will be a number representing the data type as follows:

Value	PLC Data Type
0	Local Data Block (%L) 90/70 Only
8	Register table (%R)
10	Analog Input table (%AI)
12	Analog Output table (%AQ)
16	Discrete Input table (%I)
18	Discrete Output table (%Q)

Value	PLC Data Type
20	Discrete Temporary (%T)
22	Discrete Internal (%M)
56	Genius Seamless (%G)
24	Special Contacts A (%SA)
26	Special Contacts B (%SB)
28	Special Contacts C (%SC)
30	System Fault , (%S) read only

- **Sub-Packet Domain Offset** - the domain start address of the data in the Sub-Packet. This field is numeric and quoted in units of the domain data type starting at zero.
- **Sub-Packet Length** - the length of data in the Sub-Packet. This field is numeric and quoted in bytes.

If the Sub-Packet domain is **%L**, then the first 8 bytes of data contain the block name for the **%L** domain.

- **Timestamp** -the time relating to the data in the Sub-Packet, in CIMPLICITY COR_STAMP format *(page 831)* .

Sample Compound Message

A compound message block at %R512 to send the value 99 to %R600 with a timestamp *(page 831)* of March 4, 1995 05:06:07.08 would be:

Address	Value	Description
%R512	**1**	Format
%R513	**0**	Reserved
%R514	**1**	Number of subpackets
%R515	**8**	Subpacket 1 domain
%R516	**600**	Subpacket 1 domain offset
%R517	**2**	Subpacket 1 length in bytes
%R518	**27360**	Low order date timestamp
%R519	**304**	High order date timestamp
%R520	**14436**	Low order time timestamp
%R521	**77**	High order time timestamp
%R522	**99**	Data value

Timestamp Calculation

The definition of the COR_STAMP structure is::

```
typedef struct cor_time_stamp {
   COR_I4   yyyymmdd;
   COR_I4   hhmmsstt;
} COR_STAMP;
```

where **COR_I4** is a four-byte integer.

For example, for the date March 4, 1995 and time 05:06:07.08, the COR_STAMP structure looks like this:

```
19950304
05060708
```

To save the date or time timestamp in Registers, you will need to do the following:

- Divide the 8-byte date or time by 65536 and put the quotient in the high order timestamp register.
- Multiply the quotient by 65536, subtract the result from the original number and put the result in the low order timestamp register.

For example, to place the timestamp of March 4, 1995 05:06:07.08 in Register memory starting at %R518:

```
%R519 = 19950304 / 65536 = 304
%R518 = 19950304 - (304*65536) = 19950304 - 19922944 = 27360
%R521 = 05060708 / 65536 = 77
%R520 = 05060708 - (77*65536) = 05060708 - 05046272 = 14436
```

Series 90 TCP/IP Triplex Advanced Configuration Topics

Series 90 TCP/IP Triplex Advanced Configuration Topics

Advanced topics include:

- TcpMaxConnectRetransmission parameter.
- Recommended configuration for VERSAMAX SE devices.
- Recommended configuration for RX7i devices.

TcpMaxConnectRetransmissions

To avoid ghost connections (connections that are established at the operating system level, but not recognized by the enabler) when the heartbeat interval is configured lower than 45 seconds, follow this procedure to change the registry configuration for **TcpMaxConnectRetransmissions**.

Warning: It is possible to cause serious damage to your operating system by using RegEdit and RegEdt32. Be careful not to modify anything that is not listed in these instructions.

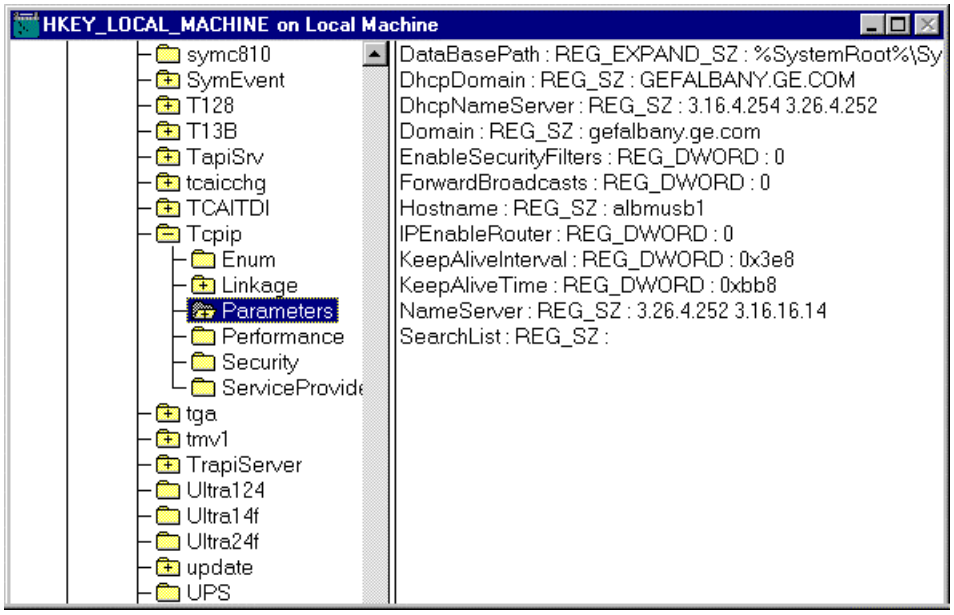
todo: To change the registry configuration for TcpMax Connect Retransmisisons:

1. Run **regedt32** from a command prompt.

   The Registry Editor window opens.

2. Follow the tree to:

   [HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters]

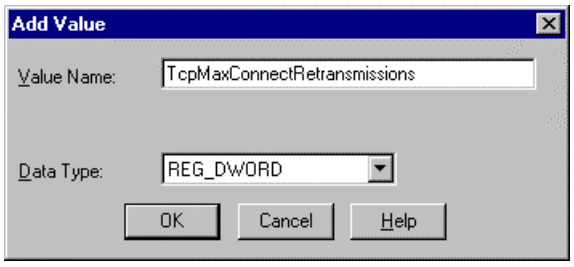


3. Click Edit on the Registry Editor menu bar.

4. Select Add Value from the menu.

5. Enter **TcpMaxConnectRetransmissions** in the **Value Name** field.

6. Select **REG_DWORD** in the **Data type** field.

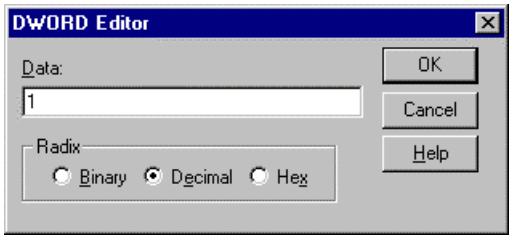


7. Click **OK**.

   The DWORD Editor opens.

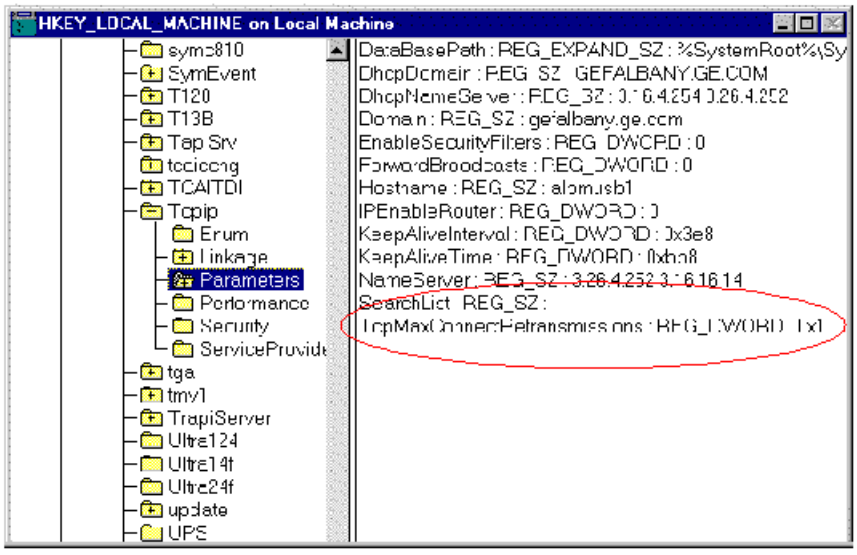
8. Enter the integer **1** in the **Data** field.

9. Check **Decimal**.



10. Click **OK**.

    The **TcpMaxConnectRetransmissions** parameter displays in the Registry Editor right pane.



11. Exit the Registry Editor and restart your computer.

Recommended Configuration for VERSAMAX SE Devices

1. Configure the timeout on the device's COM port and SE Interface to 60 seconds.
2. Add the global parameter S90TCP_<DEVICE_ID>_IS_SE with a value of "Y" for each SE device.
3. The connection and request time out values on the TCP/IP tab of the port configuration need to be set to a value that ensures consistency in establishing and maintaining a connection. A value of 55 seconds (55000 ms) is recommended as an initial starting point. An initial value of 5 to 10 seconds may be configured for the heartbeat timer to reduce the possibility of timeout due to inactivity. It is desirable that the request timeout plus the network latency be less than the time out value configured in the SE and PLC Port configurations.
4. Add the global parameter S90TCP_RECONNECT_DELAY with a value of "10". This is the time by which to attempt to perform a reconnect.
5. Open the Triplex project in Workbench and then, from the Tools menu, select Command Prompt.
6. Type "idtpop device" and press Enter.
7. Open the device.idt file in a text editor. The device.idt file is located in the project folder (e.g. C:\Program Files\Proficy\Proficy CIMPLICITY\MyTriplexProject).
8. For the device in question, change the dead_scan_rate from 45 to a smaller number, such as 10, and save the file.
9. In the Command Prompt window, type "scpop device" and press Enter.
10. In Workbench, from the Project menu, select Configuration Update. A message appears asking to copy the master configuration data to run-time data.
11. Click OK.

    When you start the project, it should recover 20 to 30 seconds after a failover/recovery.

    Note:

    - Where SE devices are configured on the port, the mode address must have a valid PLC address even if there are no redundant devices configured on the port.
    - The first time that a connection is established via the S90 Triplex in a project after loading a project later than 6.0 SP1, the memory sizes from the PLC will be read at startup regardless of the setting in the device configuration.

Recommended Configuration for RX7i Devices

1. Open the Triplex project in Workbench and, from the Tools menu, select Command Prompt.
2. Type "idtpop device" and press Enter.
3. Open the device.idt file in a text editor. The device.idt file is located in the project folder (e.g. C:\Program Files\Proficy\Proficy CIMPLICITY\MyTriplexProject).
4. For the device in question, change the dead_scan_rate from 45 to a smaller number, such as 10, and save the file.
5. In the Command Prompt window, type "scpop device" and press Enter.
6. In Workbench, from the Project menu, select Configuration Update. A message appears asking to copy the master configuration data to run-time data.
7. Click OK.

   When you start the project, it should recover 20 to 30 seconds after a failover/recovery.

Series 90 TCP/IP Triplex Enabler used with Server Redundancy

Series 90 TCP/IP Triplex Enabler used with Server Redundancy

The Series 90 TCP/IP Triplex enabler is among the communication interfaces that are supported in CIMPLICITY Server Redundant configurations. The enabler on the standby node will heartbeat all communication paths of the configured devices. If a problem is detected, an alarm will be generated to indicate the communication path in error.

On the Primary Server, alarms indicating individual communication path errors are NOT generated automatically. Diagnostic points that indicate the status of the individual paths can be configured to have an Alarm condition. Since diagnostic points do not reflect the status of the communication paths on the Standby Server, the enabler on the standby node generates alarms to reflect status.

Failure Conditions and Actions using Triplex Enabler

The enabler responds to fault condition as follows.

Failure conditions and actions using Triplex enabler include the following.

- Server failure.
- Server recovery.
- Communication path failure detected by the standby enabler.
- Communication path recovery detected by the standby enabler.

Server Failure

A Server failure occurs when the Secondary Server can no longer communicate to the Primary Server; for example; the project on Primary Server stopped.

The enabler on the standby node becomes the active server and begins collecting data from the highest ranking running Healthy PLC with the Health and active mode bits set.

The enabler updates the diagnostic points to reflect the status of connections on the new active node.

Server Recovery

A Server recovery occurs when the Configured Primary server becomes active and a transition takes place. .

The enabler on the new standby node suspends polling the devices, and returns to heartbeat processing.

The enabler on the recovered node becomes the active and begins collecting data from the highest ranking running Healthy PLC with the Health and mode bits set.

Communication Path Failure detected by the Standby Enabler

A communication path failure occurs when the enabler on the Secondary Server can no longer communicate over a configured communication path to a PLC. .

The enabler generates a $DEVICE alarm to indicate the communication path in error. After all paths to the device are tested, a "summary" alarm is generated, if necessary. The text of the alarm message contains an error code, which is an integer that indicates the connection status of each path. If you turn the error code into a bit string, the bits can be interpreted as follows:

bit 0: primary PLC primary cable

bit 1: primary PLC secondary cable

bit 2: secondary PLC primary cable

bit 3: secondary PLC secondary cable

bit 4: third PLC primary cable

bit 5: third PLC secondary cable

If the bit is equal to 1, then the enabler cannot communicate over that path. As an example, if both cables to the Primary PLC are disconnected, but all other paths are operational, bits 0 and 1 would equal 1, so the error code would equal 3 (000011 = 3).

Communication Path Recovery detected by the Standby Enabler

A communication path recovery occurs where the enabler re-establishes communication over a network path to a PLC..

If all communication paths are now operational to the device, a $DEVICE alarm is generated to indicate full recovery. If a different communication path still is in error, a $DEVICE alarm is generated as described above.

Series 90 TCP/IP Triplex Global Parameters

Series 90 TCP/IP Triplex Global Parameters

- Allow multiple messages.
- Disable/Enable KEEPALIVE
- Maximum outstanding messages.
- Maximum cache.
- Maximum request cache.
- Allow unsolicited communication.
- Device Bit Reverse.
- Device ID is SE
- Reconnect Delay
- DEVICE_TIMESTAMP_UTC
- <PORT>_DEVICE_TIMESTAMP_UTC

Allow Multiple Messages

The following global parameter is used to allow multiple outstanding messages to be sent over a single TCP/IP connection to a Series 90-70 controller via a Series 90-70 Ethernet card.

- S90TCP_ALLOW_MULTIMSG
- <PORT>_ALLOW_MULTIMSG

S90TCP_ALLOW_MULTIMSG

For	Project
Purpose	To allow multiple messages to be sent on all devices on all ports in the project.
Value	Y
Default Value	N

<PORT>_ALLOW_MULTIMSG

Where <PORT> is the number of the port.

For	Port
Purpose	To allow multiple messages to be sent on all devices on a specific port within the project.
Value	Y
Default Value	N

Disable/Enable KEEPALIVE

The following project level global parameters enable you to disable or enable KEEPALIVE.

S90TCP_DISABLE_KEEPALIVE

For	Project	
Purpose	Enable or disable KEEPALIVE for all S90TCP/IP ports.	
Value	The values are:	
	Y	Disables KEEPALIVE for all S90TCP/IP ports.
	N	Enables KEEPALIVE for all S90TCP/IP ports.
Default Value	N	

<PORT>_DISABLE_KEEPALIVE

For	Project	
Purpose	Enable or disable KEEPALIVE for a specified S90TCP/IP port..	
Comments	The specified device communications will log to status log that Keepalives are disabled for confirmation.	
Value	The values are:	
	Y	Disables KEEPALIVE for a specified S90TCP/IP port.
	N	Enables KEEPALIVE for a specified S90TCP/IP port.

Default Value	N	

Maximum Outstanding Messages

If multiple messages is enabled, the following global parameter is used to define the maximum number of outstanding messages that can be sent.

- S90TCP_MAX_POLL_MSG
- <PORT>_MAX_POLL_MSG
- <DEVICE>_MAX_POLL_MSG

S90TCP_MAX_POLL_MSG

For	Project
Purpose	To define the maximum number of outstanding messages that can be sent to all devices on all ports in the project.
Value	1 to 64
Default Value	1

<PORT>_MAX_POLL_MSG

Where <PORT> is the number of the port.

For	Port
Purpose	To define the maximum number of outstanding messages that can be sent to all devices on a specific port within the project.
Value	1 to 64
Default Value	1

<DEVICE>_MAX_POLL_MSG

Where <DEVICE> is the name of the device.

For	Device
Purpose	To define the maximum number of outstanding messages that can be sent to a specific device.
Value	1 to 64
Default Value	1

The following table identifies the order of precedence for valid globals.

Global...	Supercedes
Device-level	Port-level
Port-level	S90TCP-level
S90TCP-level	Default

Note: As long as the communication load is within the capability of the Series 90-70 (as it is configured with the existing PLC application), communications remains reliable without dropouts. If the communication load is beyond the capability of the Series 90, then there may be loss of communications or it may be observed that the Triplex interface will switch between viable communication paths. If this occurs, reduce the number of concurrent messages.

When confirming that the number of concurrent messages chosen is compatible with the PLC application, be sure to validate the application with the Series 90-70 in run mode running the complete PLC application.

Maximum Cache

The following global parameter is used to determine the maximum number of caches available on a project or port basis.

Note: <PORT>_MAX_CACHE has a higher precedence than does the S90TCP_MAX_CACHE.

- S90TCP_MAX_CACHE
- <PORT>_MAX_CACHE

S90TCP_MAX_CACHE

For	Project
Purpose	To set the maximum number of caches for the project.
Value	0 - 2000 (New projects can be configured for up to 2000 caches; existing projects can be configured for up to 1024 caches.)
Default Value	1024

Use the global parameter S90TCP_MAX_CACHE to change the cache size for the project from its default size.

<PORT>_MAX_CACHE

For	Port
Purpose	To set the maximum number of caches for a single port.
Value	0 - 2000 (New projects can be configured for up to 2000 caches; existing projects can be configured for up to 1024 caches.)

Comments	<PORT> is the name of the S90 Triplex port to be configured such as S90TRI0, S90TRI1, etc. The value assigned to the global is the maximum port size. If a value larger then the maximum is specified, the largest allowed value will be used.
Default Value	1024

Use the global parameter <PORT>_MAX_CACHE to change the cache size for a specific port from its default size. Where <PORT> is the number of the port; for example S90TR10_MAX_CACHE.

Note: In some configurations, supporting a larger message size may be beyond the capability of Series 90-70 (as it is configured with the existing PLC application). This is more likely to occur where the IC697CMM741 Ethernet cards are used for communications on the Series 90-70. If communications becomes unstable as a result of using the larger message size, it is recommended that the application continue to use a cache size of 1024 bytes.

When confirming that the number of concurrent messages chosen is compatible with the PLC application, be sure to validate the application with the Series 90-70 in run mode running the complete PLC application.

Maximum Request Cache

By default, the Triplex device communication interface will queue up to 100 requests per device for processing. For projects with a large number of caches (due to a large number of regions to read or a large number of scan rates for a single device), it is possible to exceed this value.

- S90TCP_MAX_REQUEST_CACHE
- <PORT>_MAX_REQUEST_CACHE
- <DEVICE>_MAX_REQUEST_CACHE

S90TCP_MAX_REQUEST_CACHE

For	Project
Purpose	To set the maximum number of queued/cached reads per device/group.
Value	0 - n
Default Value	100

<PORT>_MAX_REQUEST_CACHE

Where <PORT> is the number of the port.

For	Port
Purpose	To set the maximum number of queued/cached reads per device/group for a single port.
Value	0 - n

Default Value	100

<DEVICE>_MAX_REQUEST_CACHE

Where <DEVICE> is the number of the device.

For	Device
Purpose	To set the maximum number of queued/cached reads per device/group for a single device.
Value	0 - n
Default Value	100

Allow Unsolicited Communication

By default, the ability to process unsolicited processing is disabled on all Triplex ports.

The following global parameter is available to enable unsolicited processing.

S90TCP_ALLOW_UNSO

For	Project	
Purpose	To enable unsolicited processing on an S90TRI0 port. **Important:** Only one S90TRI0 port should be used per computer when unsolicited processing is enabled.	
Value	Y	Enables unsolicited processing
	N	Disables unsolicited processing
Default Value	N	

Device Bit Reverse

By default, bit ordering in BYTE, WORD, or DWORD starts with the least significant bit first.

The following global parameter is available to modify the default behavior.

S90TCP_DC_BIT_REVERSE

For	Project
Purpose	To set bit pattern from least significant first to most significant first.
Value	Y
Default Value	N

<DEVICE_ID> Is SE

The following global parameter is used to indicate whether a specific device is a VersaMax SE device.

S90TCP_<DEVICE_ID>_IS_SE

For	Project
Purpose	To indicate if a specific device is a VersaMax SE device, where <DEVICE_ID> is the name of the device in the project.
Value	Y
Default Value	N

Reconnect Delay

The following global parameter is used to define the time to delay between a connection failure and reconnect attempt.

S90TCP_RECONNECT_DELAY

For	Project
Purpose	The time frame to delay before attempting to reconnect after a connection failure.
Value	n (where n is a value greater than zero)

Chapter 50. Series 90 Triplex Installation Verification Procedures

Series 90 Triplex Installation Verification Procedures

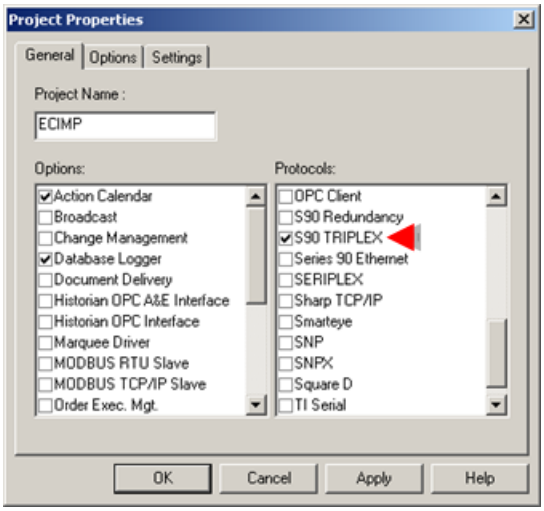
The Series 90 Triplex Diagnostic Program (srtpdiag.exe) is a program provided with the Series 90 Triplex Communications enabler that you can use to check the basic operation and configuration of your network without starting CIMPLICITY software.

For this program to function, CIMPLICITY software's Series 90 Triplex Communications enabler must be successfully installed, and you must have configured the S90TRI port and Series 90 devices using CIMPLICITY application configuration functions.

- Start the diagnostic program.
- Connect to a Series 90 programmable controller in the S90TRI network.
- Display data for a different domain.
- Work in the SRTPDiag window.

Start the diagnostic program

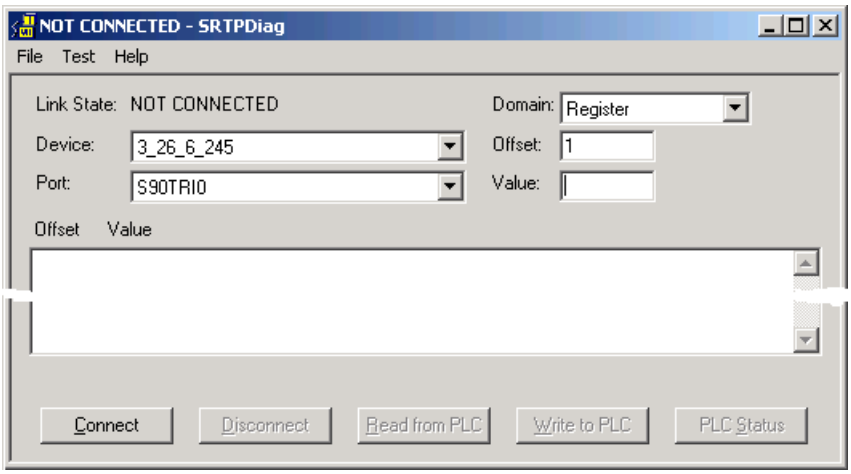
1. Make sure that S90 TRIPLEX is checked on the General tab in the Project Properties dialog box.



2. Select **Project>Equipment>Diagnostics** in the Workbench left-pane.

3. Double-click **Series 90 Diagnostic**.

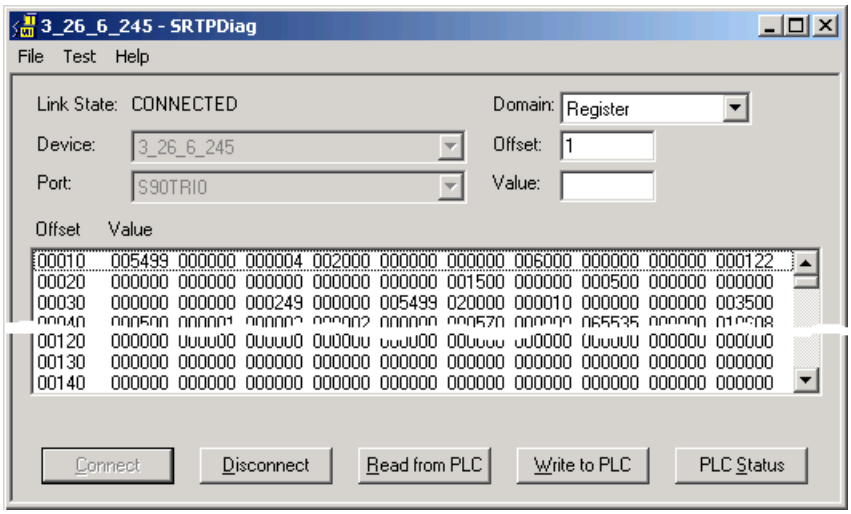
Result: The SRTPDiag window opens.



Connect to a Series 90 programmable controller in the S90TRI network:

4. Click the drop-down list button to the right of the Device field to display the list of currently configured devices, and select a Series 90 Triplex device from the list.
5. Click the drop-down list button to the right of the **Port** field to display the list of currently configured ports, and select the S90TRI port to use.
6. Click the drop-down list button to the right of the **Domain** field to display the list of currently supported domains (memory types *(page 849)*), and select a domain to display.
7. Click Connect.

The program makes the connection with the programmable controller, and displays the values contained in the requested domain.



Display the data for another domain:

Select the domain from the drop-down list in the Domain input field.

Result:The data for the new domain is automatically read and displayed in the data box.

You can use the menus and push buttons to do the following:

- Toggle between manual and automatic data update modes.
- Request manual reads of domain data
- Request writes to the programmable controller
- Display programmable controller status
- Perform a read load test
- Perform a write load test

Work in the SRTPDiag window

- Toggle between data update modes.
- Request manual reads of domain data
- Write test data to the programmable controller.
- Check Series 90 Triplex status.
- Perform a read load test.
- Perform a write load test.
- Series 90 TCP/IP supported memory types.

Toggle Between Data Update Modes

Normally, domain data is only read when the domain display is initiated or when you click on the Read from PLC or Write to PLC buttons.

todo: To update data automatically:

1. Click on the File menu.
2. Select Live Update from the submenu.

   Result: When Live Update is enabled, the domain values are read and updated automatically approximately every 3 seconds.

   todo: To return to manual mode:

3. Click the File menu. Note that Live Update has a check mark next to it.

4. Select Live Update again to turn off the Live Update mode.

Request Manual Reads of Domain Data

If you are not in Live Update mode, you can manually request reads of the domain by selecting Read from PLC.

Write Test Data to the Programmable Controller

1. Select the domain to write to.

   Note: If you select a domain that is read-only, Write to PLC is disabled (grayed out).

2. Enter the memory location offset that you want to write to in the **Offset** field.
3. Enter the value you want to write in the **Value** field.
4. Select Write to PLC.

The value is written to the offset location, and the data display box updates to reflect your write request.

Check Series 90 Triplex Status

1. Click PLC Status to display the current status of the programmable controller. An SRTP Diagnostic dialog box opens.
2. Click OK to close the dialog box.

Perform a Read Load Test

You can use srtpdiag.exe to perform read load tests on your network. The read load test is performed in two parts:

- The first part tests the number of words per second that can be read over the network.

- The second part tests the number of read requests per second that can be made over the network.

The test reads data from all domains on the currently selected Series 90 device.

todo: To perform a read load test:

1. Click the Test menu.
2. Select Read Load Test... from the submenu.

   The Read Load Test dialog box opens.

3. Enter the number of seconds you want to run the test in the **Number of Seconds** field.
4. Click OK to start the test.

   The test cycles through reads of all memory types on the current device for the number of seconds you specify. At the end of the time period, an SRTP Diagnostic dialog box opens, telling you the average number of words read per second over the test period.

5. Click OK to proceed to the second part of the read test.

   The test again cycles through reads of all memory types on the current device for the number of seconds you specified on the Read Load Test dialog box. At the end of the time period, an SRTP Diagnostic dialog box opens, telling you the average number of read requests made per second over the test period.

6. Click OK to terminate the test.

Perform a Write Load Test

Warning: This test will overwrite the contents of the register domain (%R). Do not run this test if the register domain of the programmable controller contains important information.

You can use srtpdiag.exe to perform write load tests on your network. The write load test tests the number of words per second that can be written over the network.

The test writes data to the register domain (%R) on the currently selected programmable controller.

todo: To perform a write load test:

1. Click on the Test menu.
2. Select Write Load Test... from the submenu.

The Write Load Test dialog box opens.

3. Enter the number of seconds you want to run the test in the **Number of Seconds** field,
4. Click OK to start the test.

   The test cycles through writes of register memory on the current device for the number of seconds you specify. At the end of the time period, an SRTP Diagnostic dialog box opens, telling you the average number of words written per second over the test period.

5. Click OK to terminate the test.

Series 90 Triplex Supported Memory Types

The Series 90 Triplex Communications enabler supports reading and writing point data to the Series 90-70 and Series 90-30 controllers.

Note: If you add a block to a programmable controller or resize domains, you must stop and restart the Series 90 Triplex Communications enabler in order for it to see the changes.

Data may be read from the following memory types:

Memory Type	Description	Memory Type	Description
%AI	Analog Input Table	%Q	Discrete Output Table
%AQ	Analog Output Table	%R	Register Memory
%G	Genius Seamless	%S	System Fault Table
%I	Discrete Input Table	%SA	Special Contacts A
%L	Local Memory *	%SB	Special Contacts B
%M	Discrete Internal	%SC	Special Contacts C
		%T	Discrete Temporary

* %L may only be used on Series 90-70

Data may be written to the following memory types:

Memory Type	Description	Memory Type	Description
%AI	Analog Input Table	%L	Local Memory *
%AQ	Analog Output Table	%M	Discrete Internal
%G	Genius Seamless	%Q	Discrete Output Table

%I	Discrete Input Table	%R	Register Table
		%T	Discrete Temporary

* %L may only be used on Series 90-70

CAUTION: Do not write data to the system registers (%S, %SA, %SB, and %SC). Doing so may interfere with the normal operation of the programmable controller.

Chapter 51. Siemens TI Serial Communications

About Siemens TI Serial Communications

The Siemens TI Communications option supports serial communications over RS-232 to Siemens/ TI series 505 PLCs. This communications option uses the NITP protocol to communication with the serial port on the TI CPU card.

Siemens TI Serial Communications Supported Devices

Siemens TI Serial Communications supports communication between Windows systems and Siemens/TI series 505 PLCs.

Siemens TI Serial Communications Supported Memory Types

The following data types can be read and written from CIMPLICITY communication interface:

Domain Name	Data Type	Direction (as seen by CIMPLICITY software)
Discrete input registers	Digital	Read/Write
Discrete output registers	Digital	Read/Write
Word input registers	Analog	Read/Write
Word output registers	Analog	Read/Write
Constant Memory	Analog	Read/Write
Variable Memory	Analog	Read/Write
Internal Control Relays	Digital	Read/Write
Timer Current values Preset values	Analog Analog	Read/Write Read/Write

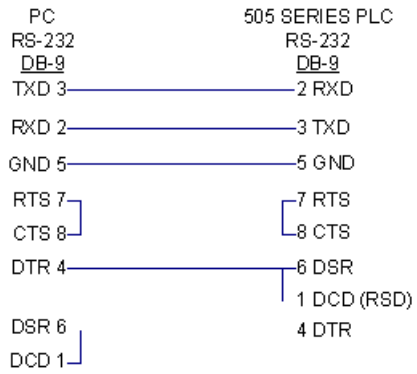
Siemens TI Serial Communications Related Documents

You should have the following documentation available when configuring this interface:

Siemens SIMATIC TI575 Task Code User Manual.

Siemens TI Serial Communications RS-232 Cable Configuration

The RS-232 cable between the CIMPLICITY computer and Siemens/TI 505 Series PLCs must be wired as follows:



CIMPLICITY Configuration for Siemens TI Serial

CIMPLICITY Configuration for Siemens TI Serial

When you are configuring ports, devices, and device points, you will be asked for some information that is specific to Siemens TI Serial communications.

Siemens TI Communications Port Configuration

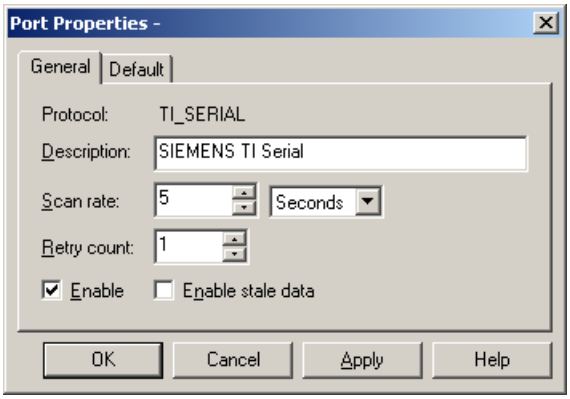
Siemens TI Communications Port Configuration

1. In the **Protocol** field, select **TI_SERIAL** from the list of available protocols
2. In the **Port** field, select that communication port that will be used for Siemens TI Serial communications.

When you click **OK** to create the port, the Port Properties dialog box for the protocol opens.

Important: You can configure a maximum of only four ports in any Siemens TI Communications project.

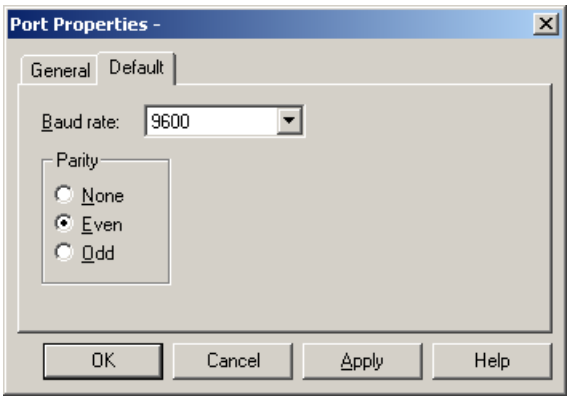
General Port Properties



Use the General tab in the Port Properties dialog box to enter general information for the port. You can define the following:

Description	Enter an optional description to help you identify the port.
Scan Rate	Enter the base scan rate for the port. Point scan rates will be multiples of the base rates. You can specify a scan rate in **Ticks** (100 ticks = 1 second), **Seconds**, **Minutes**, or **Hours**.
Retry Count	If communications cannot be established to a device on this port, the device is considered to be down and a $DEVICE_DOWN alarm is generated. Once a device is down, periodic attempts are made to resume connection to it.
	Enter the number of scans to wait before attempting to reconnect to a device on this port after a communications error is detected.
Enable	Check this check box if you want the port to be enabled when the project starts. If you clear this check box, the port will not be enabled when the project starts, and points will not be available for devices on the port.

Default Port Properties



Use the Default tab in the Port Properties dialog box to enter communications information for the port. You can define the following:

Baud Rate	Enter the baud rate for communications. Click the drop-down list to the right of the input field and select a rate from the list.
Parity	Select the parity to be used for the communications.

Siemens TI Communications Device Configuration

Siemens TI Communications Device Configuration

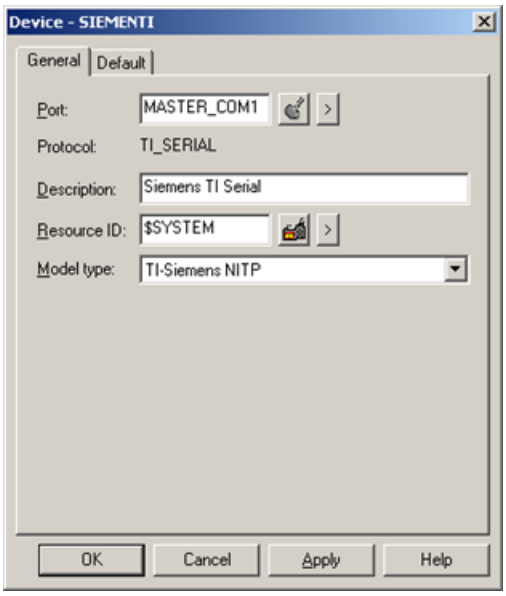
You must configure one device for each Siemens TI device from which you will collect data using Siemens TI Serial communications.

When you create a new device for Siemens TI Serial communications, enter the following information in the New Device dialog box:

1. In the **Device** field, enter the name of the device you are configuring.
2. In the **Port** field, select the Siemens TI Serial port to be used by the device.

   When you click **OK** to create the device, the Device Properties dialog box opens.

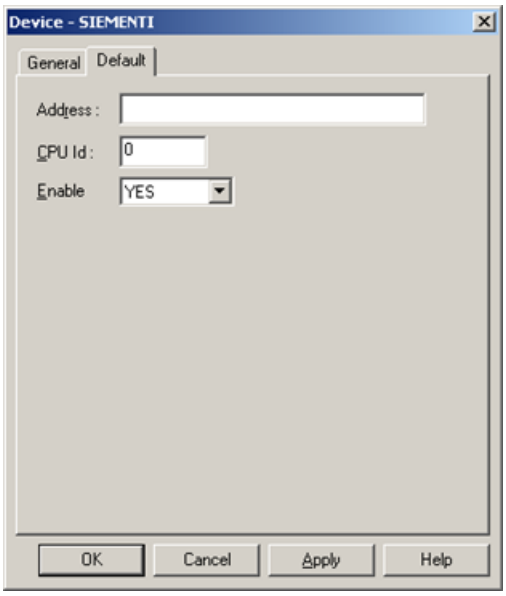
General Device Properties



Use the General tab in the Device dialog box to enter general information for the device. You can define the following:

Port	Select the port for this device. Click buttons to the Port field to select the port as follows.	
		Display the list of ports and select one.
		Create a new port, edit the current port, or select a port from the list of ports.
Description	Enter an optional description to help you identify the device.	
Resource	Enter a resource that can be associated with this device for alarm generation. Click buttons to the right of the **Resource** field to select the resource, as follows.	
		Display the list of resources and select one.
		Create a new resource, edit the current resource, or select a resource from the list of resources.
Model Type	Select the type of device. Click the drop-down button to the right of the input field to display your choices, then make a selection. For this protocol, the only choice is:	
	TI – Siemens NITP	

Default Device Properties



Use the Default tab in the Device dialog box to enter information about communications for the device. You can define the following:

Address	The field is not used.
CPU Id	Enter **0** in the CPU Id field.
Enable	Select YES to enable the device when the project starts. If you enter NO, the device will not be enabled and points associated with the device will be unavailable.

Siemens TI Communications Point Configuration

Siemens TI Communications Point Configuration

Once your devices are configured, you may configure points for them. Fields in the Point Properties dialog box have configuration values that are unique to or have special meaning to Siemens TI device communications.

General Point Properties

On the General tab of the Point Properties dialog box, clear the **Read only** check box for all the memory types.

Device Point Properties

On the Device tab in the Point Properties dialog box:

Update Criteria	The update criteria field determines how the data will be requested.

	Select On Change or On Scan for points whose values should be polled by the Siemens TI driver at regular intervals.
	Select On Demand On Scan or On Demand On Change for points whose values should be polled by the Siemens TI driver at regular intervals while users are viewing them.
Address	A Siemens TI device point address consists of a one to three character alphabetic field followed by an n-digit decimal-numeric field. The numeric field reflects the offset within an addressing domain of the particular point. The alphabetic fields map to domains according to the following table:
	Enter the point address.

Valid data types, address formats, memory ranges and data types for point addresses are:

Data Memory Type	Address Format	Memory Range	Data Type
Discrete input registers	Xn	**1 - 2048**	Digital
Discrete output registers	Yn	**1 - 2048**	Digital
Word input registers	WXn	**1 - 2048**	Analog
Word output registers	WYn	**1 - 2048**	Analog
Constant Memory	Kn	**1 - 2048**	Analog
Variable Memory	Vn	**1 - 32768**	Analog
Internal Control Relays	Cn	**1 - 4095**	Digital
Timer/counter current values	TCCn	**1 - 1024**	Analog
Timer/counter preset values	TCPn	**1 - 1024**	Analog

Note: Since discrete inputs (X) and discrete outputs (Y) share the same memory type, do not use the same register number for a discrete input and a discrete output (for example X0001 and Y0001) or unexpected results may occur.

Since word inputs (WX) and word outputs (WY) share the same memory type, do not use the same register number for a word input and a word output (for example WXOOOI and WYOOOI) or unexpected results may occur.

Normally TCC and TCP memory types share the number. For example, TCC5 and TCP5 share the same address.

Siemens TI Performance Notes

Optimize Point Configuration

Point performance is based on the number of points being processed. If your project is set up with 2 points, for example, v1 and v125, it will get every point in between and process each one, thus slowing performance.

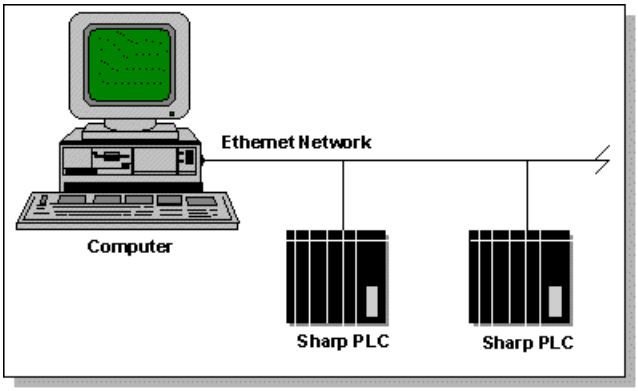
You can do the following to optimize the performance of points:

- Use staggered scan rates.
- Avoid gaps in allocated data addresses or use small gaps. This will reduce the transfer of "meaningless" values.

Chapter 52. Sharp TCP/IP Communications

About Sharp TCP/IP Communications

The Sharp TCP/IP communications module lets you exchange data between a CIMPLICITY project and Sharp PLCs over an Ethernet network using TCP/IP protocol.



This enabler supports the following CIMPLICITY features:

- Polled reads at user defined rates
- Poll after setpoint
- Triggered reads
- Unsolicited data
- Analog deadband
- Alarm on communications failure
- Server Redundancy configurations

This enabler supports the following data types:

- Signed 8, 16, and 32 bit integers
- Unsigned 8, 16, and 32 bit integers
- Text
- Arrays

Sharp TCP/IP Communications Supported Devices

The following models of the Sharp New Satellite JW Series Programmable Controllers are supported by this communications interface:

JW-50CU

JW-70CU

JW-100CU

Sharp TCP/IP Communications Supported Memory Types

The following data memory types are supported by this communications interface:

File 0 register (includes I/O relay, Retentive relay, General purpose relay, TMR, CNT time limit contact, TMR, CNT, MD, and Register memory)

File-1 register

File-2 register

File-3 register

File-4 register

File-5 register

File-6 register

File-7 register

Write Enable Status

Sharp TCP/IP Communications Hardware Configuration Requirements

To use Sharp TCP/IP communications, you must have the following installed and configured in your computer:

- An Ethernet card
- TCP/IP communications

You must also have an Ethernet Interface Module type JW-50CM installed in the programmable controller.

Sharp TCP/IP Communications Related Documents

You should have the following documentation available when configuring this interface:

New Satellite JW series Programmable Controller A User's Manual Ethernet Interface Module type JW-50CM Rev 1.0.

Sharp Programmable Controller New Satellite JW Programming Manual

Sharp Programmable Controller New Satellite JW Instruction Manual

Sharp PLC Communications Configuration Checklist

Prior to using this communication interface, you must perform the following steps:

1. Install an Ethernet card in your computer.
2. Configure the minimum TCP/IP Communications for your computer, including assignment of a name and TCP/IP address for the computer.
3. Assign a TCP/IP address for each Sharp programmable controller.

   If you are on a company-wide or standardized network, obtain your TCP/IP addresses from your network administrator. Please refer to the PLC documentation for assistance in developing the needed ladder logic.

4. For each programmable controller communicating with CIMPLICITY software via Ethernet, assign a socket port number for the connection.

   If you are on a company-wide or standardized network, obtain your Ethernet port number from your network or system administrator. If you need to receive unsolicited data messages, use port number 6000H, 6001H, 6002H or 6003H, depending on the channel configuration. Please refer to the PLC documentation for assistance in developing the needed ladder logic.

5. The CIMPLICITY project always initiates the TCP/IP connection. Configure all TCP/IP channels in the Sharp PLC used by CIMPLICITY software as **TCP Passive**.
6. Configure the Windows TCP/IP address for each programmable controller on your computer.

   Use **C:\Windows\System32\drivers\etc\hosts** for the configuration.

In the **hosts** file, add one line for each programmable controller using the following format:

<IP address> <PLC Name> <Alias Name>

Separate the fields with at least one space. For example, if the IP address configured for a programmable controller is 1.2.3.4 and its host name is "SHARP", its line in the hosts file is:

1.2.3.4 SHARP sharp

Sharp TCP/IP Communications Test Program

Sharp TCP/IP Communications Test Program

You can use the Sharp Test Program (**sharp_diag.exe**), provided with the Sharp TCP/IP communications option, to check your network configuration without starting a CIMPLICITY project. You need to run this program from a command prompt window for your project.

todo: To open a command prompt window for your project:

1. Open your project in the CIMPLICITY Workbench.
2. Select Command Prompt... from the Tools menu.
3. Enter the command you want at the MS-DOS prompt in the Command Prompt window.
   - If the command succeeds a series of information messages displays
   - If the command fails, then an appropriate error message displays. In this case, check your Ethernet connections, ladder logic and your computer configuration to ensure that all are configured correctly. Any error or inconsistency will prevent successful communication.

   You can find detailed information about the Sharp PLC communication commands in the "New Satellite JW series" Programmable Controller A User's Manual Ethernet Interface Module type JW-50CM Rev 1.0 manual.

Verify the PLC Connection

Type the following command at the MS-DOS prompt in your command window to verify the connection between your computer and a Sharp PLC:

```
sharp_diag <Host Name> <Socket Port Number>
```

If the test succeeds, output with the following format displays:

```
Host name: <Computer name>     PLC name: <PLC name>     Port number: <Port
 number>
<Computer name> is bound to the socket.
<Computer name> is connected to <PLC name>.
Communication between <Computer name> and <PLC name> is disconnected
 (normal exit).
```

Example-Connection Verification

For example, if the computer name is ABC123, the configured host name is SHARP and the configured port number is 24576, the output for a command:

```
sharp_diag ABC123 24576
```

looks like this:

```
Socket port: 24576
Host name: ABC123     PLC name: SHARP     Port number: 24576
ABC123 is bound to the socket.
ABC123 is connected to SHARP.
Communication between ABC123 and SHARP is disconnected (normal exit).
```

Test PLC Communications

Type the following command at the MS-DOS prompt to test the network configuration and the communication between your computer and the PLC:

```
sharp_diag <Host Name> <Socket Port Number> COM <command> [COM <…> (up to
 10 commands)]
```

If the test succeeds, output with the following format displays:

```
Host name: <Computer name>     PLC name: <PLC name>     Port number: <Port
 number>
<Computer name> is bound to the socket.
<Computer name> is connected to <PLC name>.
Command #1: <…>
Command #1 has been sent to the PLC successfully.
Command #2:…
…
Successfully read data fro   (Sharp TCP/IP)m the PLC. The length of the
 response message = <…>
…
Response for command #1: <…>
Response for command #2: …
…
```

```
Communication between <Computer name> and <PLC name> is disconnected
 (normal exit).
```

Example - Communication Verification

For example, if the computer name is ABC123, the configured host name is SHARP and the configured port number is 24576, with two commands:

```
sharp_diag ABC123 24576 COM a2 com 24 0 0 2 1 0
```

the output looks like this:

```
Host name: ABC123     PLC name: SHARP     Port number: 24576
ABC123 is bound to the socket.
ABC123 is connected to SHARP.
Command #1: a2
Command #1 has been sent to the PLC successfully.
Command #2: 24 0 0 2 1 0
Command #2 has been sent to the PLC successfully.
Successfully read data from the PLC. The length of the response message =
 98
Response for command #1: a2 0 97 3 28 5
Response for command #2: 24 0 0 0 2 1 0 1
Communication between ABC123 and SHARP is disconnected (normal exit).
```

CIMPLICITY Configuration for Sharp TCP/IP

CIMPLICITY Configuration for Sharp TCP/IP

When you are configuring ports, devices, and device points, you will be asked for some information that is specific to Sharp TCP/IP communications.

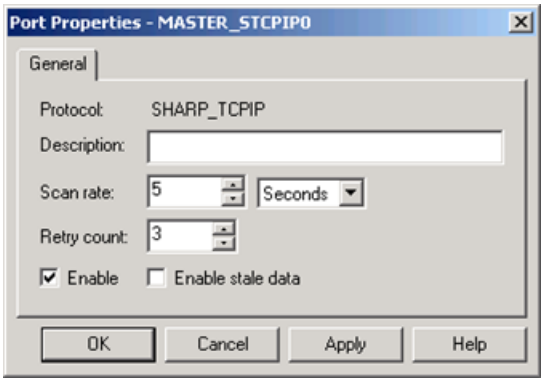
Sharp TCP/IP Port Configuration

Sharp TCP/IP Port Configuration

1. In the **Protocol** field, select **SHARP_TCPIP** from the list of available protocols.
2. In the **Port** field, select the communication port that will be used for Modbus TCP/IP communications.

When you select OK to create the port, the Port Properties dialog box for the protocol opens.

General Port Properties



Use the General tab in the Port Properties dialog box to enter general information for the port. You can define the following:

Description	Enter an optional description to help you identify the port.
Scan Rate	Enter the base scan rate for the port. Point scan rates will be multiples of the base rate. You can specify a scan rate in **Ticks** (100 ticks = 1 second), **Seconds**, **Minutes**, or **Hours**.
Retry Count	If communications cannot be established to a device on this port, the device is considered to be down and a $DEVICE_DOWN alarm is generated. Once a device is down, periodic attempts are made to resume connection to it.
	Enter the number of scans to wait before attempting to reconnect to a device on this port after a communications error is detected.
Enable	Set this check box if you want the port to be enabled when the project starts. If you clear this check box, the port will not be enabled when the project starts, and points will not be available for devices on the port.

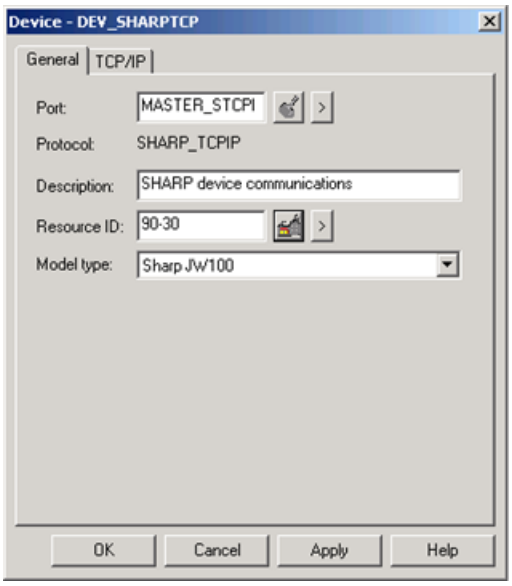
Sharp TCP/IP Device Configuration

Sharp TCP/IP Device Configuration

1. In the **Device** field, enter the name of the device you are configuring.
2. In the **Port** field, select the Sharp TCP/IP port to be used by the device.

   When you click **OK** to create the device, the Device Properties dialog box opens.

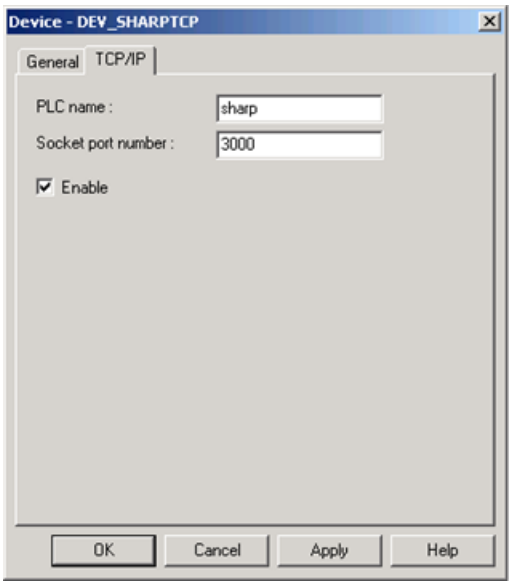
General Device Properties



Use the General tab in the Device dialog box to enter general information for the device. You can define the following:

Port	Select the port for this device. Click buttons to the Port field to select the port as follows.	
		Display the list of ports and select one.
		Create a new port, edit the current port, or select a port from the list of ports.
Description	Enter an optional description to help you identify the device.	
Resource	Enter a resource that can be associated with this device for alarm generation. Click buttons to the right of the **Resource** field to select the resource, as follows.	
		Display the list of resources and select one.
		Create a new resource, edit the current resource, or select a resource from the list of resources.
Model Type	Select the type of device. Click the drop-down button to the right of the input field to display your choices, then make a selection. For this protocol, the choices are:	
	Sharp JW50 Sharp JW70 Sharp JW100	

Device Properties



Use the TCP/IP tab in the Device dialog box to enter information about communications for the device. You can define the following:

PLC name	The PLC name must be configured exactly as it is configured under step 5 of Sharp PLC Communications Configuration Checklist.
Socket port number	The socket port number is a decimal number that must match the port number configured on the Sharp PLC.
	For unsolicited data messages, the socket port number must be one of the following: 24576 (6000H) 24577 (6001H) 24578 (6002H) 24579 (6003H) Base your selection on the channel configuration on your Sharp PLC.
Enable	Set this check box to enable the device when the project starts. If you clear this check box, the device will not be enabled and points associated with the device will be unavailable.

Sharp TCP/IP Point Configuration

Sharp TCP/IP Point Configuration

Once your devices are configured, you may configure points for them. Fields in the Point Properties dialog box have configuration values that are unique to or have special meaning to Sharp TCP/IP communications.

General Point Properties

On the General tab in the Point Properties dialog box, you may configure points for **Read** or **Read/ Write** access.

Note: All writes are disabled by default. To enable data memory write on a device, you must first set its Write Enable Status.

For Write Enable Status points, set the following:

- **Access** must be Read/Write.
- **Point type** must be **Boolean**.
- **Element** must be set to **1**.

Device Point Properties

On the Device tab in the Point Properties dialog box:

Update Criteria	The update criteria determine how the data will be requested.
	Select **On Change** or **On Scan** for points whose values should be polled by the Sharp TCP/IP driver at regular intervals.
	Select **On Demand On Scan** or **On Demand On Change** for points whose values should be polled by the Sharp TCP/IP driver while users are viewing them.
	Select **Unsolicited** for points whose value will be sent by the programmable controller as necessary.
Address	Enter the point address of the data in the following format: <File Register #>:<OFFSET> *(page 868)*
	The offset is entered in octal.
	Please refer to the Sharp PLC documentation for the detailed information on these memory types.
	Use the following address for a Write Enable Status point:
	WRITE_MODE

Point Address Formats

Use the table below to determine how to specify your file register number and offset.

File Register Type	Data Memory Type	Offset (Octal)
0	I/O relay	**0 - 77 (JW50) 0 - 177 (JW70)**
0	Retentive relay	**700 - 777**
0	General purpose relay	**1000 - 1577**
0	TMR, CNT time limit contact	**1600 -1777**
0	TMR, CNT, MD	**2000 - 3777**
0	Register	**4000 - 17777**
1	File-1 register	**0 - 177777**

2	File-2 register	**0 - 177777**
3	File-3 register	**0 - 177777**
4	File-4 register	**0 - 177777**
5	File-5 register	**0 - 177777**
6	File-6 register	**0 - 177777**
7	File-7 register	**0 - 177777**

Sharp TCP/IP Unsolicited Data

Sharp TCP/IP Unsolicited Data

Unsolicited data is supported for the Sharp TCP/IP communication interface.

- You must develop ladder logic to send the unsolicited data from a Sharp PLC to the CIMPLICITY host computer.
- Use the Sharp SEND/RECEIVE instruction to send data from the PLC to the CIMPLICITY host computer.
- The ladder logic should also check to ensure that the connection has been made and that the requests are acknowledged.
- The destination file address used in the SEND/RECEIVE instruction must exactly match the unsolicited point address in the CIMPLICITY project.
- For your CIMPLICITY project to receive the data, you must configure the computer's IP address at the correct location in the "Station Number Correspondence Table" of the Ethernet module in the Sharp PLC.
- Any CIMPLICITY point with the same starting address as the unsolicited data and whose size is less than or equal to the size of the unsolicited data updates when the PLC sends the unsolicited data.

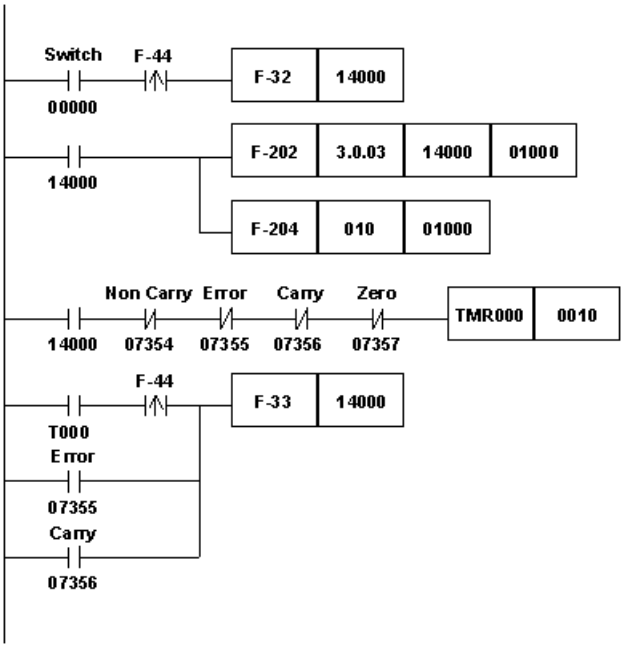
If you are implementing polled and unsolicited communications in Sharp Ethernet module JW-50CM, do the following:

- Configure one device in your CIMPLICITY project for polled data.
- Configure a second device at the same IP address, and a different socket port number for unsolicited data.

See the New Satellite JW Series Programmable Controller - A User's Manual - **Ethernet** Interface Module type JW-50CM Rev 1.0 for details about configuring the Ethernet module.

Sample Ladder

The sample ladder logic, shown below, sends eight (8) bytes of data, starting at address 1000, from File 0 through channel 0 (address 6000H) to configured station number 3.Logic



The data is sent to the CIMPLICITY computer whenever the **Switch** in the ladder diagram is activated.

Any CIMPLICITY point with a starting point address **0:1000** whose point size is less than or equal to 8 bytes updates when the PLC sends the unsolicited data. For example, if the address 1000 in PLC contains a value of 5, the following point updates occur:

- An ANALOG_8 point at 0:1000 updates to 5.
- An ANALOG_32 point at 0:1000 updates to 5.
- A Boolean point at 0:1000 with an address offset 0 updates to 1.

Sharp TCP/IP Communications Troubleshooting

Please check the CIMPLICITY Status Log Viewer first if you encounter problems with your device communications. It often provides helpful hints on communication problems that you might encounter, especially during initial configuration.

This section lists some of those messages, and where appropriate, actions you can take to resolve the problems that cause them.

Other messages for this communications option may also appear in the CIMPLICITY Status Log Viewer. If you need help interpreting the messages in the CIMPLICITY Status Log Viewer, please contact the CIMPLICITY Technical Support Hotline.

Skip Polling

Skipping polling around <POINT ID> because request is already pending.

Cause	The Scan Rate you selected is faster than the data can be collected. This is not an error.

Gethostname for <hostname> failed errno <number>

Cause	The host name and socket port number entered in the TCP/IP property tab are not configured in your host file.
Resolution	To resolve this problem, configure the host name into your computer system as described in step 5 of the Sharp PLC Communications Configuration Checklist. Remember that the names or aliases must match exactly.

Unable to Get A Socket

Unable to get a socket errno = <number>

Cause	You do not have enough sockets in your system.
Resolution	Contact your system administrator for help in tuning the operating system.

Bind Failed

Bind() for <hostname> port <port number> failed REFUSED

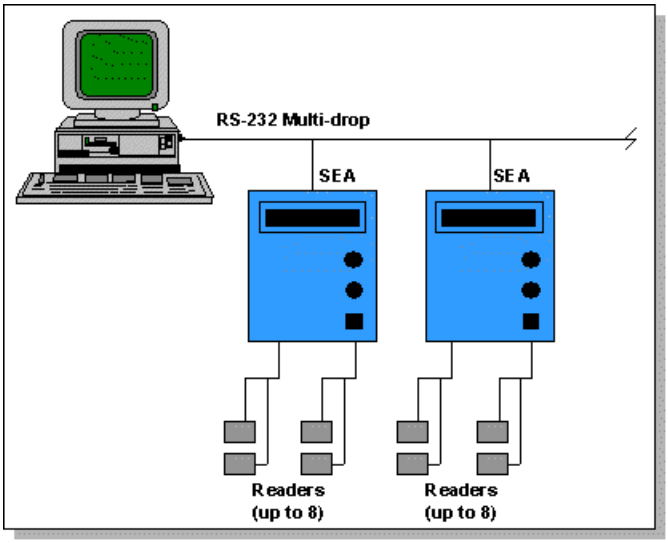
Cause	You can get this message under any of the following circumstances: • You have not properly configured the socket port number within your CIMPLICITY application with that configured on the PLC. • At startup, another application is using your port or the socket has not cleared from a previous use. • The CIMPLICITY configuration for the device is inconsistent with that of the programmable controller.

Resolution	To determine if the socket is in use, open a Command Prompt window from the project's Workbench, then type the following command: **netstat** Under "Foreign address", find the line with your configured **<**PLC_name**>:<**Port_number**>** combination. If the last column says **TIME_WAIT**, the connection was previously in use and is in the process of cleaning up. No further action is needed. If the last column says **ESTABLISHED**, the socket port is in use. Check to make sure that you do have a connection (you can usually tell by examining the LEDs on your programmable controller's Ethernet card). If you do not have a connection, contact your system administrator for assistance in locating the application using the port. If the command yields no information, this usually indicates that the IP address or socket port number configured on the computer does not match the configuration in the programmable controller. Review the configuration on both the computer and programmable controller and correct the mismatch.

Chapter 53. Smarteye Electronic Assembly Communications

About Smarteye Electronic Assembly Communications

The Smarteye Electronic Assembly (SEA) Communications enabler uses the computer's serial port to communicate with Smarteye Electronic Assemblies on a multi-drop RS-232 or RS-422 network.



Using this option, CIMPLICITY software can:

- Poll points at a user-defined scan rate
- Read and write single points
- Issue alarms to users when a device goes down or a reader error occurs
- Monitor reader calibration

This enabler can:

- Support up to thirty-two (32) SEAs per serial port.
- Support up to eight (8) Smarteye readers per SEA.
- Support Polled, Handshake, and Open Loop modes of the SEA.

This enabler supports the following CIMPLICITY features:

- Polled read at user defined rates
- Poll after setpoint

- Triggered reads
- Unsolicited data (only in RS-232 point-to-point configurations)
- Alarm on communications failure
- Server Redundancy configurations

This enabler supports the following data types:

- Signed 8, 16, and 32 bit integers
- Unsigned 8, 16, and 32 bit integers
- Floating point
- Text
- Arrays

Smarteye Electronic Assembly Supported Data Types

The Smarteye Electronic Assembly communicates using a change-driven, message oriented protocol. The Smarteye Electronic Assembly device communication enabler processes the SEA messages and uses them to update a virtual device with five regions of memory.

The first four regions each have one location for each reader on the SEA.

Labels

Label values are available in one numeric and 12 string formats. Eight of the string formats include a conveyance system ID.

Errors

Error values are available as a raw error numbers or translated into error messages.

Diagnostics

Diagnostic values correspond to reader calibration values. They are displayed as strings.

Conveyance System Ids

Conveyance System IDs are stored internally in the enabler and are used to build some of the label string formats.

For example, if a label string format includes the conveyance system ID "ABCD" as a suffix, and a label "1234" passes the reader, the format gives a value of "1234ABCD".

System Data

The System region has only two locations: the poll point and the command point.

- **Poll Point**

Each time the Poll point is polled, the enabler queries the SEA for new data and updates other region values with unsolicited data. The value of the Poll point is the number of responses received on one poll cycle.

- **Command Point**

Data written to the Command point is parsed as a command to the SEA or a Smarteye reader. The corresponding command is issued to the SEA.

Smarteye Electronic Assembly Related Documents

You should have the following documentation available when configuring this interface option:

Smarteye Electronic Assembly User Manual

This document is shipped with each SEA. It discusses the issues involved in configuring the communication and operation parameters of the SEA.

Smarteye Electronic Assembly Configuration

Configure the Smarteye Electronic Assembly with the following communication and operational parameters.

Physical Protocol	RS-232
Baud Rate	9600
Parity	None
Communication Mode	Poll, Open Loop or Handshake

See the Smarteye Electronic Assembly User Manual for details on setting these parameters

CIMPLICITY Configuration for Smarteye Electronic Assembly

CIMPLICITY Configuration for Smarteye Electronic Assembly

When you configure ports, devices, and points that use the Smarteye Electronic Assembly Communications enabler, some fields must contain unique values for the communications to work successfully. These are detailed below.

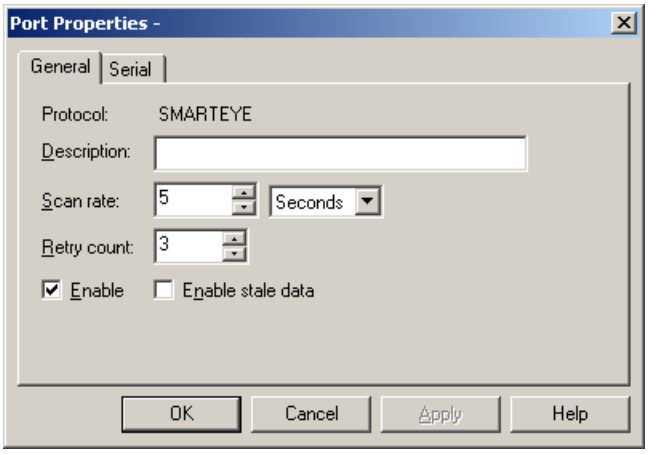
SEA Port Configuration

SEA Port Configuration

1. In the **Protocol** field, select **SMARTEYE** from the list of protocols.
2. In the **Port** field, select the communication port that will be used for Smarteye Electronic Assembly communications.

   When you click **OK** to create the port, the Port Properties dialog box opens.

General Port Properties

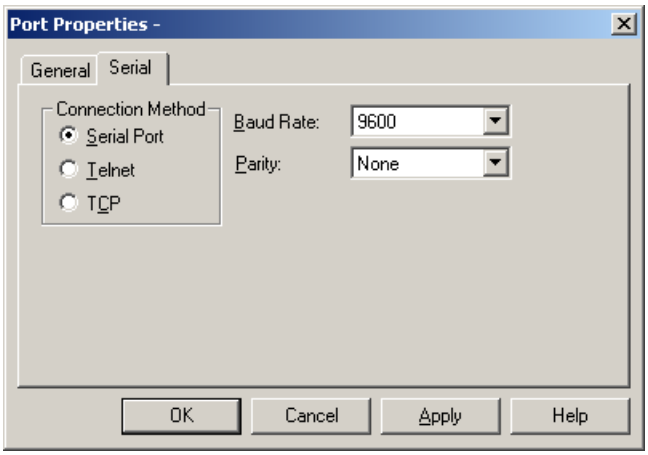


Use the General properties to enter general information for the port. You can define the following:

Description	Enter an optional description to help you identify the port.

Scan Rate	Enter the base scan rate for the port. Point scan rates are multiples of the base rate. You can specify a scan rate in **Ticks** (100 ticks = 1 second), **Seconds**, **Minutes**, or **Hours**.
Retry Count	If communications cannot be established, the device is considered to be down and a **$DEVICE_DOWN** alarm is generated. Once a device is down, periodic attempts are made to resume connections to it.
	Enter the number of scans to wait before attempting to a device on this port after a communications error is detected.
Enable	Set this check box if you want the port to be enabled when the project starts. If you clear this check box, the port will not be enabled when the project starts, and points will not be available for devices on the port.

Serial Port Properties - Serial

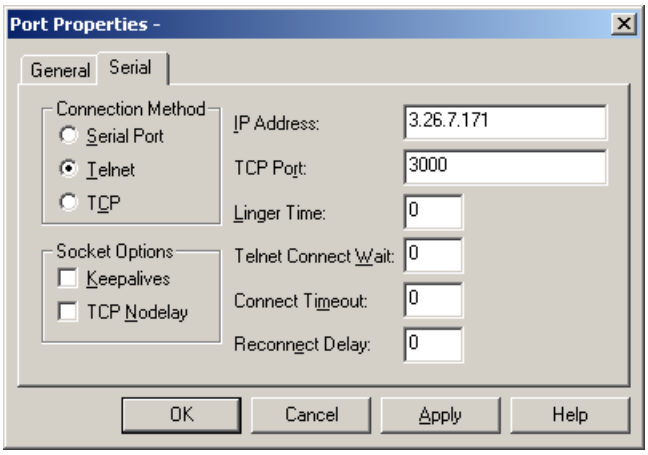


If you select the serial port connection method, you need to define the following:

Baud Rate	Enter the baud rate for communications. Click the drop-down list to the right of the input field and select a rate from the list.
Parity	Select the parity to be used for communications.

Remember that you must configure the same baud rate, data bits, parity, stop bits and flow control for all PLCs using the serial port.

Serial Port Properties - Telnet or TCP Connection



If you select the Telnet or TCP connection method, you need to define the following:

Socket Options	Set the **Keepalives** check box to use keepalives to detect the loss of the terminal server. Clear the check box if you do not want to use keepalives to detect the loss of the terminal server. Set the **TCP Nodelay** check box if you want to set the Nodelay flag on the socket. Clear the check box if you do not want to set the Nodelay flag.
IP Address	IP address of the terminal server.
TCP Port	Port number of the TCP port on the terminal server.
Linger Time	Time in seconds to wait after closing the socket before aborting the socket.
Telnet Connect Wait	Time in seconds to wait for the Telnet protocol to initialize.
Connect Timeout	Time in seconds to wait for the TCP connection to form.
Reconnect Delay	Time in seconds to wait before attempting to reconnect to a device. If you set this value to zero and the terminal server is not available, no attempts will be made to reconnect to the terminal server.

Terminal Server Setup

1. Log into your terminal server, and set privileged mode by issuing the following command. Enter the privileged password when prompted (typically "system"):

   > set priv

   Password>

2. Enter the following command so that the permanent database and the operational database will be updated when a **define** command is issued:

>> **set server change enabled**

3. For each port to which a marquee is attached, enter the following set of commands where:

   <port> is the actual port number to which the marquee is attached. Note that it is possible to specify a range of port numbers instead of a single port (for example, 1-10 signifies ports 1 through 10).

   <ip_port> is the telnet port number.

   <parity> is odd, even, or none.

   <stop bits> is the number of stop bits.

   <speed> is the baud rate of the marquee.

```
>> define port <port> access remote
>> define port <port> telnet remote <ip_port>
>> define port <port> telnet transmit immediate
>> define port <port> parity <parity>
>> define port <port> character size 8
>> define port <port> stop bits <stop bits>
>> define port <port> autobaud disabled
>> define port <port> speed <speed>
>> define port <port> line editor disabled
>> define port <port> telnet csi escape enabled
>> define port <port> telnet binary session mode passall
>> define port <port> loss notification disabled
>> define port <port> autoprompt disabled
>> define port <port> verification disabled
>> define port <port> outboundsecurity disabled
>> define port <port> broadcast disabled
>> define port <port> default session mode transparent
>> define port <port> internet tcp keepalive 2
```

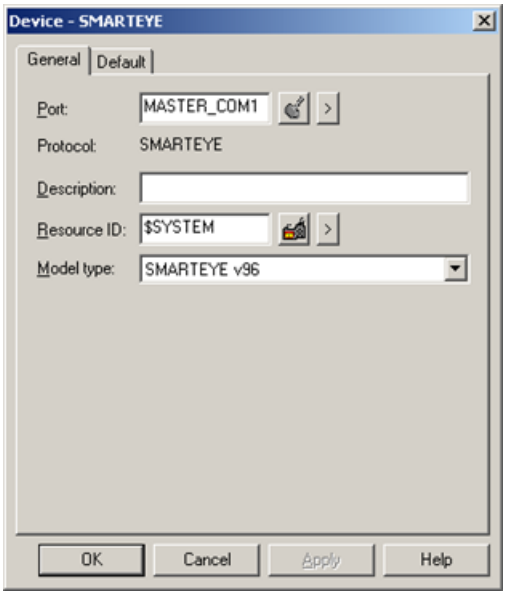
SEA Device Configuration

SEA Device Configuration

1. In the **Device** field, enter the name of the device you are configuring.
2. In the **Port** field, select the SEA port to be used by the device.

   When you click **OK** to create the port, the Device Properties dialog box opens.

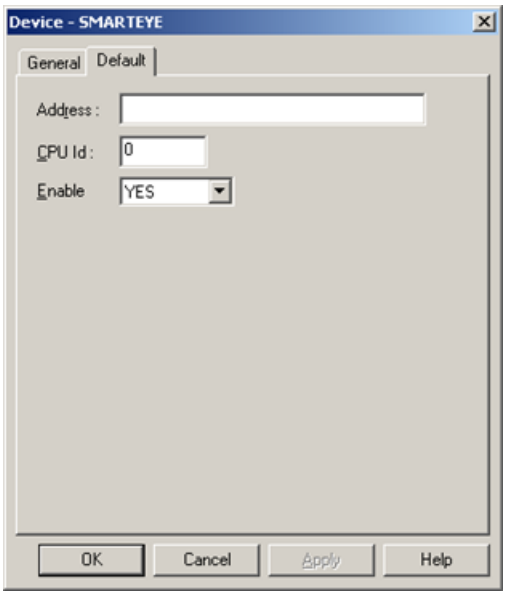
General Device Properties



Use the General tab in the Device dialog box to enter general information for the device. You can define the following:

Port	Select the port for this device. Click buttons to the Port field to select the port as follows.	
		Display the list of ports and select one.
		Create a new port, edit the current port, or select a port from the list of ports.
Description	Enter an optional description to help you identify the device.	
Resource	Enter a resource that can be associated with this device for alarm generation. Click buttons to the right of the **Resource** field to select the resource, as follows.	
		Display the list of resources and select one.
		Create a new resource, edit the current resource, or select a resource from the list of resources.
Model Type	There is only one model, **SMARTEYE v96**.	

Default Device Properties



Use the Default tab in the Device dialog box to enter information about Smarteye Electronic Assembly communications for the device. You can define the following:

Address	Enter the octal address for the device. The address is an octal value between 000 and 370 ending in 0.
CPU ID	Not used.
Enable	Select YES in this box to enable the device when the project starts. Select NO in this box to disable the device. Points associated with the device will be unavailable.

SEA Point Configuration

Smarteye Electronic Assembly Device Configuration Point Configuration

When you define a point, fields in the Point Properties dialog box values are unique to or have special meaning for Smarteye Electronic Assembly communications.

General Point Properties

On the General tab in the Point Properties dialog box, you may configure points for **Read** or **Read/ Write** access as follows:

Region	Access type
Label	Read
Error *(page 874)*	Read
Diagnostic *(page 874)*	Read

Conveyance System *(page 874)*	Read/Write
System *(page 874)*	Read/Write

Note: In the System region, only the command point is write-able.

Device Point Properties

On the Device tab in the Point Properties dialog box:

Update Criteria	The update criteria you use determines how the data will be requested.
	Enter **On Scan** or **On Change** for points whose values should be polled by the Smarteye Electronic Assembly driver at regular intervals.
	Enter **On Demand On Scan** or **On Demand On Change** for points whose values should be polled by the Smarteye Electronic Assembly driver at regular intervals while users are viewing them. >
	Enter **Poll Once** for conveyance system IDs and command points.
	Enter **Unsolicited** for all other points.
Address	Point address requirements are region-dependent. Valid regions are: Labels *(page 882)*Errors *(page 883)*Diagnostics *(page 884)*Conveyance System IDs *(page 885)*System data *(page 884)*
Scan Rate	For poll points, set this to some low number as required.
	For all other points, this field is ignored.
Poll After Set	For conveyance system IDs and command points, set this check box

Point Address Formats

Point Address Formats

The Smarteye Electronic Assembly supports the following address formats:

- Labels
- Errors
- Diagnostics
- Conveyance System IDs
- System Data

Labels

The address format for Labels is:

```
L<READER>.<FORMAT>
```

where:

<READER> is the reader number from 0 to 7.

<FORMAT> is one of the following 13 string format specifiers. The string format specifiers indicate the justification of the label value (Left or Right), fill character (Blank or Zero), and the position of the conveyance system ID (None, Prefix, or Suffix).

Format	Data Type	Elements	Example
`NUM`	INT	1	1234
`LBN`	STRING	5	"1234 "
`LBP`	STRING	9	"ABCD1234 "
`LBS`	STRING	9	"1234ABCD "
`LZN`	STRING	5	"1234 "
`LZP`	STRING	9	"ABCD1234 "
`LZS`	STRING	9	"1234ABCD "
`RBN`	STRING	5	" 1234"
`RBP`	STRING	9	"ABCD 1234"
`RBS`	STRING	9	" 1234ABCD"
`RZN`	STRING	5	"01234"
`RZP`	STRING	9	"ABCD01234"
`RZS`	STRING	9	"01234ABCD"

For example, `L0.NUM` is the label number for reader 0.

Note: If conveyance system IDs are less than four characters, string values are padded on the right with blanks to fill the required number of spaces.

Errors

The address format for Errors is:

```
E<READER>.<FORMAT>
```

Where:

<READER> is the reader number from 0 to 7

<FORMAT> is one of two format specifiers in the following table:

Format	Data Type	Elements
`NUM`	INT	1
`STR`	STRING	80

Note: See Smarteye Reader Errors *(page 889)* for details on interpreting error codes and configuring alarms for reader errors.

Diagnostics

The address format for Diagnostics is:

```
D<READER>
```

Where <READER> is the reader number from 0 to 7.

Diagnostic strings are 28-element STRING points.

System Data

The address format for System Data is:

```
S<LOCATION>
```

where <LOCATION> is 0 or 1 to select the Poll or Command point.

0 to select the Poll point

1 to select the Command point

Note: Exactly one poll point must be configured for each system device.

Additional Poll Point Configuration

You must also configure the following for a Poll point:

Data Type is **INT**.

Elements is **1**.

Additional Command Point Configuration

You must also configure the following for a Command point:

Data type is **STRING**.

Elements is **2**.

Conveyance System IDs

The Conveyance System ID is a string of up to four characters, which is appended to label values for certain label formats. Each label reader has a Conveyance System ID associated with it.

The device communication enabler initializes Conveyance System IDs to an empty string when it starts up and accepts setpoint requests to assign them at run time.

One mechanism for these setpoints is to use the Basic Control Engine, create events to check for the IDs being blank, and associated actions to set them to the desired value. In this way, each time the enabler restarts, the desired Conveyance System IDs are initialized.

Command Points

Commands may be sent to the Smarteye Electronic Assembly via command points. A command point is configured as a 2-character string point at address S1.

The first character of the command point is the command code. The second character specifies the reader number for certain commands.

The following commands are directed at the SEA and do not require a reader ID.

Code	Command Description
R	Restart. Initialize all readers on-line.
B	Box poll. Request message from any reader in the SEA.
A	Ack. Acknowledge the previous transmission.
N	Nak. Negatively acknowledge the previous transmission.

The communication enabler uses these commands to interact with the SEA. Generally, they should not be used interactively.

The following commands are directed at specific readers and do require a reader ID.

Code	Command Description
I	Initialize the reader on-line

S	Status inquiry regarding reader
C	Create a diagnostic message for the next label at the reader
L	Lock in diagnostic for all labels at the reader
U	Unlock diagnostic at reader
F	Take reader off-line

For example, sending **"C2"** to the command point of an SEA causes the value for the next label to pass reader two to be followed by a diagnostic message.

Advanced Configuration Requirements

Advanced Configuration Requirements

Advanced Configuration Requirements

The operation of the Smarteye Electronic Assembly Communication enabler can be controlled through global parameters. This section discusses the global parameters used by the enabler and their settings.

`<port>_LOG_WARNING`

By default, all Warning messages are logged to the project's Status Log file.

To eliminate or redirect Warning messages, add the following line to your global parameters file:

```
<port>_LOG_WARNING|1|<log_file_location>
```

where:

<port> is the name of the port used by the enabler.

<log_file_location> identifies the file to which the Warning messages will be redirected. You can enter one of the following in this field:

- COR_STATUS-Warning messages are logged to the projects Status Log (this is the default).
- STD_OUT-Warning messages are logged to the processes standard output file
- NONE-Warning messages are not logged.

- <port>_MODE
- <port>_POLL_LIMIT
- <port>_RESTART_SEA
- SEA_HANDSHAKE_TIMEOUT

- SE_LABEL_LEN

<PORT>_MODE

Ordinarily, the enabler polls each configured device at intervals determined by the port base scan and the device's poll point scan rate multiplier. This is Polled mode.

The enabler may also be configured for Listen Only or Handshake mode.

In Listen Only mode, the enabler listens for messages from SEAs but does not poll for data or process command points. This is useful if the SEA is configured for Open Loop mode or if another system is polling the SEAs during an initial installation or changeover between systems.

In Handshake mode, the enabler listens for and acknowledges messages and processes command points, but does not poll for data. This may reduce communication and processing overhead.

To configure the mode that the enabler operates in, add the following line to your global parameters file:

<PORT>_MODE|1|<mode>

where <port> is the name of the port used by the enabler, (for example, **COM1**) and <mode> is **LISTEN** or **HANDSHAKE**.

<port>_POLL_LIMIT

Each poll cycle, the enabler will poll each device for up to eight new messages. This is the poll limit. In some circumstances, performance may be improved by adjusting this poll limit.

To set the poll limit, add the following line to your global parameters file:

<PORT>_POLL_LIMIT|3|<limit>

where <port> is the name of the port used by the enabler and <limit> is the number of messages you want processed each poll cycle.

<port>_RESTART_SEA

Each time the enabler starts up in Polled or Handshake mode, and each time there is a communication error in Polled mode, the enabler sends a RESTART command to the Smarteye Electronic Assembly (SEA). This RESTART command flushes any buffered label messages in the SEA.

To preserve messages for the labels that are read while the enabler is not running, add the following line to your global parameters file:

<PORT>_RESTART_SEA|1|FALSE

where <port> is the name of the port used by the enabler.

To restore the default behavior of restarting the SEA when the enabler starts or detects an error, change the FALSE to TRUE or remove the line from the global parameters file.

<port>_SEA_HANDSHAKE_TIMEOUT/ SEA_HANDSHAKE_TIMEOUT

When the SmartEye device is configured within CIMPLICITY to be in handshake mode, the device communication interface will query the device for status if no data has been received from the device within the configured interval.

If the device does not respond to the status, the SmartEye device will be marked down.

Global parameters to change the default (100 seconds) are as follows.

SEA_HANDSHAKE_TIMEOUT

For	Project
Purpose	To specify the number of seconds of silence (on a project basis) that should pass before querying for status of the device to determine device availability.
	If a port and project global are defined, the port level defined global will take precedence.
Value	Enter a value that is equal to the number of seconds of silence that should pass before querying for status of the device to determine device availability.
Default Value	100 seconds (of silence).

<PORT>_SEA_HANDSHAKE_TIMEOUT

For	System
Purpose	To specify the number of seconds of silence (on a port basis) that should pass before querying for status of the device to determine device availability.
	If a port and project global are defined, the port level defined global will take precedence.
Value	Enter a value that is equal to the number of seconds of silence that should pass before querying for status of the device to determine device availability.

Default Value	100 seconds (of silence).

SE_LABEL_LEN

The default label length is 5 characters. Labels are zero-filled or blank-filled to this length.

If all the labels in your project are all the same length, you can use this global parameter to set the label length and eliminate zero and blank filling of labels. To configure a new label length for your project, add the following line to your global parameters file:

SE_LABEL_LEN|1|<label_length>

where <label_length> is 3, 4 or 5. If a label is read that has a length greater than this, a FAILURE error message will be logged to the Status Log.

Smarteye Reader Errors

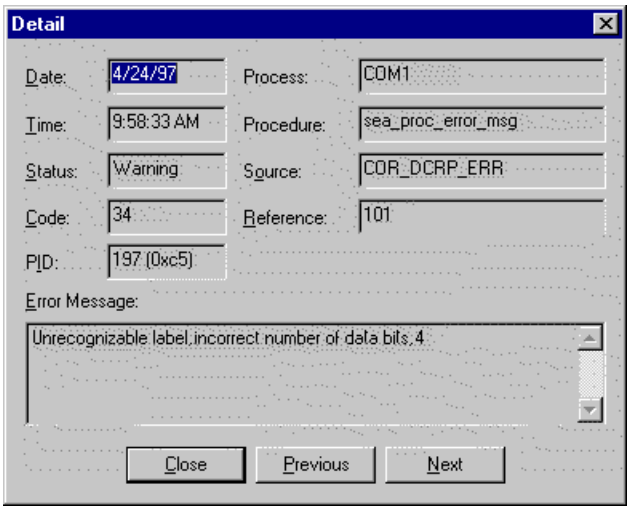
Smarteye Reader Errors

By default, all reader errors are logged in the Status Log. In addition, reader errors may be monitored in via CIMPLICITY points and alarms:

- You may configure points to hold the error code and error message for each reader. These points may be displayed on a CimView screen, logged using the data logger, etc.
- You may configure alarms for error code points that will be displayed in the Alarm Viewer.

Status Log Entries

A typical Status Log entry looks like:

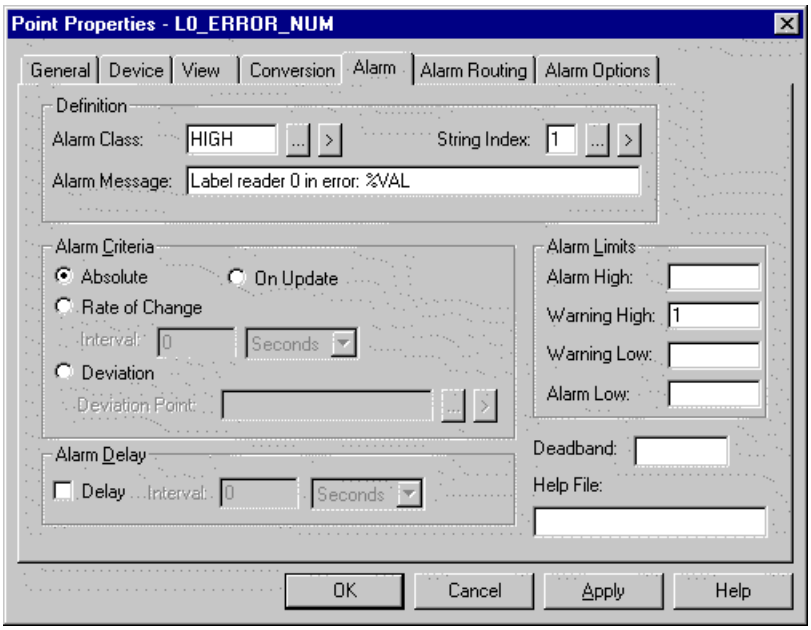


The following fields have special meaning for Smarteye reader errors

Process	The port used to communicate with the reader
Code	The error code returned by the Smarteye Electronic Assembly. Refer to the Smarteye Electronic Assembly User Manual for help in interpreting this code.
	This same code appears in a **NUM** format error point configured for the reader.
Reference	The octal address of the Smarteye reader. The first two digits indicate the Smarteye Electronic Assembly, and the last digit is the reader number on that SEA. For example, a reference of 245 would indicate an error on reader 5 of the SEA at address 240.
Error Message	This is the message that appears in a **STR** format error point configured for the reader.

Configuring Reader Error Alarms

If you configure numeric points to hold reader error codes, you may wish to configure alarms on those points to generate entries in the Alarm Viewer when a reader error occurs. The following is an example of such a configuration:


Point Properties - LO_ERROR_NUM
General
Device
View
Conversion
Alarm
Alarm Routing
Alarm Options
Definition
Alarm Class:
HIGH
String Index:
1
Alarm Message:
Label reader 0 in error: %VAL
Alarm Criteria
Absolute
On Update
Rate of Change
Interval:
0
Seconds
Deviation
Deviation Point:
Alarm Delay
Delay
Interval:
0
Seconds
Alarm Limits
Alarm High:
Warning High:
1
Warning Low:
Alarm Low:
Deadband:
Help File:
OK
Cancel
Apply
Help

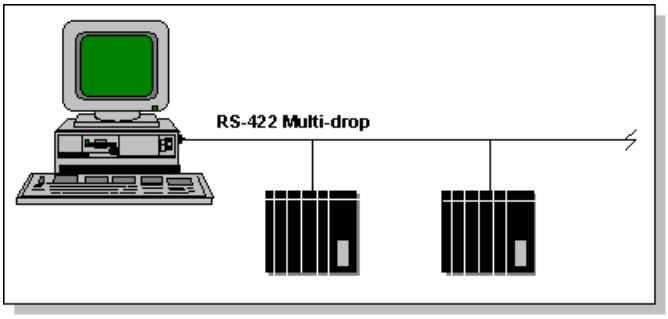
Chapter 54. SNP and SNPX Communications

About SNP and SNPX Communications

The SNP Communications enabler uses the Series Ninety Protocol (SNP) to communicate with Series 90 controllers over a serial line.

The SNPX Communications enabler uses the SNPX protocol to communicate with Series 90-30 controllers over a serial line.

SNP and SNPX are half-duplex protocols that uses the RS-485 (enhanced version of RS-422) and RS-232 electrical interfaces. These protocols permit communication between one server and one or more clients. Communication between CIMPLICITY software and Series 90 controllers may be set up as a point-to-point connection or a multi-drop configuration. Connections may be established on the RS-485 CPU port of the Series 90 or via the CMM card.



Both enablers support the following CIMPLICITY features:

- Polled reads at user defined rates
- Poll after setpoint
- Triggered reads
- Analog deadband
- Alarm on communications failure
- Server Redundancy configurations

Both enablers support the following data types:

- Signed 8, 16, and 32 bit integer
- Unsigned 8, 16, and 32 bit integer
- Floating point
- Text
- Arrays

SNP and SNPX Performance Comparison

For point-to-point communication, the SNPX protocol is 5 to 10 percent faster than SNP protocol if the average packet size is greater than 100 bytes. The difference becomes smaller when the average packet size get bigger.

The SNPX protocol is 5 or even 10 times faster than the SNP protocol in multi-drop communications.

- In multi-drop communications, the SNPX protocol supports broadcast-attach which makes the SNPX device communications option attach to all the PLCs at the same time. After attaching to all the PLCs, the SNPX device communications option can send a request to any PLC in the network without re-attaching to it.
- Currently, it takes approximately 0.6 second for the SNP communications option to attach to a PLC each time the communications option sends a data request.

Supported SNP Devices

The SNP Communications enabler supports reading and writing point data to the following Series 90 controllers:

- PACSystems RX3i
- PACSystems RX7i
- Series 90-30 / 90 Micro
- Series 90-70
- VersaMax

If you are using a Series S90-20 Controller, select device model Series 90-30 / 90 Micro.

Supported SNPX Devices

The SNPX Communications enabler supports reading and writing point data to Series 90-30 and VersaMax controllers.

Supported SNP and SNPX Memory Types

If you add a block to the programmable controller or resize domains, you must stop and restart the SNP Communications Enabler in order for it to see the changes.

Data may be read from the following memory types:

%AI	Analog Inputs	**%R**	Registers
%AQ	Analog Outputs	**%S**	Discretes
%G	Genius Global Data	**%SA**	Discretes
%I	Discrete Inputs	**%SB**	Discretes
%M	Discrete Internals	**%SC**	Discretes
%Q	Discrete Outputs	**%T**	Discrete Temporaries

Data may be written to the following memory types:

%AI	Analog Inputs	**%M**	Discrete Internals
%AQ	Analog Outputs	**%Q**	Discrete Outputs
%G	Genius Global Data	**%R**	Registers
%I	Discrete Inputs	**%T**	Discrete Temporaries

CAUTION: Do not write data into the system registers (%S, %SA, %SB, and % SC). Doing so may interfere with the normal operation of the programmable controller.

SNP Hardware Configuration Requirements

Original CMM cards did not support the SNP protocol. Other modules in the Series 90 CPU rack may need to be upgraded when upgrading the CPU firmware.

PACSystems Family
Series 90-30
Series 90-70 family
Series 90 Micro PLCs
VersaMax

PACSystems Family

It is recommended that the firmware revision for both RX7i and RX3i be at V2.0 or higher.

Machine Edition v5.0 or higher is required as the programming software.

Series 90-30

It is recommended that the firmware revision for all CPU models be at V6.5 or higher.

The versions for Series 90-30, with CMM communication, use IC693CMM311 with a Series 90-30 Model 331 (CPU IC693CPU331) or higher model number.

Machine Edition is required as the programming software. Version 3.50 or higher version of Logicmaster 90-30 Software Package Programmer and Configurator. This is optional if a Hand-Held Programmer for Series 90-30 and 90-20 Programmable Controllers is available. The Hand-Held Programmer may be used to configure the CMM. VersaPro 2.02 or higher may also be used.

Series 90-70 family

It is recommended that firmware revision for all CPU models be at V4.12 or higher.

Machine Edition is required as the programming software. Version 7.0 or higher of the Logicmaster 90-70 Software Package Programmer and Configurator is required to configure a CMM card.

Series 90 Micro PLCs

All CPU models require V2.1 or higher firmware.

The following PLCs have been qualified for use with the SNP Communications option.

CPU	**IC693UDR001HP1**
	IC693UDR001GP1

VersaMax

All models with a serial interface supporting SNP.

Version 2.02 or higher of VersaPro can be used to configure the serial ports.

SNPX Hardware Configuration Requirements

SNPX supports communications with Series 90-30 controllers that support the SNPX protocol on one or more serial ports. These include Series 90-30 Models 301, 311, 313, 323, 331, 341, 35x Series, and 36x Series CPUs and with a CPU firmware revision of 6.5 or higher. VersaMax CPUs with serial ports that can be configured for use with the SNPX protocol are also supported.

Machine Edition is required as the programming software. Version 3.50 or higher version of Logicmaster 90-30 Software Package Programmer and Configurator. This is optional if a Hand-

Held Programmer for Series 90-30 and 90-20 Programmable Controllers is available. The Hand-Held Programmer may be used to configure the CMM. VersaPro V2.02 and higher may also be used.

Required SNP and SNPX Documents

You should have one or more of the following documents available when configuring your SNP or SNPX communications network:

PACSystems CPU Reference Manual (GFK 2222 — for both RX3i and RX7i)

PACSystems RX3i System Manual (GFK 2314)

PACSystems RX7i Installation Manual (GFK 2223)

Series 90-20 Programmable Controller User's Manual (GFK 0501)

Series 90-30 Programmable Controller Installation Operator's and User's Manual (GFK 0356)

Series 90-70 Programmable Controller Installation Manual (GFK 0262)

If you are using a CMM card, you should also have the following manual available:

Series 90 Programmable Controller Serial Communications User's Maual (GFK 0582)

If you are using a VersaMax processor, you should have the following manual available:

VersaMax PLC User’s Manual (GFK-1503)

If you are using a VersaMax Nano or Micro

VersaMax Micro and Nano PLC User’s Manual (GFK-1645)

SNP and SNPX Hardware Installation (External)

SNP and SNPX Hardware Installation (External)

The CIMPLICITY SNP and SNPX Communications enablers support point-to-point and multi-drop configurations of Series 90 controllers.

When you configure the SNP port, you will need to know the baud rate and parity being used by the Series 90 programmable controllers. In addition, CIMPLICITY software uses 8-bit characters with 1 stop bit for communicating with Series 90 controllers.

If a CMM card is used, it may be desirable to configure an SNP ID different from that of the attached port. The Series 90 Programmable Controller Serial Communications User's Manual (GFK-0582) describes the ladder logic required to change the SNP ID. If this is done, be sure that the Series 90 is in a RUN state, and the SNPIO ladder logic has successfully completed before attempting to communicate with the device via CIMPLICITY software or device communications will fail.

In general, a baud rate of 9600 or less is recommended. If both ports on the CMM module are in use, the baud rate must be 9600 or less.

CAUTION: Serial ports on the Series 90 family are not isolated. Potential differences above 7 volts will cause damage. Isolators should be employed on long distance runs to guard against equipment damage.

The Installation manuals shipped with the Series 90 controller provide information under Cabling for connecting to the attached port of Series 90 controllers. The Serial Communications manual provides information for connecting to the CMM port available for some Series 90s.

When you refer to this documentation, use the following guidelines:

- Computers with 9-pin serial connectors should follow the cabling diagrams for IBM-AT compatible Personal Computers.

On the following pages are several sample configurations with cabling directions. A more comprehensive description is available in the Installation manuals listed above. Refer to these manuals for cable and connector information as well as information regarding the converters.

Cabling Diagrams for Point-to-Point Configurations

Cabling Diagrams for Point-to-Point Configurations

The following diagrams illustrate point-to-point configurations using the RS-232/RS-422 Converter (IC690ACC900) or the RS-422/RS-232 Isolated Repeater/Converter (IC655CCM590):

Diagram	Converter	Connected to:	Cables
A *(page 898)*	**IC690ACC900**	Series 90 RS-485 CPU port	A *(page 905)* or B *(page 906)*, C *(page 906)*
B *(page 898)*	**IC690ACC900**	Series 90 RS-485 CPU port	A *(page 905)* or B *(page 906)*, D *(page 906)*
C *(page 898)*	**IC690ACC900**	Series 90 RS-485 CPU port	A *(page 905)* or B *(page 906)*, E *(page 907)*

D *(page 899)*	**IC690ACC900**	Series 90 RS-485 CPU port	A *(page 905)* or B *(page 906)*, F *(page 908)*
E *(page 899)*	**IC690ACC900**	Series 90 RS-485 CMM port	A *(page 905)* or B *(page 906)*, G *(page 908)*
F *(page 899)*	**IC655CCM590**	Series 90 RS-485 CPU port	A *(page 905)* or B *(page 906)*, H *(page 909)*
G *(page 900)*	**IC655CCM590**	Series 90 RS-485 CPU port	A *(page 905)* or B *(page 906)*, I *(page 910)*
H *(page 900)*	**IC655CCM590**	Series 90 RS-485 CMM port	A *(page 905)* or B *(page 906)*, J *(page 910)*

Diagram A

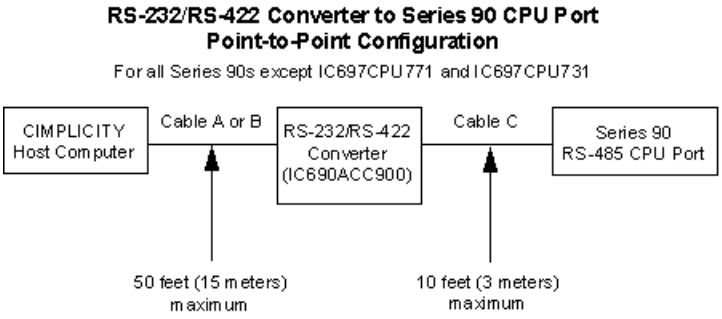


Diagram B

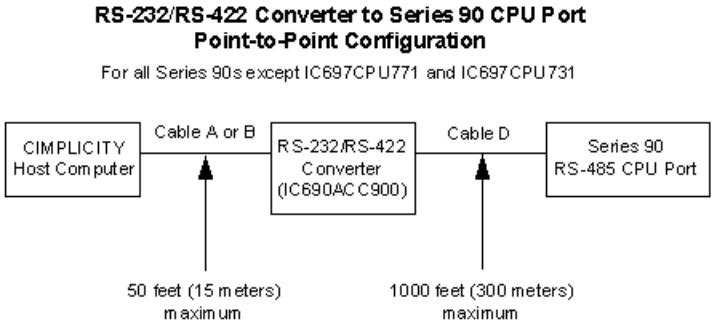


Diagram C

RS-232/RS-422 Converter to Series 90 CPU Port
Point-to-Point Configuration

For Series 90 IC697CPU771 and IC697CPU731 only

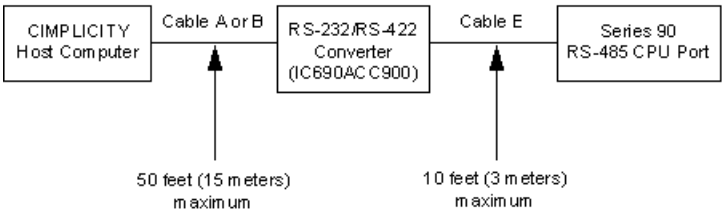


Diagram D

RS-232/RS-422 Converter to Series 90 CPU Port
Point-to-Point Configuration

For Series 90 IC697CPU771 and IC697CPU731 only

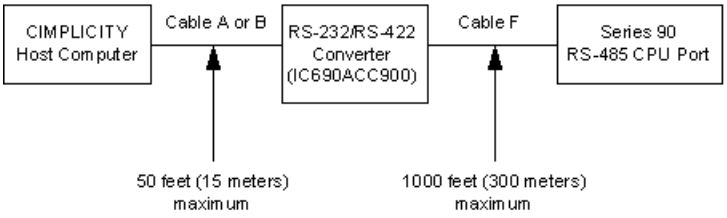
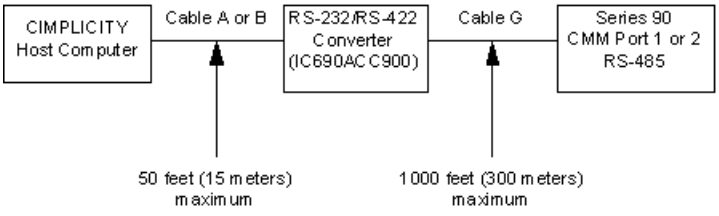


Diagram E

RS-232/RS-422 Converter to Series 90 CMM Port
Point-to-Point Configuration



Note: on the Series 90-30 CMM, only Port 2 supports RS-422/RS-485

Diagram F

RS-232/RS-422 Isolated Converter to Series 90 CPU Port
Point-to-Point Configuration
For all Series 90s except IC697CPU771 and IC697CPU731

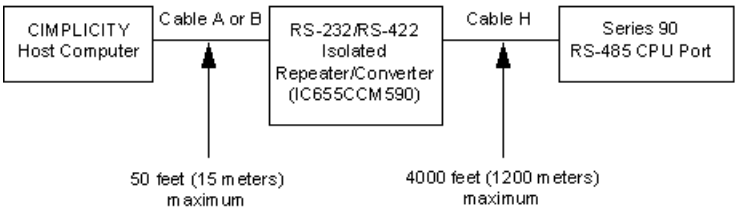


Diagram G

RS-232/RS-422 Isolated Converter to Series 90 CPU Port
Point-to-Point Configuration
For IC697CPU771 and IC697CPU731 only

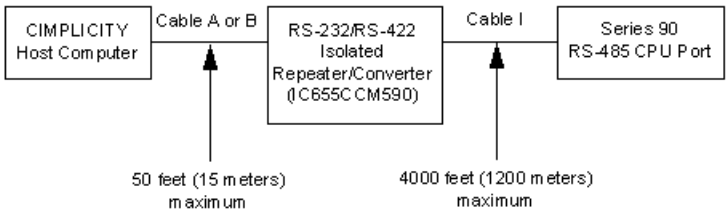
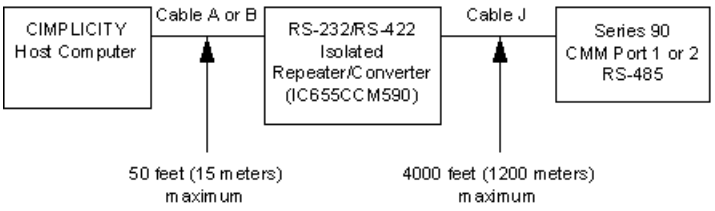


Diagram H

RS-232/RS-422 Isolated Converter to Series 90 CMM Port
Point-to-Point Configuration



Note: on the Series 90-30 CMM, only Port 2 supports RS-422/RS-485

Cabling Diagrams for Multi-drop Configurations

Cabling Diagrams for Multi-drop Configurations

The following diagrams illustrate multi-drop configurations using the RS-232/RS-422 Converter (IC690ACC900) or the RS-422/RS-232 Isolated Repeater/Converter (IC655CCM590)

Diagram	Converter	Last Station:	Cables

I *(page 901)*	**IC690ACC900**	Series 90 RS-485 CPU port (not IC697CPU771 or IC697CPU731)	A *(page 905)* or B *(page 906)*, K *(page 911)*
J *(page 901)*	**IC690ACC900**	Series 90 RS-485 CPU port (IC697CPU771 or IC697CPU731	A *(page 905)* or B *(page 906)*, L *(page 911)*
K *(page 902)*	**IC690ACC900**	Series 90 RS-485 CMM port	A *(page 905)* or B *(page 906)*, M *(page 912)*
L *(page 902)*	**IC655CCM590**	Series 90 RS-485 CPU port (not IC697CPU771 or IC697CPU731)	A *(page 905)* or B *(page 906)*, N *(page 913)*
M *(page 903)*	**IC655CCM590**	Series 90 RS-485 CPU port (IC697CPU771 or IC697CPU731)	A *(page 905)* or B *(page 906)*, O *(page 914)*
N *(page 903)*	**IC655CCM590**	Series 90 RS-485 CMM port	A *(page 905)* or B *(page 906)*, P *(page 915)*

Diagram I

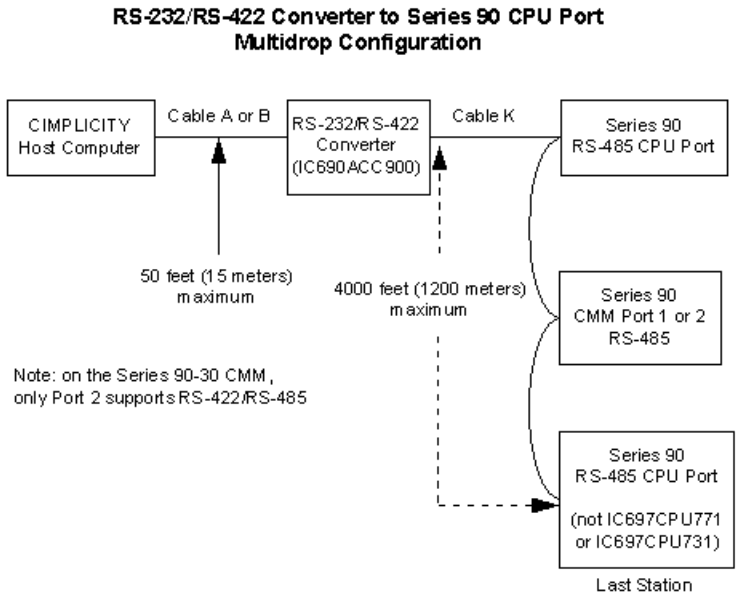


Diagram J

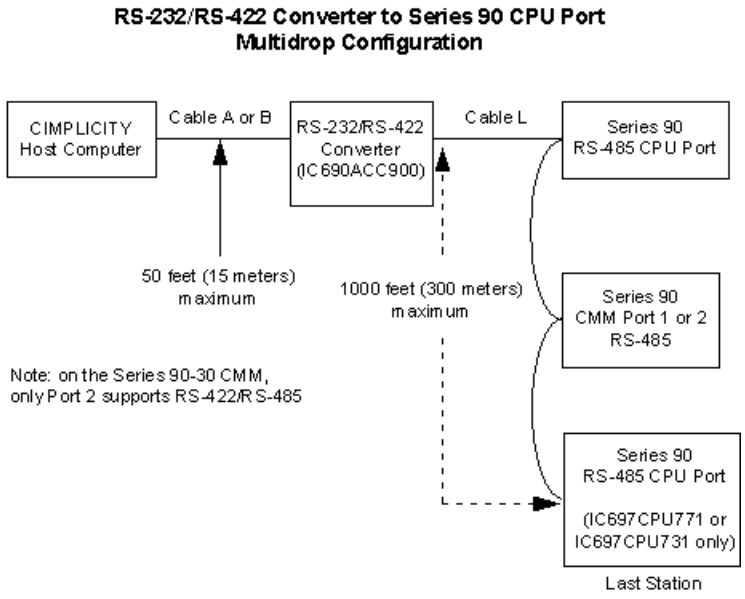


Diagram K

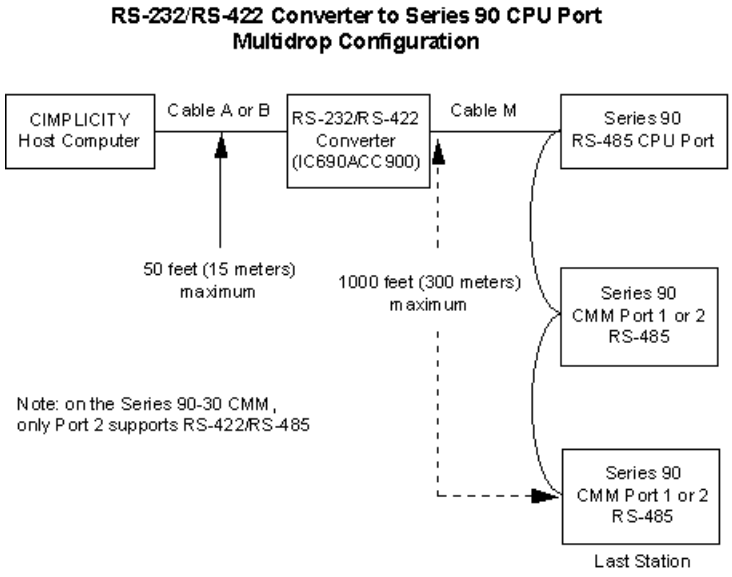


Diagram L

RS-422/RS-232 Isolated Repeater/Converter to Series 90 PLC Ports Multidrop Configuration

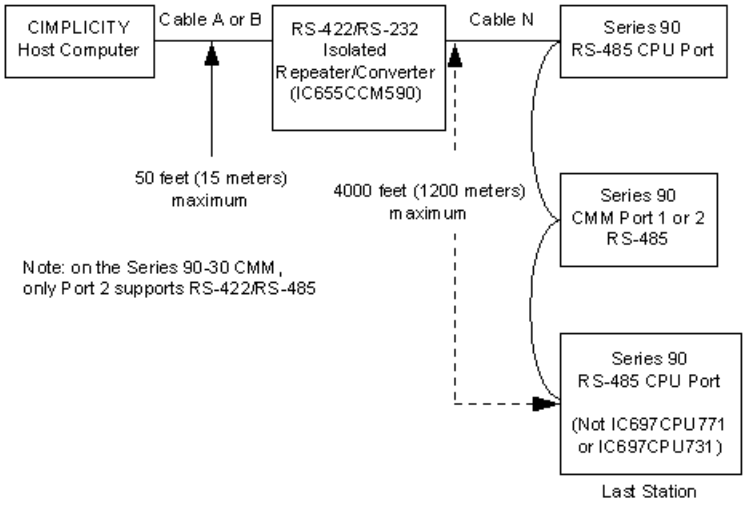


Diagram M

RS-422/RS-232 Isolated Repeater/Converter to Series 90 PLC Ports Multidrop Configuration

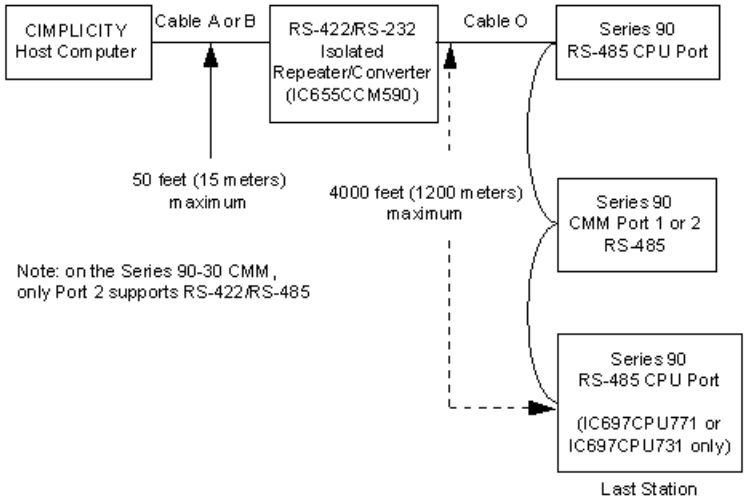
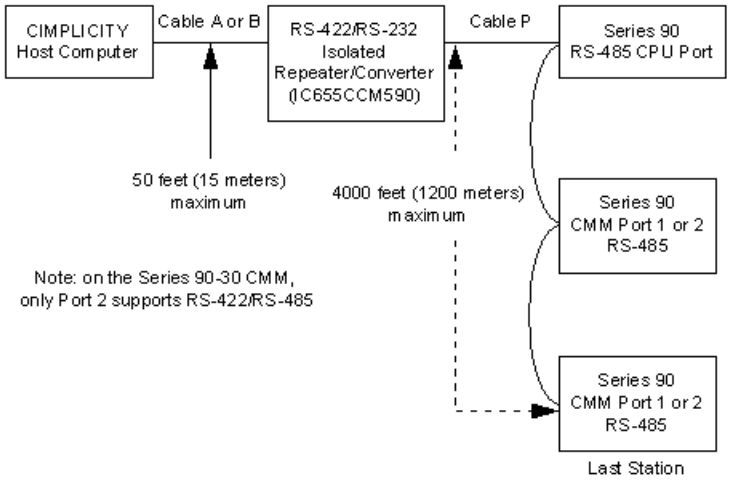


Diagram N

RS-422/RS-232 Isolated Repeater/Converter to Series 90 PLC Ports Multidrop Configuration



Cable Configurations

Cable Configurations

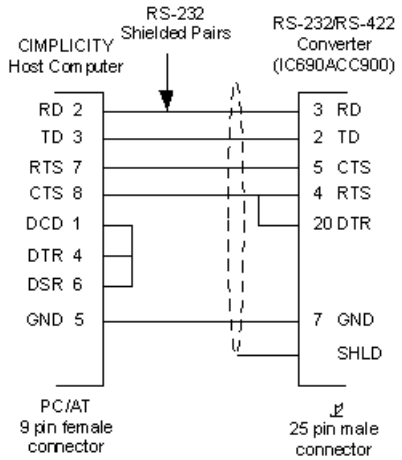
The drawings on the next pages detail the cable configuration for cables A through P in the diagrams.

Cable

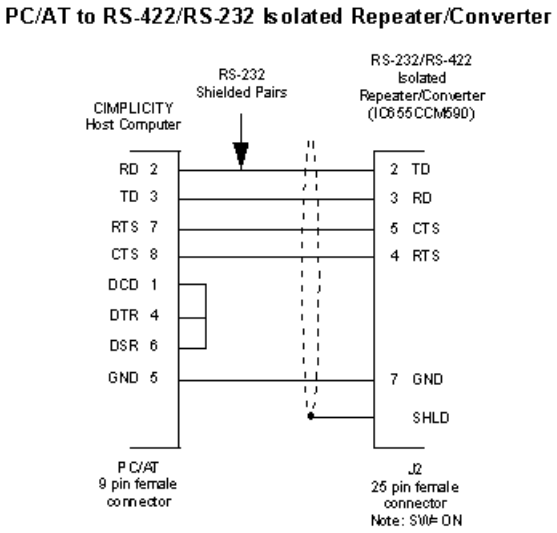
A *(page 905)*
B *(page 906)*
C *(page 906)*
D *(page 906)*
E *(page 907)*
F *(page 908)*

Cable A

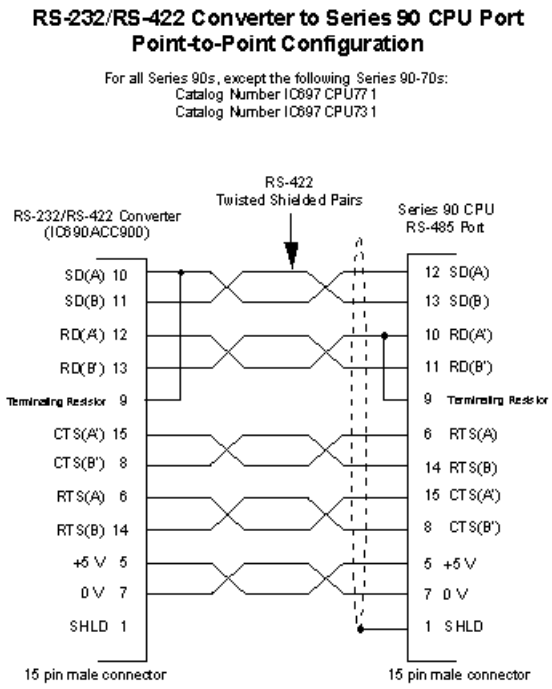
PC/AT to RS-232 Converter



Cable B



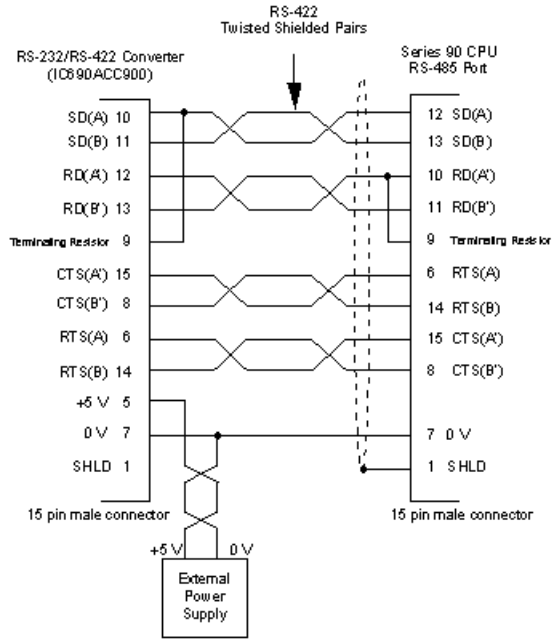
Cable C



Cable D

RS-232/RS-422 Converter to Series 90 CPU Port
Point-to-Point Configuration

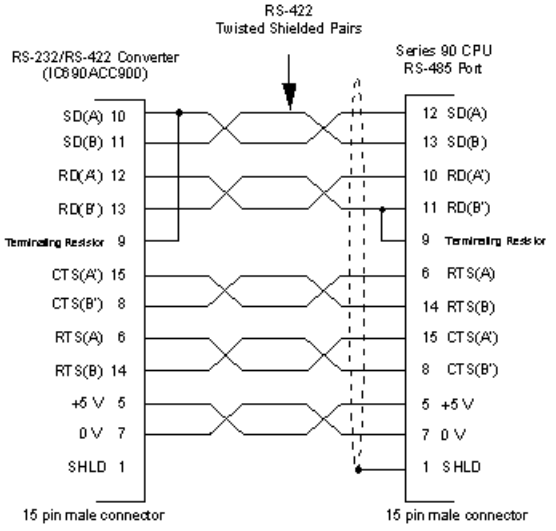
For all Series 90s, except the following Series 90-70s:
Catalog Number IC697CPU771
Catalog Number IC697CPU731



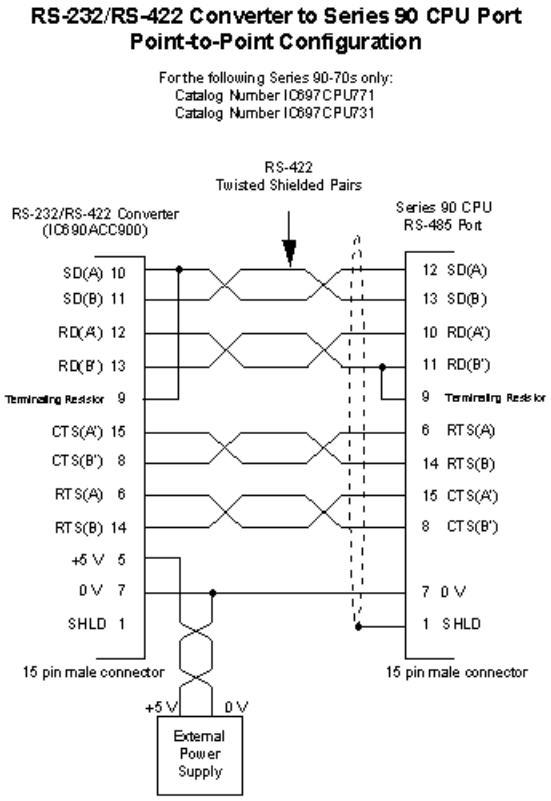
Cable E

RS-232/RS-422 Converter to Series 90 CPU Port
Point-to-Point Configuration

For the following Series 90-70s only:
Catalog Number IC697CPU771
Catalog Number IC697CPU731



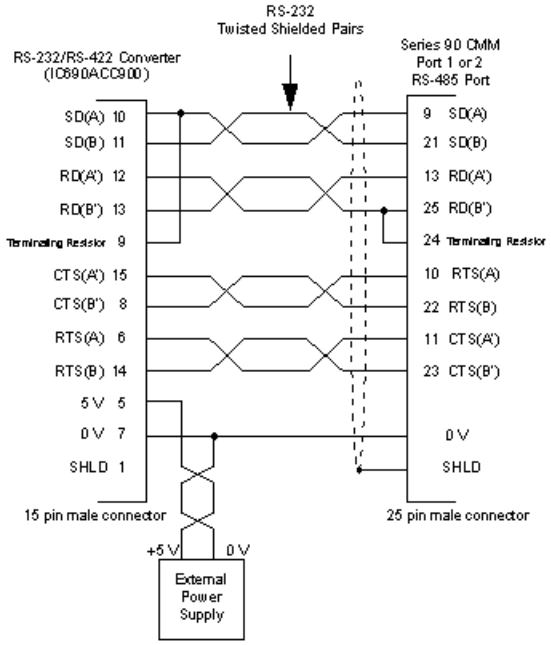
Cable F



Cable G

RS-232/RS-422 Converter to Series 90 CMM Point-to-Point Configuration

For the following Series 90-70s only:
Catalog Number IC697CPU771
Catalog Number IC697CPU731

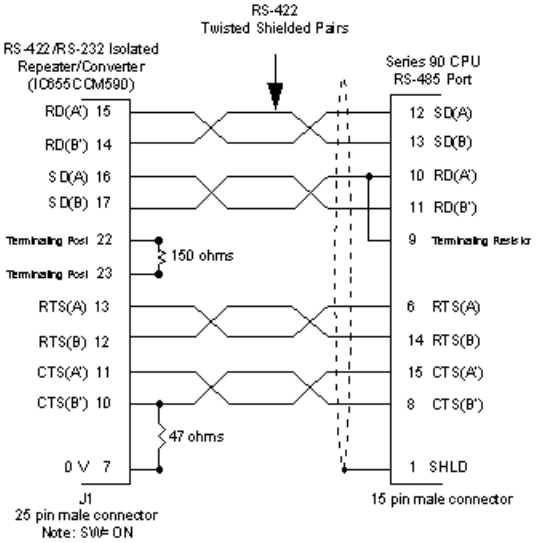


Note: On the Series 90-30 CMM, only Port 2 supports RS-422/RS-485

Cable H

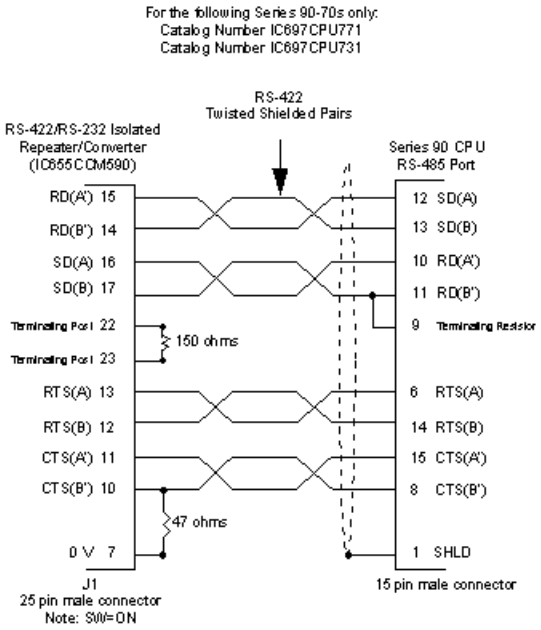
RS-422/RS-232 Isolated Repeater/Converter to Series 90 CPU Port Point-to-Point Configuration

For all Series 90s, except the following Series 90-70s:
Catalog Number IC697CPU771
Catalog Number IC697CPU731



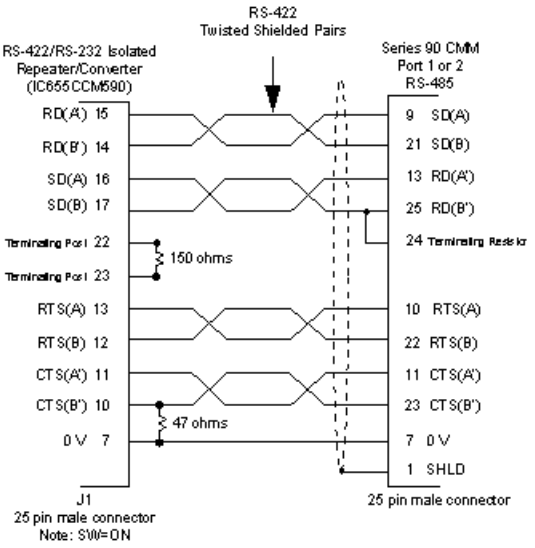
Cable I

RS-422/RS-232 Isolated Repeater/Converter to Series 90 CPU Port
Point-to-Point Configuration



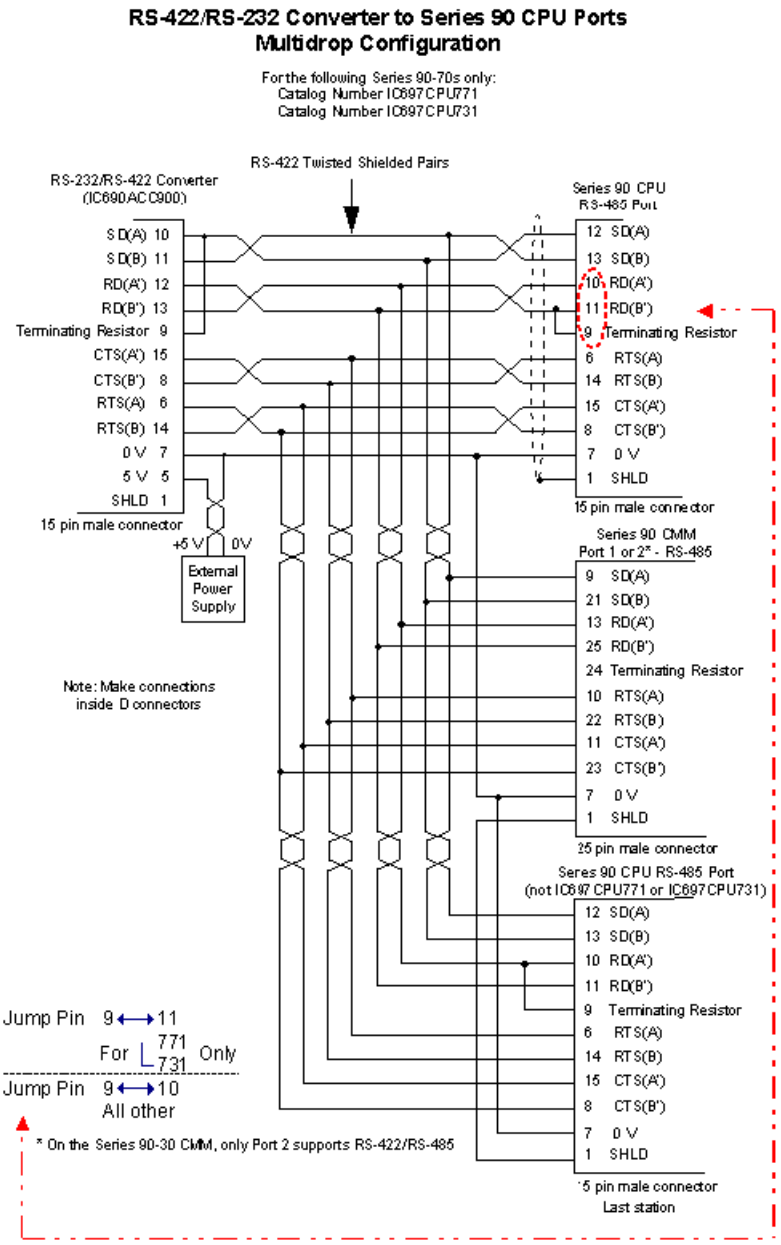
Cable J

RS-422/RS-232 Isolated Repeater/Converter to CMM Port
Point-to-Point Configuration



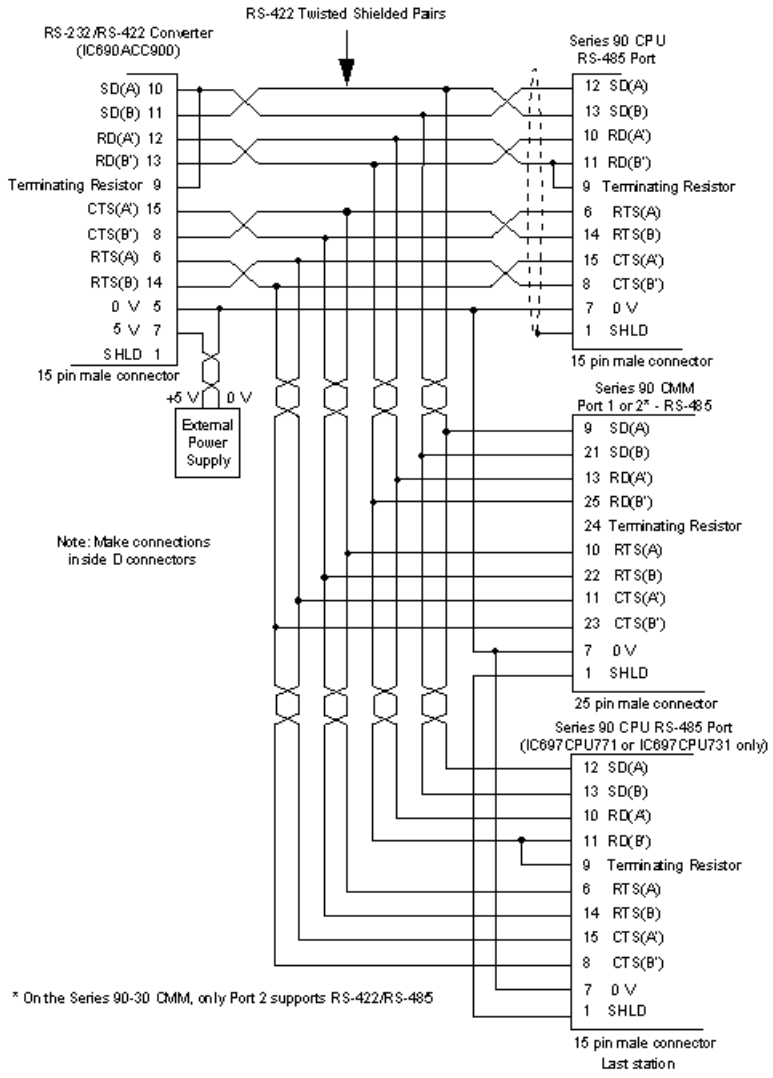
Note: on the Series 90-30 CMM, only Port 2 supports RS-422/RS-485

Cable K



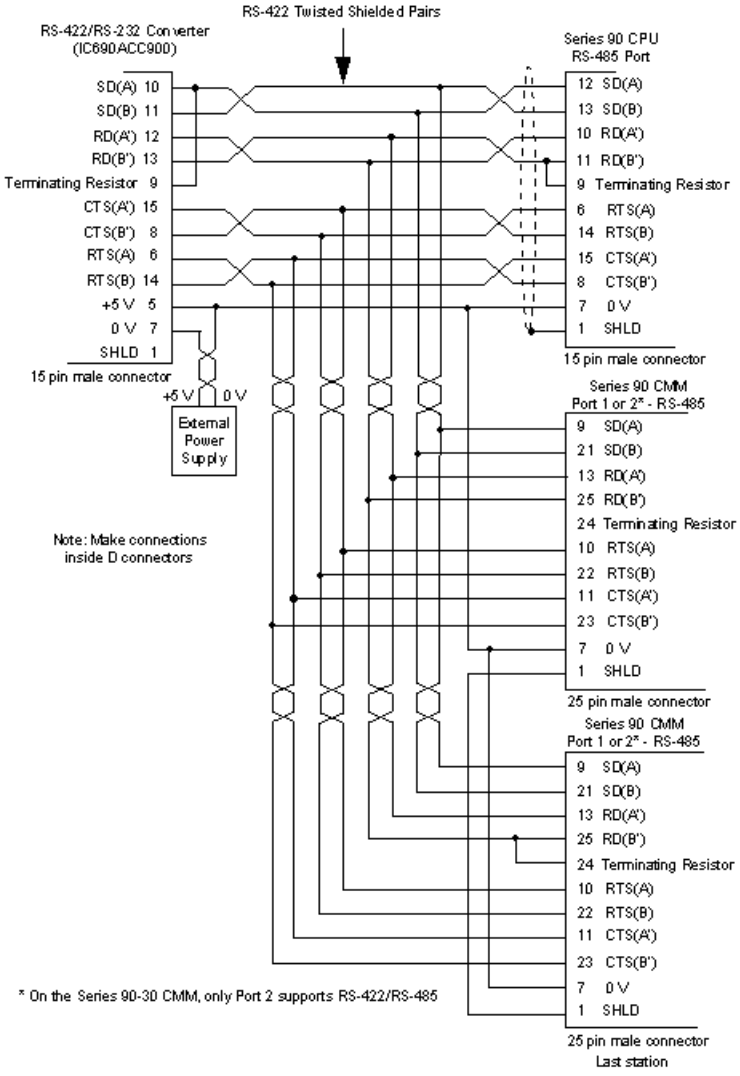
Cable L

RS-422/RS-232 Converter to Series 90 CPU Ports
Multidrop Configuration



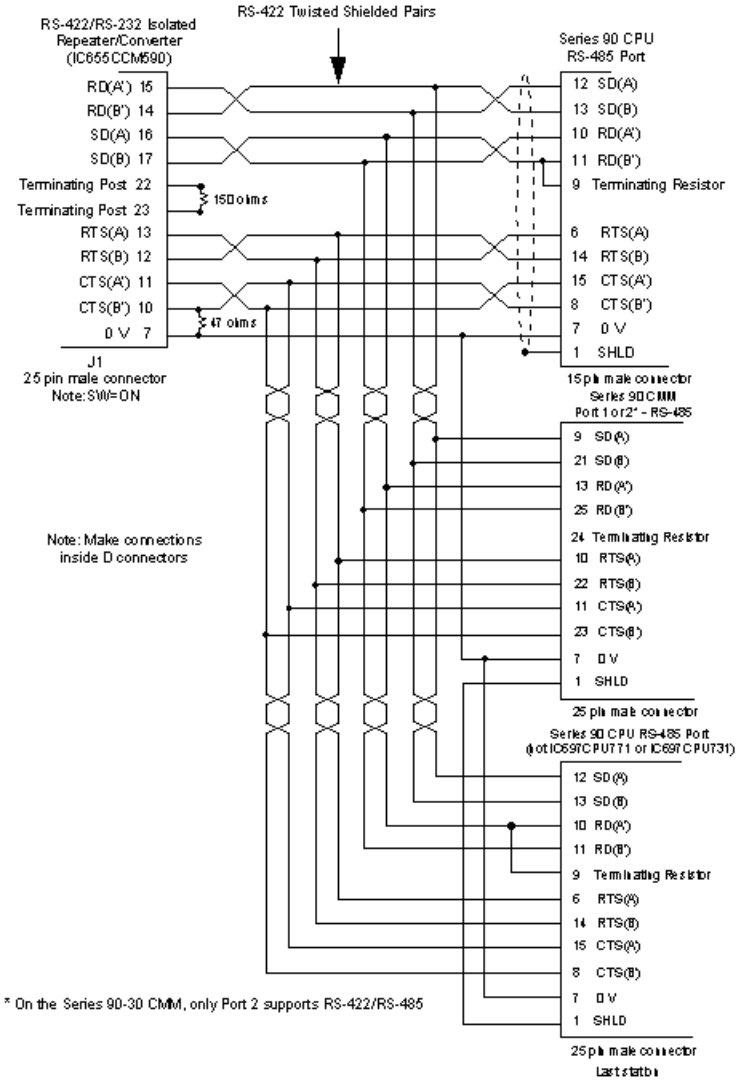
Cable M

RS-422/RS-232 Converter to Series 90 CPU Ports
Multidrop Configuration



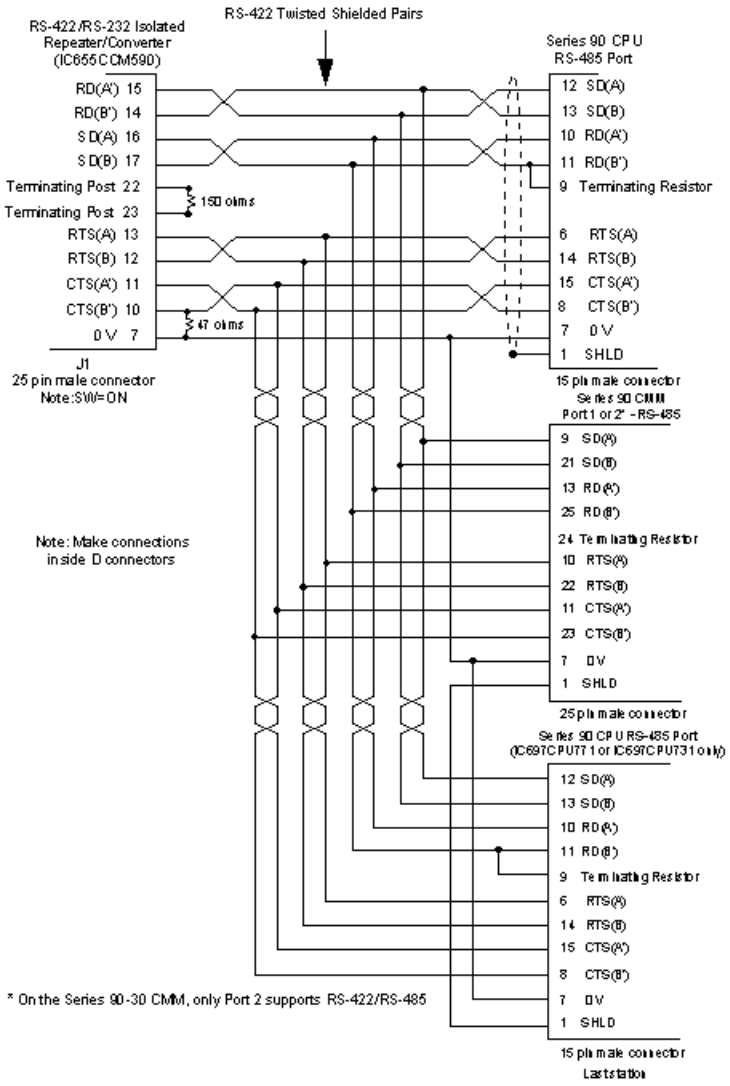
Cable N

RS-422/RS-232 Isolated Repeater/Converter to Series 90 Ports
Multidrop Configuration

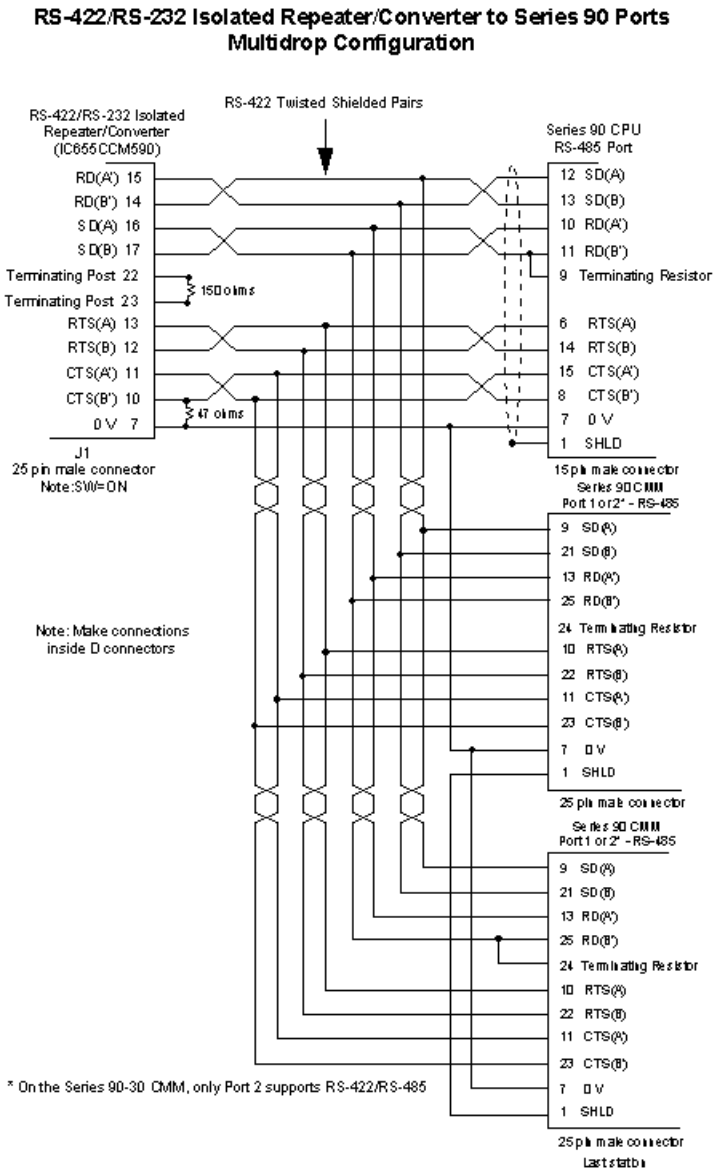


Cable O

RS-422/RS-232 Isolated Repeater/Converter to Series 90 Ports Multidrop Configuration



Cable P



SNP and SNPX Hardware Installation

Serial communication ports are used to communicate between the Series 90 controllers and CIMPLICITY software. You may use standard serial ports on your PC.

If you are using SNP or SNPX communications with a Series 90-30, the ground connection at the PC COM port has to be removed for communications to work.

CIMPLICITY Configuration for SNP and SNPX

CIMPLICITY Configuration for SNP and SNPX

When you are configuring ports, devices, and device points, you will be asked for some information that is specific to SNP or SNPX device communications.

SNP and SNPX Port Configuration

SNP and SNPX Port Configuration

- SNP Port Configuration
- SNPX Port Configuration

SNP Port Configuration

SNP Port Configuration

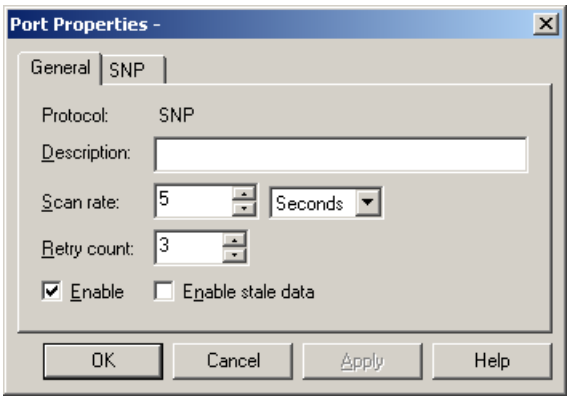
1. Select the following in the New Port dialog box.

Field	Description
Protocol	Select SNP in the list of available protocols.
Port	Select the communication port that will be used for SNP communications.

2. Click OK.

The Port Properties dialog box for SNP Communications opens.

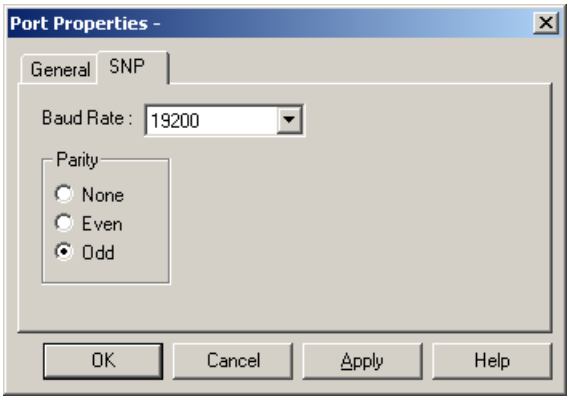
General Port Properties



Use the General tab in the Port Properties dialog box to enter general information for the port. You can define the following:

Description	Enter an optional description to help you identify the port.
Scan Rate	Enter the base scan rate for the port. Point scan rates will be in multiples of the base rate. You can specify a scan rate in Ticks (100 ticks = 1 second), Seconds, Minutes, or Hours.
Retry Count	If communications cannot be established to a device on the port, the device is considered to be down and a $DEVICE_DOWN alarm is generated. Once a device is down, periodic attempts are made to resume connections to it.
	Enter the number of scans to wait before attempting to reconnect to a device after a communications error has been detected.
	Configure the Retry Count to 3 or less under the following circumstances: For a Series 90-70 with a CPU firmware version prior to V4.02. For a Series 90-30 with a CPU firmware version prior to V3.03. For a Series 90-20.
Enable	Set this check box if you want the port to be enabled when the project starts. If you clear this check box, the port will not be enabled when the project starts, and points will not be available for devices on the port.
Enable stale data	Check to keep the point available in most circumstances that would have made it unavailable. However the point value will be stale, meaning it is the last known value and may or may not have changed in the device.

Port Properties



Use the SNP tab in the Port Properties to enter information about the SNP communications for the port. You can define the following:

Baud Rate	Enter the baud rate for communications. Click the drop-down list to the right of the input field and select a rate from the list.
Parity	Select the parity for communications.

The default values are:

Baud Rate	19200
Parity	Odd

Series 90 controllers are normally shipped with the RS-485 and CMM ports configured for a baud rate of 19.2 K baud and odd parity.

Make sure the baud rate and parity you select match those on the Series 90s with which you are communicating.

SNPX Port Configuration

SNPX Port Configuration

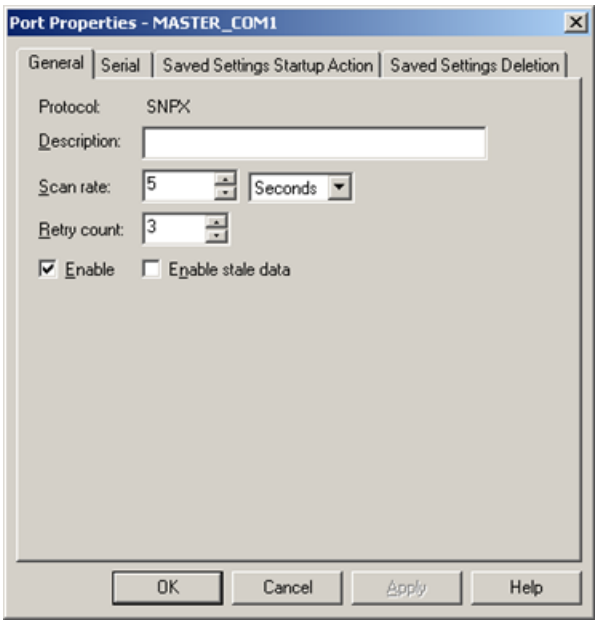
1. Select the following in the New Port dialog box.

Field	Description
Protocol	Select SNPX in the list of available protocols.
Port	Select the communication port that will be used for SNPX communications.

2. Click OK.

The Port Properties dialog box for SNPX Communications opens.

General Port Properties

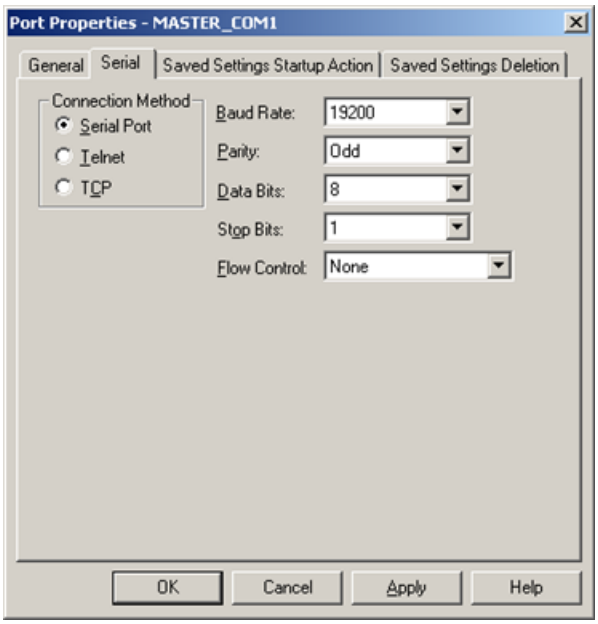


Use the General tab in the Port Properties dialog box to enter general information for the port. You can define the following:

Description	Enter an optional description to help you identify the port.
Scan Rate	Enter the base scan rate for the port. Point scan rates will be multiples of the base rate. You can specify a scan rate in **Ticks** (100 ticks = 1 second), **Seconds**, **Minutes**, or **Hours**.
Retry Count	If communications cannot be established for a device on the port, the device is considered to be down, and a $DEVICE_DOWN alarm is generated. Once a device is down, periodic attempts are made to resume connections to it.
	Enter the number of scans to wait before attempting to reconnect to a device after a communications error has been detected.
Enable	Check if you want the port to be enabled when the project starts. If you clear this check box, the port will not be enabled when the project starts, and points will not be available for devices on the port.
Enable stale data	Check to keep the point available in most circumstances that would have made it unavailable. However the point value will be stale, meaning it is the last known value and may or may not have changed in the device.

Serial Connection Port

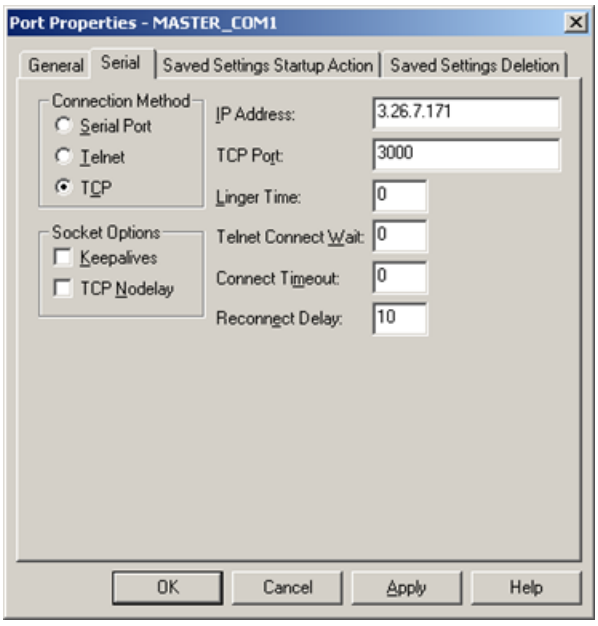
If you check the Serial Port connection method, define the following.



Baud Rate	Enter the baud rate for communications. Click the drop-down list to the right of the input field and select a rate from the list.
Data Bits	Select the number of data bits per word to be used for communications.
Parity	Select the parity to be used for communications.
Stop Bits	Select the number of stop bits to be used for communications
Flow Control	Select the type of flow control to be used for communications. If you change the Flow Control type, you must reboot the PC for the changes to take affect.

Telnet or TCP Connection Port

If you check the Telnet or TCP connection method, define the following.



Socket Options	Check the Keepalives check box to use keepalives to detect the loss of the terminal server. Clear the check box if you do not want to use keepalives to detect the loss of the terminal server. Set the TCP Nodelay check box if you want to set the Nodelay flag on the socket. Clear the check box if you do not want to set the Nodelay flag.
IP Address	Enter the IP address of the terminal server.
TCP Port	Enter the port number of the TCP port on the terminal server.
Linger Time	Enter the time in seconds to wait after closing the socket before aborting the socket.
Telnet Connect Wait	Enter the time in seconds to wait for the Telnet protocol to initialize.
Connect Timeout	Enter the time in seconds to wait for the TCP connection to form.
Reconnect Delay	Enter the time in seconds to wait before attempting to reconnect to a device. If you set this value to zero and the terminal server is not available, then no attempts will be made to reconnect to the terminal server.

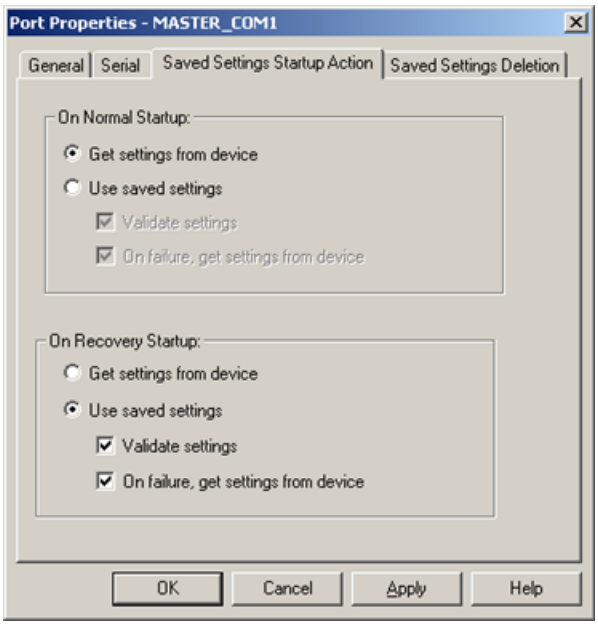
Saved Settings Startup Action

SNPX Communications provides you with the option to reduce normal and/or recovery start up time by saving device characteristics for subsequent re-use.

Note: Saved Settings Startup Action is intended for devices whose characteristics, such as the memory size, do not change. If the settings change, make sure they are deleted.

Select the Saved Settings Startup Action tab in the SNPX Port Properties dialog box.

Selections can be made for the following: Normal startup or On Recovery Startup.



rect 28, 55, 129, 78 *(page 923)*
rect 28, 197, 129, 220 *(page 924)*
rect 34, 82, 170, 105 *(page 923)*
rect 31, 220, 167, 243 *(page 923)*
rect 31, 106, 237, 171 *(page 923)*
rect 31, 242, 237, 307 *(page 923)*

Normal Startup

- Normal Startup occurs when CIMPLICITY starts.
- CIMPLICITY is started either when:
- The computer is booted up or
- A CIMPLICITY user starts the project.

Options for each of the startups are as follows.

Get settings from device

Defines the actions that the device communication interface takes to determine the supported memory types and ranges for a specific device.

The methods vary by device communication interface.

Use saved settings

Checked: The device communication interface will use the stored settings to define the device-specific memory types and ranges.

- The device configuration data is recorded and stored for later use.
- Options if Use saved settings is checked are:

Option	Description	
Validate settings	Checked	The device communication interface will query the high range of each memory range as a quick check to confirm that the memory type and range represented is valid.
	Clear	The device communication interface will not do a quick check.
On failure, get settings from device	When using saved settings, failures may occur. Two typical failures are: • An integrity error is detected in the saved information. • There is a failure to verify a memory range when Validate settings is selected.	
	Checked	When a failure occurs the device communication interface will immediately attempt to determine the valid device information using the standard method for obtaining the information.
	Clear	When the failure occurs, the: • Device will immediately be marked down. • Valid device information will not be collected during initial startup. It will be determined later during retry processing.

On Recovery Startup

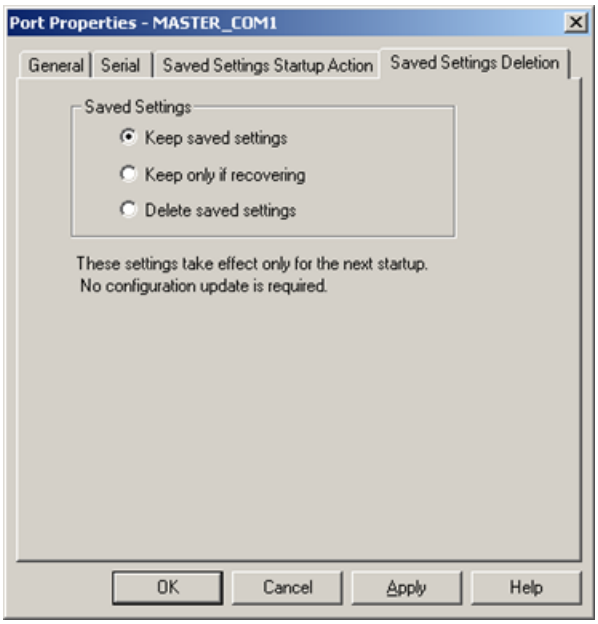
Recovery startup occurs after:

- A cluster fails.
- Process health kills a project as a result of a process failure.

Options *(page 923)* for On Recovery Startup are the same as they are for On Normal Startup.

Saved Settings Deletion

Check one of the following to specify what SNPX Communications should do with saved settings.



Option	Description
Keep saved settings	Do not automatically delete the saved settings on the next startup
Keep only if recovering	Delete the current saved settings unless the next startup is in recovery mode.
Delete saved settings	Deletes: • The current saved settings at the next startup. • Settings for all the devices that are configured for the port. **Note:** If the configuration of the device is changed, make sure to check and apply Delete saved settings.

Note: These settings affect the next startup only. The Saved Settings Deletion tab will always default to the Keep saved settings option for subsequent startups.

SNP and SNPX Device Configuration

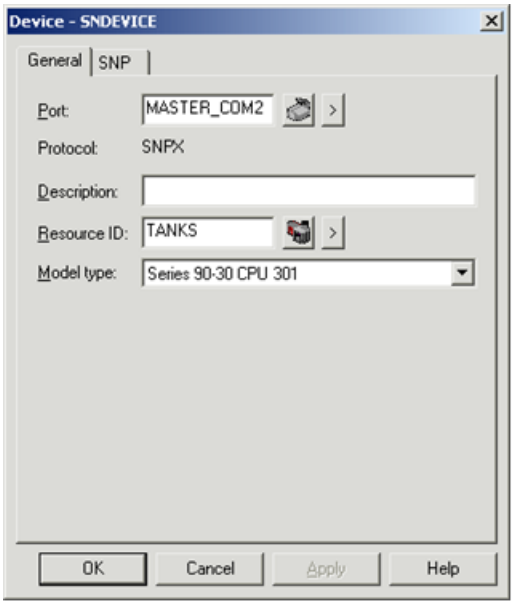
SNP and SNPX Device Configuration

1. In the **Device** field, enter the name of the device you are configuring.
2. In the **Port** field, select the SNP port to be used by the device.

When you click OK to create the port, the Device Properties dialog box opens.

Note: The following special characters cannot be used for a SNPX device name: \, /, :, *, ?, ", <. >, |, [,]. If the device name contains any of these characters, the device settings will not be preserved and saved startup will not be possible.

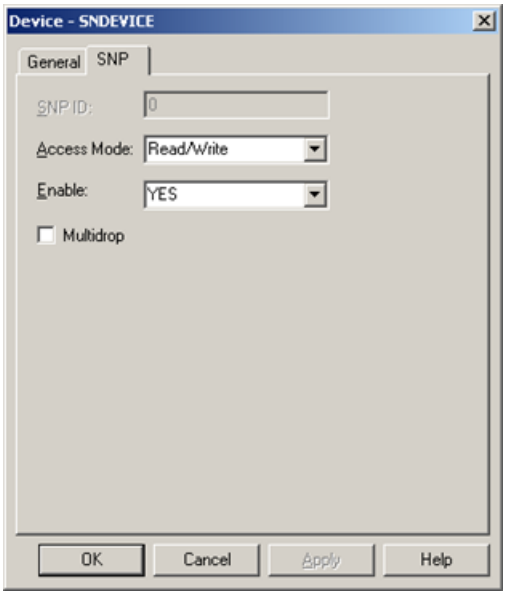
General Device Properties



Use the General tab in the Device dialog box to enter general information for the device. You can define the following:

Port	Select the port for this device. Click buttons to the Port field to select the port as follows.	
		Display the list of ports and select one.
		Create a new port, edit the current port, or select a port from the list of ports.
Description	Enter an optional description to help you identify the device.	
Resource	Enter a resource that can be associated with this device for alarm generation. Click buttons to the right of the **Resource** field to select the resource, as follows.	
		Display the list of resources and select one.
		Create a new resource, edit the current resource, or select a resource from the list of resources.
Model Type	Select the type of device. Click the drop-down button to the right of the input field to display your choices, then make a selection.	
	For SNP communications, the choices are: PACSystems RX3i PACSystems RX7i Series 90-30 Series 90-70 Series 90 Micro VersaMax For SNPX communications, the choices are: Series 90-30 CPU 301 Series 90-30 CPU 311 10 slot Series 90-30 CPU 311 5 slot Series 90-30 CPU 313 Series 90-30 CPU 323 Series 90-30 CPU 331 Series 90-30 CPU 341 Series 90-30 CPU 35x, 36x VersaMax	

SNP/SNPX Device Properties



Use the SNP tab in the Device dialog box to enter information about SNP or SNPX device communications for the device. You can define the following:

SNP ID	For multi-drop communications: • Check the **Multidrop** check box. • Enter the ID of the programmable controller in SNP ID. The SNP ID may contain ASCII (A-Z) characters as well as numbers (0-9).
Access Mode	For point-to-point communications: • Clear the **Multidrop** check box. • Select Read or Read/Write in Access Mode.
Enable	For all communications • Click Yes to enable the device when the project starts. • Click No to disable the device. The points associated with the device will be unavailable.

SNP and SNPX Point Configuration

When you define a point, the following fields on the Device tab in the Point Properties dialog box, the point has values that are unique to or have special meanings for SNP and SNPX device communications:

Access	A point may be configured for READ or WRITE access.

Update Criteria	The update criteria determine how the data will be requested.
	Enter **On Change** or **On Scan** for points whose values should be polled by the SNP or SNPX communications enabler at regular intervals. Enter **On Demand On Change** or **On Demand On Scan** for points whose values should be polled by the SNP or SNPX communications enabler at regular intervals while users are viewing them. Enter Poll Once for points that are to be polled once when the SNP or SNPX communications enabler starts.
Address	Point address requirements are device-dependent. For SNP and SNPX device communications, the address format is <memory_type><offset> The memory types you can use are listed in the Supported SNP and SNPX Memory Types *(page 893)* section

When you are configuring Boolean points, you must also enter data in the following fields:

Address Offset	Enter the bit offset that corresponds to the bit position within the word.

SNP Performance Notes

SNP Performance Notes

- Impact on Series 90 performance.
- Performance using the SNP Communications enabler.
- Tune SNP protocol throughput.
- SNP special notes.

Impact on Series 90 Performance

When the attached port of a Series 90 controller is being used to support serial communications, a maximum of 5 milliseconds will be added to the scan on the controller. This should be taken into account in any application.

Performance Using the SNP Communications Enabler

The CIMPLICITY SNP communications enabler is a serial communications enabler that supports access to most memory types in a Series 90 controller. The amount of data that can be transferred between CIMPLICITY software and the Series 90 controller is limited by the baud rate and some protocol level tunables (discussed in the next section).

In addition, when communicating to multiple Series 90 controllers in a multi-drop configuration, certain transactions between the CIMPLICITY software and the Series 90 controller must occur each time communication is switched to a different Series 90 controller. These transactions occupy a

significant amount of time (typically 0.4 to 1 seconds). When designing a CIMPLICITY application, the throughput via SNP should be taken into account.

Tune SNP Protocol Throughput

The protocol supports the ability to tune certain protocol level parameters that significantly impact performance. These parameters, along with the baud rate, are dominant factors in polling throughput.

- SNP data size.
- Time between SNP messages.
- Time before SNP attach messages.

SNP Data Size

This parameter controls the amount of data that can be sent between the host and the Series 90 controller. For CIMPLICITY projects, CIMPLICITY software will attempt to negotiate a data size of 4000 bytes. This value was selected in order to optimize the I/O performance in the computer where CIMPLICITY software is running. If a lower value is configured on the Series 90 controller, the lower value will be used.

Time between SNP Messages

This parameter controls the time between SNP messages. By default, CIMPLICITY software attempts to negotiate a value of 5 milliseconds. You can configure a higher value by defining a logical named BSM$SNP_T1_TIME.

To define this logical, add a line to the project's **data\log_names.cfg** file with the following format:

```
BSM$SNP_T1_TIME|S|default|3|<time>
```

Where <time> is the desired time in milliseconds between messages.

If you create or modify the logical, you must stop and restart the project for the new value to take effect.

As with the data size parameter, the time between SNP messages is a negotiated value. A higher value on the Series 90 controller will result in the higher value being used.

Time before SNP Attach Messages

This parameter controls the time delay before SNP attach messages. By default, there is a 1-millisecond delay before the SNP attach messages. If the SNP attach fails while the PLCs are running, a minimum value of 5 milliseconds is recommended to resolve the connect failure situation. You can configure a longer delay value by defining a logical named **BSM $SNP_ATTACH_DELAY**.

To define this logical, add a line to the project's **data\log_names.cfg** file with the following format:

```
BSM$SNP_ATTACH_DELAY|P|default|3|<time>
```

where <time> is the desired delay time in milliseconds before the SNP attach messages.

If you create or modify the logical, you must stop and restart the project for the new value to take effect.

SNP Special Notes

- Log controller information.
- Enable protocol level debugging.
- Disable protocol level debugging.

Log Controller Information

When a CIMPLICITY project first establishes communication with a Series 90 controller, it prints out the Series 90 controller's model, SNP timer values, the internal data transfer size and the number of elements for each memory type available in CIMPLICITY software. This information appears in the **<PORT>.out** file in the project's **log** directory, where <PORT> is the name of the port configured for SNP communications. For example, if SNP is configured to run on COM1, the output file will be **COM01.out**.

Enable Protocol Level Debugging

In addition to this information, the SNP communications enabler supports the ability to dynamically enable/disable the logging of protocol level errors as they occur. This information is logged in the Status Log file. By default, the logging of protocol level errors is disabled. When difficulties are encountered in communicating with a device, it may be useful to enable the protocol level debugging. To do this, create the **<PORT>.debug** file in the project's **\log** directory, where <PORT> is the name of the port configured for SNP communications.

To create the file, select Command Prompt... from the Workbench Tools menu, then type the following commands to create the file:

```
cd .\log
copy nul <PORT>.debug
```

where <PORT> is the name of the port configured for SNP communications. For example, if SNP is configured to run on COM1, the file name will be **COM01.debug**.

The debugging begins immediately. You do not need to restart the CIMPLICITY project.

Disable Protocol Level Debugging

To turn off the logging of protocol level information, remove the **COM01.debug** file from the project's **\log** directory.

To remove the file, select Command Prompt... from your project's Tools menu, then type the following commands to create the file:

```
cd .\log
del <PORT>.debug
```

where **<PORT>** is the name of the port configured for SNP communications.

SNP Global Parameters

The SNP Device Communication protocol supports disabling BREAK when using the Firmware version 8.2 of PLC Firmware on Series 90 devices.

The following project level global parameters have been added:

- SNP_SEND_BREAK
- SNP_IDLE_TIME

SNP_SEND_BREAK

For	SNP Project	
Purpose	To enable or disable sending BREAK.	
	SNP_SEND_BREAK can be superceded on a port basis by defining <PORT>_SEND_BREAK.	
Value	Choose one of the following.	
	Y	Enables BREAK.
	N	Disables BREAK.
Default Value	Y	

SNP_IDLE_TIME

For	SNP Project
Purpose	To set the number of seconds for idle time when BREAK is disabled.
	SNP_IDLE_TIME an be superceded on a port basis by defining <PORT>_IDLE_TIME.

Value	Number of seconds.
Default Value	10 seconds.

SNPX Global Parameters

- SNPX_BROADCAST_TIME
- SNPX_VALIDATE
- <DEVICE>_VALIDATE

SNPX_BROADCAST_TIME

Time after SNPX Broadcast Attach Messages

This parameter controls the time delay after SNPX broadcast attach messages. By default, there is a 20-millisecond delay after the SNPX broadcast attach messages. If the SNPX fails to read data from PLCs while the PLCs are running, a longer delay is recommended to resolve the read request failure situation. You can configure a longer delay value by defining a logical named **SNPX_BROADCAST_TIME**.

To define this logical, add a line to the project's **data\log_names.cfg** file with the following format:

```
SNPX_BROADCAST_TIME|P|default|10|<time>
```

where < time > is the desired delay time in milliseconds after the SNP-X broadcast attach messages.

If you create or modify the logical, you must stop and restart the project for the new value to take effect. Please refer to Series 90 PLC Serial Communications User's Manual for more information regarding this broadcast delay timer.

SNPX_VALIDATE

When an INI file is used and SNPX_VALIDATE was defined to a value of N, device communication will always resume following a drop in the communications.

In SNPX, one may choose to define the memory sizes for a given device by creating a file named <PORT>.INI and including information about the memory sizes for one or more devices configured on the given CIMPLICITY port. The format of the file is as follows:

```
[YOUR_DEVICE_ID]
MemoryType=CountOfMemory Type
Below is a sample file for COM1 (MASTER_COM1.INI) where one Device, DEVICE1
 has been configured:
[DEVICE1]
%AI=2048
```

```
%AQ=512
%R=9999
%I=2048
%Q=2048
%T=256
%M=4096
%G=1280
%S=32
%SA=32
%SB=32
%SC=32
```

By default, the communication interface will evaluate the information and confirm that the last element in each specified range exists. If the items doesn’t exist, then the communication interface will, by default, attempt to determine the number of items configured for the given memory type.

To override this behavior, a global can created at the device, or project level to override the behavior. When defined, the device level global overrides the project level global which overrides the default.

The project level global is defined as SNPX_VALIDATE.

<DEVICE>_VALIDATE

The device level global parameter is< DEVICE>_VALIDATE.

Where

Device is the name of the device.

The default value is Y.

To override the default, define the global to have a value of N.

Chapter 55. Square D SY/MAX Communications

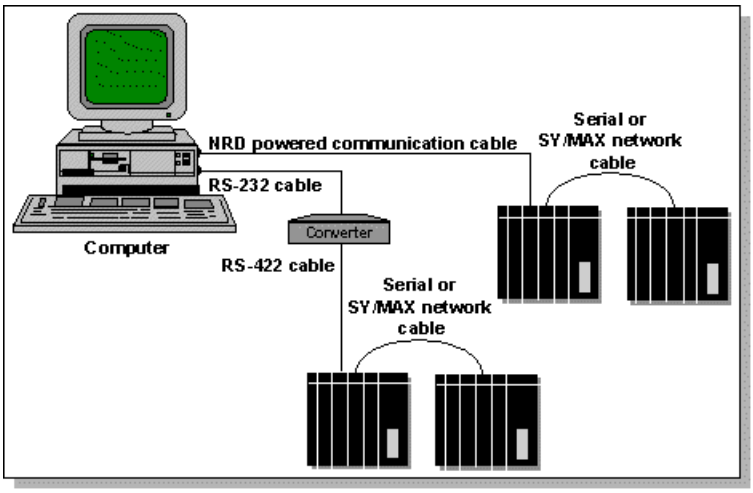
About Square D SY/MAX Communications

The Square D SY/MAX Communications option lets you exchange data between CIMPLICITY software and Square D family PLCs using Square D's SY/MAX protocol.

Communications between a CIMPLICITY computer and Square D PLCs via SY/MAX protocol is accomplished by connecting a serial port on the CIMPLICITY computer to a SY/NET Network Interface Module or the direct serial port on the first PLC in the network. Routing information, as defined in the Processor Instruction Bulletin for your PLC model(s), is used to route requests to the devices.

This protocol was validated using an NRD powered communication cable.

If you are using an RS-232 cable from the serial port on your computer, you will need an RS232 to RS422 converter box and an RS-422 cable to connect to the first PLC. The use of an isolated converter box is recommended.



You can connect up to 199 programmable controllers to a CIMPLICITY computer on a single network connection. Networks can also be linked.

This enabler supports the following CIMPLICITY features:

- Polled reads at user defined rates
- Triggered reads
- Unsolicited data
- Analog deadband

- Alarm on communications failure

This enabler supports the following data types:

- Signed 8, 16, and 32 bit integers
- Unsigned 8, 16, and 32 bit integers
- Floating point
- Text
- Arrays

Square D SY/MAX Required Documents

You should have the following documentation, as it applies to your hardware configuration, available when configuring this interface:

Square D SY/MAX Protocol Reference Instruction Bulletin 3059871301

Square D SY/NET Network Interface Module Instruction Bulletin 3059871301

Square D SY/NET Network Interface Module Technical Overview TO25703

Square D SY/MAX Model 400 Processor Instruction Bulletin 3059850301

Square D SY/MAX Model 600 Processor Instruction Bulletin 3059860901

Square D SY/MAX Supported Devices

The Square D SY/MAX Communication option supports data collection and modification of memory values in target devices that adhere to the protocol specified in the Square D SY/MAX Protocol Reference Instruction Bulletin 3059871301.

Devices supporting this protocol include:

- SY/MAX Model 600 Processor; Class 8020, Type SCP631, and 632
- SY/MAX Model 500 Processor; Class 8020, Type SCP501, 522, 523, and 544
- SY/MAX Model 400 Processor; Class 8020, Type SCP401, 423, 424, and 444
- SY/Matic Weld Controllers (EQ5200, EQ5110, EQ5100)
- Power Logic Circuit Monitor; Class 3020

Square D SY/MAX Supported Memory Types

Square D does not reserve areas of user memory for special purposes or data types. Memory locations (registers) contain 16 bits and are addressed by their location numbers.

Square D SY/MAX Installation Requirements

You may use any available serial communication port on your computer for Square D SY/MAX communications. No special port configuration is required.

Square D SY/MAX Test Program

Square D SY/MAX Test Program

A stand-alone test program, **sd_test.exe**, is provided. This program provides direct communication between CIMPLICITY software and a SY/NET Network Interface Module, or between the CIMPLICITY software and a Programmable Logic Controller.

The test program can be used to:

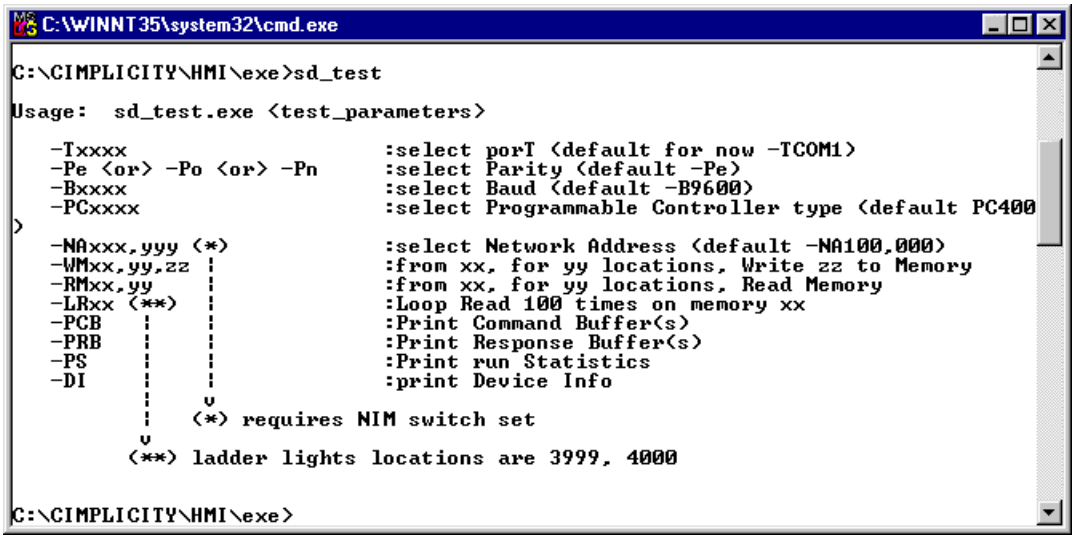
- Verify port, network, and interface unit communication parameters.
- Verify line, network, and hardware functionality and operability.
- Directly access (for reading or writing) a specific PLC register location.
- Dump or clear some or all of the PLC user registers.
- Examine the send and receive message buffer content.
- Collect transmission statistics of the current configuration.
- Verify device information response.

You cannot use the test program while the CIMPLICITY Square D Communications enabler is running.

If the protocol for Square D is copyrighted or declared to be trademark or proprietary, or there are very restrictive conditions on copying the manual, obtain written permission from Square D to display the command or response buffers in the test program.

Test Program Banner Page

To display the test program banner page, enter the test program name, without any test parameters. The resulting display is as follows:



Test Program Applications Examples

Ensure that the port, parity, baud, PLC type, and network address are specified on the command line, if different that the default.

Examples

```
sd_test.exe –TCOM1 –B4800 –PC423
```

Selects the COM1 port, sets the baud rate to 4800 and selects the 423 programmable controller type.

```
sd_test.exe –B2400 –NA100.077
```

Sets the baud rate to 2400 and the network address to 100.077

Use the WM, RM, or LR commands to write or read to specified register (memory) locations.

Examples

```
sd_test.exe –RM1,8000
```

Initiates a memory dump of a Series 600 PLC.

```
sd_test.exe –WM18,3981,0
```

Writes zeroes in locations 18 through 3981 of a Series 400 PLC.

```
sd_test.exe –RM1011
```

Reads memory location 1011 in the PLC.

```
sd_test.exe –LR3999
```

Executes a loop and read 100 times on location 3999 in the PLC.

Use the PCB, PRB, and PS commands for additional diagnostic information.

Examples

```
sd_test.exe –PRB –PCB
```

Prints the command and receive messages of specified operation

```
sd_test.exe – PS
```

Prints statistics pertaining to the specified operation

Use DI to verify or ascertain PLC type and memory size recognition.

Example

```
sd_test.exe –PC500 –DI
```

Prints the device information for a Series 500 PLC.

CIMPLICITY Configuration for Square D SY/MAX

CIMPLICITY Configuration for Square D SY/MAX

As with other CIMPLICITY communication options, you must complete the Port, Device, and Device Point configuration to set up Square D SY/MAX device communications.

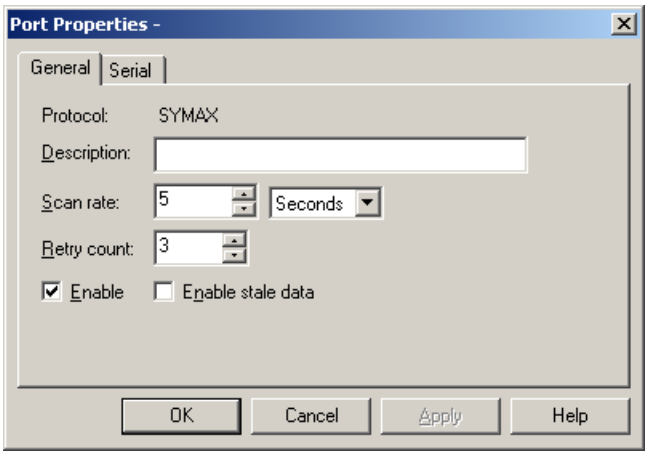
Square D SY/MAX Port Configuration

Square D SY/MAX Port Configuration

1. In the **Protocol** field, select **SY/MAX** from the list of available protocols.
2. In the **Port** field, select the communication port that will be used for Square D communications.

When you select **OK** to create the port, the Port Properties dialog box for the protocol opens.

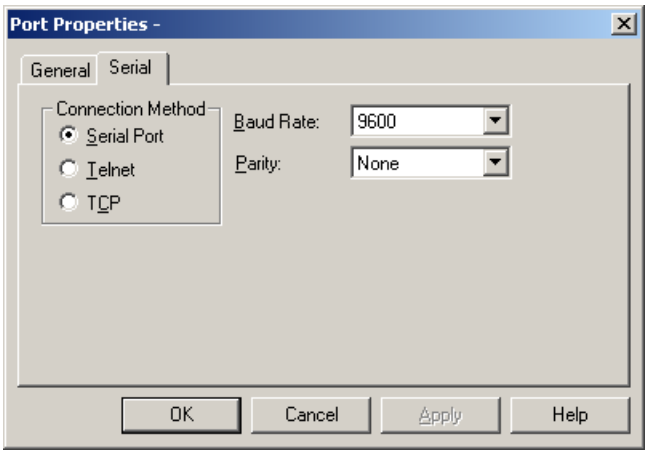
SY/MAX General Port Properties



Use the General tab in the Port Properties dialog box to enter general information for the port. You can define the following:

Description	Enter an optional description to help you identify the port.
Scan Rate	Enter the base scan rate for the port. Point scan rates will be multiples of the base rate. You can specify a scan rate in Ticks (100 ticks = 1 second), Seconds, Minutes, or Hours.
Retry Count	If communications cannot be established for a device on the port, the device is considered to be down, and a $DEVICE_DOWN alarm is generated. Once a device is down, periodic attempts are made to resume connections to it.
	Enter the number of scans to wait before attempting to reconnect to a device after a communications error has been detected.
Enable	Set this check box if you want the port to be enabled when the project starts. If you clear this check box, the port will not be enabled when the project starts, and points will not be available for devices on the port.

Serial Port Properties: Serial Port Connection

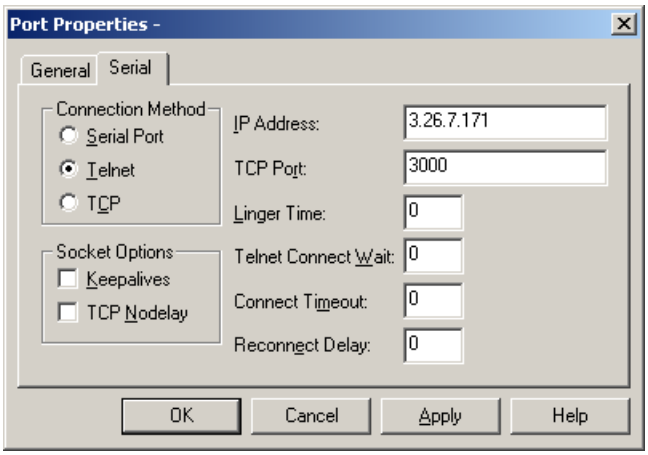


If you select the serial port connection method, you need to define the following:

Baud Rate	Enter the baud rate for communications. Click the drop-down list to the right of the input field and select a rate from the list.
Data Bits	Select the number of data bits per word to be used for communications.
Parity	Select the parity to be used for communications.
Stop Bits	Select the number of stop bits to be used for communications
Flow Control	Select the type of flow control to be used for communications. If you change the Flow Control type, you must reboot the PC for the changes to take affect.

Remember that you must configure the same baud rate, data bits, parity, stop bits and flow control for all PLCs using the serial port.

Serial Port Properties: Telnet or TCP Connection



If you select the Telnet or TCP connection method, you need to define the following:

Socket Options	Check the Keepalives check box to use keepalives to detect the loss of the terminal server. Clear the check box if you do not want to use keepalives to detect the loss of the terminal server. Check the TCP Nodelay check box if you want to set the Nodelay flag on the socket. Clear the check box if you do not want to set the Nodelay flag.
IP Address	IP address of the terminal server.
TCP Port	Port number of the TCP port on the terminal server.
Linger Time	Time in seconds to wait after closing the socket before aborting the socket.
Telnet Connect Wait	Time in seconds to wait for the Telnet protocol to initialize.
Connect Timeout	Time in seconds to wait for the TCP connection to form.
Reconnect Delay	Time in seconds to wait before attempting to reconnect to a device. If you set this value to zero and the terminal server is not available, then no attempts will be made to reconnect to the terminal server.

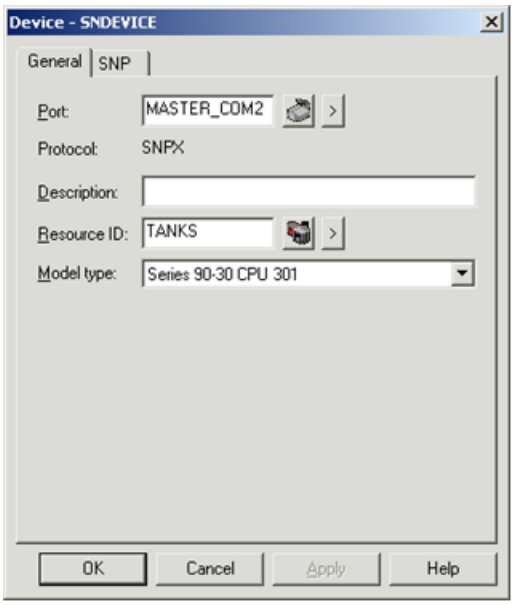
Square D SY/MAX Device Configuration

Square D SY/MAX Device Configuration

1. In the **Device** field, enter the name of the device you are configuring.
2. In the **Port** field, select the Square D SY/MAX port to be used by the device.

   When you click **OK** to create the device, the Device Properties dialog box opens.

General Device Properties

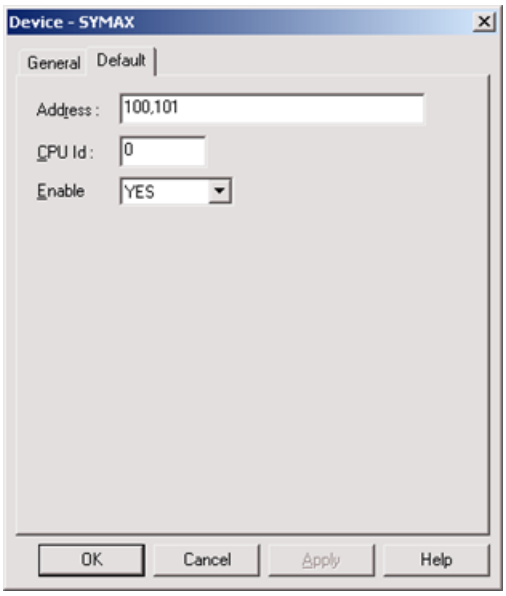


Use the General tab in the Device dialog box to enter general information for the device. You can define the following:

Port	Select the port for this device. Click buttons to the Port field to select the port as follows.	
		Display the list of ports and select one.
		Create a new port, edit the current port, or select a port from the list of ports.
Description		Enter an optional description to help you identify the device.
Resource	Enter a resource that can be associated with this device for alarm generation. Click buttons to the right of the **Resource** field to select the resource, as follows.	
		Display the list of resources and select one.
		Create a new resource, edit the current resource, or select a resource from the list of resources.
Model Type	Select the type of device. Click the drop-down button to the right of the input field to display your choices, then make a selection. For this protocol, the only choice is:	

	SY/MAX SY/MAX-300 SY/MAX-400	SY/MAX-500 SY/MAX-600 SY/MAX-AS42C

Default Device Properties



Use the Default tab in the Device dialog box to enter information about Square D SY/MAX communications for the device. You can define the following:

Address	Each device to be accessed by the Square D SY/MAX Device Communications interface will require a sequence of one (1) or more 3-digit numbers representing the network route (for example, 100 101).
CPU ID	Not used.
Enable	Select Yes to enable the device when the project starts. If you select No, the device will not be enabled and points associated with the device will be unavailable.

Square D SY/MAX Point Configuration

Square D SY/MAX Point Configuration

Once your devices are configured, you may configure data points for them. Fields in the Point Properties dialog box have values that are unique or have specific meaning for Square D SY/MAX Device Communications.

General Point Properties

On the General tab in the Point Properties dialog box, you may configure points for **Read** or **Read/ Write** access.

Device Point Properties

On the Device tab in the Point Properties dialog box:

Update Criteria	The update criteria you use determines how the data will be requested.
	Select **On Change** or **On Scan** for points whose values should be polled by the SY/MAX driver at regular intervals.
	Select **On Demand On Scan** or **On Demand On Change** for points whose values should be polled by the SY/MAX driver at regular intervals while users are viewing them. A Square D SY/MAX address consists of a register (memory location) number. The ranges on the various models are: 300 Series PLC = 1112 400 Series PLC = 14000 500 Series PLC = 14193 600 Series PLC = 18000 Generic SYMAX model = 14000 Addresses must be specified as numerical values within the PLC range. The FLOATING Point Type occupies two adjacent registers and must be specified as an odd number within the PLC range.

SY/MAX Advanced Configuration Topics

You can control some of the SY/MAX Communications options through global parameters. This section discusses those global parameters and their settings.

SYMAX_<PORT>_WORD_SWAP

By default, the SY/MAX Communications enabler swaps byes for digital word points.

To disable byte swapping, you can add the following to the global parameter file:

SYMAX_<PORT>_WORD_SWAP|1|NO

Where

<port_name> is the name of the SY/MAX port.

To enable byte swapping again, you can delete the global parameter, or change the parameter value from NO to YES.

Chapter 56. Toyopuc Ethernet Device Communication Module

About Toyopuc Ethernet Device Communication Module

This communication enabler supports data collection from Toyoda Machine Work's Toyopuc PC2 Series programmable controllers via Ethernet. From an Ethernet connection established with a Toyopuc programmable controller, it is possible to collect data from the local programmable controller as well as from other PC2 Series programmable controllers accessible via ME-NET or High Speed PC Link (HPC).

Toyopuc Supported Devices

Toyoda Toyopuc PC2 Series Programmable Controllers that:

- Have installed an Ethernet EN-IF card at revision level V2.1or later.
- Are networked via ME-NET or HPC to a PC2 Series Programmable controller with an installed EN-IF Ethernet card at revision level V2.1 or higher.

Toyopuc Required Hardware/Software

You must have the following:

- Toyopuc PC2 Series Programmable Controller with EN-I/F card (V2.1)
- PC with Ethernet card installed.

Toyopuc Programmable Controller Configuration Requirements

Hardware Setup

All installed I/O, communications cards, must be configured and downloaded to your programmable controller.

Toyopuc Ethernet Card Configuration Requirements

The EN-IF Ethernet Card must be configured into the programmable controller via Hellowin or GL1 programming software. The EN-IF Ethernet card is setup by writing ladder logic to configure the card's IP address and Ethernet communication port. Non-specific Passive Communication setup with a non-zero non-reception timer is recommended.

Toyopuc Module Related Documents

Programmable Controller TOYOPUC PC2 series EN-I/F Instruction Manual – T-8765N

Toyopuc Programmable Logic Controller INSTRUCTION MANUAL FOR your Toyopuc Processor

Toyopuc Programmable Logic Controller INSTRUCTION MANUAL FOR GL1 PROGRAMMER FOR PC 2 Series

Toyopuc Programmable Logic Controller GL1 Common Module for IBM PC/AT COMPATIBLE COMPUTERS – INSTRUCTION MANUAL

Toyopuc HelloWin Programmer for PC3/2 Series Instruction Manual

Toyopuc Programmable Logic Controller – High Speed PC Link

Toyopuc Programmable Logic Controller – ME-NET

Communication Configuration Checklist

Please follow these instructions for configuring your Ethernet configuration prior to trying to use this communication interface. If your PC and/or programmable controllers are to be connected to a standardized or company-wide network, please contact your network administrator for the specific hostnames, IP addresses and socket port numbers used in your setup. Install an Ethernet card in your PC.

1. Configure the minimum TCP/IP Communications for your PC. This includes assigning a name and TCP/IP address for the computer.

Note: By default, if your computer has multiple Ethernet cards installed, the first connection configured for access by network services is used. An alternate card may be configured for each Toyopuc port. See Toyopuc Global Parameters *(page 954)* for more details.

2. Assign a TCP/IP address and socket port for each TCP/IP connection to be used on your Toyopuc PC2 Series programmable controller.
3. Setup each EN-IF Ethernet card that will be used to communicate with CIMPLICITY .
4. Configure the TCP/IP address for each connection to a Toyopuc PC2 series programmable controller in the host file. (See Device Address specification for additional information)
5. Verify Ethernet connectivity to the each Toyopuc PC2 connection that has a unique IP address by pinging the processor. Using the name configured for the programmable controller, execute the command

   ping < name>

6. Using the test program, toyopuc_diag, included with this software distribution, verify communications with each PC2 programmer for each connection that will be used in your CIMPLICITY application.
7. Develop and run your CIMPLICITY application that collects data from Toyopuc PC2 Series programmable controllers.

Toyopuc_diag

The diagnostic program, toyopuc_diag, may be use to read and write data to a Toyopuc PC2 series programmable controller, accessible via Ethernet (including those networked to such a programmable controller via ME-NET or HPC).

To use the diagnostic program, you will first need to go to the configuration cabinet for your project and do the following:

1. Open your project in the CIMPLICITY Workbench.
2. Select Command Prompt, from the Tools menu, to open an MS-DOS window.
3. In the MS-DOS window, type the following at the MS-DOS prompt:

   toyopuc_diag < device_address> <memory_address> <element_count> <command> **[data]**

   (The typed line is on two lines here for display purpose. Type it on one line at the MS-DOS prompt.)

Note: Please review the sections of this document titled Device Address Specification *(page 950)* for the format of the device address, and the section titled Toyopuc Supported Memory Types *(page 954)* for the format and range of memory addresses.

The available commands are:

-r	Read the data and display results in decimal
-h	Read the data and display results in hex
-w	Write the data. Values to be written are entered in decimal.
-x	Write the data. Values to be written are entered in hex

To read the first 3 data registers on the programmable controller Toyopuc accessible through socket port 4096, the following command would be entered at the DOS prompt.

```
toyopuc_diag toyopuc/4096 d0 3 –r
```

For example, the following command would be used to write to the first 3 data registers on the Toyopuc programmable controller (setting d0 to 100, d1 to 200 and d2 to 300):

```
Toyopuc_diag toyopuc/4096 d0 3 –w 100 200 300
```

Multiple Ethernet Cards

If you have multiple Ethernet cards and do no want to use the first one configured for network access:

4. Configure a hostname to correspond with the TCP/IP address for the Ethernet card to be used by the diagnostic.
5. Set the environment variable **TOYOPUC_BIND_ADDR** to the hostname configured in Step 1.

   For example, if the second Ethernet card is configured to the TCP/IP address 192.168.1.100 and the hostname for that address is MYPC2, then the following command is entered in the command prompt:

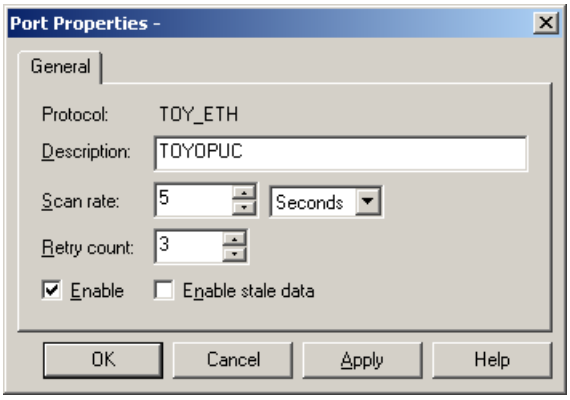
   ```
   SET TOYOPUC_BIND_ADDR=MYPC2
   ```

CIMPLICITY Configuration for Toyopuc

CIMPLICITY Configuration for Toyopuc

As with other CIMPLICITY software communications enablers, you must complete the Port, Resource, Device and Device Point Configuration to setup the Toyopuc Ethernet Communications.

Toyopuc Port Configuration



When you configure a port for Toyopuc Ethernet communications, the New Port dialog box opens.

Use the General Properties tab to enter general information for the port. You can define the following:

Description	Enter an optional description to help you identify the port.
Scan Rate	Enter the base scan rate for the port. Point scan rates will be multiples of the base rate.
	You can specify a scan rate in Ticks (100 ticks = 1 second), Seconds, Minutes, or Hours.
Retry Count	If communications cannot be established to a device on this port, the device is considered to be down and a $DEVICE_DOWN alarm is generated. Once a device is down, periodic attempts are made to resume connection to it.
	Enter the number of scans to wait before attempting to reconnect to a device on this port after a communications error is detected.
Enable	Set this check box if you want the port to be enabled when the project starts. If you clear this check box, the port will not be enabled when the project starts, and points will not be available for devices on the port.

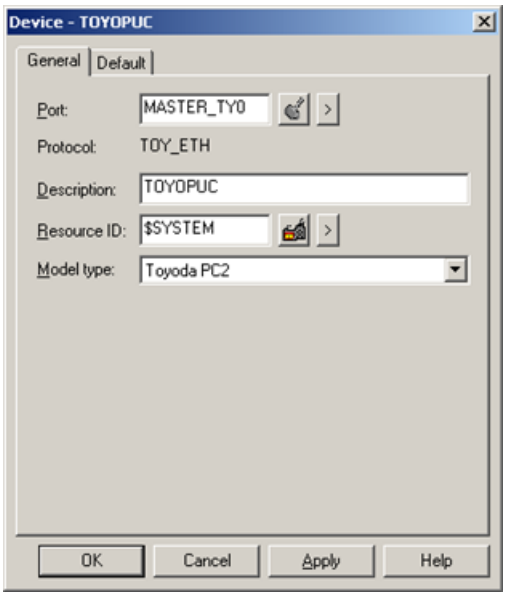
Toyopuc Device Configuration

Toyopuc Device Configuration

1. In the **Device** field, enter the name of the device you are configuring.
2. In the **Port** field, select the Toyopuc port to be used by the device.

When you click **OK** to create the port, the Device Properties dialog box opens.

General Device Properties



When you configure a Toyopuc device, the Device Properties dialog for devices using this protocol opens.

Use the General Properties tab to enter general information for the device. You can define the following:

Port	Select the port for this device. You can click the Browser button to the right of the field to display the list of ports and select one. You can click the Pop-up Menu button-to create a new port, edit the current port, or select a port from the list of ports.
Description	Enter an optional description to help you identify the device.
Resource	Enter a resource that can be associated with this device for alarm generation.
Model type	Enter Toyoda PC2
	Use the default properties to enter information about communications for the device. You can define the following:
Address	Enter the device address following the specification in the Device Addresses *(page 950)* section.
CPU	Enter 0
Enable	Set this check box to enable the device when the project starts. If you clear this check box, the device will not be enabled and points associated with the device will be unavailable.

Device Address Specification

The **device address** should have one of the following formats:

For a programmable controller directly connected to the Ethernet network.

< HOSTNAME>/<SOCKET PORT NUMBER>

For a programmable controller that is accessible through another PC2 programmable controller that is connected via Ethernet, the format is:

< HOSTNAME>/<ROUTING INFORMATION>/<SOCKET PORT NUMBER>

Routing for a hierarchy of up to 4 levels is supported. The format of the routing information for a one level hierarchy is:

< LINK NUMBER>**:**<EXCHANGE NUMBER>

On a PC2 that is indirectly accessed via an HPC network, the Exchange Number is the Station Number.

The format of the routing information for a 4 level hierarchy is:

< LINK NUMBER>**:**<EXCHANGE NUMBER>**.**<LINK NUMBER>**:**<EXCHANGE NUMBER>**.**<LINK NUMBER>**:**<EXCHANGE NUMBER>**.**<LINK NUMBER>**:**<EXCHANGE NUMBER>

(The routing information above is on three lines for display purposes. When you are using it, type it on one line.)

Periods (.) separate the routing information for each level of the hierarchy.

Host Name Specification

The hostname for each connection must exist in the Windows file named "hosts".

The Windows host file is located in:

%Systemroot%\System32\drivers\etc

where %Systemroot% is the default location of the system folder. For example, Windows XP, Windows 2003, and later versions typically use \WINDOWS.

The file may be edited using any text editor.

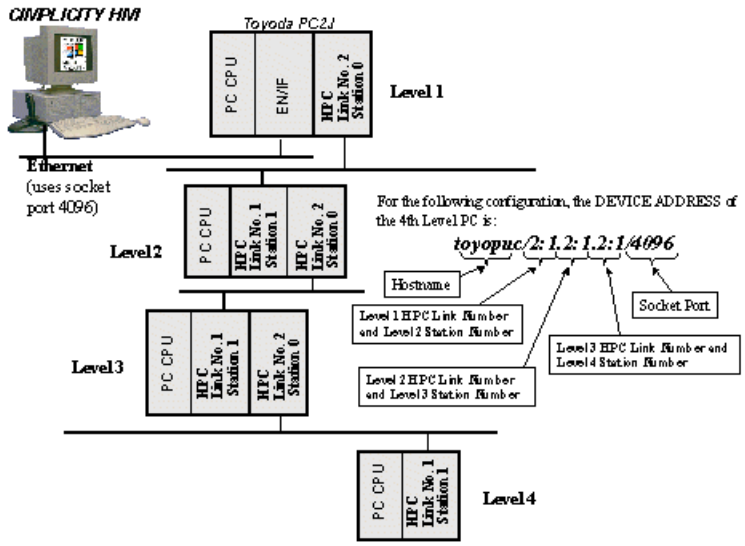
The format of the host files is:

<IP_address> <name> #<comment>

IP addresses are written as xxx.xxx.xxx.xxx, where x is a number from 0 through 255.

Important: : The last line in the hosts file must be left blank, (i.e. the file must end with a blank line) or the last line with data in it will be ignored.

Device Address Specification Example



Toyopuc Point Configuration

Toyopuc Point Configuration

Once your devices are configured, you many configure points for them. Fields in the Point Properties dialog box have configuration values that are unique to or have special meaning to Toyopuc Ethernet communications.

General Point Properties

On the General tab in the Point Properties dialog box, you may configure points for Read or Read/ Write access.

Device Point Properties

On the Device tab in the Point Properties dialog box:.

Update Criteria

The update criteria determine how the data will be collected and/or updated in your CIMPLICITY software application.

- Enter **On Change** or **On Scan** for points whose values should be polled by the Toyopuc Ethernet Communications Driver at regular intervals.
- Enter **On Demand On Scan** or **On Demand On Change** for points whose values should be polled by the Toyopuc Ethernet at regular intervals while users are viewing them.

IMPORTANT: OPC UA servers running on devices, such as PLCs, may not support the #Unsolicited# update criteria. If this is the case the points will not update in the PCP even though it is possible to browse the server from the Device configuration dialog and #Test# button reports success. Changing the update criteria to #On Demand On Poll# or #On Demand On Scan# should allow data collection to start. Note that these update criteria are less efficient and should only be used when there are a small number of points.

Address

Enter the point address. For device points, valid address and ranges are described under Toyopuc Ethernet Communications Supported Memory Types**.**

This communication interface supports the diagnostic points defined in the CIMPLICITY User's Manual under point configuration. In addition, the following diagnostic points may be configured for devices that use this communication interface:

Point Address	Description	Valid Point Types
$CONNECT_STATUS	Shows status of connect **0** = CONNECTED **1** = NO CONNECTION	SINT, USINT
$AVG_CONNECT_TIME	Average amount of time it took to successfully connect to the device in the last **$SAMPLE_CONNECT_SIZE** attempts. (Time unit is in seconds and fractions of seconds.)	REAL
SAMPLE_CONNECT_SIZE	Number of data values used for average connect time calculation	INT, UINT
$HI_CONNECT_TIME	Longer amount of time to connect to device since the communication enabler was started (Time unit is in seconds and fractions of seconds.)	REAL
$LO_CONNECT_TIME	Smallest amount of time to connect to device since the communication enabler was started (Time unit is in seconds and fractions of seconds.)	REAL
$AVG_READ_TIME	Average amount of time it took to successfully read from the device in the last **$SAMPLE_READ_SIZE** attempts. (Time unit is in seconds and fractions of seconds.)	REAL
$SAMPLE_READ_SIZE	Number of data values used for average read time calculation	INT, UINT

$HI_READ_TIME	Longer amount of time to read from device since the communication enabler was started (Time unit is in seconds and fractions of seconds.)	REAL
$LO_READ_TIME	Smallest amount of time to read from the device since the communication enabler was started (Time unit is in seconds and fractions of seconds.)	REAL
$AVG_WRITE_TIME	Average amount of time it took to successfully write to the device in the last **$SAMPLE_WRITE_SIZE** attempts. (Time unit is in seconds and fractions of seconds.)	REAL
$SAMPLE_WRITE_SIZE	Number of data values used for average write time calculation	INT, UINT
$HI_WRITE_TIME	Longer amount of time to write to device since the communication enabler was started (Time unit is in seconds and fractions of seconds.)	REAL
$LO_WRITE_TIME	Smallest amount of time to read from the device since the communication enabler was started (Time unit is in seconds and fractions of seconds.)	REAL

Toyopuc Supported Memory Types

The following elements are accessible from this communication interface.

Name	I/O Address And Maximum Range	Access
Edge Relay	**P000 – P1FF**	Read/Write
Keep Relay	**K000 – K2FF**	Read/Write
Specific Relay	**V00 – VFF**	Read/Write
Timer/Counter	**T000 - T1FF or C000 – C1FF**	Read/Write
Link Relay	**L000 – L7FF**	Read/Write
I/O Relay	**X0000 – X7FF or Y000 – Y7FF**	Read/Write
Internal Relay	**M000 – M7FF**	Read/Write
Specific Register	**S000 – S3FF**	Read/Write
Timer/Counter Current Value	**N000 – N1FF**	Read/Write
Link Register	**R000 – R7FF**	Read/Write
Data Register	**D0000 – D2FFF**	Read/Write
File Register	**B000 – B1FFF**	Read/Write
CPU Status .	**SY0 – SY8**	Read

Toyopuc Global Parameters

You can adjust the timing and performance characteristics of the Toyopuc communications by changing or defining the global parameters listed below.

- <PRCNAM>_CONNECTION_TIMEOUT
- <PRCNAM>_READ_WRITE_TIMEOUT
- <PORT>_BIND_ADDR

<PRCNAM>_CONNECTION_TIMEOUT

For	Toyopuc Project
Purpose	Specify how long the communications enable should wait for a connect operation to complete.
Value	Number of seconds
Recommended	5
Default Value	10

<PRCNAM>_READ_WRITE_TIMEOUT

For	Toyopuc Project
Purpose	Specify how long the communications enable should wait for a read or write operation to complete
Comment	

Value	Number of seconds	
Recommended	3	
Default Value	10	

<PORT>_BIND_ADDR

For	Toyopuc Project
Purpose	To assign the host name that corresponds to the Ethernet card to be used for Toyopuc communications.
Comment	When there are multiple Ethernet cards in the computer and the first connection configured for access by network services is not used, a host name needs to be configured for each card that will by used by a CIMPLICITY Toyopuc port. For each CIMPLICITY Toyopuc port that is using an alternate connection, define a CIMPLICITY project parameter.
Value	Host name that corresponds to the Ethernet card to be used for Toyopuc communications.
Default Value	First card in the TCP/IP binding order is used.