

# AADvance Controller

Catalog Numbers T9110 T9300 T9310 T9401/2 T9431/2 T9451 T9481/2



---

# Important User Information

Read this document and the documents listed in the additional resources section about installation, configuration, and operation of this equipment before you install, configure, operate, or maintain this product. Users are required to familiarize themselves with installation and wiring instructions in addition to requirements of all applicable codes, laws, and standards.

Activities including installation, adjustments, putting into service, use, assembly, disassembly, and maintenance are required to be carried out by suitably trained personnel in accordance with applicable code of practice.

If this equipment is used in a manner not specified by the manufacturer, the protection provided by the equipment may be impaired.

In no event will Rockwell Automation, Inc. be responsible or liable for indirect or consequential damages resulting from the use or application of this equipment.




The examples and diagrams in this manual are included solely for illustrative purposes. Because of the many variables and requirements associated with any particular installation, Rockwell Automation, Inc. cannot assume responsibility or liability for actual use based on the examples and diagrams.

No patent liability is assumed by Rockwell Automation, Inc. with respect to use of information, circuits, equipment, or software described in this manual.

Reproduction of the contents of this manual, in whole or in part, without written permission of Rockwell Automation, Inc., is prohibited

Throughout this manual, when necessary, we use notes to make you aware of safety and other considerations.

---

	<b>WARNING:</b> Identifies information about practices or circumstances that can cause an explosion in a hazardous environment, which may lead to personal injury or death, property damage, or economic loss.
	<b>ATTENTION:</b> Identifies information about practices or circumstances that can lead to personal injury or death, property damage, or economic loss. Attentions help you identify a hazard, avoid a hazard, and recognize the consequence.
	<b>CAUTION:</b> Identifies information about practices or circumstances that can cause property damage or economic loss.
<b>IMPORTANT</b>	Identifies information that is critical for successful application and understanding of the product.
<b>NOTE</b>	Provides key information about the product or service.
<b>TIP</b>	Tips give helpful information about using or setting up the equipment.

---

---

Labels may also be on or inside the equipment to provide specific precautions.

---



**SHOCK HAZARD:** Labels may be on or inside the equipment, for example, a drive or motor, to alert people that dangerous voltage may be present.

---



**BURN HAZARD:** Labels may be on or inside the equipment, for example, a drive or motor, to alert people that surfaces may reach dangerous temperatures.

---



**ARC FLASH HAZARD:** Labels may be on or inside the equipment, for example, a motor control center, to alert people to potential Arc Flash. Arc Flash will cause severe injury or death. Wear proper Personal Protective Equipment (PPE). Follow ALL Regulatory requirements for safe work practices and for Personal Protective Equipment (PPE).

---





### Issue Record

This manual contains new and updated information as indicated in the following table.

Issue	Date	Comments
01	Jan 2009	First Issue
01A	Aug 2009	Release 1.1 Issue
01B	Aug 2009	Updated issue for per review comments
02	Nov 2009	Release 1.1.1
03	July 2010	Update for CRs
04*	Oct 2011	Release 1.2
05	Apr 2012	Updated Release 1.2 version with Analogue Output Module information added.
06	June 2012	Release 1.3 & 1.3.1
07	June 2013	Update to add information to on-line update topics
08	Feb 2015	Update for R1.32
09	April 2015	Final version for WB 1.3, R1.34
H	April 2018	Update and reformat manual for firmware release version 1.40

\* was designated Issue: Release 1.2

### Summary of changes in this Document Issue

Topic	Page
New Front-sheet for Configuration Guide Workbench 1.x	1
Table of Contents - Setting Up a Project and Application is now Chapter 4 (was Chapter 6)	11
Table of Contents - Using the Dictionary is now Chapter 6 (was Chapter 7)	12
Update to sub-section About Discover Communications in Chapter 3	39
Insert new sub-section Online Recovery and Update from Controller to New Computer	54
Addition of reference to Process Safety Time	59
Update to Network Firewall Table 6-AADvance Communication Ports	77
Addition of Note and hyper-link to Knowledgebase Article 605753	143
Change to Connection requirements of a produced or Consumed Tag	165
Change to Example	166
Modification to section Data Types for CIP	169
Modification to section Obtaining the Connection Status for a Consumed Variable	174
Addition of section AADvance Producer RunMode	175
Change to MODBUS message Scheduling	180

Topic	Page
Changes to <a href="#">Table 23 Message Sequence for Example MODBUS Master</a>	182
Changes to <a href="#">Table 38 Controller 1 - Dual Peer-to-Peer Net Control, Network 1, Subnet 1</a>	257
Changes to <a href="#">Table 39 Controller 1 - Dual Peer-to-Peer Net Control, Network 1 Subnet 2</a>	257
Changes to <a href="#">Table 44 Controller 1 and 2</a>	260
Changes to <a href="#">Table 45 Controller 2 and 3</a>	260
Changes to <a href="#">Table 46Controller 4</a>	260 & 261
Update to references for other publications in the document suite	263

In no event will Rockwell Automation be responsible or liable for indirect or consequential damages resulting from the use or application of this equipment. The examples given in this manual are included solely for illustrative purposes. Because of the many variables and requirements related to any particular installation, Rockwell Automation does not assume responsibility or reliability for actual use based on the examples and diagrams.

No patent liability is assumed by Rockwell Automation, with respect to use of information, circuits, equipment, or software described in this manual.

All trademarks are acknowledged.

### **DISCLAIMER**

It is not intended that the information in this publication covers every possible detail about the construction, operation, or maintenance of a control system installation. You should also refer to your own local (or supplied) system safety manual, installation and operator/maintenance manuals.

### **REVISION AND UPDATING POLICY**

This document is based on information available at the time of its publication. The document contents are subject to change from time to time. The latest versions of the manuals are available at the Rockwell Automation Literature Library under "Product Information" information "Critical Process Control & Safety Systems".

### **DOWNLOADS**

The product compatibility and download center is

[www.rockwellautomation.com/rockwellautomation/support/pcdc.page?](http://www.rockwellautomation.com/rockwellautomation/support/pcdc.page?)

Select the Find Downloads option under Download

In the Product Search field enter "AADvance" and the AADvance option is displayed.

Double click on the AADvance option and the latest version is shown.

Select the latest version and download the latest version.

### **AADVANCE RELEASE**

This technical manual applies to AADvance Release: 1.40

## LATEST PRODUCT INFORMATION

For the latest information about this product review the Product Notifications and Technical Notes issued by technical support. Product Notifications and product support are available at the Rockwell Automation Support Center at

<http://rockwellautomation.custhelp.com>

At the Search Knowledgebase tab select the option "By Product" then scroll down and select the ICS Triplex product AADvance.

Some of the Answer ID's in the Knowledge Base require a TechConnect Support Contract. For more information about TechConnect Support Contract Access Level and Features please click on the following link:

[https://rockwellautomation.custhelp.com/app/answers/detail/a\\_id/898272](https://rockwellautomation.custhelp.com/app/answers/detail/a_id/898272)

This will get you to the login page where you must enter your login details.

---

**IMPORTANT** A login is required to access the link. If you do not have an account then you can create one using the "Sign Up" link at the top right of the web page.

---

## PURPOSE OF THIS MANUAL

Starting with a new panel and a software distribution DVD, this manual explains how to install the AADvance Workbench. This manual defines the process you must follow to configure your system. It includes background information and step-by-step instructions for the following:

- Installing the software and setting up the software licensing
- Setting the controller IP Address and achieving communications with the controller
- Defining the processor configuration
- Defining the application variables
- Configuring the I/O modules and channels
- Setting up MODBUS, CIP and SNCP

The instructions are based on using Release 1.4 of the AADvance Workbench under the Windows XP operating system; other Windows operating systems are supported - refer to the topic Chapter 2 "Software Installation" for a full list of supported Windows operating systems.

This manual includes reference information about module status parameters and I/O variables, to help you make a decision about which variable types to use and full descriptions of the data values supplied by the I/O modules.

- If the information in this document does not agree with the applicable project codes and standards, the system integrator must find a solution for the mismatch.

- If AADvance is used for a safety function, the system integrator must apply the requirements specified in the AADvance Safety Manual.

## **WHO SHOULD USE MANUAL**

The data in this manual is written for system integrators who know about building and setting up new systems. The system integrator must make sure that the system complies with the local, national and international standards for the application that AADvance is being used for.

### **Environmental compliance**

Rockwell Automation maintains current product environmental information on its website at:

<http://www.rockwellautomation.com/rockwellautomation/about-us/sustainability-ethics/product-environmental-compliance.page>



## **Chapter 1**

### **Introduction**

AADvance Workbench and Software Development Environment.	17
Operating Systems .....	18
About The Configuration Process .....	18
Integrating the AADvance Controller with Other Systems .....	20
Application Scan Model .....	21
Scan Times.....	21

## **Chapter 2**

### **Software Installation**

AADvance Workbench Licensing Options.....	25
Install the AADvance Workbench and Utilities.....	25
Add and Activate a New AADvance Workbench License.....	28
Updating or Upgrading an Existing License .....	31
Update or Upgrade a Hardware/Software License Key.....	32
Using a Floating License Server .....	34
Set Up a Server for Hardware Floating Licenses .....	34
Set Access to Floating Licenses.....	36

## **Chapter 3**

### **Connecting the Workbench to the Controller**

Setting the Controller IP Address for AADvance Workbench ....	39
Controller IP Address.....	39
About Discover Communications .....	39
AADvance Discover Utility .....	40
Configure the Controller Resource Number in the Controller	42
Configure the IP Address in the Controller.....	43

## **Chapter 4**

### **Setting Up a Project and Application**

Create a New Project .....	49
Configure Controller Type (Standard or Eurocard) .....	49
Compiler Verification Tool .....	50
Enable the Compiler Verification Tool (CVT).....	51
Back-up and Restore a System Configuration .....	53
Back-up a repository project .....	53
Restore a repository project.....	53
Online Recovery and Update from Controller to New Computer .	54
Allocate IP Addresses for Network Communications .....	56
Configure the IP Address of the Target Controller .....	57

	<b>Chapter 5</b>	
<b>Configuring the Processor Modules</b>	Configure the Top-level Process Safety Time (PST).....	61
	Configure the Processor Battery Alarm.....	62
	View Module Firmware Versions .....	62
	Processor Firmware Upgrades .....	65
	Configure the Serial Ports.....	66
	Serial Port Protocols .....	66
	Serial Port Parameters.....	66
	Time Synchronization (SNTP).....	67
	Configure the Controller as an SNTP Client .....	67
	Configure the Controller as an SNTP Server.....	68
	Using the Controller as a MODBUS Slave .....	69
	Support for MODBUS Slave Exceptions .....	69
	Configure the Controller MODBUS Slaves .....	70
	MODBUS Slave Communication Parameters .....	71
	Transparent Communication Interface (TCI).....	72
	TCI Configuration .....	72
	Differential Services (DiffServ) .....	73
	Configure DiffServ .....	75
	Network Firewall.....	76
	Ethernet Forwarding .....	79
	Configure Ethernet Forwarding .....	80
	About T9110 Processor Variables .....	82
	Wire Processor Variables .....	83
	Unwire Processor Variables.....	84
	Status Integers.....	84
	Control Integers.....	85
	Status Booleans.....	86
	Control Booleans.....	93
	RTC Status Variables .....	94
	RTC Program Variables.....	96
	RTC Control Variables .....	98
	Set the Processor Clock .....	103
	<b>Chapter 6</b>	
<b>Using the Dictionary</b>	About the Dictionary.....	105
	Properties for AADvance Variables .....	105
	Create or Modify Variables in the Dictionary .....	107
	Edit the Contents of a Cell in the Dictionary .....	108
	Edit the Contents of a Row in the Dictionary .....	110
	SOE Service Parameters .....	111
	<b>Chapter 7</b>	
<b>Configuring the Controller I/O</b>	About Configuring I/O Modules.....	113



Defining the I/O Hardware Architecture.....	114
Example Controller Configuration.....	114
Assign I/O Modules to I/O Bus Slots .....	116
Change the I/O Configuration with an On-line Update .....	120
Configure the I/O Module Process Safety Time.....	123
Wire Status Variables to I/O Modules.....	123
T9K_TA_GROUP_STATUS (I/O Module Status Information)	124
About Configuring I/O Channels .....	125
Wire Variables to Digital Input Channels .....	126
Wire Variables to Analogue Input Channels .....	127
Wire Variables to Digital Output Channels.....	128
Wire Variables to Analogue Output Channels .....	129
Configuring Digital Inputs.....	129
T9K_DI_COMPACT and T9K_DI_FULL (Digital Inputs) ...	130
Faulted State for Digital Inputs .....	131
Threshold Values for Digital Inputs .....	131
Configuring Analogue Inputs .....	134
T9K_AI_COMPACT and T9K_AI_FULL (Analogue Inputs)..	135
Faulted State for Analogue Inputs.....	136
Threshold Values for Analogue Inputs .....	136
Configuring Digital Outputs.....	139
T9K_DO_COMPACT and T9K_DO_FULL (Digital Outputs)	140
The State Variable for Digital Outputs .....	141
Overcurrent Protection for Digital Outputs .....	142
Faulted State for Digital Outputs .....	143
Configure Advanced Channel Settings for Digital Outputs ..	143
Configuring Analogue Outputs .....	145
The State Variable for Analogue Outputs.....	145
T9K_AO_COMPACT and T9K_AO_FULL (Analogue	
Outputs) .....	146
Faulted State for Analogue Outputs.....	147
Configure Advanced Channel Settings for Analogue Output	
Channels .....	147
Status Variables for Digital Output Modules .....	148
Wire Status Variables to a Digital Output Module .....	149
Unwire Status Variables from a Digital Output Module .....	149
Status Booleans.....	150
Field Power Status Integers.....	151
HART .....	152
HART Features .....	153
HART .....	153
Precautions for HART in a Safety System .....	154
Configure HART for Field Device Monitoring .....	154
T9K_AI_HART and T9K_AI_HART_FULL.....	155

HART Pass-Through.....	157
Using HART Pass-Through.....	157
HART Pass-Through Features .....	158
Precautions for HART Pass-Through in a Safety System.....	158
Install the 9033 DTM .....	158
Enable HART Pass-Through in the Controller .....	159
Configure HART for Pass-Through Communication Monitoring .....	160
Use FactoryTalk Asset Center with an Analogue Input Module .....	161

## Chapter 8

### Using CIP over EtherNet/IP

CIP over EtherNet/IP.....	165
Connection Requirements of a Produced or Consumed Tag.....	165
Example:.....	166
Memory Considerations.....	166
Define a CIP Network.....	167
Data Types for CIP.....	169
Configure an AADvance Variable as a Producer .....	170
Defining an AADvance Controller as a CIP Producer.....	170
Using the Dictionary with CIP.....	171
CIP in the Dictionary View .....	171
Parameters for CIP Producer and Consumer Variables .....	171
Configure an AADvance Variable as a Consumer .....	172
Obtaining the Connection Status for a Consumed Variable.....	174
AADvance Producer RunMode .....	175
CIP within the Application Scan Cycle .....	176
About the RSLogix 5000 Configuration.....	176
Set the RSLogix UNICAST Configuration.....	176
Further Information on CIP over Ethernet/IP.....	176

## Chapter 9

### Configuring MODBUS Master

MODBUS Master .....	177
MODBUS Standards.....	177
MODBUS Master Hardware and Physical Connections.....	178
MODBUS Master Command Set .....	179
MODBUS Data Types and Addressing .....	179
MODBUS Message Scheduling .....	180
Handling MODBUS Communication Errors.....	182
MODBUS Statistics.....	182
MODBUS Service Parameters.....	183
Diagnosing MODBUS Communications and Slave Devices .....	183
MODBUS Exception Responses .....	183
AADvance Objects for MODBUS Master.....	184
MODBUS Master Capacities .....	184

Planning for MODBUS Master .....	185
Physical Connections for MODBUS RTU .....	185
Connect a Slave Device, Full Duplex .....	186
Connect Multiple Slave Devices, Full Duplex .....	187
Connect a Slave Device, Half Duplex .....	189
Connect Multiple Slave Devices, Half Duplex .....	190
Configure the Serial Ports for MODBUS Master .....	191
Serial Port Parameters .....	192
Physical Connections for MODBUS TCP .....	192
Setting Up the Project for MODBUS Master Operation .....	192
About the MODBUS Master Bus .....	193
Insert the MODBUS Master Bus .....	193
Create a MODBUS Master Object .....	194
MODBUS Master Communication and Control Settings ...	195
Configure a MODBUS Master Object for MODBUS RTU .	196
Configure a MODBUS Master Object for MODBUS TCP..	197
Controlling a MODBUS Master Object .....	200
MODBUS Ping Mode, Interval and Address .....	200
Configure Statistics for a MODBUS Master Object .....	207
Create Links to MODBUS Slaves .....	208
MODBUS Slave Link Identification and Control Settings ...	209
Configure a MODBUS Slave Link for MODBUS RTU .....	210
Configure a MODBUS Slave Link for MODBUS TCP .....	212
Choosing Names for MODBUS Objects .....	213
MODBUS Slave ID .....	213
MODBUS Slave Wait Interval .....	213
MODBUS Ping Mode, Interval and Address .....	213
MODBUS Slave Commands .....	214
Serial Port .....	214
Ethernet and Serial Port .....	214
MODBUS Slave Link Control and Status Registers .....	216
MODBUS Slave Link Control Register .....	216
MODBUS Slave Link Status Register .....	217
Add Messages for a MODBUS Slave .....	217
MODBUS Slave Link Message Parameters .....	218
Controlling a MODBUS Message .....	219
Configure Statistics for a MODBUS Slave Link .....	220
MODBUS Statistics Parameters .....	220
Remove a Slave Link .....	221
Remove all Slave Links .....	221
Remove a MODBUS Master Object .....	222

## Chapter 10

### SNCP

Bindings and the SNCP Network .....	223
SNCP Networks .....	224
Set Up Multiple SNCP Networks .....	226

SNCP Link Properties .....	229
Configure Bindings .....	232
SNCP Binding Error Variables .....	233

## **Chapter 11**

### **Peer-to-Peer Network**

Peer-to-Peer Features .....	235
Peer-to-Peer Configuration Process .....	235
Create a Peer-to-Peer Network .....	236
Peer-to-Peer Subnet Controller Configuration .....	238
Set up the Peer IP Addresses and Status Variable .....	241
Peer-to-Peer Data Boards .....	241
Configure Input Boards .....	242
Configure Analogue Input Boards .....	243
Configure Digital Input Boards .....	247
Configure Output Boards .....	250
Configure Analogue Output Boards .....	251
Configure Digital Output Boards .....	253
Peer-to-Peer Configuration Example 1 .....	255
Peer-to-Peer Controller Setting Summary .....	256
Peer-to-Peer Data Summary .....	260

## **Chapter 12**

### **Additional Resources**

Associated AADvance Publications .....	263
Regional Offices .....	264

<b>Glossary .....</b>	<b>265</b>
-----------------------	------------

## Introduction

This chapter provides a brief overview of the AADvance Workbench and this manual.

### AADvance Workbench and Software Development Environment

The Engineering Workstation runs the AADvance Workbench developed specifically for the AADvance system that enables you to configure your system, design the complete control strategy as one, then to target parts of the strategy at each controller. Interaction between the resources is automatic, significantly reducing the complexity of configuration in a multi-resource solution.

The workstation software is compliant with IEC 61131 industrial standard and has the following powerful features:

- the regulation of the flow of control decisions for an interacting distributed control system
- providing for the consistency of data
- providing a means for synchronous operation between devices
- eliminating the need to have separate synchronous schemes
- easing the development and maintenance of robust systems

The Workbench is a software development environment for a controller. It lets you create local and distributed control applications using the five languages of IEC 61131-3. Engineers can use one language or a combination that best suits their knowledge and programming style and the type of application.

It is a secure development environment that requires a hardware or software license to run on a PC. There is also a Program Enable key (not applicable to a Euro Controller) that must be plugged into the processor base unit to allow the user to modify and download the application resource or access the AADvance Discover tool to set or change the controller IP address. The Program Enable key when it is removed protects the application from unauthorized access.

The development environment includes:

- tools for program development
- program documentation
- function block library management
- application archiving
- database configuration
- import/export utilities

- on-line monitoring
- off-line simulation and controlled on-line changes.

Programs can be simulated and tested on the computer before downloading to the controller hardware. Also provided is a set of configuration tools that enables you to define the hardware architecture in the software; set up the processor functionality; and connect application variables to the Workbench application resource program that will monitor processor and I/O module status information and report I/O channel data values to the Workbench. Resource Control applications can be distributed across several hardware platforms, communicating with each other through secure networks.

---

**IMPORTANT** Apply the safety application guidelines contained in the AADvance Safety Manual to use the Workbench for a safety related application.

---

## Operating Systems

The workstation operating requirements for this application development software is as follows:

### Operating Systems:

- Windows XP with Service Pack 3



**CAUTION:** Do not use XP Professional x64 Edition.

---

- Windows Vista, Windows 7 & Server 2003 in both 32-bit and 64-bit versions

---

**IMPORTANT** Network Licensing - Windows 64-bit version will only work with the USB licensing method. Windows 64-bit will not recognize Workbench software licensing.

---

- Network port (10/100 Base T Ethernet)
- Access to a CD-ROM drive, for software installation.

If the application adopts the dongle licensing option for the software, the workstation will also require one free USB port.

## About The Configuration Process

The process begins by creating a project and allocating the IP addresses for communication with the AADvance controllers. You can then configure the network communications parameters for the project.

You then define the hardware set up for your controller. During this process you assign I/O modules to slot numbers on the processor bus. There are two

I/O Busses and each can be assigned up to 24 I/O modules and you can assign the modules in single slots or in adjacent slots as groups of two or three.

---

**IMPORTANT** If you change the physical arrangement of the hardware after you have configured a controller using the AADvance Workbench, you must change the AADvance Workbench configuration to match the changed hardware arrangement.

From Release 1.3 you can change the I/O module configuration with an on-line update and in some cases without having to stop the data flow. Where an on-line update stops the data flow refer to the Safety Manual if it is a safety related application. However, if you are still using an earlier product release the I/O module configuration cannot be changed with an on-line update.

---

You now define your module status parameters and the I/O channel variables and their properties in the project Dictionary. The AADvance Workbench provides you with a wide range of variables types to choose from including a set of structured variables. Set up enough variables to cover all the I/O points and module status variables for your controller architecture. You can add new variables at any time during configuration of a system and the AADvance Workbench or after reconfiguration.

You allocate tag names to the variables you want to use. If you chose structured variables for I/O channels, the AADvance Workbench will add a tag name and automatically generates a set of additional variable elements with the same tag name for each element type.

In the next stage of the process you define the T9110 processor module functionality and set up connections to a group of processor module status parameters. Here you will enter values for functions such as the serial port settings, process safety time, and SNTP and MODBUS services.

The AADvance Workbench provides pre-defined I/O module status parameters for each module to which you assign application variables.

Finally you connect (wire) each I/O channel to structured variables. These structured variables report the channel status and define output data values.

You define hardware redundancy in the AADvance Workbench when you define the hardware configuration. During the allocation of I/O modules to empty slots, you are presented with the option to add two or three modules. When you choose the two or three option the AADvance Workbench automatically allocates the modules to a group of adjacent slots. The AADvance Workbench then only allows you to configure one set of I/O channels to the group. You do not need to define redundancy for the processors. The AADvance Workbench automatically connects to all three processors after their IP addresses have been set up in the AADvance Workbench.

## Integrating the AADvance Controller with Other Systems

The AADvance controller connects to existing control systems and plant monitoring equipment. This connection enables a third-party control system to read the state of sensors connected to the controller.

The connection interfaces are through the controller network ports and serial ports, and use the following protocols:

- CIP over Ethernet/IP
- MODBUS RTU
- Open MODBUS/TCP
- OLE for Process Control (OPC).

---

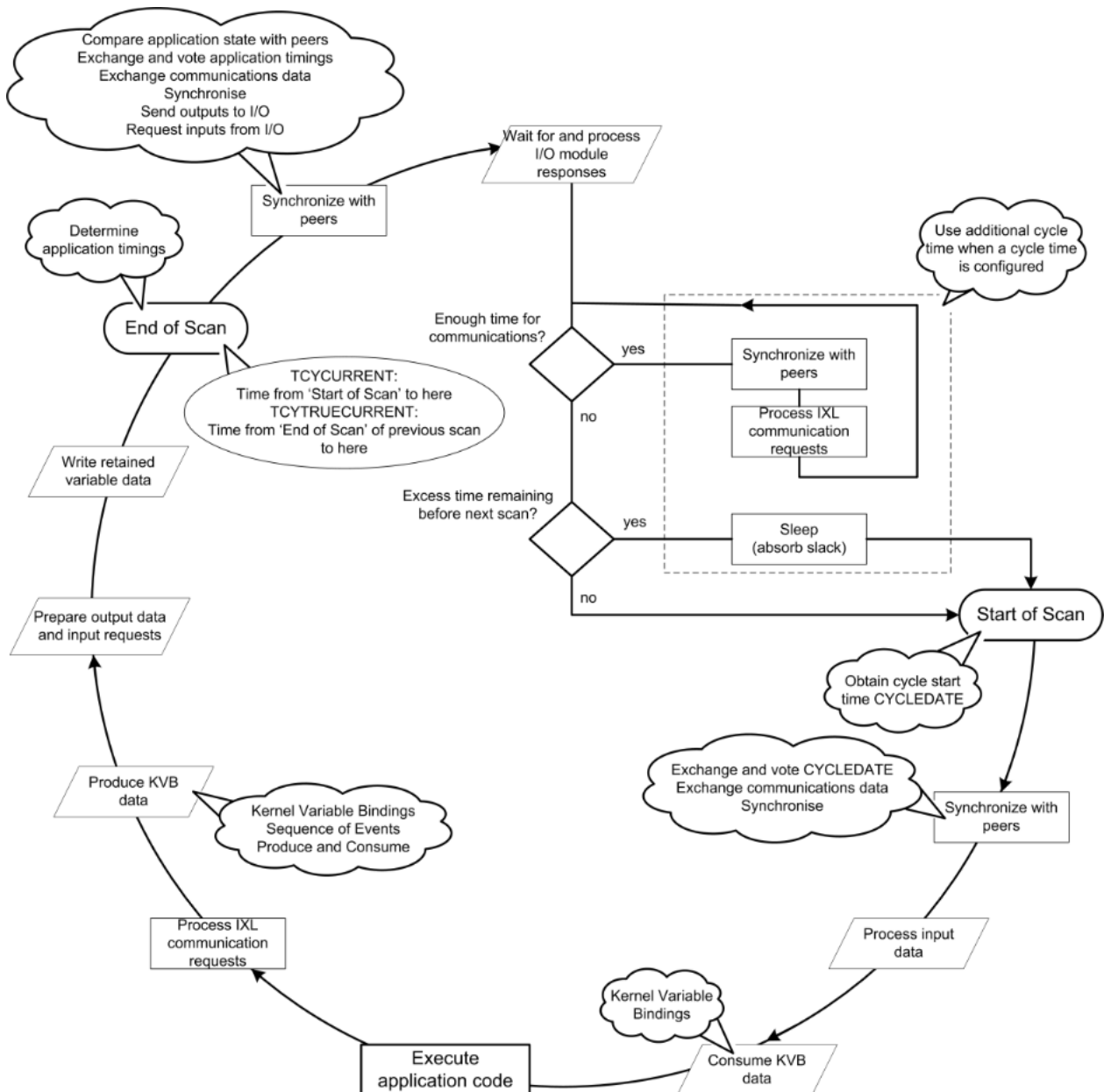
**IMPORTANT** To ensure the integrity of the AADvance system activating network firewalls and Windows operating system firewalls is strongly advised.

---



## Application Scan Model

The application scan model for the AADvance controller is shown in the following illustration:



## Scan Times

The controller processing scan times listed in the table are taken from a test system which used only production modules. The tests which were used to measure the scan times did not measure the effects of logic complexity and communications loading.

**Table 1 - Typical Module Scan Times**

Module		Scan Time
9402	Digital input module, 24 Vdc, 16 channel	
	Simplex	0.924 ms
	Dual	1.676 ms
	Triple	2.453 ms
9432	Analogue input module 24 Vdc, 16 channel	
	Simplex	1.170 ms
	Dual	1.965 ms
	Triple	2.656 ms
9451	Digital output module, 24 Vdc, 8 channel	
	Simplex	1.174 ms
	Dual	2.202 ms
9482	Analogue output module, 24 Vdc, 8 channel	
	Simplex	0.981 ms
	Dual	1.761 ms
Minimum cycle time overhead <sup>1</sup>		39.3 ms
Scan overhead for each module		0.04 ms

<sup>1</sup> The minimum overhead to the cycle time is a feature of the AADvance Workbench.

The Scan time is:

$$\begin{aligned}
 \text{Scan time} &= 39.3 \text{ ms} \\
 &+ \text{Sync time} \\
 &+ \text{Total number of modules} * 0.04 \text{ ms} \\
 &+ ? (\text{Number of module groups} \times \text{scan time shown above})
 \end{aligned}$$

Where:

Sync time is a function of the total number of modules defined according to the following:

0..10 modules	20 ms
11..20 modules	22 ms
21..30 modules	24 ms
31..40 modules	27 ms
41..48 modules	32 ms.

Though the average scan time will be within 1 ms of the Scan time calculated above the calculation does not take into account the effects of application logic

and network communication, and individual scans can vary by up to +/- 4 ms around the average scan time.

Throughput time is the time from input change to output action. For asynchronous inputs the throughput times can be derived from the Scan time calculated above according to the following formulae:

- Minimum throughput time = Scan period + 7 ms
- Maximum throughput time = 2 x Scan time + 13 ms.

**An example configuration scan time:**

- System configuration includes T9432 Analogue input simplex modules x 30 and T9451 Digital output simplex modules x 18.
- Total I/O modules = 48
- Sync time = 32 ms
- Scan time = 39.3 ms + 32 ms + (48 x 0.04) ms + (30 x 1.170) ms + (18 x 1.174) ms => 129.5 ms
- Minimum throughput time = 129.5 ms + 7 ms => 136.5 ms
- Maximum throughput time = (2 x 129.5) ms + 13 ms = 272.0 ms.



# Software Installation

This chapter provides the instructions to install and license the AADvance Workbench.

## AADvance Workbench Licensing Options

You can use the AADvance Workbench for a trial period of 30 days with a promotional license. To use a fully operational version of the AADvance Workbench you must purchase a license from Rockwell Automation.

License keys come in two forms:

- T9082/3U Single User Hardware License: a hardware license key is a dongle that is delivered with the software. To operate the license insert the dongle into the USB port. This hardware key allows you to move the license to another PC, but only the PC with the USB Dongle installed allows the Workbench to be started.
- T9082/3D Single User Software License: a software license key (hard disk key) is obtained and activated through the AADvance License Manager. This establishes the license on a PC; but only the PC with the software key activated allows the Workbench to be started.

When you purchase a single user software license you can select from the following:

- T9082 Multiscan (PRS): Single user, single controller license
- T9083 Distributed (PRD): Single user, multiple controllers' license.

Network licenses are also available:

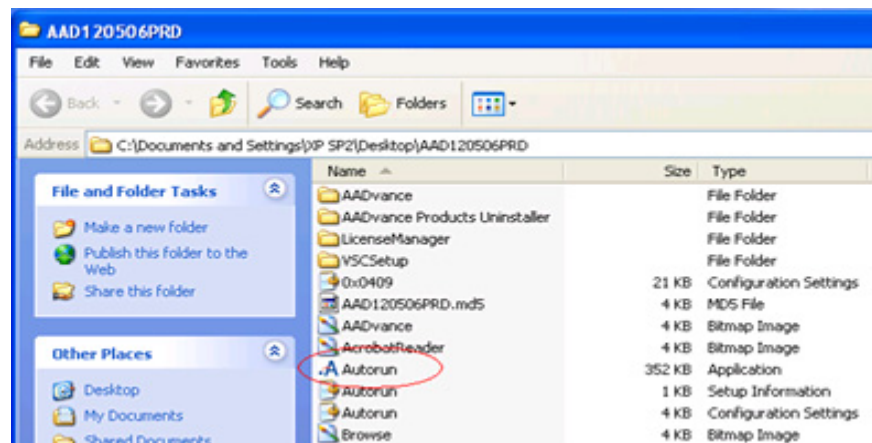
- T9084U Network User License: A network license (USB dongle) allows the users to license copies of the AADvance Workbench on several PC's so long as they have a continuous network connection to a license server. As it is a USB license option and will work with Windows XP Service pack 3; Windows Vista; Windows 7 and server 2003 in both 32-bit and 64-bit versions.

## Install the AADvance Workbench and Utilities

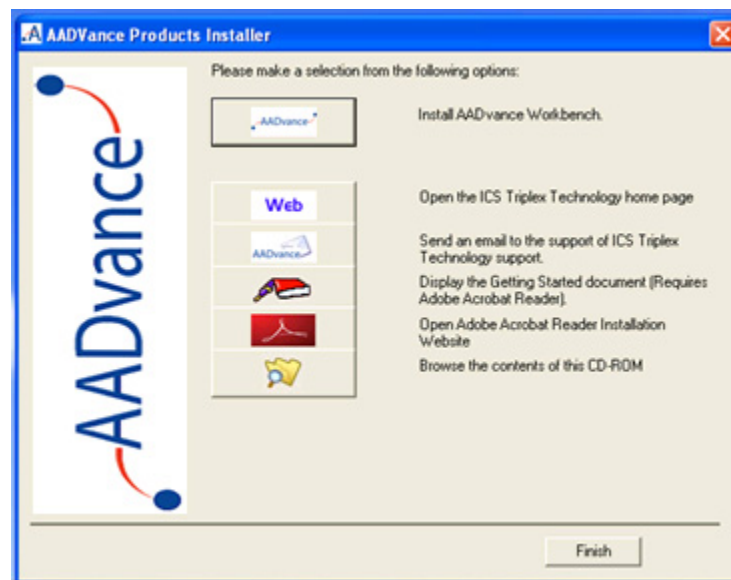
When installing a single user license you do not need to install the License Manager separately unless you are running a network installation; just run the License Manager to activate or enable a license.

## Installing Workbench Part Numbers: T9082U, T9082D, T9083U and T9083D

- NOTE**
- You require "Admin" rights on the PC the Workbench is to be installed on.
  - The Install CD has an Autorun application, so simply inserting the CD will launch the AADvance Products Installer.
  - If however the Workbench is being installed using a downloaded CD image, to launch the AADvance Products Installer you must run the Autorun application from the following folder.



To install the Workbench and its Utilities do the following:

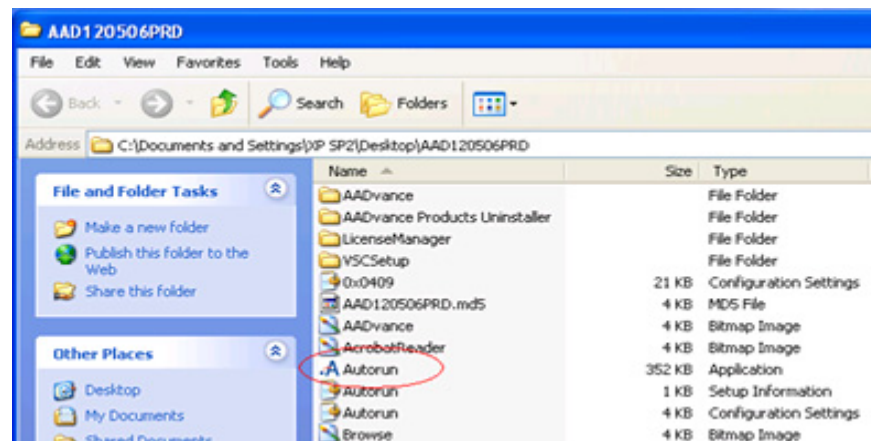


1. Click on the **Install AADvance Workbench** button.
2. Accept the License Agreement and the installation will commence.

- The default is to install both the Workbench and the License Manager components; we recommend proceeding with the defaults.
- When the install commences you will be given the option of changing the location for the install, we recommend you accept the default settings.
- During the install you will be asked if you want shortcuts placed on your desktop, simply answer yes or no depending on your preference.
- When the installation is complete you will be prompted to restart the computer.

### Installing Workbench Part Numbers T9084U Network License

- NOTE**
- You require "Admin" rights on the PC the Workbench is to be installed on.
  - The Install CD has an Autorun application, so simply inserting the CD will launch the AADvance Products Installer.
  - If however the Workbench is being installed using a downloaded CD image, to launch the AADvance Products Installer you must run the Autorun application from the following folder.



- To use Network based licensing, you have to install the Workbench and the License Manager on any PC's you wish to run the Workbench on and you have to install the License Manager on the PC or Server that will act as the network License Server.

#### 1. Click on the **Install AADvance Workbench** button.

- You must accept the License Agreement before the installation will commence.
- The default is to install both the Workbench and the License Manager components; we recommend proceeding with the defaults.
- When the install commences you will be given the option of changing the location for the install, we recommend you accept the default settings.

- During the install you will be asked if you want shortcuts placed on your desktop, simply answer yes or no depending on your preference.
- When the installation is complete you will be prompted to restart the computer, this must be done prior to using the Workbench.

### Install the License Manager on the Network License Server

1. Click on the **Install AADvance Workbench** button at the top of the list of the AADvance Products Installer.
  - You must accept the License Agreement before the installation will commence.
2. Select the License Manager Only.

---

**NOTE** The License Server can also function as a Workbench, if this is the intended method of operation proceed with the defaults.

---

- When the install commences you will be given the option of changing the location for the install, we recommend you accept the default settings.
- During the install you will be asked if you want shortcuts placed on your desktop, simply answer yes or no depending on your preference.
- When the installation is complete you will be prompted to restart the computer, this must be done prior to using the Workbench.

## Add and Activate a New AADvance Workbench License

You use the following AADvance License Manager window to add and activate a new Software (Disk Based) AADvance Workbench licenses. USB licenses are detected automatically.

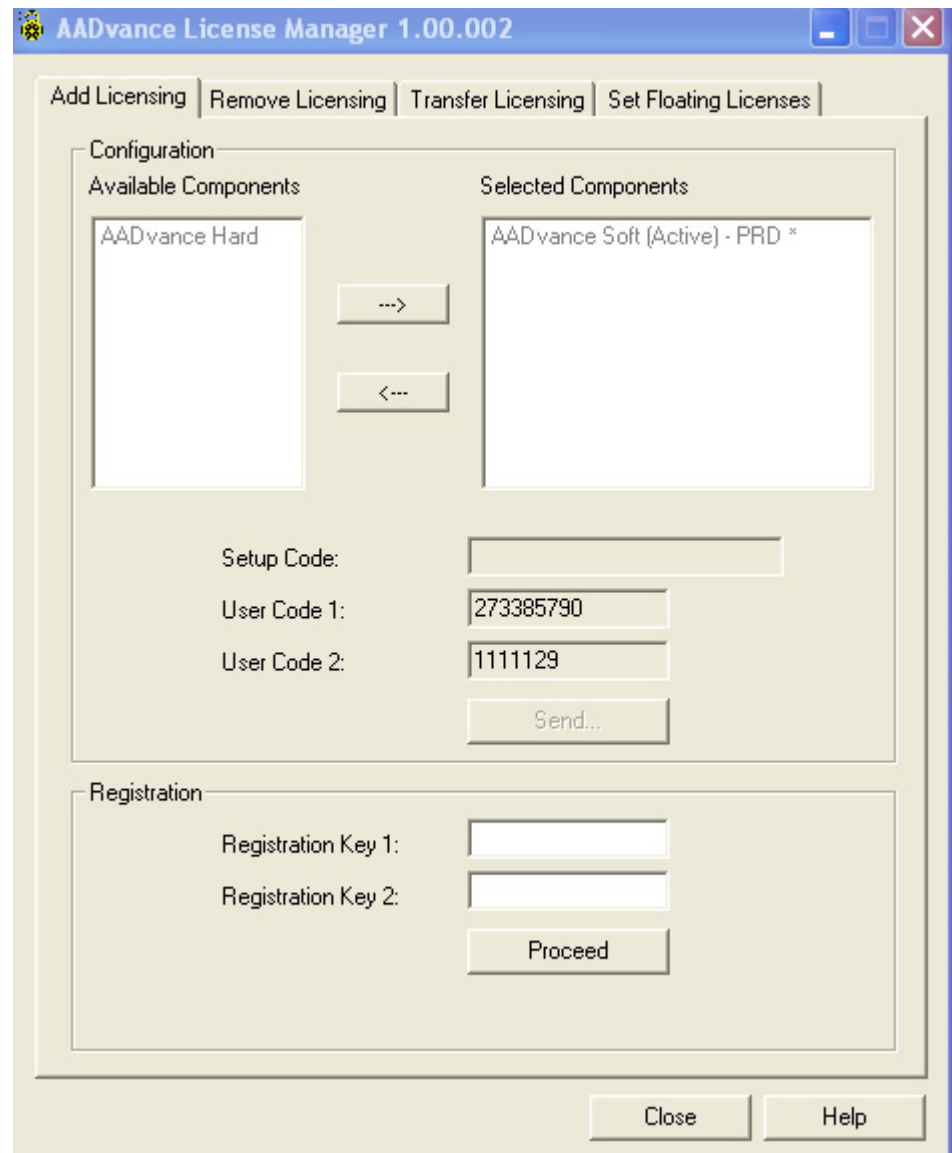
---

**NOTE** You need only one set of user codes and registration keys to activate the license key, irrespective of the feature set or number of licenses ordered.

---

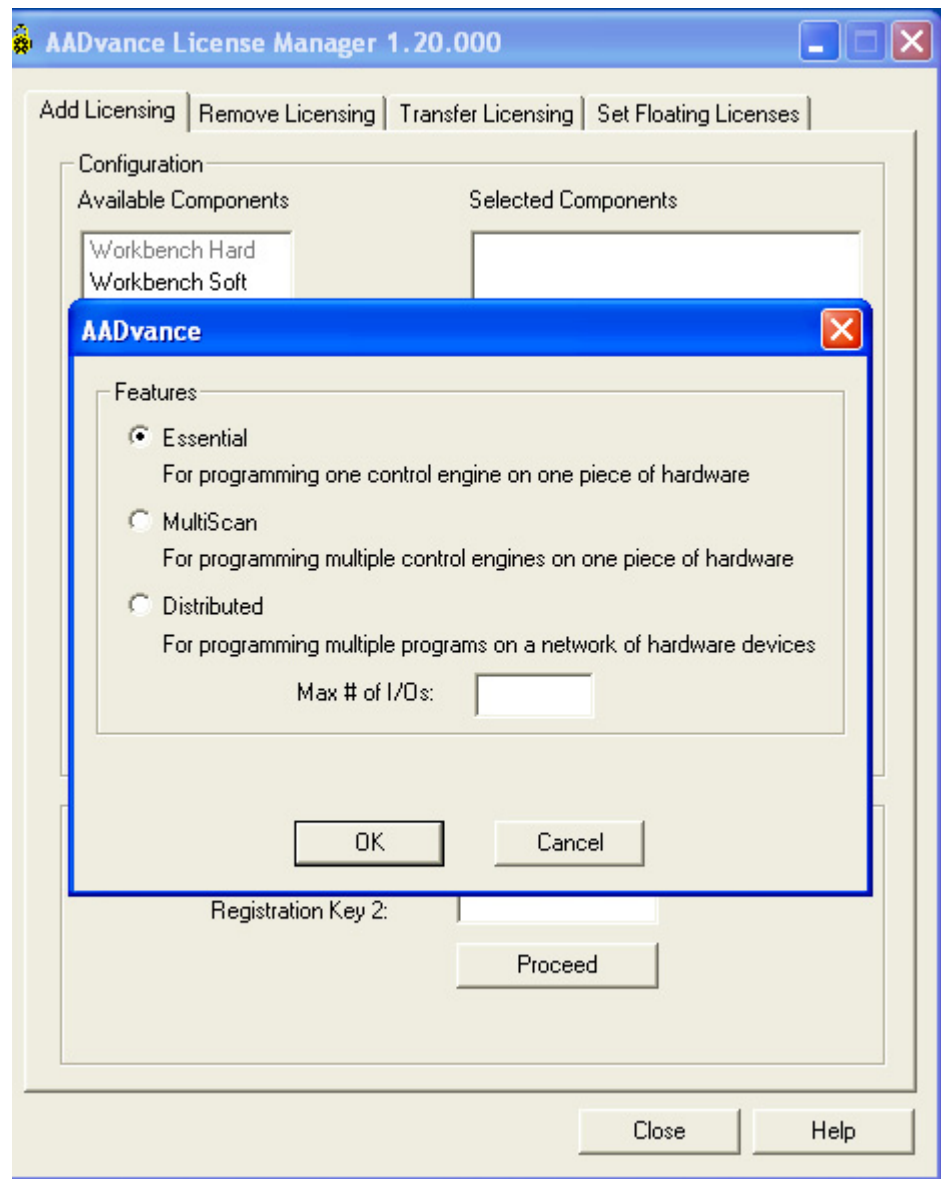


To add a new license do the following:



1. From the Start menu select **AADvance ➔ Licensing AADvance**.
  - The AADvance License Manager dialog box opens.
2. Select the **Add Licensing** tab.
3. Select an AADvance Soft component from the Available Components, then click the ➔ button to move the selection to the Selected Components window.

4. The AADvance license Features options appears.



5. Select the license feature and leave the Max # of I/Os blank. When you purchase a single user license you can choose from the following feature sets:
  - T9082 Multiscan (PRS): Single user, single controller license
  - T9083 Distributed (PRD): Single user, Multiple Controllers license.
6. A Setup Code and two User Codes appear in their respective fields.
7. Click **Send**.
  - A License Manager window opens telling you to register by email, click **Yes**.

8. At this point your email client may ask your permission to access an external email address, click the **Yes** button
  - An addressed email form opens containing the following:
    - the selected components details
    - setup code and both user codes.
9. Enter the contact information and the purchase order number into the email form.
10. Send the email.

---

**NOTE** If the computer does not have an email client configured, copy the text and send from another computer.

---

- The original Setup Codes and User Codes together with the new Registration Keys are returned to you in an email.
11. Check that the Setup Code and User Codes are the same as the original ones.
  12. Enter the Registration Keys into their respective fields, click **Proceed**.
    - The selected components appear greyed out in the Selected Components list.
  13. Restart AADvance.

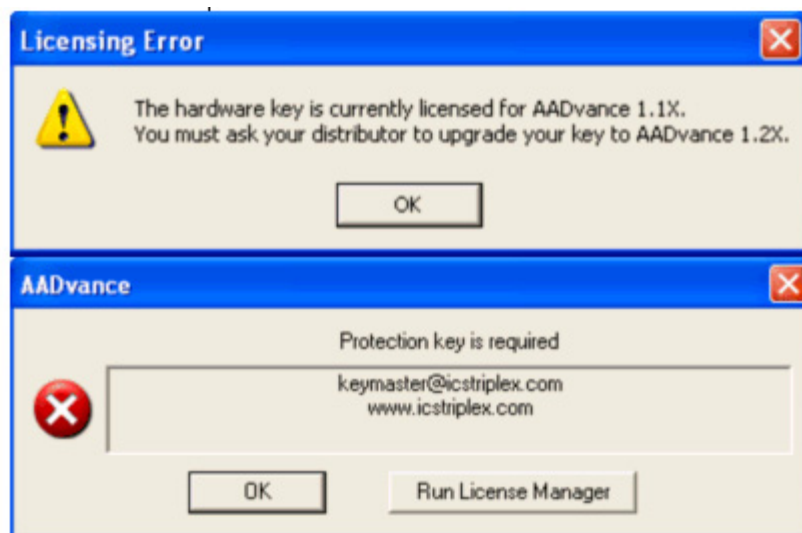
## Updating or Upgrading an Existing License

You can update your current license by moving to a newer version or upgrading your license by changing your feature set.

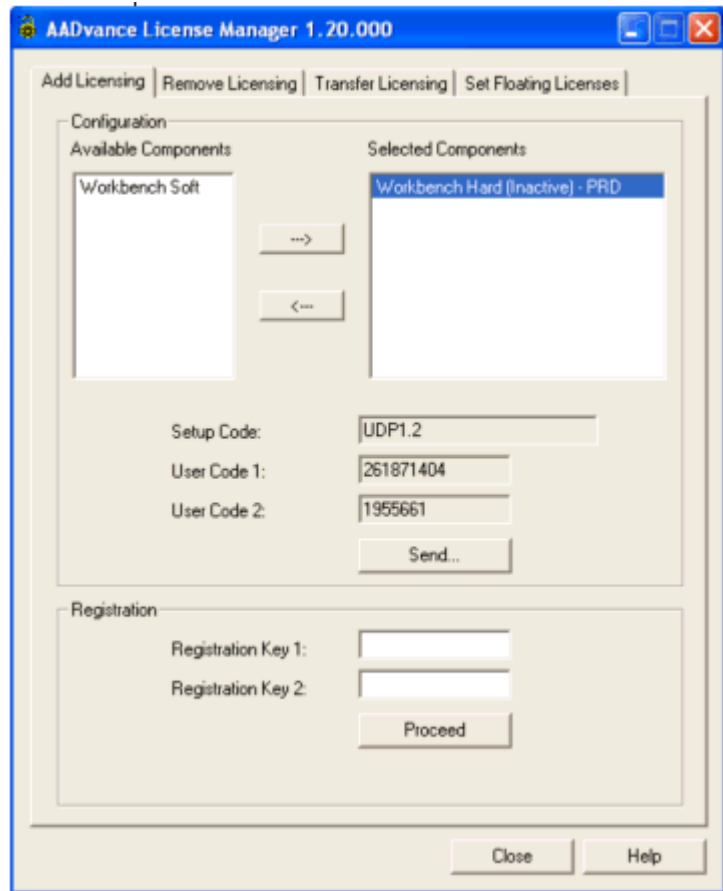
- If you are using a hardware or software license you can update or upgrade it using the AADvance License Manager.

## Update or Upgrade a Hardware/Software License Key

After upgrading the Workbench to Release 1.2 and opening it for the first time, a message box will appear showing that the license needs to be upgraded.



1. From the Start menu select **AADvance → Licensing AADvance**.
  - The AADvance License Manager dialog box opens with the UPP1.2 Setup Code for a software license and the UDP1.2 Setup Code for a Hardware license and the User Codes 1 and 2.



2. Send the User Codes by email to [keymaster@ra.rockwell.com](mailto:keymaster@ra.rockwell.com).
3. If you are upgrading a hardware license key, leave it fitted.
  - You will be sent the Registration Keys 1 and 2.
4. Enter these keys in the two boxes and click **Proceed**.
  - Your license will be upgraded.

### Multi-User Floating Hardware License

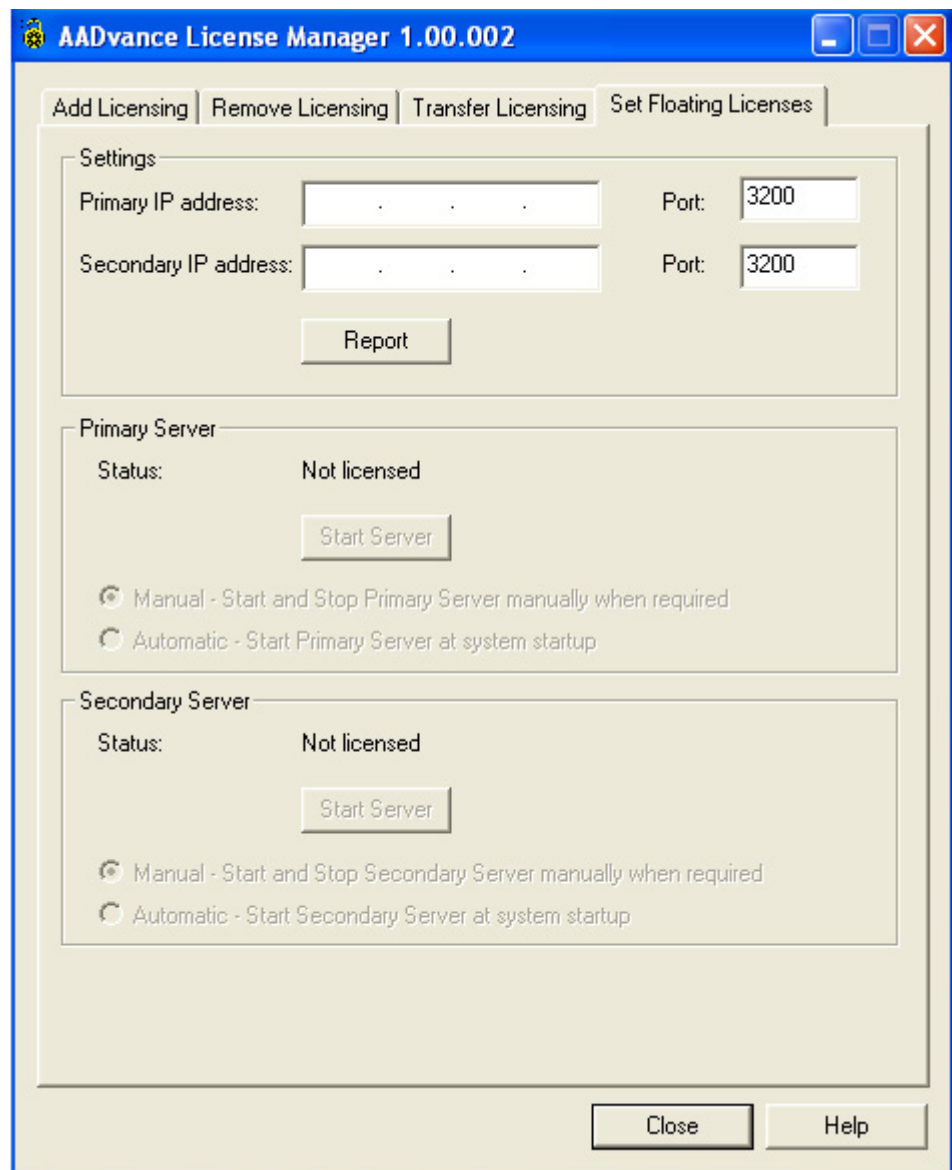
1. The Licensing Error message may not be automatically displayed, but you still need to upgrade your license.
2. A 2-stage procedure must be followed:
  - Follow the above procedure to upgrade a license for a single user.
  - Send a second email to [keymaster@ra.rockwell.com](mailto:keymaster@ra.rockwell.com) with the user codes generated in the first step to create a multi-user license.

## Using a Floating License Server

You can set up a networked PC as a server to provide floating AADvance licenses to workstations on the same network. You can also set up a secondary server to provide additional floating licenses or an alternative floating license source.

## Set Up a Server for Hardware Floating Licenses

To set up a primary server for hardware floating licenses, do the following:

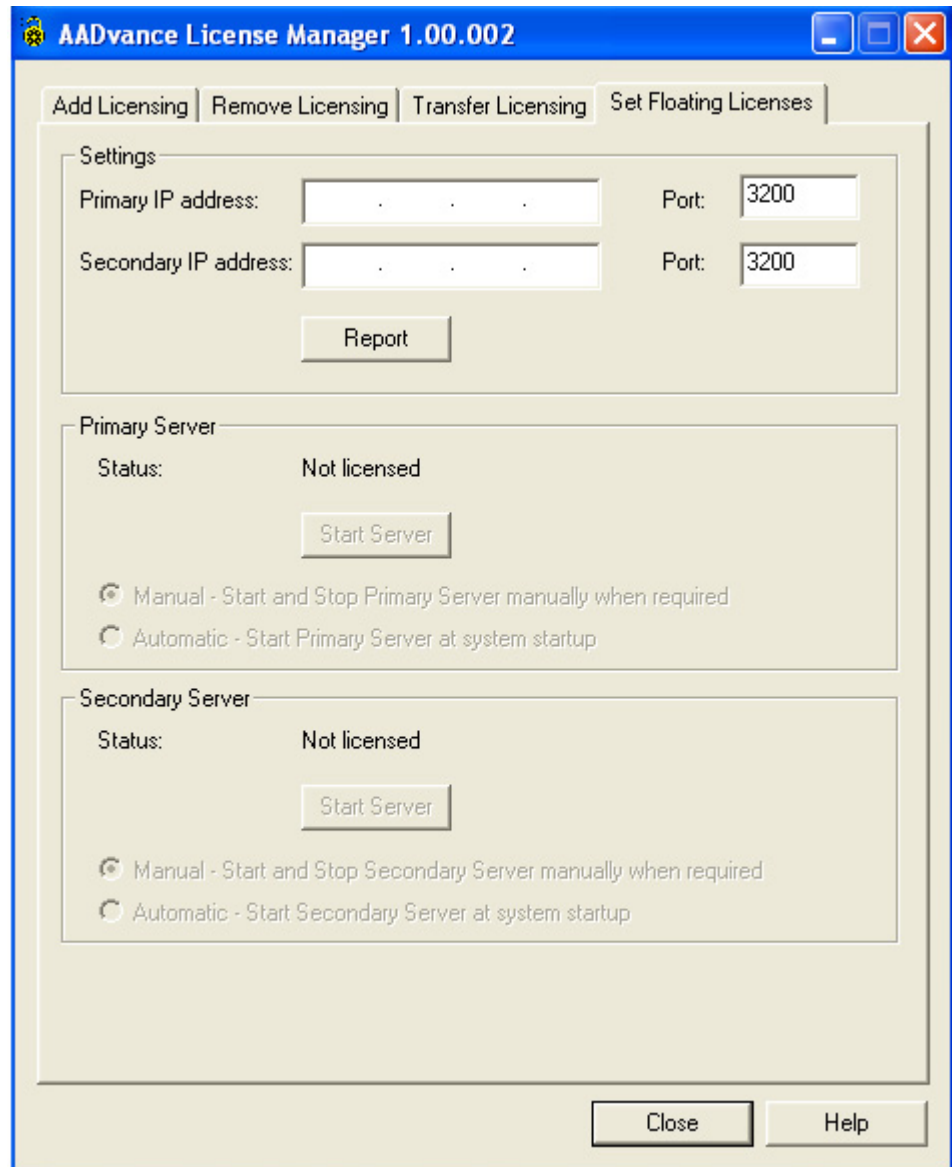


1. Insert the dongle supplied with the CD set into a USB or parallel port.

2. From the Start menu select **AADvance ➔ Licensing**.
  - The License Manager dialog box opens.
3. Select the **Set Floating Licenses** tab.
4. Set the IP Address of the primary server and a port number.
5. Select one of the two options to start the server Manual or Automatic (Automatic is recommended).
  - The Primary Server Status will show Stopped.
6. Click **Start Server** button.
  - The Server Status will show Licensed.
7. Repeat this procedure to set up a secondary server, if you have two floating license servers acting as a backup pair.

## Set Access to Floating Licenses

To set access a floating license, do the following:



1. From the Start menu select **AADvance** ➔ **Licensing**.
  - The AADvance License Manager dialog box opens.
2. Select the **Set Floating Licenses** tab.
  - The Primary Server Status will show Licensed.
3. Set the IP address of the primary server and the port number.
4. Click **Report** to claim a license.
  - The Primary Server Status will show Licensed.
5. Repeat this procedure to set up a secondary server if two license servers have been set up entering the Secondary IP Address and Port number.



6. Click **Report** to claim a license.
  - The Secondary Server will show Licensed.
7. Click **Close**.



## Connecting the Workbench to the Controller

This chapter describes the procedures for connecting the AADvance Workbench to the controller so that the application can be downloaded.

### Setting the Controller IP Address for AADvance Workbench

The AADvance system uses Internet Protocol (IP) to carry communications between the controller and the AADvance Workbench. This chapter shows you how to set up the IP address in the controller. It is convenient to set up the controller resource number at the same time.

#### Controller IP Address

The AADvance controller stores its IP address data in non-volatile memory in the 9100 processor base unit. The data is independent of the 9110 processor modules in the controller, and so the controller keeps the address information when you remove a processor module.

- You must set up the IP address data when you create a new system, or if you fit a new processor base unit.
- After you have set up the IP address data in the controller, you can configure the AADvance Workbench to find the controller on the network.

#### About Discover Communications

AADvance Discover uses DCP (Discovery & Configuration Protocol), transported within UDP. This is a general purpose protocol which allows AADvance controllers without an IP address to be configured on a network without having to set switches, or connect a serial cable to supply an initial IP address.

DCP communications may be blocked or mis-routed by Windows or by other network devices. You may find that systems known to be on the network do not appear in the list, or appear only randomly. A system that has not been successfully supplied with an IP address will not respond to a ping.

---

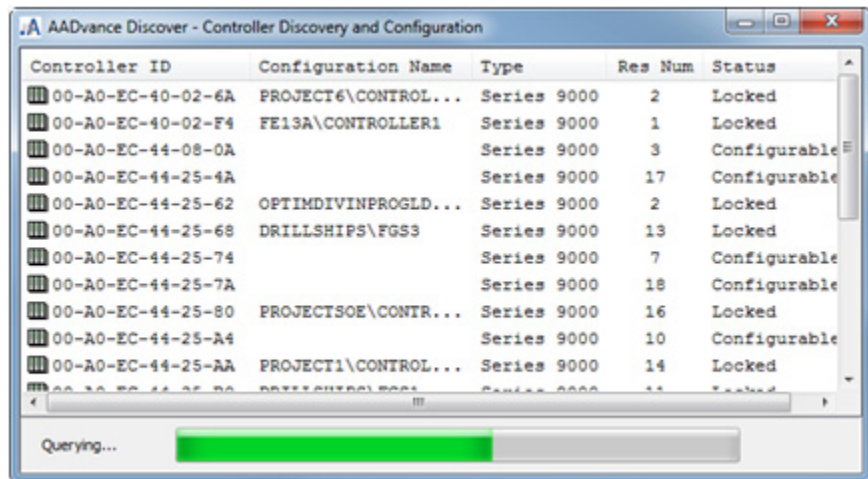
**NOTE** DCP relies on limited broadcasts and so may not work across network bridges and routers.

---

## AADvance Discover Utility

The utility uses a discovery and configuration protocol (DCP) to make a scan of the broadcast domain for AADvance controllers, and lets you configure the resource number and IP Address to be stored in a controller. You can use the utility to save configurations and re-load them in the future.

The AADvance Discover utility is installed when you install the AADvance Workbench, and appears on the Windows Start menu of the computer. Click **AADvance Discover** on the menu to start the utility. The utility displays a list of the AADvance controllers on the broadcast network, and reports a status for each one.



Double-clicking on an entry in the list lets you examine the resource and IP address settings for a controller. There is also a **Refresh** button, which will rescan the network and create a new list.

**TIP** If a controller known to be on the network does not show in the list, look for communication blocking or mis-routing by Windows or by other network devices. DCP communications will not go through network bridges and routers.

The DCP is proprietary to Rockwell Automation. It uses the first MAC address of the 9100 processor base unit to identify each individual controller - the MAC address is the 'Controller ID' in the list.

The controller status will be 'No Response', 'Locked' or 'Configurable':

- 'Configurable' means that the controller can be configured with its IP address. The utility has established communications with the controller, the program enable key is present (this plugs into the KEY connector on the 9100 processor base unit) and either no application is loaded or an application is loaded but not running.
- 'Locked' means that the utility has established communications with the controller, but one or more of the criteria for 'Configurable' status is not there.

- 'No response' means that the controller is turned off, or the communications between the computer running the utility and the controller have failed.

The status must be shown as 'Configurable' before you can set the controller configuration. If you change the configuration of a controller, click the **Refresh** button to make a new list.

A status bar is displayed at the bottom of the window, below the list. This will show the message 'Initializing' as the tool starts followed by 'Searching,' locating all the controllers connected to the network, then 'Querying' and finally 'Ready':

- 'Querying' scans the network for controllers and creates a list.
- 'Ready' means the controller status and IP addresses are displayed ready for new settings.

A **Refresh** button allows you to repeat the Querying process.

### **Troubleshooting AADvance Discover Communications**

This procedure describes how to activate communications using the Discover tool. After completing the steps, refresh the Discover tool's list of modules to test for communications.

1. Ensure that the Ethernet cable is plugged into a socket above a fitted AADvance controller – communications will not 'pass through' unused slots.
2. Ensure that the controller is activated by turning the locking bar. Wait for the Ready LED to go green before refreshing the Discover tool. (The communications tasks are not active until it is 'Ready').
3. Do not use an office network. Use an isolated hub or switch between the computer and AADvance controller. Check that the hub/switch has LEDs lit for the ports to both computer and controller, showing that the ports are working.
4. Open the Network Connections window. Open the Properties of the computer's network adapter (as used for configuring AADvance). Un-tick all protocol "items" which are not immediately necessary, especially "Check Point SecuRemote" and "iPass Protocol" (if present). You will need to leave "Internet Protocol" (or) "IPv4" and "IPv6", "Client for Microsoft Networks", "File and Printer Sharing" and "Network Monitor Driver" (if present) for normal Windows operation.
5. Disable the Windows Firewall, or any third-party firewalls and shields.
6. If you are using a laptop, disable Wireless. If you have more than one network connection, disable those you are not using. The Discover tool installed with AADvance Workbench release 1.2 (1.20.109) will not discover with more than one network connection available.

## Configure the Controller Resource Number in the Controller

When you assemble a new AADvance controller (or install a new 9100 processor base unit) you have to configure the resource number stored in the controller. The resource number is a type of device address, it must be configured to be the same value as in the application.

The procedure to configure the resource number uses the AADvance Discover utility. To set the resource number do the following:

1. Write down the controller's first MAC address (the Controller ID); displayed on a label on the processor base unit. Install at least one 9110 processor module into the processor base unit.
2. Make sure the program enable key is inserted in the KEY connector on the processor base unit.
3. Start the AADvance Discover tool from the Start menu:
  - **Start → All Programs → AADvance → AADvance Discover**
  - The AADvance Discover utility scans the network for controllers, and creates a list.
4. Locate the controller in the list and make sure that the status of the controller is **Configurable** (any installed application must be stopped).
5. Double-click on the MAC address in the Controller ID field.

- The Resource and IP Address dialog box opens.

6. Enter the resource value into the Resource Number field, click **Apply**.
  - Returning to the main window of the utility, the controller status will show Pending Restart.
7. To finish the update, turn off the power to the controller.
8. Start the controller. Refresh the screen to confirm that the new resource number is shown in the Resource Number field, and that the controller status is configurable.

---

**NOTE** The same value as used in the Resource Number field must also be configured in the application's Resource Properties.

---

## Configure the IP Address in the Controller

When you assemble a new AADvance controller, or install a new 9100 processor base unit, you have to configure the IP Address stored in the controller. See [Chapter 4, Allocate IP Addresses for Network Communications](#) for more information.

The procedure to configure the IP Address uses the AADvance Discover utility. Changes happen immediately and you do not have to start the controller again. To set the IP Address do the following:

1. Make a note of the controller's first MAC address (the Controller ID); displayed on a label on the processor base unit. Install at least one 9110 processor module into the processor base unit.
2. Make sure the program enable key is inserted in the KEY connector on the processor base unit.
3. Start the AADvance Discover tool from the Start menu:
  - **Start → All Programs → AADvance → AADvance Discover.**
  - The AADvance Discover utility scans the network for controllers, and creates a list.
4. Locate the controller in the list and make sure that the status of the controller is **Configurable** (any installed application must have been stopped).
5. If controller not immediately detected, rescan with the AADvance Discover utility.
6. Double-click on the MAC address in the Controller ID field.
  - The Resource and IP address dialog box opens.
7. Enter the IP Address and Subnet Mask into the fields for each Ethernet port.

Series 9000 - 00-A0-EC-40-03-2A (INITIALTRAININGPROJECT\TRAINING...)

Resource Number:

	IP Address	Subnet Mask
Ethernet E1-1	10 . 75 . 105 . 244	255 . 255 . 252 . 0
Ethernet E1-2	10 . 75 . 109 . 244	255 . 255 . 252 . 0
Gateway	10 . 75 . 104 . 1	Gateway Port: <input checked="" type="radio"/> E1-1 <input type="radio"/> E1-2
Ethernet E2-1	10 . 75 . 105 . 245	255 . 255 . 252 . 0
Ethernet E2-2	10 . 75 . 109 . 245	255 . 255 . 252 . 0
Gateway	10 . 75 . 104 . 1	Gateway Port: <input checked="" type="radio"/> E2-1 <input type="radio"/> E2-2
Ethernet E3-1	10 . 75 . 105 . 246	255 . 255 . 252 . 0
Ethernet E3-2	10 . 75 . 109 . 246	255 . 255 . 252 . 0
Gateway	10 . 75 . 104 . 1	Gateway Port: <input checked="" type="radio"/> E3-1 <input type="radio"/> E3-2

☐ Enable Ethernet Forwarding



8. Enter the Gateway's IP address for each processor module (these can be left as the null address if there is no gateway) and set the Gateway Port (Subnet) on which the gateway is located, click **Apply**.
  - Returning to the main window of the utility, the controller status will show In Progress and then Configurable.
  - The controller uses the new settings.

### Save and Load a Configuration

When you have entered the IP Address details you can now save the configuration:

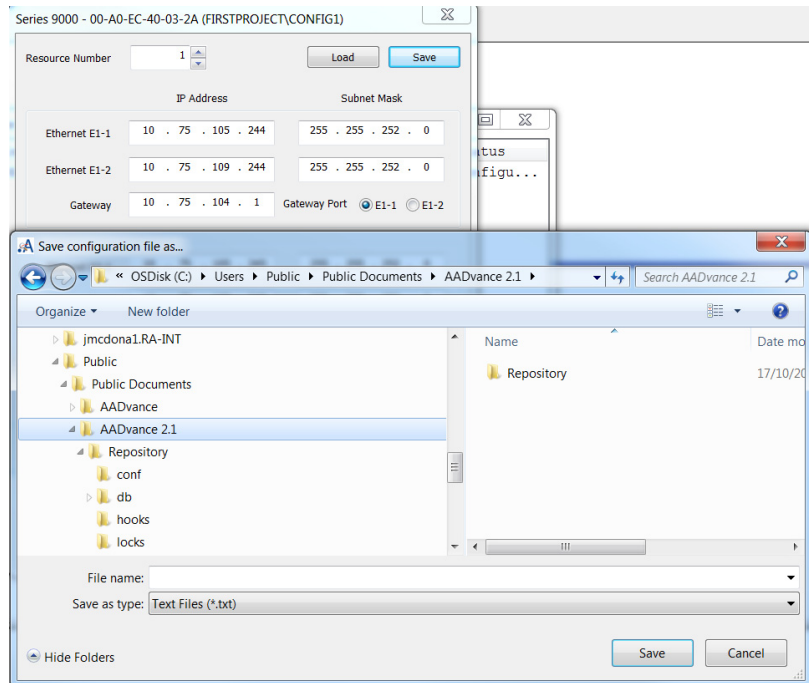
Series 9000 - 00-A0-EC-40-03-2A (INITIALTRAININGPROJECT\TRAINING...

Resource Number

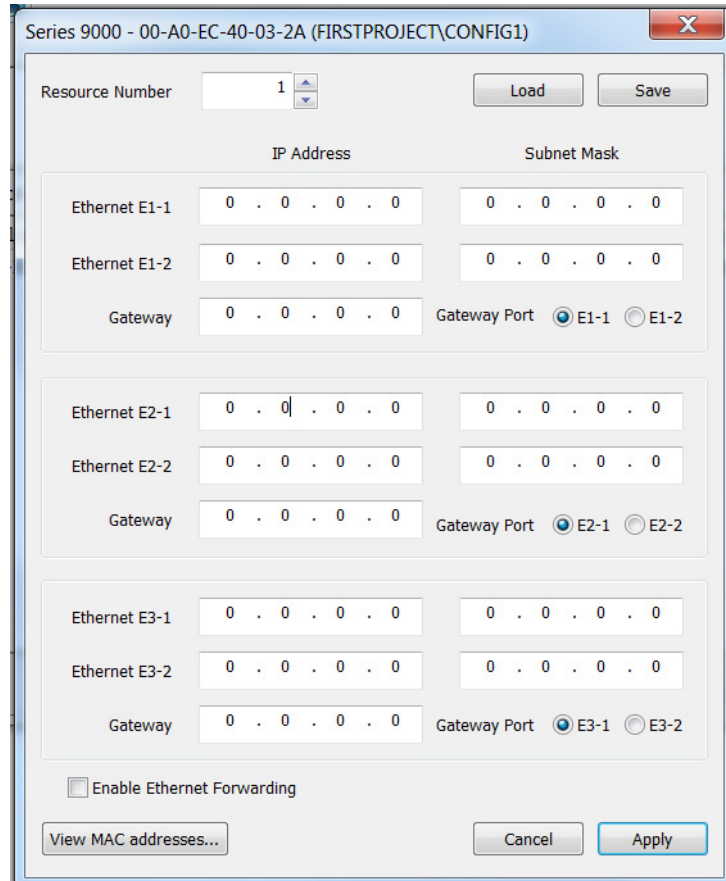
	IP Address	Subnet Mask
Ethernet E1-1	10 . 75 . 105 . 244	255 . 255 . 252 . 0
Ethernet E1-2	10 . 75 . 109 . 244	255 . 255 . 252 . 0
Gateway	10 . 75 . 104 . 1	Gateway Port <input checked="" type="radio"/> E1-1 <input type="radio"/> E1-2
Ethernet E2-1	10 . 75 . 105 . 245	255 . 255 . 252 . 0
Ethernet E2-2	10 . 75 . 109 . 245	255 . 255 . 252 . 0
Gateway	10 . 75 . 104 . 1	Gateway Port <input checked="" type="radio"/> E2-1 <input type="radio"/> E2-2
Ethernet E3-1	10 . 75 . 105 . 246	255 . 255 . 252 . 0
Ethernet E3-2	10 . 75 . 109 . 246	255 . 255 . 252 . 0
Gateway	10 . 75 . 104 . 1	Gateway Port <input checked="" type="radio"/> E3-1 <input type="radio"/> E3-2

☐ Enable Ethernet Forwarding

1. Click **Save** after you have entered your required configuration.
  - Give the configuration a name and save it to a suitable location.



To reload a saved configuration file:



1. Open the AADvance Discover utility
2. Double-click on a MAC address to open the Configuration dialog box.
3. Browse to the desired Configuration file.
4. Click **Load** to load the saved configuration.

Series 9000 - 00-A0-EC-40-03-2A (INITIALTRAININGPROJECT\TRAINING...

Resource Number

	IP Address	Subnet Mask
Ethernet E1-1	10 . 75 . 105 . 244	255 . 255 . 252 . 0
Ethernet E1-2	10 . 75 . 109 . 244	255 . 255 . 252 . 0
Gateway	10 . 75 . 104 . 1	Gateway Port <input checked="" type="radio"/> E1-1 <input type="radio"/> E1-2
Ethernet E2-1	10 . 75 . 105 . 245	255 . 255 . 252 . 0
Ethernet E2-2	10 . 75 . 109 . 245	255 . 255 . 252 . 0
Gateway	10 . 75 . 104 . 1	Gateway Port <input checked="" type="radio"/> E2-1 <input type="radio"/> E2-2
Ethernet E3-1	10 . 75 . 105 . 246	255 . 255 . 252 . 0
Ethernet E3-2	10 . 75 . 109 . 246	255 . 255 . 252 . 0
Gateway	10 . 75 . 104 . 1	Gateway Port <input checked="" type="radio"/> E3-1 <input type="radio"/> E3-2

☐ Enable Ethernet Forwarding

Once the IP address of the target controller has been configured, as described in [Configure the IP Address of the Target Controller](#) in [Chapter 4](#), a built application can be downloaded to it and started executing. Once an application is running, the AADvance Discover tool shows the controller as locked. Additionally, the configuration dialog box is grayed (read only), indicating that the controller cannot be configured while the application is running.

:

Series 9000 - 00-A0-EC-40-03-2A (INITIALTRAININGPROJECT\TRAINING... X

Resource Number 

1

Load

Save

	IP Address	Subnet Mask
Ethernet E1-1	10 . 75 . 105 . 244	255 . 255 . 252 . 0
Ethernet E1-2	10 . 75 . 109 . 244	255 . 255 . 252 . 0
Gateway	10 . 75 . 104 . 1	Gateway Port <input checked="" type="radio"/> E1-1 <input type="radio"/> E1-2

Ethernet E2-1	10 . 75 . 105 . 245	255 . 255 . 252 . 0
Ethernet E2-2	10 . 75 . 109 . 245	255 . 255 . 252 . 0
Gateway	10 . 75 . 104 . 1	Gateway Port <input checked="" type="radio"/> E2-1 <input type="radio"/> E2-2

Ethernet E3-1	10 . 75 . 105 . 246	255 . 255 . 252 . 0
Ethernet E3-2	10 . 75 . 109 . 246	255 . 255 . 252 . 0
Gateway	10 . 75 . 104 . 1	Gateway Port <input checked="" type="radio"/> E3-1 <input type="radio"/> E3-2

☐ Enable Ethernet Forwarding

View MAC addresses...

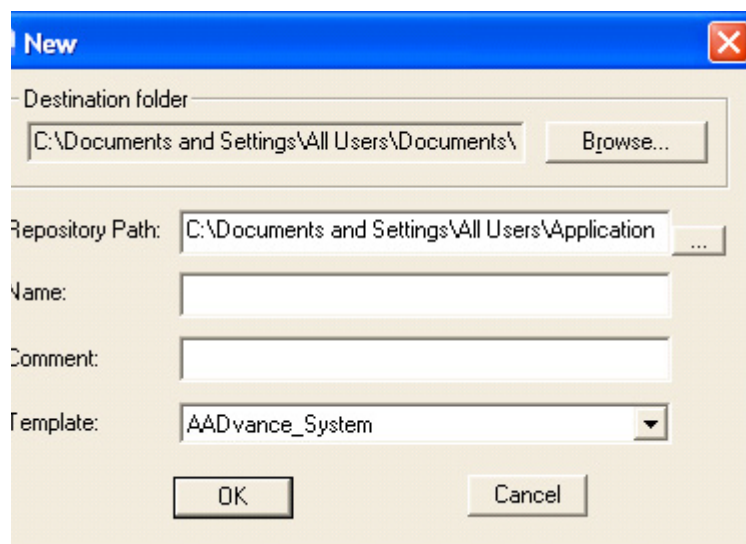
OK

## Setting Up a Project and Application

This chapter provides an overview of the setting up process for the AADvance Workbench, including creating a new project.

### Create a New Project

The configuration process starts by creating a new AADvance project. To create a new project do the following:



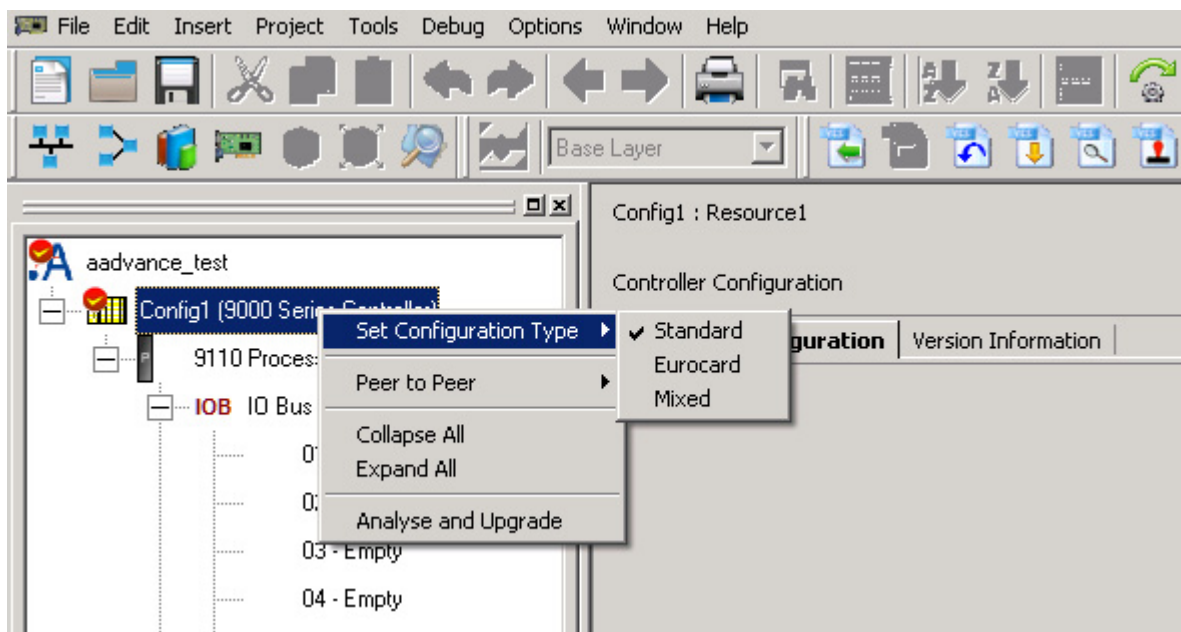
1. Start the AADvance Workbench.
2. Select **File** then New Project/Library (<ctrl>+N).
  - The New dialog box opens.
3. Enter a project name (maximum of 128 characters) and add a comment line.
4. Choose the AADvance\_System template from the drop down list, click **OK**.
  - The AADvance Workbench creates a project.

### Configure Controller Type (Standard or Eurocard)

The controller can be configured as a standard AADvance controller, a Eurocard controller or a mixture of both.

To configure the controller type, do the following:

1. Select the I/O Wiring icon .
2. Right click on Config1 (9000 Series Controller).
3. Select **Set Configuration Type** → Standard or Eurocard or Mixed.



Selecting Standard or Mixed will configure 48 empty IOB IO slots for I/O modules (IOB IO Bus 1 and IO Bus 2); selecting Eurocard will configure 18 empty slots on IOB IO Bus 1.

## Compiler Verification Tool

The Compiler Verification Tool (CVT) is a software utility that validates the output of the application compilation process. It is automatically enabled for resources when a project is created and when you add a resource to an existing project. This process in conjunction with the validated execution code produced by the AADvance Workbench confirms that there are no errors introduced by the Compiler during the development of the application.

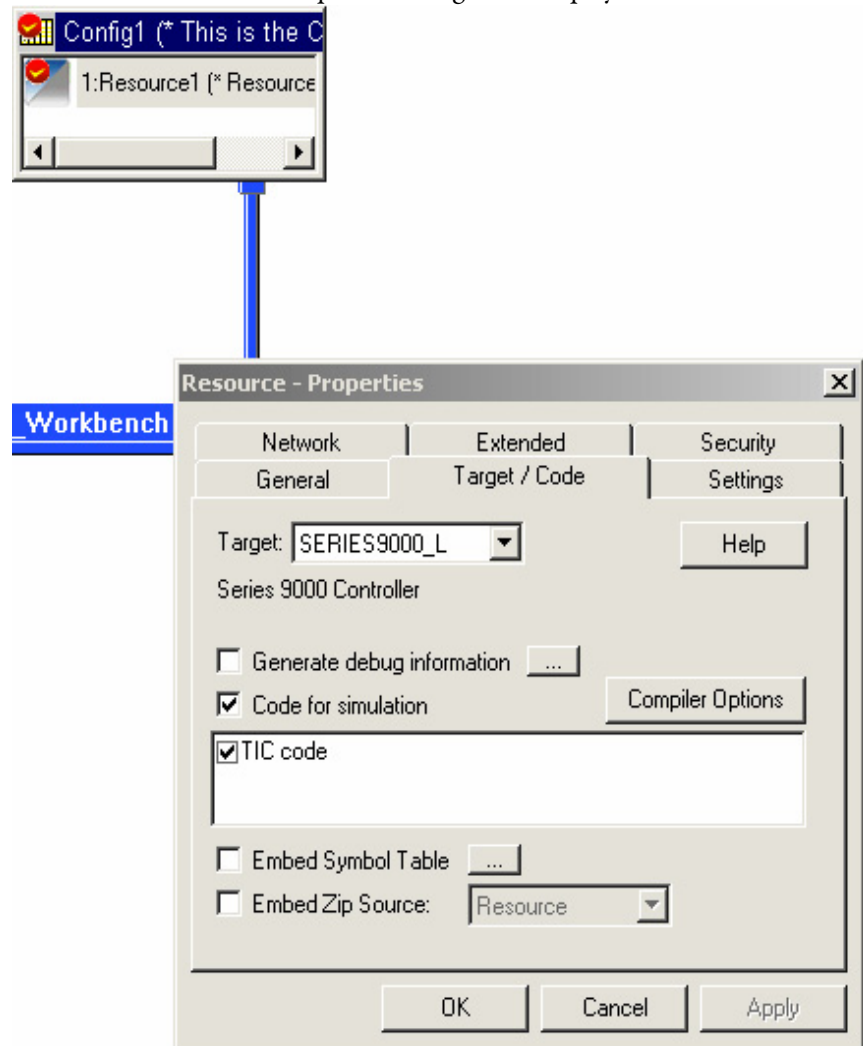
To achieve this CVT decompiles the application project file and then compares each individual application project (POU) source files with its decomposed version. The CVT analysis is displayed in the Workbench window.

## Enable the Compiler Verification Tool (CVT)

CVT is automatically enabled when you create a project. However, if your system is used for a Safety Related application you must ensure that the Compilation Verification Tool is enabled (and has not been disabled at some time) before you compile your POU files.

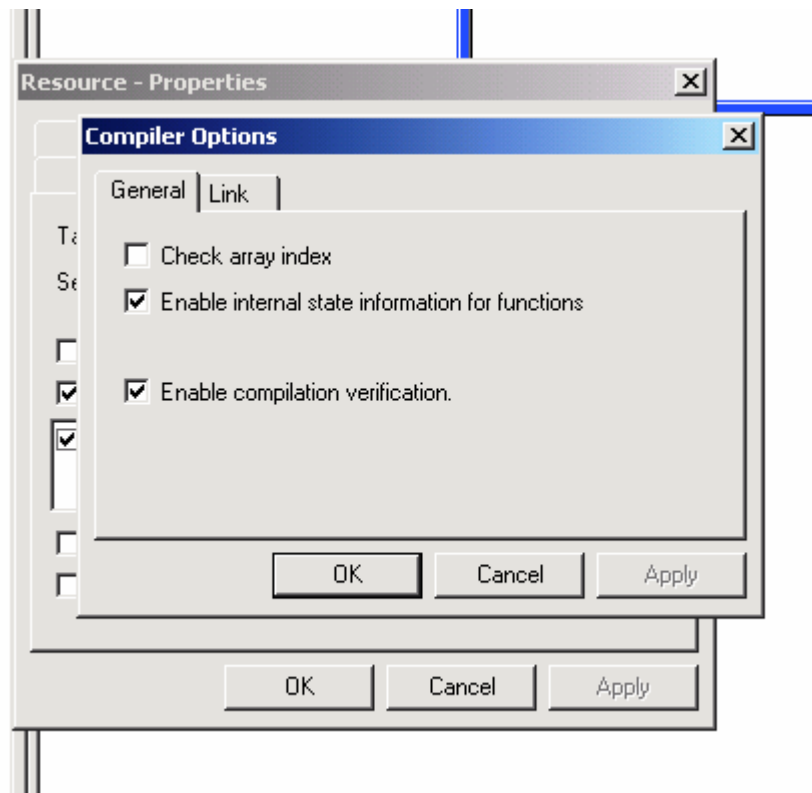
This procedure will show you how to check to see if it is enabled and tell you how to enable it if required.

1. Select the Resource and then the Resource Properties.
  - The Resource-Properties dialog box is displayed.

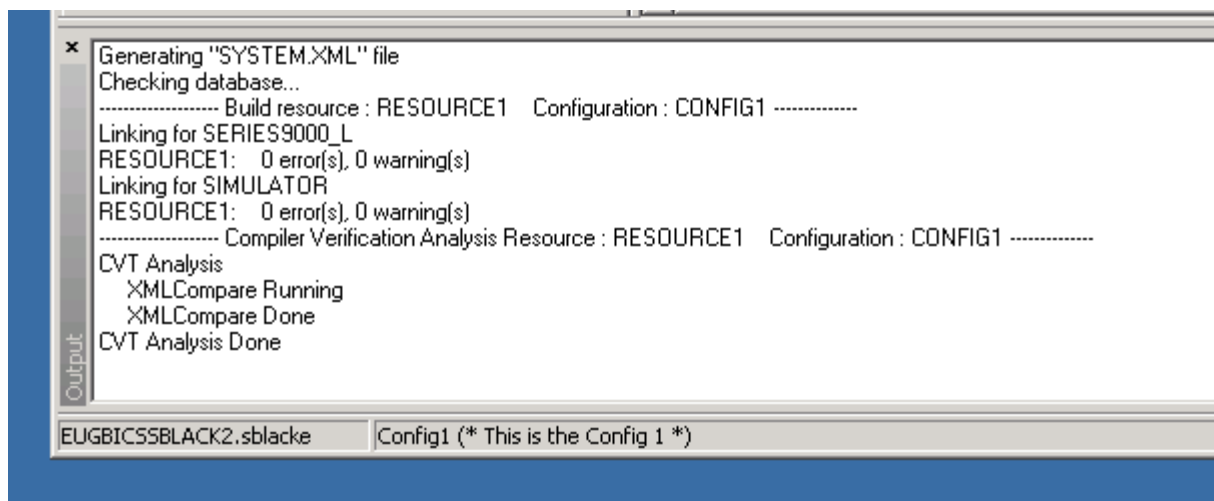


2. Click on **Compiler Options** button.
  - The Compiler Options dialog box is displayed.

3. If the Enable compilation verification box has a tick in it then click **OK** and proceed with your compilation.



4. If the box does not have a tick in it, click on the box so a tick appears.
  - You have now enabled CVT.
  - When you compile your application successfully the window will display the following message:







## Back-up and Restore a System Configuration

You can back-up (archive) system configurations (Projects) contained in a repository and you can restore them and others that were previously archived. Backing-up a repository project means copying a project and its modification history and placing it in a compressed file (.VSC). You can then restore a compressed file and the original modification history in another repository. The only way to test it is to connect it to an identical controller set up and confirm that it is correct.

### Back-up a repository project



The Workbench automatically points to the default source repository.

1. From the **File** menu, choose **Archive/Restore Project**.
2. In the Archive/Restore Project dialog, select the **Archive** option.
3. To select a repository other than the default, indicate the alternate source repository.
4. In the Repository Path field, click .
5. In the Browse for Folder dialog, specify the repository location, then click **OK**.
6. Specify the destination and a name for the archive file.
7. In the Archive File field, locate the directory in which to save the archive file by clicking .
8. In the Save As dialog, browse to select the required location, specify the name of the archive file using the .vsc extension, then click **Save**.
9. Select the required project from the repository:
10. In the Project Name field, select the project from the drop-down combo-box.
11. In the Archive Type field, select one of the following options, then click **Archive**.
  - To save all previously checked-in versions, select **Complete History**.
  - To save only the latest checked-in version, select **Latest Version Only**.

### Restore a repository project

The Workbench automatically points to the default destination repository. When restoring a project you should restore it to a different repository. If you restore it to a repository containing one having the same name, the existing project must be checked in recursively. Such a restore operation overwrites the history of the existing project.

1. From the **File** menu, choose **Archive/Restore Project**.

2. In the Archive/Restore Project dialog, select the **Restore** option.
3. To select a repository other than the default, indicate the alternate source repository.
4. In the Repository Path field, click .
5. In the Browse for Folder dialog, specify the repository location, then click **OK**.
6. Specify the source archive file.
7. In the **Archive File** field, locate the directory in which to save the archive file by clicking .
8. In the Open dialog, browse to select the required archive file, then click **Open**.
9. Select a project from the archive file by choosing one from the Project Name field, then click **Restore**.

## Online Recovery and Update from Controller to New Computer

In AADvance Workbench the versioned projects in the repository do not store the application binaries. Therefore it is not possible to perform an online update after changing its programming environment (e.g. new computer). This section illustrates the steps required to successfully perform an online update on a fresh programming environment with an existing repository.

Get the running project from the repository using AADvance Workbench.

1. Open the project from the repository.
2. Go in a debug/online session. The AADvance Workbench retrieves the last downloaded project binaries of each controller from the repository.
3. While still online, open the local project folder in Windows Explorer and copy the “\_\_Debug” folder. In this example the project name is Demo\_IP206  
*“C:\Users\Public\Documents\AADvance\Projects\AADvance\Prj\Demo\_IP206\\_\_Debug”*
4. Go offline and paste the folder into a temporary folder of your choice.

- Open the file “*Resource1\_MdfLinkReport.mtc*”, which is located at “...\\\_Debug\\Demo\_IP206\\Config1\\Resource1\\**Resource1\_MdfLinkReport.mtc**” and verify the value of the VER\_NEW flag. This number indicates the current build version running in the controller.

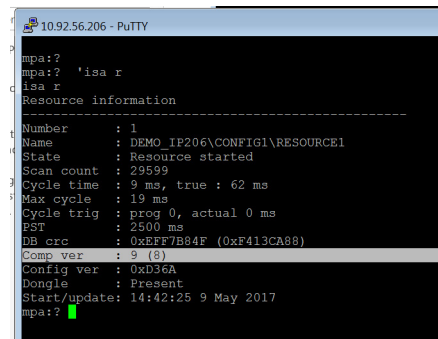
```

15 [RESULT]
16 ON_LINE=Y
17 TBL_CONF=Y
18 CRC_ORG=0xF413CA88
19 CRC_OLD=0xF413CA88
20 CRC_NEW=0xEFF7B84F
21 DAT_ORG=1494340728
22 DAT_OLD=1494340728
23 DAT_NEW=1494341126
24 VER_ORG=8
25 VER_OLD=8
26 VER_NEW=9
27 CRC_ALL_OLD=0x1209FF4D
28 CRC_ALL_NEW=0x3C5D4767

```

**Figure 1 - Build version indicated by the software**

- Use Telnet or the AADvance Collection Tool to verify the running build version..



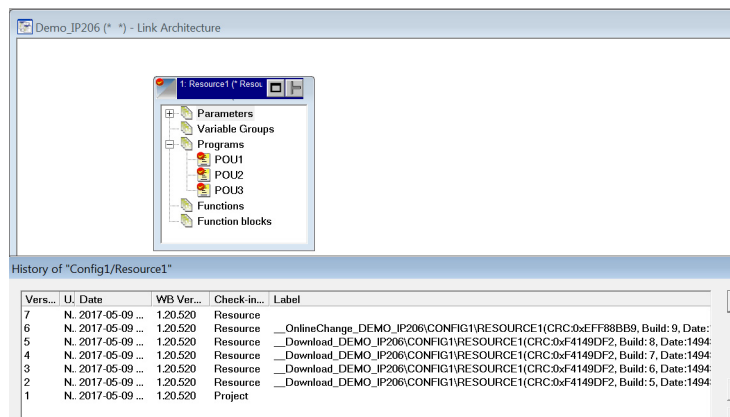
```

10.92.56.206 - PuTTY
mpa:?
mpa:? 'isa r
isa r
Resource information
-----
Number      : 1
Name        : DEMO_IP206\CONFIG1\RESOURCE1
State       : Resource started
Scan count  : 29599
Cycle time  : 9 ms, true : 62 ms
Max cycle   : 19 ms
Cycle trig  : prog 0, actual 0 ms
PST         : 2500 ms
DB crc      : 0xEFF7B84F (0xF413CA88)
Com2 ver    : 9 (0)
Config ver  : 0xb36a
Dongle      : Present
Start/update: 14:42:25 9 May 2017
mpa:?

```

**Figure 2 - Build version indicated by the controller**

- In the AADvance Workbench, from Version Source Control, “**Get**” the version of interest. In the following example, **Get** Version 6:



**Figure 3 - History of versions available in the repository**

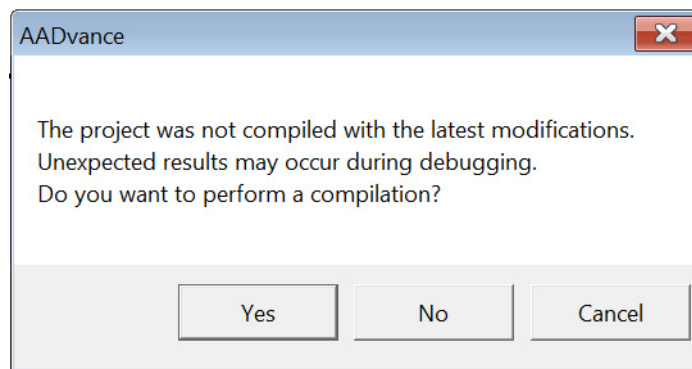
8. Once the version of interest is retrieved, close the workbench and open the project folder using the windows explorer:  
`"C:\Users\Public\Documents\AADvance\Projects\AADvance\Prj\Demo_IP206\Config1\Resource1"`

---

**NOTE** 8.1: only the source files were brought back from the repository.

---

9. Copy the pasted folder from step 4:  
`"... \_Debug\Demo_IP206\Config1\Resource1"` and paste it into :  
`"C:\Users\Public\Documents\AADvance\Projects\AADvance\Prj\Demo_IP206\Config1\Resource1"`
10. Reopen the project. At this point, the Resource1 files are the ones currently running in the controller. Starting an online session prompts a message to rebuild the project. This is as expected when a project is restored from the repository.



**Figure 4 - Compilation request after a Get operation**

11. Click **Yes** to proceed with the compilation.
12. Add/Modify the required changes within the project and perform a build. The version should be VER\_NEW or Com\_ver plus 1.

Online updates should now be permitted.

## Allocate IP Addresses for Network Communications

The AADvance system uses Internet Protocol (IP) for all communications between the controller and the AADvance Workbench. This includes downloading the application to the controller and real-time monitoring of the system in operation.

For many systems, the administrator of the local area network will allocate the address for the controller. If this is not the case, choose an address from the ranges allocated to private networks:

- 0.0.0.0 to 10.255.255.255 (10/8 prefix)
- 172.16.0.0 to 172.31.255.255 (172.16/12 prefix)
- 192.168.0.0 to 192.168.255.255 (192.168/16 prefix).

Each controller on a particular local area network must have a unique IP address.

---

**IMPORTANT** You must ensure that the two Ethernet ports on each T9110 processor module are on different subnets.

---

### **Example**

As an example you can use subnet masks to ensure that the two ports on a processor module are on different subnets:

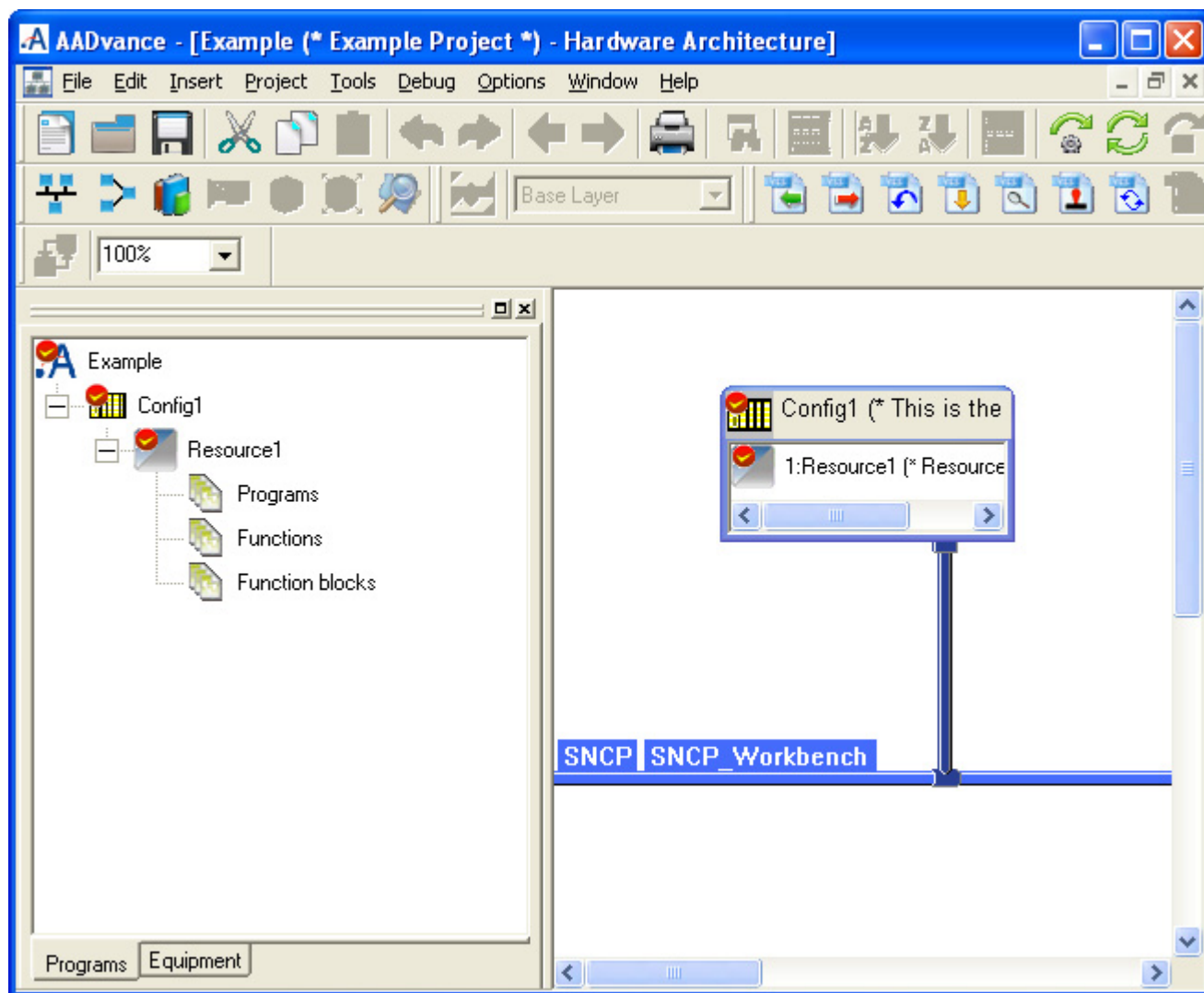
- Ethernet port E1-1 Address: 10.10.1.1
- Subnet Mask: 255.255.255.0
- Ethernet port E1-2 Address: 10.10.2.1
- Subnet Mask: 255.255.255.0.

The subnet mask defines the first three digits of the IP address, in this case 10.10.1 and 10.10.2.

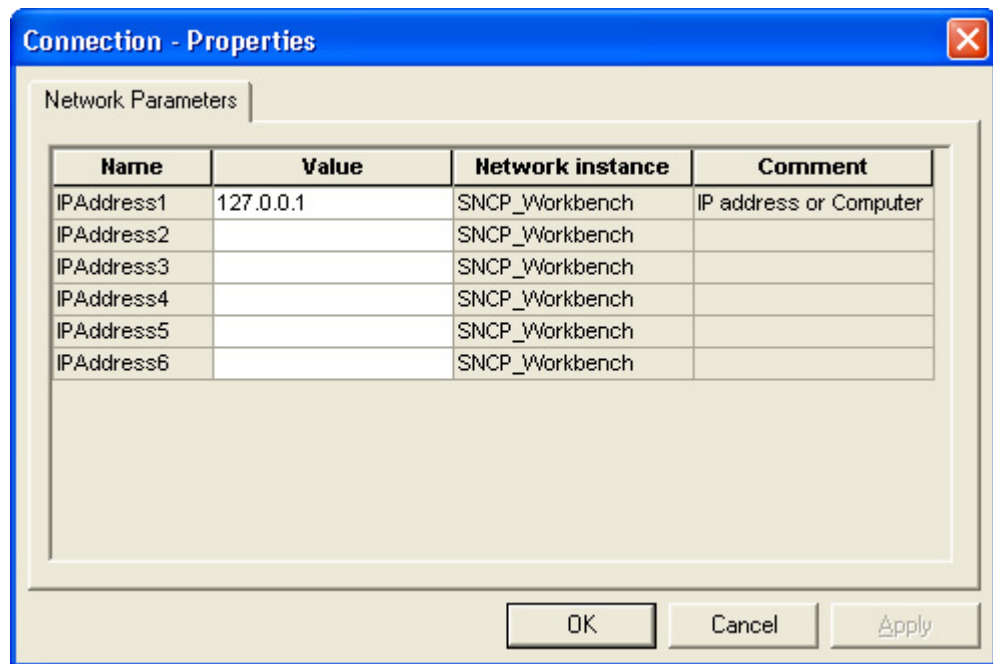
## **Configure the IP Address of the Target Controller**

To connect the AADvance Workbench project to the target controller you have to indicate the IP addresses allocated for the controller to the project.

1. Select the Hardware Architecture view then double-click on the vertical connection between the SNCP network line and the configuration.



2. The Connection - Properties dialog box opens.



3. Enter the IP Addresses in the Value field for each of the required Ethernet network. Press Enter after typing each IP address.

---

**NOTE** The value shown above is a default value. Enter a value that you require.

---

4. Click **OK**.

You have now configured the IP addresses of the configuration to match the controller.

5. Build the application.
6. See Section [Configure the Top-level Process Safety Time \(PST\)](#) in [Chapter 5](#) to choose a top-level process safety time that must be larger than the application execution time.

---

**IMPORTANT** If you specify a process safety time below the application execution time, the application will not run. The default is 2,500 ms.

---

Choose from the following range of values:

- Minimum 20 ms
- Maximum 60,000 ms (which is 60 seconds or 1 minute).

---

**IMPORTANT** A PST of 20 ms is not sufficient to let the controller run, short PSTs must be tested, including testing the addition of processors.

For a large installation, set the top-level PST to 1,500 ms (1.5 s) or larger. This ensures there is sufficient time for a second or third processor module to educate.

---

7. Download the application to the controller.



## Configuring the Processor Modules

This chapter describes the process to configure the processor modules. The first step in the process is to set the top level PST and the battery alarm. Then you must make sure that all processor modules are using the same firmware version and if necessary update the firmware version.

### Configure the Top-level Process Safety Time (PST)

The PST setting defines the maximum time that the processor will allow the outputs to remain in the ON state in the event of certain internal diagnostic faults or systematic application faults. If PST expires the system will go to its safe state.

You have to specify the PST for the whole controller. This is a top level setting, which you make once for all the T9110 processor modules.

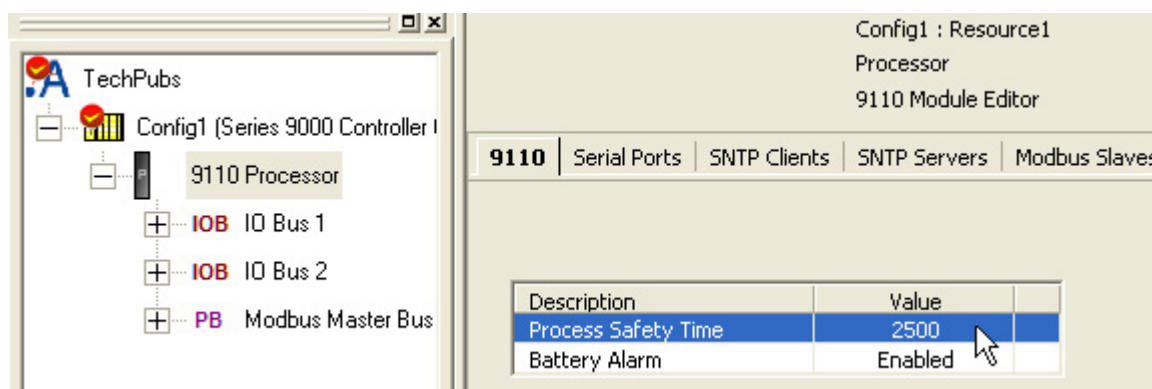
---

**IMPORTANT** Groups of I/O modules can inherit this setting, or use an I/O module specific PST setting instead, however this setting cannot exceed the top level setting. For a large installation with a high number of I/O modules it is recommended that you set the PST or the Processor a minimum of 1.5s. This is because for a larger installation the education process for a second or third processor module can take quite a long time.

---

### Set the Top Level PST

To set the top-level process safety time do the following:



1. Select the 9110 Processor in the Equipment tree view.
  - The 9110 Module Editor opens.
2. Select the **9110** tab.

3. Enter the time value into the Process Safety Time field. Choose from the following range of values:
  - Minimum: 1500 ms
  - Maximum: 60,000 ms (60 seconds)
4. Click **Apply**.

---

**IMPORTANT** If you specify a process safety time below the application execution time, the application will not run. The default is 2,500 ms.

---

### Set PST to its Default Value

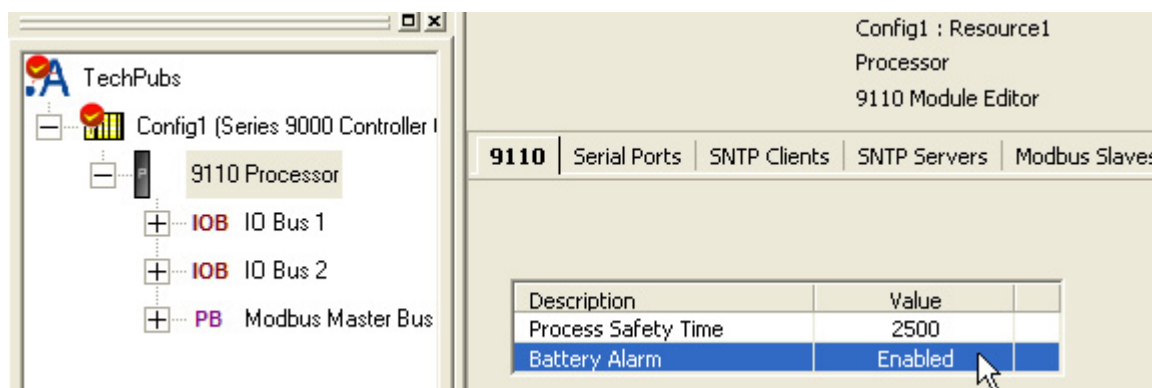
You can reset the PST to its default value, do the following:

1. Select the 9110 Processor in the Equipment tree view.
  - The 9110 Module Editor opens.
2. Select the **9110** tab.
3. Clear the entry in the Process Safety Time field.
4. Press return.
  - The Process Safety Time will change to its default value.

## Configure the Processor Battery Alarm

The 9110 Module Editor includes a configuration setting for the battery alarm. It can be set to Enabled or Disabled as required.

1. Select the 9110 Processor in the Equipment view.
2. Select the **9110** tab on the 9110 Module editor.
3. Select the Battery Alarm row and the required option.



## View Module Firmware Versions

Using the AADvance Workbench you can view the module firmware information on screen and save this information with your project. Using an update function view the latest information and save it as an external text file.

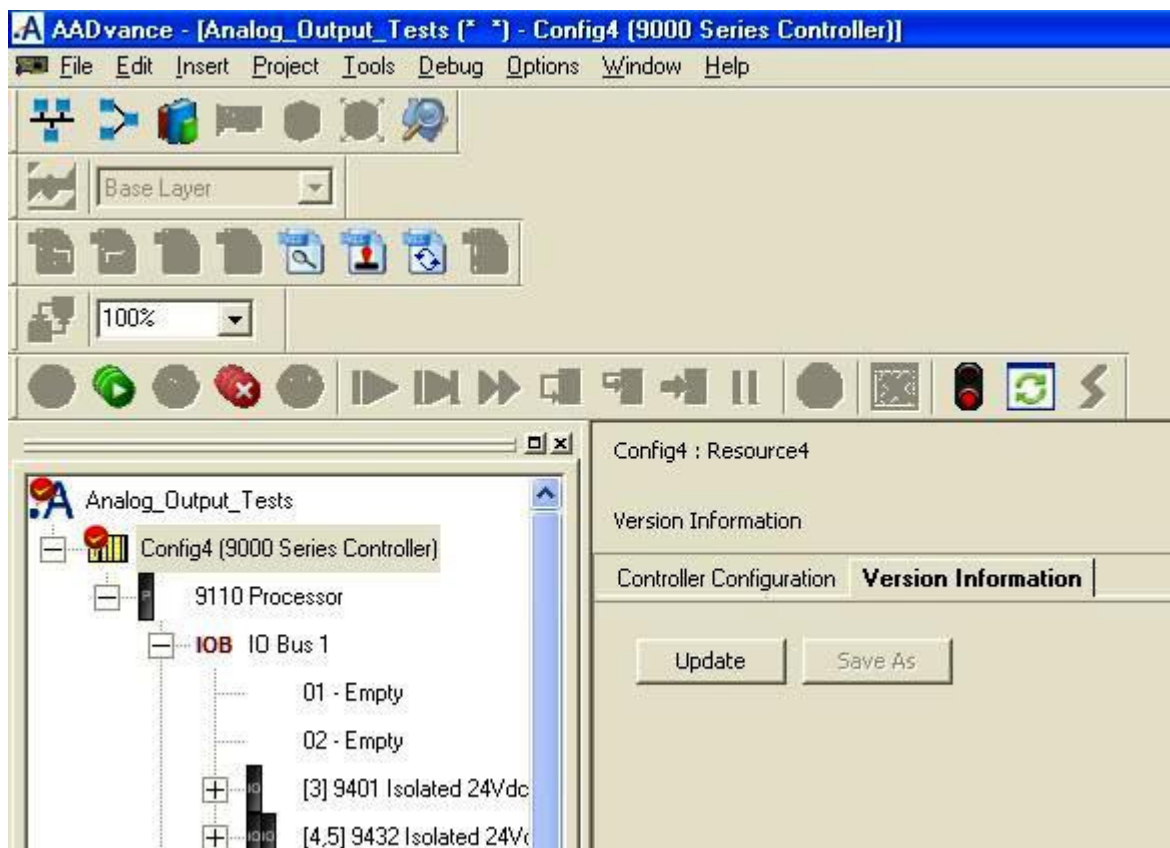
To do this proceed as follows:

---

**IMPORTANT** To view the firmware version numbers of the modules you must be connected using Debug to a running controller.

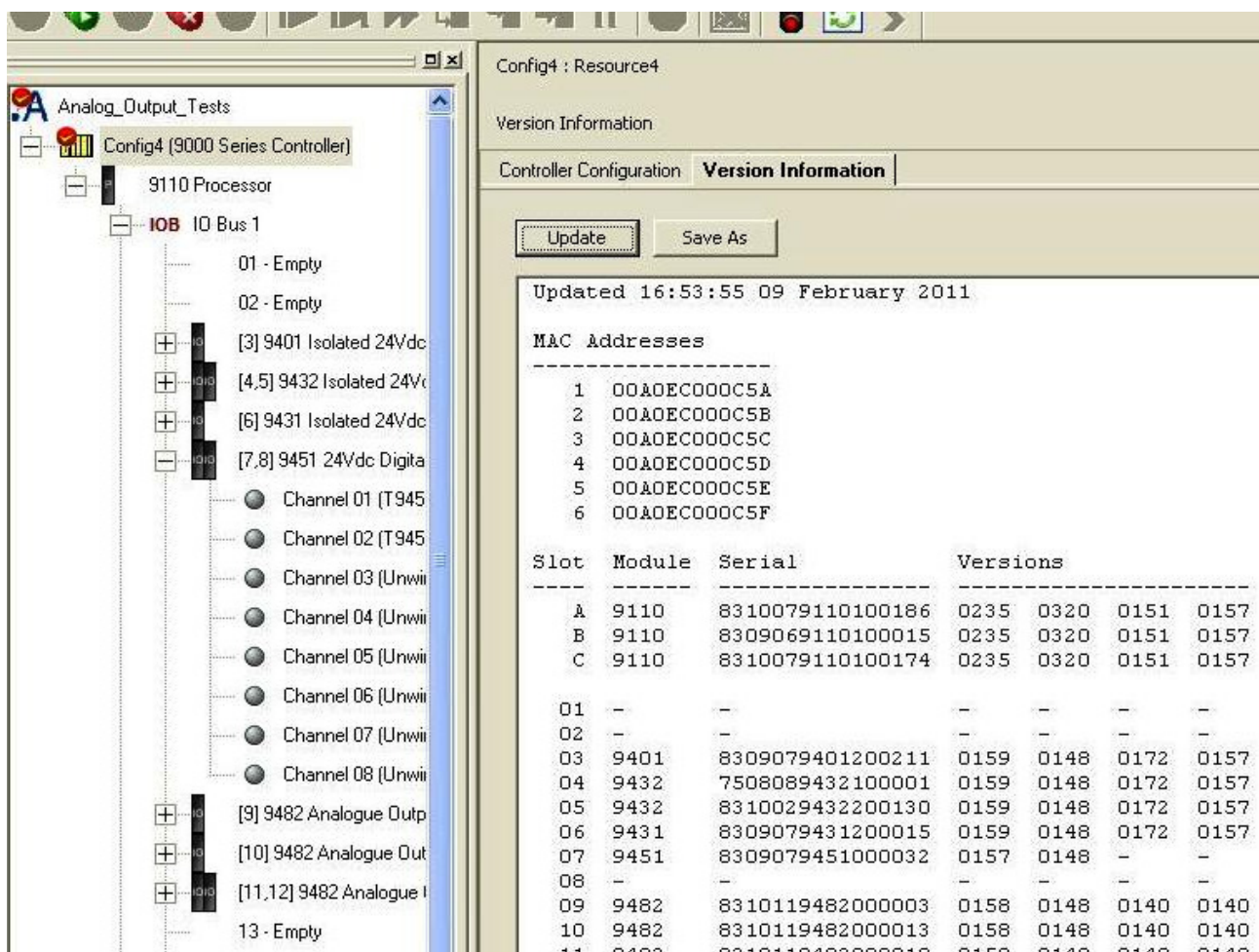
---

1. Select the **Equipment View** tab.
2. Select the desired configuration node. This is Config1 (9000 Series Controller) in the example shown below.
3. Select the **Version Information** tab.
  - The version information window appears. If the version data has previously been requested and saved (applied) then it will be displayed in this window.



4. Click the **Update** button.

- The window now shows your controller's current firmware version information.



The information displayed is as follows:

- MAC Addresses - MAC addresses for the controllers. There are two addresses per controller set by the BUSP chip inserted into the processor base unit. However, 6 MAC addresses are displayed regardless if the system is a Single, Dual or Triple processor system.

The module Information is as follows:

- Slot - the slot the module has been allocated
- Module - the module identity
- Serial - the module hardware serial number
- Versions - The firmware versions in the module

1. Click **Apply**.

- The data is saved with the project so that next time the project is opened you can view it.

2. To save data to a text file click **Save As**.

3. A window opens with a default text file name Version\_Report.txt; Enter your own file name and click **Save**.



- The saved text file can be viewed in Notepad.

## Processor Firmware Upgrades

Refer to The [AADvance System Build Manual](#).  
[Document Number; ICSTT-RM448-EN-P](#) for firmware upgrade details

## Configure the Serial Ports

The AADvance controller provides up to six serial communication ports, two for each T9110 processor module present.

The serial port settings define the protocol ('type') and the data characteristics of each of the serial ports. To configure the serial ports do the following:

Config1 : Resource1  
Processor  
9110 Serial Ports Editor

9110 **Serial Ports** | SNTP Clients | SNTP Servers | Modbus Slaves | TCI | DiffServ | Variables

Name	Baud	Data Bits	Parity	Stop Bits	Type
S1-1	19200	8	None	1	RS485hdmux
S1-2	19200	8	None	1	RS485hdmux
S2-1	19200	8	None	1	RS485hdmux
S2-2	19200	8	None	1	RS485hdmux
S3-1	19200	8	None	1	RS485hdmux
S3-2	19200	8	None	1	RS485hdmux

1. Select the **Serial Ports** tab.
  - The Serial Ports Editor dialog box opens.
2. Select the communication parameters from the drop down lists, click **Apply**.
3. To restore the default values, click **Default** then **Apply**.

## Serial Port Protocols

The serial ports support the protocols listed in the table.

**Table 2 - Serial Port Protocols**

Type	Description
<b>RS485fd</b>	<b>Full-duplex, 4-wire connection with a separate bus for transmit and receive</b>
RS485fdmux	Full-duplex, 4-wire connection with a separate bus for transmit and receive and tri-state outputs on the transmit connections
RS485hdmux	Half duplex, 2-wire connection

## Serial Port Parameters

Each serial port on the AADvance controller supports the set of control parameters as detailed in the table.

**Table 3 - Controller Serial Port Parameters**

Description	Value(s)	Default	Remarks
Baud	1,200; 2,400; 4,800; 9,600; 19,200; 38,400; 57,600; 76,800 or 115,200	19,200	
Data Bits	5 to 8	8	
Parity	None, Odd or Even	None	
Stop Bits	1 or 2	1	
Type	RS485fd RS485fdmux RS485hdmux	RS485hdmux	'fd' means 'full duplex' 'hd' means 'half duplex'

**TIP** Most systems use two bits after each data byte. The two bits are either a parity bit (odd or even) and one stop bit, or no parity and two stop bits.

## Time Synchronization (SNTP)

The AADvance controller supports the Simple Network Time Protocol (SNTP) service that can circulate an accurate time around the network. It can be configured to operate as a SNTP client or server.

As an SNTP client the controller will accept the current time from external Network Time Protocol (NTP) and SNTP network time servers. The SNTP clients settings tell the controller the IP address of the external server; the version of SNTP offered by the server; and the operating mode for the time synchronization signal that the processors will use for their real time clock. As a client the processor module can be configured as a unicast or broadcast client.

The AADvance controller can also fulfill the role of one or more SNTP servers (one for each processor module) to provide a network time signal throughout the network. To enable server time on an interface it is necessary to specify the direct broadcast address for that interface. This works for broadcast or unicast modes and when configured as a broadcast server it can respond to Unicast requests from clients.

To set up SNTP you need to connect your controller to a suitable network using one of the Ethernet ports. The network must be connected to an external NTP server or have NTP loaded on to it.

## Configure the Controller as an SNTP Client

To configure the Clients service do the following:

1. Select the **Clients** tab.
  - The Clients Editor dialog box opens.

2. Set the E1-1 and E1-2 Address fields to the IP addresses of the network time server.

---

**IMPORTANT** The first address represents that of the primary server and the second one the secondary server for each processor module. At start up the SNTP client will choose the primary server of the "lowest" slice; if no primary signal is valid the SNTP client looks for an active secondary server signal.

---

- For non-fault tolerant operation, define one SNTP server for only one processor. The other processors will automatically synchronize to it and will inherit the time.
  - For fault-tolerant SNTP client operation, define more than one server address.
3. Select the server Version.
    - Choose SNTPv1, SNTPv2, SNTPv3, SNTPv4 or Unknown.
    - If you do not know the version of NTP/SNTP that the server offers, choose Unknown. This will disable some validation of the incoming signal.
  4. Set the Mode to Unicast or Broadcast as required.
    - In Broadcast mode the SNTP client will passively wait for regular broadcasts from the server. This reduces network traffic and hence the load on the servers.
    - In Unicast mode the SNTP client will actively poll as many servers as are configured every few seconds and use their responses. The polling rate (19s) is based on the drift rate of the real-time clock and cannot be configured.
  5. Click **Apply**.

## Configure the Controller as an SNTP Server

To configure the SNTP servers service do the following:

1. Select the **SNTP Servers** tab.
  - The SNTP Server Editor dialog box opens.
2. Select the **Unicast** or **Broadcast** mode.
  - If you select Unicast mode for a processor the controller will wait to be polled by a client and then respond with a time signal; it will not broadcast any time signals.

**TIP** If you select Broadcast mode for a processor the controller will regularly broadcast: it will also respond to unicast polling requests on that interface.

3. Set the Broadcast IP Address for the network.
4. Repeat steps 2 and 3 for each additional processor module.
5. Click **Apply**.



## Using the Controller as a MODBUS Slave

The AADvance controller can operate as a MODBUS Slave, supporting up to ten MODBUS Slaves on each 9110 processor module. This gives a capacity of thirty MODBUS Slaves for a controller with three processor modules.

---

**NOTE** As a MODBUS Slave device, the controller only transmits data upon a request from a MODBUS Master, and does not communicate with other slaves

---

### Support for MODBUS Slave Exceptions

When the AADvance controller operates as a MODBUS Slave, it can raise these exception codes:

#### Code 01: Illegal Function

The function code received in the query is not an allowable action for the slave. If a Poll Program Complete command was issued, this code indicates that no program function preceded it. Code 01 represents a function that the AADvance controller does not recognize or does not support.

#### Code 02: Illegal Data Address

The data address received in the query is not an allowable address for the slave.

The AADvance controller raises code 02 when a request specifies an address outside the 16-bit range 0 to 65,535. The exception occurs if the request specifies the address implicitly ('give me the 20 registers from address 65,530') or explicitly ('give me the register at address 65,536').

#### Code 03: Illegal Data Value

A value contained in the query data field is not an allowable value for the slave.

The AADvance controller can raise code 03 only on Boolean (coil) writes.

#### Code 04: Slave Device Failure

An unrecoverable error occurred while the slave was attempting to perform the requested action.

#### Code 05

Represents an internal error within the AADvance controller.

#### Code 06: Slave Device Busy

The slave is engaged in processing a long-duration program command. The master should retransmit the message later when the slave is free.

The AADvance controller can be 'busy' and thus raise code 06 while it is waiting for its application to download or to start. The controller can be report itself to be busy for up to 30 seconds; after this period, the controller will cease to respond.

## Configure the Controller MODBUS Slaves

You have to configure the communication parameters for each MODBUS Slave you implement within the AADvance controller.

Config1 : Resource1  
Processor  
9110 Modbus Slaves Editor

9110 | Serial Ports | SNTP Clients | SNTP Servers | **Modbus Slaves** | TCI | DiffServ | Variables

Name	Connection	Id	Port	Protocol
Processor A Slave 1	Not Configured			
Processor A Slave 2	Not Configured			
Processor A Slave 3	Not Configured			
Processor A Slave 4	Not Configured			
Processor A Slave 5	Not Configured			
Processor A Slave 6	Not Configured			
Processor A Slave 7	Not Configured			
Processor A Slave 8	Not Configured			
Processor A Slave 9	Not Configured			
Processor A Slave 10	Not Configured			
Processor B Slave 1	Not Configured			
Processor B Slave 2	Not Configured			
Processor B Slave 3	Not Configured			
Processor B Slave 4	Not Configured			
Processor B Slave 5	Not Configured			
Processor B Slave 6	Not Configured			
Processor B Slave 7	Not Configured			
Processor B Slave 8	Not Configured			
Processor B Slave 9	Not Configured			
Processor B Slave 10	Not Configured			
Processor C Slave 1	Not Configured			
Processor C Slave 2	Not Configured			
Processor C Slave 3	Not Configured			
Processor C Slave 4	Not Configured			
Processor C Slave 5	Not Configured			
Processor C Slave 6	Not Configured			
Processor C Slave 7	Not Configured			
Processor C Slave 8	Not Configured			
Processor C Slave 9	Not Configured			
Processor C Slave 10	Not Configured			

To configure a MODBUS Slave do the following:

1. Select the **Modbus Slaves** tab.
  - The 9110 Modbus Slaves Editor dialog box opens.
2. In the Name column, locate the processor and slave you wish to configure.
3. Set the Connection field, click **Apply**.
  - The Id, Port and Protocol fields are set to their default values.
4. If you set the Connection to a serial port, the Id field represents the Slave ID; set the Id field or accept the default value. The Port field does not apply for a serial connection, and is disabled.
5. If you set the Connection to Ethernet, do the following:
  - Set the Protocol field, click **Apply**.

**TIP** As a MODBUS Slave, the controller supports MODBUS RTU, using a serial or Ethernet connection; and MODBUS TCP, using an Ethernet connection. You can configure a combination of connections for the MODBUS Slaves, subject to a limitation of no more than two MODBUS RTU slaves using serial communications for each processor.

- The Id field represents the Unit ID; set the Id field or accept the default value.
- The default setting for the Port field suits most systems; occasionally, you will have to adjust it. Make sure the Port field matches the port expected by the MODBUS Master.

If the Port field is set to 502, MODBUS TCP protocol is enabled. All other settings of the Port field will enable MODBUS RTU protocol packaged as a serial stream over Ethernet. This can be converted to a serial connection using a standard terminal server.

The range of values accepted for the Id field, and the default value for the Port field, vary according to the protocol selected.

6. Click **Apply**.

## MODBUS Slave Communication Parameters

Each MODBUS Slave has a series of communication parameters as detailed in the tables.

**Table 4 - MODBUS RTU Slave Parameters**

Description	Value(s)	Default	Remarks
Connection	Not Configured, Sn-1, Sn-2, Ethernet (†)	Not Configured	
Id	1 to 247 (serial); 1 to 255 (Ethernet)	1 (serial); 255 (Ethernet)	Represents the Slave ID for a serial connection Represents the Unit ID for an Ethernet connection
Port	0 to 65,535	2000	Only used with Ethernet connections

**NOTE** (†) The letter 'n' identifies the processor module:  
1 = processor A, 2 = processor B, 3 = processor C.

**Table 5 - MODBUS TCP Slave Parameters**

Description	Value(s)	Default	Remarks
Connection	Not Configured, Ethernet	Not Configured	
Id	1 to 255	1	
Port	0 to 65,535	502	

## Transparent Communication Interface (TCI)

The AADvance controller processor module provides a Transparent Communications Interface (TCI) function. This functionality will establish a pass-through communications link between an Ethernet link to a Serial port allowing devices attached to a serial port to be communicated with and for them to reply. The controller does not tamper with or inspect the data passed over the channel.

TCI uses a TCP port number to represent a serial port. All six serial ports are represented by each controller, so any serial port can be reached from any controller. Traffic is routed through TCP to the relevant serial port and in reverse. However, TCI communication from the serial ports is only available when the controller is not executing an application.

Users can enable and disable the function and set the Inactivity Timeout and Idle Time values.

**IMPORTANT** To use the TCI function you must stop the resource. This will have a serious effect on a Safety Related application.

## TCI Configuration

To configure the TCI function use the following procedure:

1. Insert the Program Enable Key into the socket on the Processor Base Unit.
2. Ensure the Application is not running.
3. Select the Equipment Tree view and the **TCI** tab on the 9110 Module Editor screen.

4. On the TCI dialog box double click on the **Enable TCI** box so a tick appears in the box.

Config1 : Resource1  
Processor  
9110 TCI Editor

9110 | Serial Ports | SNTP Clients | SNTP Servers | Modbus Slaves | **TCI** | DiffServ | Variables

☒ Enable TCI

Name	TCP Port	Inactivity (s)	Idle (ms)
S1-1	10001	600	3
S1-2	10002	600	3
S2-1	10003	600	3
S2-2	10004	600	3
S3-1	10005	600	3
S3-2	10006	600	3

5. Click on the S1-1 row and set the required Inactivity and Idle values.
  - Inactivity values range from 1s minimum to 65535s maximum with a default value of 600s. After this period, if no data has flowed in either direction, the TCP connection will be closed.
  - Idle values range from 1 ms minimum to a maximum of 1000 ms with a default value of 3 ms. This is the time between characters arriving at the serial port that defines when a "serial packet" has ended and should be passed on to TCP.
  - Remember that TCP connections are also subject to the keep-alive period.
6. Click on the **Apply** button.
  - You have now enabled the TCI function.
7. To use TCI, the resource must be stopped.

## Differential Services (DiffServ)

Differentiated services (DiffServ) gives a simple and coarse method to classify the services of different applications, and thus specify the priority of IP traffic. DiffServ is useful to make sure that high priority services are not delayed (or less delayed) during periods of network congestion. When applied, the service uses bit patterns in the "DS-byte" of IP, which for IPv4 is Type-of-Service (ToS) octet.

When you configure DiffServ you apply a priority value to a service and therefore mark it as different to less important services. You do this by arranging routers or switches able to inspect IP headers and prioritize them by the ToS header octet. The network devices will then apply their rules to prioritize IP traffic. The AADvance controller maintains the priority when it responds to incoming messages, and sets a priority according to the configuration for the messages it sends out.

If you use DiffServ, the controller scan rate can be up to 5 ms larger or smaller than the scan rate when DiffServ feature is disabled.

The TCP/IP stack can apply the user-specified ToS data in its datagrams during the TCP negotiation (this is the 3-way handshake, RFC 793). You can specify this behavior when you set up DiffServ.

The DiffServ feature is available with release 1.3 onwards of the AADvance Workbench and controller:

- You can use a Workbench at release 1.3 with older controller firmware, but DiffServ will not be available.
- If you use the release 1.3 Workbench to configure DiffServ and then go back to use an older version of the Workbench, DiffServ will continue to work on the controller but you will not be able to alter its configuration until you use the release 1.3 Workbench again.

## Configure DiffServ

**IMPORTANT** Although you can set all 8 bits of the ToS octet, your networking equipment may interpret this as one byte or as two fields, DiffServ and ECN.

You can set the IP ToS octet (the DiffServ priority) for up to 16 services. You use the DiffServ editor in the Workbench to configure DiffServ for a controller, do the following:

Config1 : Resource1  
Processor  
9110 DiffServ Editor

9110 | Serial Ports | SNMP Clients | SNMP Servers | Modbus Slaves | TCI | **DiffServ** | Variables

☒ Enable DiffServ

Description	Value
TCP Negotiation	No

Index	Protocol	Port	Peer Address	ToS
1	TCP	80	0.0.0.0	Low
2	Unused			
3	Unused			
4	Unused			
5	Unused			
6	Unused			
7	Unused			
8	Unused			
9	Unused			
10	Unused			
11	Unused			
12	Unused			
13	Unused			
14	Unused			
15	Unused			
16	Unused			

1. Select the **DiffServ** tab.
  - The DiffServ Editor dialog box opens.
2. Put a tick in the box labelled **Enable DiffServ**.
3. Select a row in the table and enter the values for a particular service and the required ToS value:
  - Protocol: TCP or UDP
  - Port: The port number. If the AADvance controller is initiating the connection, the port is the number of the service of the device you are connecting to.

- **Peer Address:** The address of the device you are connecting with, for a TCP connection. This field is optional. It is only required if you need to set a priority for traffic from one specific device.

**TIP** If you want to set priority to traffic from all devices that use the same port to communicate with the AADvance controller, leave the peer address field empty.

- **ToS:** The priority value; a higher value sets a higher priority.  
Range: 0 to 255, or choose a predefined value from the drop-down list.  
Default: 0
4. Repeat step 3 for each port you need to configure.
    - If you enter identical details for the same service more than once, a warning symbol shows at the end of the row after the ToS column.
  5. Select a value for TCP Negotiation:
    - When the controller is acting as a server and TCP Negotiation is set to 'no', the controller uses the default ToS value 0 (zero) during the three-way TCP negotiation, and applies the ToS value from the table when the connection is made.
    - When TCP Negotiation is set to 'yes', the new socket created takes the ToS value of its parent, and the accepting socket keeps this value during the 3-way negotiation.
    - In all cases, the ToS value changes when the 3-way handshake is completed if a row in the table gives a more specific rule for the port.

**TIP** If you want a connection to be applied by ToS during the TCP negotiation phase (setting a 'yes' value) you must have Protocol and Port values set but no Peer Address value.

### Examples

1. The user configures ToS for TCP on port 80 (HTTP). All IP datagrams from the connected local port 80 are sent at the specified ToS/DSCP value.

The user configures ToS for UDP port 123 (NTP). All IP datagrams from the local UDP port 123 are sent at the specified ToS value.

## Network Firewall

If the network used by the AADvance system is connected to another network, the connections should pass through a firewall.

The following transport layer ports (services) are supported by AADvance; some ports are always open, others are only open when configured.



**Table 6 - AADvance Communication Ports**

Protocol	Port Number	Port Open	Purpose	Port Open When
TCP	502	When configured	MODBUS TCP Slave	Open if Controller is configured as a MODBUS TCP Slave.
TCP	1132	Always	ISaGRAF, application downloads, debug, SoE etc.	N/A
TCP	2000	When configured	MODBUS RTU Slave	Open if controller is configured as a MODBUS RTU Slave. The default port, 2000, is given in this table.
TCP	10001- 10006	When configured	Transparent Comms Interface (Serial Tunneling)	Serial tunneling must be enabled AND the resource is not currently loaded. The ports will be closed when the resource is restarted.
TCP	44818	Always	CIP Produce & Consume	N/A
TCP	55555	Always	Telnet (diagnostic interface) <sup>(1)</sup>	N/A
UDP	123	When configured	(S)NTP	The controller is configured as either a SNTP client or server. The ports are otherwise closed.
UDP	1123,1124	When configured	IXL bindings	Application uses IXL bindings <ul style="list-style-type: none"> <li>• 1123 open on a producer</li> <li>• 1124 open on a consumer</li> </ul>
UDP	2010	Always	Discovery and configuration protocol (DCP, Rockwell Automation)	N/A
UDP	2222	When configured	CIP Produce & Consume I/O	Open if CIP Produce & Consume I/O traffic is active
UDP	5000	When configured	Trusted peer-to-peer (P2P)	At least one P2P network / subnet has been configured and is enabled.
UDP	44818	Always	CIP Producer & Consume	N/A

(1) The Telnet service provides various commands to gather diagnostic information about the controller and interact with it. It is intended to be used by RA Support Engineers, or users acting under the guidance of RA Support Engineers. It is important that this service should only be available to authorised users whose access is limited to the control network. The service must not be made available over the corporate network or the internet. When not required, the service should be blocked, even within the control network. Additionally, the programme enable key should be removed from the controller in normal use.

Those ports that are always open, even when not configured or unused, are open to unauthorized access. Use the following guidelines to protect all open transport layer ports.

1. If the network used by the AADvance system is connected to another network, the connection should pass through a firewall, to protect the AADvance system from potential threats from the other networks. Techniques to protect a control network from the rest of the plant's network are described in the RA/Cisco CPwE Design and Implementation Guide (ENET-TD001-EN-P). See below for some advice from the CPwE.

2. The firewall should be configured to block all communication ports. If necessary a specific port may be enabled to a device that needs to communicate with other devices on other networks.
3. Ingress rate limiting should be used to protect the AADvance from network storms. The limit chosen should not impede the expected peak ingress rate for that controller and should be determined by calculation or observation of the system's network traffic when running.
4. The SNCP port must only be allowed to pass through the firewall if the Windows PCs running the AADvance Workbench are on a separate network.
5. The variable bindings ports must only be allowed to pass through the firewall if the AADvance controller is communicating with another AADvance controller on a separate network.
6. The other communication ports (e.g. MODBUS, SNTP) must only be allowed to pass through the firewall if the AADvance controller or Windows PC communicates with other devices on other networks.

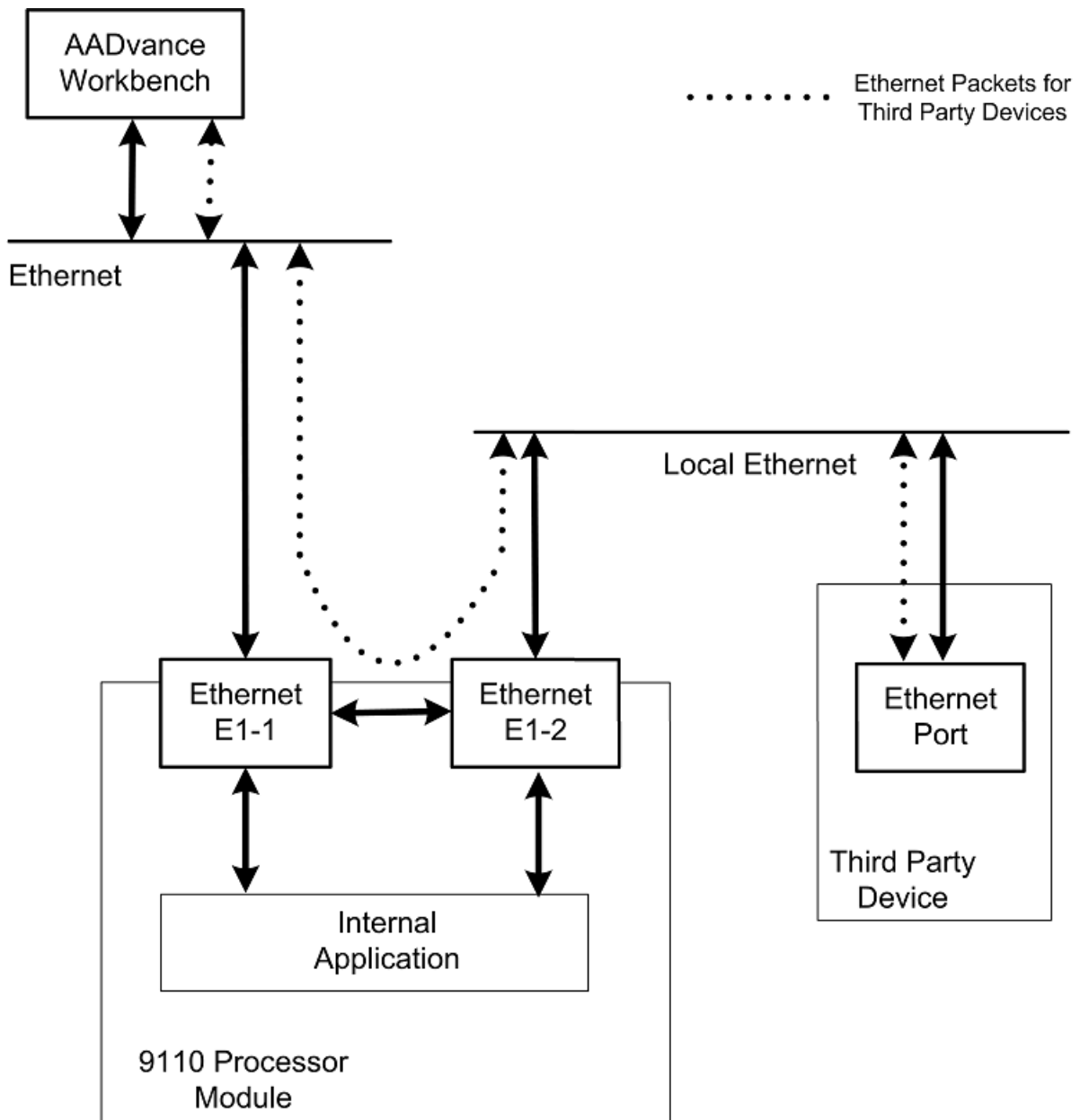
From the CPwE advice, techniques for securing a control network include (but are not limited to):

- Physical access to the Cell/Area zone should be restricted to authorized users. Especially, physical access within the Cell/Area zone to AADvance systems should be tightly controlled.
- Switches within the Cell/Area zone should be configured to permit only authorized devices (eg by MAC address).
- A firewall should be used to prevent access to ports at the boundary to the Cell/Area zone.
- When it is known that services are not required to a device within the Cell/Area zone, then for enhanced security, firewall devices can be employed in “transparent mode” to protect the AADvance controllers, blocking access to the ports requiring protection.

## Ethernet Forwarding

The Ethernet forwarding feature lets an AADvance controller re-send (that is, forward) Ethernet packets intended for a third party device, as shown in the illustration, together with all broadcast and multicast messages.

**Figure 5 - Ethernet Forwarding**



When Ethernet forwarding is enabled, each 9110 processor module in the controller forwards unicast messages intended for other devices, and all broadcast and multicast messages, between its two Ethernet ports. A device connected through the processor module can get its IP configuration through BOOTP or DHCP, or statically.

The processor module in the first position (slot) in the 9100 processor base unit forwards these messages from port E1-1 to E1-2, and in the opposite

direction from port E1-2 to E1-1. Similarly (if fitted), the processor module in the second position in the 9100 processor base unit forwards traffic from port E2-1 to E2-2, and from port E2-2 to E2-1. The third processor module (if fitted) forwards traffic from port E3-1 to E3-2, and from port E3-2 to E3-1. In each case, the second of these ports represents an uplink to the rest of the network or (if applicable) to another network. A device connected to this port sees all the traffic which can be relevant to it: broadcasts, multicasts, and unicast traffic not destined for the 9110.

The processor module continues to consume the unicast messages intended for itself, and all broadcast and multicast messages, as it does when Ethernet Forwarding is disabled.

Ethernet forwarding is not designed to make links from one processor module to another processor module, for example from ports E1-2 to E2-1 and E2-2 to E3-1. Do not do this.

The controller keeps its Ethernet forwarding setting if you change one or more of the 9110 processor modules. You do not have to change the setting during corrective maintenance.

## Configure Ethernet Forwarding

---

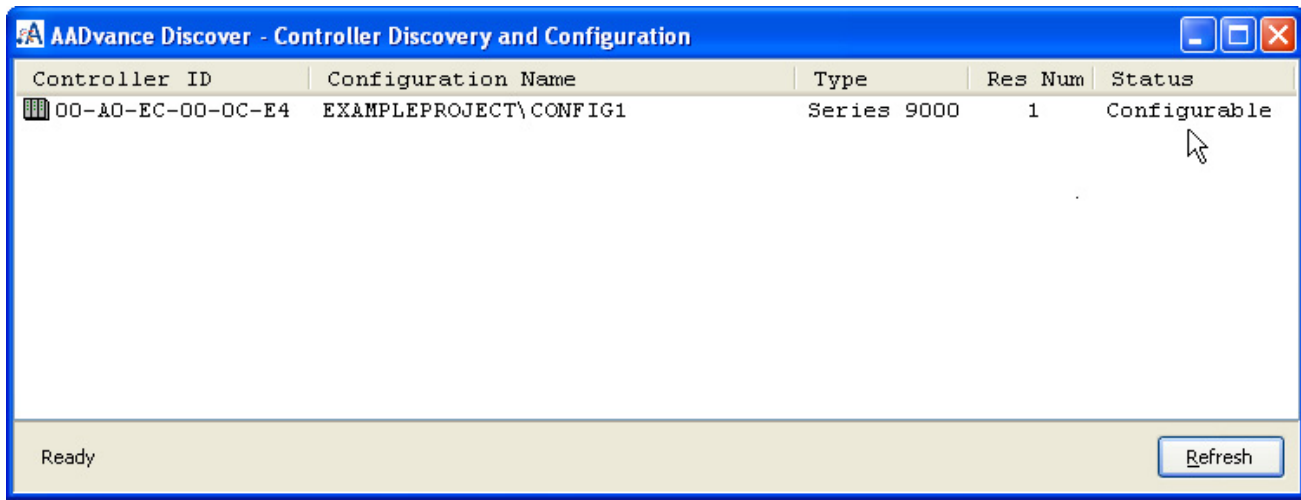
<b>IMPORTANT</b>	Make sure you do not use Ethernet forwarding to inadvertently make a bridge between segregated safety and non-safety networks. If you do this, the non-safety network will send non-safety data to a safety network. This will not cause a change to the integrity of safety-related data (which always uses a safety network as a 'black channel'), but it will add unwanted traffic and consume bandwidth on the safety network.
------------------	--

---

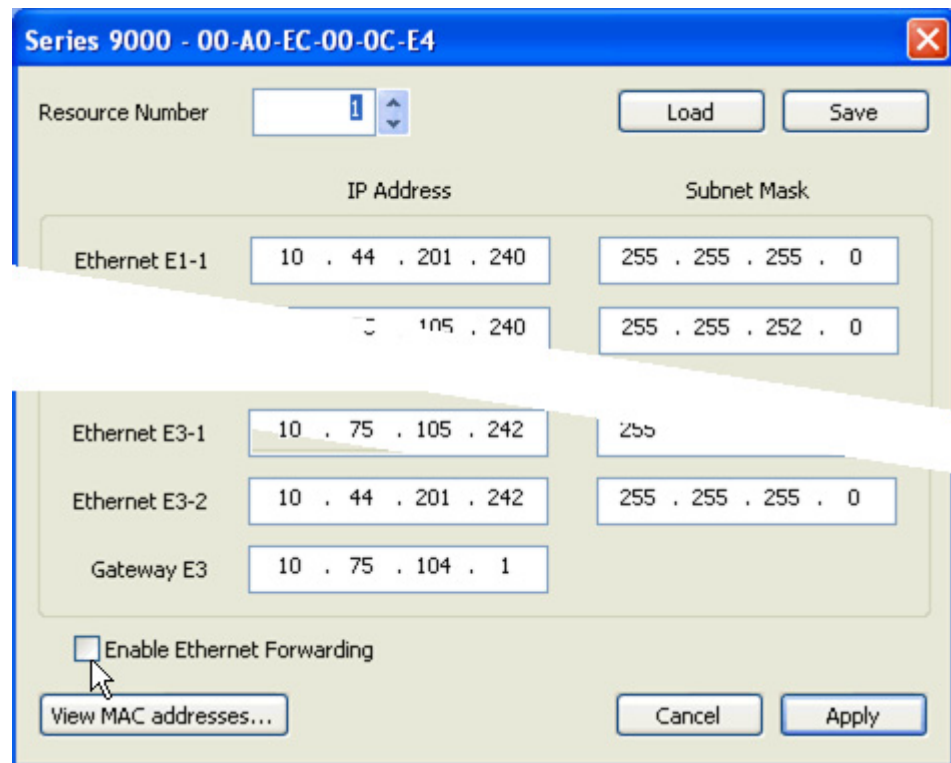
The AADvance Discover utility lets you enable and disable Ethernet Forwarding for each applicable controller on the network. Ethernet forwarding is disabled for a new controller, you have to enable it to override the default behavior. Do the following:

1. Insert the Program Enable Key into the KEY socket on the 9100 processor base unit.
2. Stop the resource running.
3. From the Windows Start menu select **AADvance Discover**.

4. Find the AADvance controller on the list and make sure that it is Configurable.



5. Double-click on the controller in the Controller ID field.
  - The Resource Number and IP Address dialog box opens.



6. Select **Enable Ethernet Forwarding**.
  - Ethernet forwarding is included in new 9110 processor modules, from release 1.3 onwards.

- If the check box is disabled, this means the controller does not support Ethernet forwarding. Use the ControlFLASH utility to upgrade the firmware in each 9110 processor module, or send an email to [icstsupport@ra.rockwell.com](mailto:icstsupport@ra.rockwell.com) (mailto:icstsupport@ra.rockwell.com) to ask for the details of your local service centre.
  - All the processor modules in a controller must have the same firmware.
7. Click **Apply**.
  8. Remove the Program Enable Key and start the resource.
    - You have now enabled Ethernet forwarding for the controller.
  9. To disable Ethernet forwarding do this procedure again, but de-select Enable Ethernet Forwarding on the dialog box.

## About T9110 Processor Variables

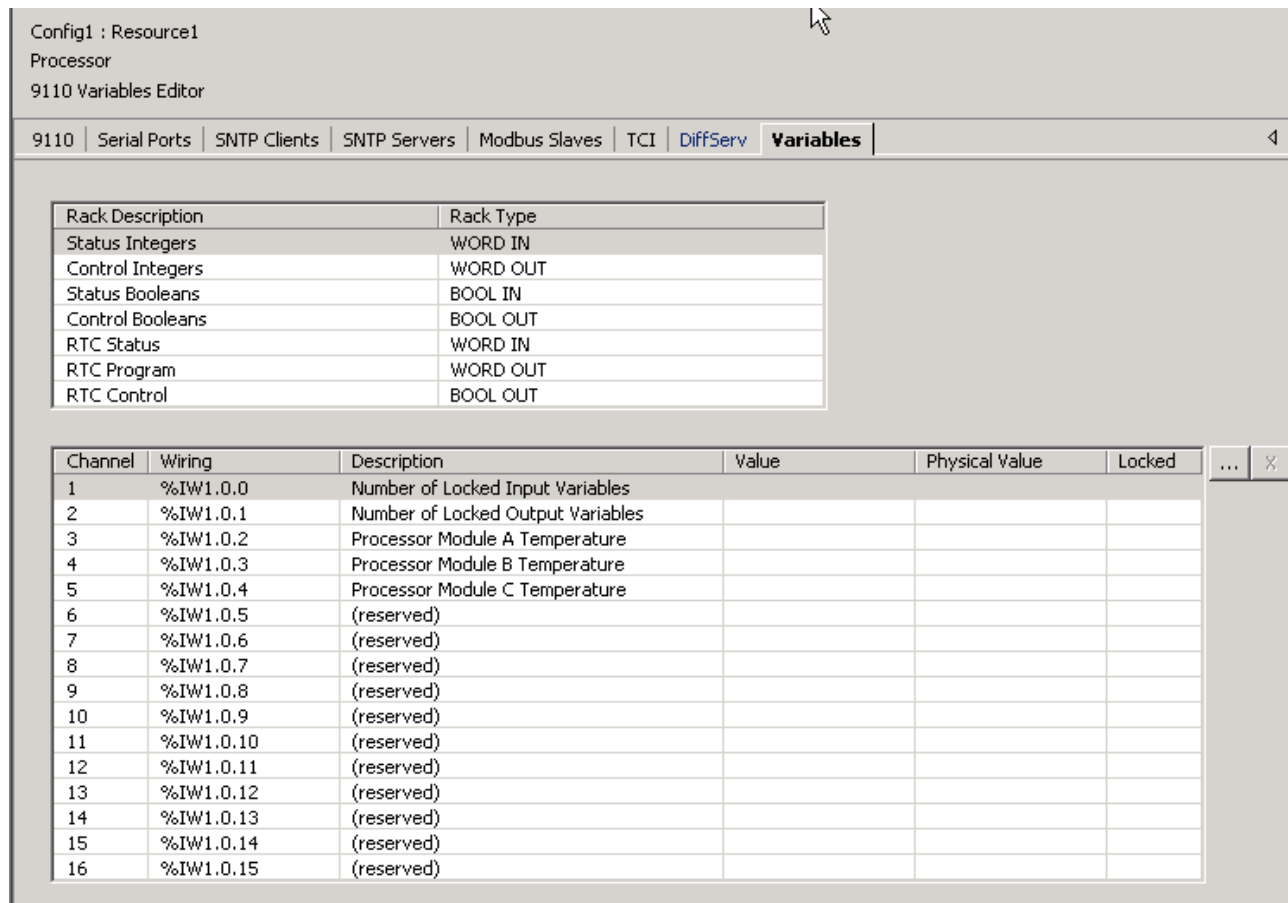
The T9110 processor module provides a number of status and control variables that are available to the application. Status variables retrieve status information; control variables set status information.


The 9110 Variables Editor presents the variables in seven collections, which it calls 'racks':

- Status Integers and Status Booleans, which provide information about the controller to the application;
- Control Integers and Control Booleans, which enable the application to send specific information to the controller;
- RTC Status variables, which provide information about the controller real-time clock to the application;
- RTC Program variables, which specify parts of the date to be written to the real-time clock;
- RTC Control variables, which set and control updates to the real-time clock.

## Wire Processor Variables

To wire a 9110 processor variable do the following:



1. Select the **Variables** tab of the 9110 Processor Editor.
  - The 9110 Variables Editor dialog box opens.
2. Select a rack, e.g. Status Registers.
  - The editor displays a list of associated channel variables.
3. Select a Channel.
4. Click the  button. The Select Variable dialog box opens.
5. From the list select an application variable to wire to the processor variable, click **OK**.
6. Repeat for each subsequent variable to be wired.
7. Return to the 9110 Processor Editor dialog box and click **Apply**. The variable will now be wired.

## Unwire Processor Variables

To disconnect a 9110 processor variable do the following:

1. Select the **Variables** tab of the 9110 Processor Editor.
  - The 9110 Variables Editor will be displayed.
2. Select the relevant rack.
  - The editor displays a list of associated variables.
3. Select the variable to be unwired, click the **X** button.
4. Click **Apply**.
  - The variable will be unwired.

**TIP** Select the **Unwire All** button and click **Apply** to disconnect all of the wired variables in the rack.

## Status Integers

The variables in the rack of status integers provide information about the controller to the application.

### *Number of Locked Input Variables*

**Direction:** input to application from controller.

**Type:** Word.

**Values:**

- 0 to 65,535.

**Description:**

Reports the quantity of input variables having been locked by the user. The top limit of 65,535 represents the capacity of the variable; the real limit is the number of variables in the application.

### *Number of Locked Output Variables*

**Direction:** input to application from controller.

**Type:** Word.

**Values:**

- 0 to 65,535.

**Description:**

Reports the quantity of output variables having been locked by the user. The



top limit of 65,535 represents the capacity of the variable; the real limit is the number of variables in the application.

#### *Processor Module A Temperature*

**Direction:** input to application from controller.

**Type:** Word.

**Values:**

- 0 to 65,535.

**Description:**

Reports the temperature of the 9110 processor module in the given slot in degrees centigrade. Set to 0 (zero) if no processor module is present.

#### *Processor Module B Temperature*

**Direction:** input to application from controller.

**Type:** Word.

**Values:**

- 0 to 65,535.

**Description:**

Reports the temperature of the 9110 processor module in the given slot in degrees centigrade. Set to 0 (zero) if no processor module is present.

#### *Processor Module C Temperature*

**Direction:** input to application from controller.

**Type:** Word.

**Values:**

- 0 to 65,535.

**Description:**

Reports the temperature of the 9110 processor module in the given slot in degrees centigrade. Set to 0 (zero) if no processor module is present.

## **Control Integers**

The variables in the rack of control integers enable the application to send specific information to the controller.

### *AUX LED Colour*

**Direction:** output from application to controller.

**Type:** Word.

**Values:**

- 0..3 (0 = off, 1 = red, 2 = green, 3 = amber)
- Default 0.

**Description:**

Sets the state of the LED indicator labelled 'Aux' on all 9110 processor modules.

## **Status Booleans**

The variables in the rack of Status Booleans provide information about the controller to the application.

### *System Health*

**Direction:** Input to application from controller.

**Type:** Boolean.

**Values:**

- TRUE = All installed processor and I/O modules are healthy and System Health variable is FALSE (See Control Booleans). The processor module System Healthy LED is green.
- FALSE = One or more of the installed processors and/or I/O modules are reporting a module health problem or the System Health variable is TRUE (See Control Booleans). The processor module System Healthy LED is red.

**TIP** The System Health alarm can be reset after a fault in a system having at least one healthy processor or I/O module in a module group.

### *System Health Reset (Voted 1oo3)*

**Direction:** input to application from controller.

**Type:** Boolean.

**Values:**

- TRUE = The **Fault Reset** button on a 9110 processor module has been pressed in the previous cycle.
- FALSE = No **Fault Reset** button is active.

**Default:** FALSE.

**Description:**

Reports that the **Fault Reset** button on any processor module has been pressed. The system health reset is triggered by pressing the button, but the value does not change to TRUE until the start of the next application cycle. The value remains TRUE for the cycle and then reverts to FALSE, even if the button has been pressed throughout.

*Dongle Detected (Voted)*

**Direction:** input to application from controller.

**Type:** Boolean.

**Values:**

- TRUE = One or more 9110 processor modules can see a program enable key at the KEY connector on the 9100 processor base unit.
- FALSE = No processor module can see a program enable key.

**Description:**

Reports the presence or absence of a program enable key.

*Processor Module A On-line*

**Direction:** input to application from controller.

**Type:** Boolean.

**Values:**

- TRUE = The 9110 processor module in the given slot is on-line
- FALSE = The processor module is off-line
- Default: TRUE

**Description:**

Reports that a processor module in a dual or triple modular redundant configuration is present and is communicating through the inter-processor link to one or both of its peers. Reports that a simplex processor module is present.

*Processor Module B On-line*

**Direction:** input to application from controller.

**Type:** Boolean.

**Values:**

- TRUE = The 9110 processor module in the given slot is on-line
- FALSE = The processor module is off-line
- Default: TRUE

**Description:**

Reports that a processor module in a dual or triple modular redundant configuration is present and is communicating through the inter-processor link to one or both of its peers. Reports that a simplex processor module is present.

*Processor Module C On-line*

**Direction:** input to application from controller.

**Type:** Boolean.

**Values:**

- TRUE = The 9110 processor module in the given slot is on-line
- FALSE = The processor module is off-line
- Default: TRUE

**Description:**

Reports that a processor module in a dual or triple modular redundant configuration is present and is communicating through the inter-processor link to one or both of its peers. Reports that a simplex processor module is present.

*Processor Module A Health*

**Direction:** input to application from controller.

**Type:** Boolean.

**Values:**

- TRUE = The 9110 processor module in the given slot is healthy and its Healthy LED indicator is green.
- FALSE = The processor module is faulty and its Healthy LED indicator is red.

**Description:**

Reports the health status of a processor module.

*Processor Module B Health*

**Direction:** input to application from controller.

**Type:** Boolean.

**Values:**

- TRUE = The 9110 processor module in the given slot is healthy and its Healthy LED indicator is green.
- FALSE = The processor module is faulty and its Healthy LED indicator is red.

**Description:**

Reports the health status of a processor module.

*Processor Module C Health*

**Direction:** input to application from controller.

**Type:** Boolean.

**Values:**

- TRUE = The 9110 processor module in the given slot is healthy and its Healthy LED indicator is green.
- FALSE = The processor module is faulty and its Healthy LED indicator is red.

**Description:**

Reports the health status of a processor module.

*Processor Module A 24 V1 Power Feed Health*

**Direction:** input to application from controller.

**Type:** Boolean.

**Values:**

- TRUE = power feed voltage is within specifications (18 to 32 Vdc).
- FALSE = power feed is outside specifications.

**Description:**

Reports the health of power feed 1 (nominal 24 Vdc) to the 9110 processor module in the given slot.

*Processor Module B 24 V1 Power Feed Health*

**Direction:** input to application from controller.

**Type:** Boolean.

**Values:**

- TRUE = power feed voltage is within specifications (18 to 32 Vdc).
- FALSE = power feed is outside specifications.

**Description:**

Reports the health of power feed 1 (nominal 24 Vdc) to the 9110 processor module in the given slot.

*Processor Module C 24 V1 Power Feed Health*

**Direction:** input to application from controller.

**Type:** Boolean.

**Values:**

- TRUE = power feed voltage is within specifications (18 to 32 Vdc).
- FALSE = power feed is outside specifications.

**Description:**

Reports the health of power feed 1 (nominal 24 Vdc) to the 9110 processor module in the given slot.

*Processor Module A 24 V2 Power Feed Health*

**Direction:** input to application from controller.

**Type:** Boolean.

**Values:**

- TRUE = power feed voltage is within specifications (18 to 32 Vdc).
- FALSE = power feed is outside specifications.

**Description:**

Reports the health of power feed 2 (nominal 24 Vdc) to the 9110 processor module in the given slot.

*Processor Module B 24 V2 Power Feed Health*

**Direction:** input to application from controller.

**Type:** Boolean.

**Values:**

- TRUE = power feed voltage is within specifications (18 to 32 Vdc).
- FALSE = power feed is outside specifications.

**Description:**

Reports the health of power feed 2 (nominal 24 Vdc) to the 9110 processor module in the given slot.

*Processor Module C 24 V2 Power Feed Health*

**Direction:** input to application from controller.

**Type:** Boolean.

**Values:**

- TRUE = power feed voltage is within specifications (18 to 32 Vdc).
- FALSE = power feed is outside specifications.

**Description:**

Reports the health of power feed 2 (nominal 24 Vdc) to the 9110 processor module in the given slot.

*Processor Module A Ready*

**Direction:** input to application from controller.

**Type:** Boolean.

**Values:**

- TRUE = The 9110 processor module in the given slot is synchronized (see description).
- FALSE = The processor module is out of synchronization or missing.

**Description:**

Reports that a processor module in a dual or triple modular redundant configuration is present and is synchronized with one or both of its peers.  
Reports that a simplex processor module is present.

*Processor Module B Ready*

**Direction:** input to application from controller.

**Type:** Boolean.

**Values:**

- TRUE = The 9110 processor module in the given slot is synchronized (see description).
- FALSE = The processor module is out of synchronization or missing.

**Description:**

Reports that a processor module in a dual or triple modular redundant configuration is present and is synchronized with one or both of its peers.  
Reports that a simplex processor module is present.

*Processor Module C Ready*

**Direction:** input to application from controller.

**Type:** Boolean.

**Values:**

- TRUE = The 9110 processor module in the given slot is synchronized (see description).
- FALSE = The processor module is out of synchronization or missing.

**Description:**

Reports that a processor module in a dual or triple modular redundant

configuration is present and is synchronized with one or both of its peers. Reports that a simplex processor module is present.

#### *Processor Module A NVRAM Battery Health*

**Direction:** input to application from controller.

**Type:** Boolean.

**Values:**

- TRUE = The back-up battery in the 9110 processor module in the given slot is present and its voltage is within acceptable limits.
- FALSE = The voltage of the back-up battery is low or the battery is missing.

**Description:**

Reports the health status of the back-up battery in a processor module. The battery voltage is checked at start up, then re-checked every 24 hours (elapsed time).

#### *Processor Module B NVRAM Battery Health*

**Direction:** input to application from controller.

**Type:** Boolean.

**Values:**

- TRUE = The back-up battery in the 9110 processor module in the given slot is present and its voltage is within acceptable limits.
- FALSE = The voltage of the back-up battery is low or the battery is missing.

**Description:**

Reports the health status of the back-up battery in a processor module. The battery voltage is checked at start up, then re-checked every 24 hours (elapsed time).

#### *Processor Module C NVRAM Battery Health*

**Direction:** input to application from controller.

**Type:** Boolean.

**Values:**

- TRUE = The back-up battery in the 9110 processor module in the given slot is present and its voltage is within acceptable limits.
- FALSE = The voltage of the back-up battery is low or the battery is missing.



**Description:**

Reports the health status of the back-up battery in a processor module. The battery voltage is checked at start up, then re-checked every 24 hours (elapsed time).

**Control Booleans**

The variables in the rack of Control Booleans enable the application to send specific information to the controller.

*Unlock All Locked Variables*

**Direction:** output from application to controller.

**Type:** Boolean

**Values:**

- TRUE = Remove all locks.
- FALSE = No effect.
- Default FALSE.

**Description:**

Removes all user locks on input and output variables.

*Set System Health Alarm*

**Direction:** output from application to controller.

**Type:** Boolean.

**Values:**

- TRUE = When the variable transitions from FALSE to TRUE, the system responds as if it had found a system level fault. The processor module System Healthy LED is set to RED and the System Health Boolean is set TRUE. When the variable is TRUE, the fault is immediately re-annunciated after pressing the **Fault Reset** button.
- FALSE = Does not send an alarm signal to the controller. After the variable transitions from TRUE to FALSE, the processor module System Healthy LED passes to GREEN after pressing the **Fault Reset** button.
- DEFAULT = FALSE.

**Description:**

Sends a System Health alarm signal from the application to the controller.

*HART Pass-Through*

**Direction:** output from application to controller.

**Type:** Boolean.

**Values:**

- TRUE = HART Pass-Through is enabled and available for an Analogue Module.
- FALSE = HART Pass-Through is disabled and not available.
- DEFAULT = FALSE.

**Description:**

Starts the HART Pass-Through feature and allows HART messages on analogue input and output modules. The system allows messages on each channel independently and together.

## RTC Status Variables

The RTC status variables provide information about the controller real-time clock to the application.

### *RTC Status: Year*

**Direction:** input to application from controller.

**Type:** Word.

**Values:**

- 2,000 to 2,399, or 0 (see description).

**Description:**

Reports the oldest value of real-time clock (RTC) year as voted by every 9110 processor module which is present and synchronized. Only updated if the real-time clock control Boolean RTC Read is set to TRUE. If RTC Read is FALSE, the value will be 0 (zero).

### *RTC Status: Month*

**Direction:** input to application from controller.

**Type:** Word.

**Values:**

- 1 to 12, or 0 (see description).

**Description:**

Reports the oldest value of real-time clock (RTC) month as voted by every 9110 processor module which is present and synchronized. Only updated if the real-time clock control Boolean RTC Read is set to TRUE. If RTC Read is FALSE, the value will be 0 (zero).

*RTC Status: Day of Month*

**Direction:** input to application from controller.

**Type:** Word.

**Values:**

- 1 to 31, or 0 (see description).

**Description:**

Reports the oldest value of real-time clock (RTC) day of the month as voted by every 9110 processor module which is present and synchronized. Only updated if the real-time clock control Boolean RTC Read is set to TRUE. If RTC Read is FALSE, the value will be 0 (zero).

*RTC Status: Hours*

**Direction:** input to application from controller.

**Type:** Word.

**Values:**

- 0 to 23.

**Description:**

Reports the oldest value of real-time clock (RTC) hours as voted by every 9110 processor module which is present and synchronized. Only updated if the real-time clock control Boolean RTC Read is set to TRUE. If RTC Read is FALSE, the value will be 0 (zero).

*RTC Status: Minutes*

**Direction:** input to application from controller.

**Type:** Word.

**Values:**

- 0 to 59.

**Description:**

Reports the oldest value of real-time clock (RTC) minutes as voted by every 9110 processor module which is present and synchronized. Only updated if the real-time clock control Boolean RTC Read is set to TRUE. If RTC Read is FALSE, the value will be 0 (zero).

*RTC Status: Seconds*

**Direction:** input to application from controller.

**Type:** Word.

**Values:**

- 0 to 59.

**Description:**

Reports the oldest value of real-time clock (RTC) seconds as voted by every 9110 processor module which is present and synchronized. Only updated if the real-time clock control Boolean RTC Read is set to TRUE. If RTC Read is FALSE, the value will be 0 (zero).

*RTC Status: Milliseconds*

**Direction:** input to application from controller.

**Type:** Word.

**Values:**

- 0 to 999.

**Description:**

Reports the oldest value of real-time clock (RTC) milliseconds as voted by every 9110 processor module which is present and synchronized. Only updated if the real-time clock control Boolean RTC Read is set to TRUE. If RTC Read is FALSE, the value will be 0 (zero).

## RTC Program Variables

The variables in the rack of RTC program variables specify parts of the date to be written to the real-time clock the next time the RTC control variable RTC Write is asserted TRUE. The values will be written only if the RTC control variable Year is TRUE.

*RTC Program: Year*

**Direction:** output from application to controller.

**Type:** Word.

**Values:**

- 2,000 to 2,399
- Default 0 (zero).

**Description:**

Specifies the year part of the date to write to the real-time clock the next time the RTC control variable RTC Write is asserted TRUE. The value is written only if the RTC control variable Year is TRUE.

*RTC Program: Month*

**Direction:** output from application to controller.

**Type:** Word.

**Values:**

- 1 to 12
- Default 0 (zero).

**Description:**

Specifies the number of the month part of the date to be written to the real-time clock the next time the RTC control variable RTC Write is asserted TRUE. The value is written only if the RTC control variable Month is TRUE.

*RTC Program: Day of Month*

**Direction:** output from application to controller

**Type:** Word.

**Values:**

- 1 to 31
- Default 0 (zero).

**Description:**

Specifies the day of the month part of the date to write to the real-time clock the next time the RTC control variable RTC Write is asserted TRUE. The value is written only if the RTC control variable Day is TRUE.

*RTC Program: Hours*

**Direction:** output from application to controller.

**Type:** Word.

**Values:**

- 0 to 23
- Default 0 (zero).

**Description:**

Specifies the time of day (in hours) to write to the real-time clock the next time the RTC control variable RTC Write is asserted TRUE. The value is written only if the RTC control variable Hours is TRUE.

*RTC Program: Minutes*

**Direction:** output from application to controller.

**Type:** Word.

**Values:**

- 0 to 59
- Default 0 (zero).

**Description:**

Specifies the time of day (in minutes) to write to the real-time clock the next time the RTC control variable RTC Write is asserted TRUE. The value is written only if the RTC control variable Minutes is TRUE.

*RTC Program: Seconds*

**Direction:** output from application to controller

**Type:** Word.

**Values:**

- 0 to 59
- Default 0 (zero).

**Description:**

Specifies the time of day (in seconds) to write to the real-time clock the next time the RTC control variable RTC Write is asserted TRUE. The value is written only if the RTC control variable Seconds is TRUE.

*RTC Program: Milliseconds*

**Direction:** output from application to controller.

**Type:** Word.

**Values:**

- 0 to 999
- Default 0 (zero).

**Description:**

Specifies the time of day (in milliseconds) to write to the real-time clock the next time the RTC control variable RTC Write is asserted TRUE. The value is written only if the RTC control variable Milliseconds is TRUE.

## RTC Control Variables

The variables in the rack of RTC control variables regulate updates to the real-time clock.

*RTC Control: RTC Write*

**Direction:** output from application to controller.

**Type:** Boolean.

**Values:**

- TRUE = Applies new values to real-time clock (see description).
- FALSE = No effect.
- Default FALSE.

**Description:**

Sets new values for the real-time clock. There are six values, all specified by the RTC program control words Year, Month, Day, Hours, Minutes and Seconds. Each value will be set only if its associated RTC control variable (which is a Boolean, and similarly named Year, Month, Day, Hours, Minutes or Seconds) is TRUE.

The change is initiated by the transition from FALSE to TRUE and actioned at the end of the application cycle. The application must hold the TRUE state at least until the end of the cycle for the clock to be updated.

There is no time limit on returning the value from TRUE to FALSE.

**Example**

Consider this scenario:

- The date is 28th October 2008, 8 hours, 12 minutes and 35 seconds
- **RTC Control RTC Read** is TRUE
- **RTC Control Year, Month** and **Day of Month** are TRUE
- **RTC Control Hours, Minutes** and **Seconds** are TRUE.

The RTC status variables will be returned, and the real-time clock will be set, like this:

- Year = 2,008
- Month = 10
- Day = 28
- Hours = 8
- Minutes = 12
- Seconds = 35.

*RTC Control: RTC Read*

**Direction:** output from application to controller.

**Type:** Boolean.

**Values:**

- TRUE = The controller updates RTC status values on each application cycle.
- FALSE = RTC status values are static (do not update).
- Default: FALSE.

**Description:**

Determines whether the RTC status variables (RTC Status: Year, RTC Status: Month, RTC Status: Day of Month, RTC Status: Hours, RTC Status: Minutes and RTC Status: Seconds) will update in real time.

All the RTC status variables must be set to TRUE when the RTC Read variable is set to TRUE, otherwise the RTC value will not be updated and reported.

*RTC Control: Year*

**Direction:** output from application to controller.

**Type:** Boolean.

**Values:**

- TRUE = RTC program year will be applied by RTC Write.
- FALSE = RTC program year will be ignored.
- Default FALSE until an initial value is specified in the application.

**Description:**

Defines whether the value of the RTC program variable named Year should be applied to the real-time clock the next time the RTC control variable named RTC Write is set to TRUE.

The RTC program variable is only updated if the RTC control variable RTC Read is set to TRUE and all other RTC Control variables are set to TRUE.

*RTC Control: Month*

**Direction:** output from application to controller.

**Type:** Boolean.

**Values:**

- TRUE = RTC program month will be applied by RTC Write.
- FALSE = RTC program month will be ignored.
- Default FALSE until an initial value is specified in the application.

**Description:**

Defines whether the value of the RTC program variable named Month should be applied to the real-time clock the next time the RTC control variable named RTC Write is set to TRUE.



The RTC program variable is only updated if the RTC control variable RTC Read is set to TRUE and all other RTC variables are set to TRUE.

#### *RTC Control: Day of Month*

**Direction:** output from application to controller.

**Type:** Boolean

**Values:**

- TRUE = RTC program day of month will be applied by RTC Write.
- FALSE = RTC program day of month will be ignored.
- Default FALSE until an initial value is specified in the application.

**Description:**

Defines whether the value of the RTC program variable named Day of Month should be applied to the real-time clock the next time the RTC control variable named RTC Write is set to TRUE.

The RTC program variable is only updated if the RTC control variable RTC Read is set to TRUE and all other RTC Control variables are set to TRUE.

#### *RTC Control: Hours*

**Direction:** output from application to controller.

**Type:** Boolean.

**Values:**

- TRUE = RTC program hours will be applied by RTC Write.
- FALSE = RTC program hours will be ignored.
- Default FALSE until an initial value is specified in the application.

**Description:**

Defines whether the value of the RTC program variable named Hours should be applied to the real-time clock the next time the RTC control variable named RTC Write is set to TRUE.

The RTC program variable is only updated if the RTC control variable RTC Read is set to TRUE and all other RTC Control variables are set to TRUE.

#### *RTC Control: Minutes*

**Direction:** output from application to controller.

**Type:** Boolean.

**Values:**

- TRUE = RTC program minutes will be applied by RTC Write.

- FALSE = RTC program minutes will be ignored.
- Default FALSE until an initial value is specified in the application.

**Description:**

Defines whether the value of the RTC program variable named Minutes should be applied to the real-time clock the next time the RTC control variable named RTC Write is set to TRUE.

The RTC program variable is only updated if the RTC control variable RTC Read is set to TRUE and all other RTC Control variables are set to TRUE.

*RTC Control: Seconds*

**Direction:** output from application to controller.

**Type:** Boolean.

**Values:**

- TRUE = RTC program seconds will be applied by RTC Write.
- FALSE = RTC program seconds will be ignored.
- Default FALSE until an initial value is specified in the application.

**Description:**

Defines whether the value of the RTC program variable named Seconds should be applied to the real-time clock the next time the RTC control variable named RTC Write is set to TRUE.

The RTC program variable is only updated if the RTC control variable RTC Read is set to TRUE and all other RTC Control Variables are set to TRUE.

*RTC Control: Milliseconds*

**Direction:** output from application to controller.

**Type:** Boolean.

**Values:**

- TRUE = RTC program milliseconds will be applied by RTC Write.
- FALSE = RTC program milliseconds will be ignored.
- Default FALSE until an initial value is specified in the application.

**Description:**

Defines whether the value of the RTC program variable named Milliseconds should be applied to the real-time clock the next time the RTC control variable named RTC Write is set to TRUE.

The RTC program variable is only updated if the RTC control variable RTC Read is set to TRUE and all other RTC Control Variables are set to TRUE.

## Set the Processor Clock

To set the processor clock you can use the RTC variables:

1. To monitor the time, wire variables to all the RTC Status points.
2. To configure the time to be set, wire variables to the RTC Program points.
  - Hours
  - Minutes
  - Seconds.
3. Preset all RTC Program variables to the time that is to be programmed.
4. It is recommended that you set the time to 08:00 (or similar). Do not set the time to midnight.
5. To control the time setting, wire variables to the RTC Control points:
  - RTC Write
  - RTC
  - Read.
6. Wire variables to RTC Control:
  - Hours
  - Minutes
  - Seconds.
7. Set RTC Read to be always True. This enables the time to be written.
8. Set RTC Control elements Hours, Minutes and seconds to be always True.
9. Use an external trigger to change RTC Write from False to True at the correct time.
  - The time will be set into the Real Time Clock.

---

**IMPORTANT** On the very first setting it will be necessary to program all the time elements manually (Year, Month, Day, etc.)

---



## Using the Dictionary

This chapter introduces the Dictionary in the AADvance Workbench. It explains how to create and modify variables, and store them in the Dictionary.

### About the Dictionary

The Dictionary is a database which holds all your application variables. You should define all your variables at the start of a new project. Include processor and I/O module variables and the I/O channel variables.

**TIP** You can define new variables while creating a program; however, you will be prompted to add the new variable to the Dictionary.

The AADvance Workbench automatically assigns a set of variable elements when you select the variable type. The Variables Grid dialog box provides fields for you to enter or change the properties of your variables. Double-clicking on each variable field will open a dialog box that allows you to enter the relevant attributes for the variable.

### Properties for AADvance Variables

**Name:** Limited to 128 characters. Must begin with a letter or underscore character followed by letters, digits and single underscore characters.

- Example: di\_full.

---

**IMPORTANT** A name cannot have two consecutive underscore characters.

---

**Alias:** A name, used in LD Editor. The variable comment text truncated before the ':' character, and limited to 16 characters including spaces.

**Wiring:** Read only cell, generated by the Workbench, gives the I/O point that the variable is wired to. Uses the syntax of a directly represented variable to represent a free channel, which is a channel that is not linked to a declared variable. The identifier of a directly represented variable is always %.

- Example: %IXs.c; where 'I' is an input, 'X' is a Boolean, 's' is the slot number of the I/O module and 'c' is the channel number.

**Type:** Standard IEC 61131 types as a set of pull down options: Boolean (Bool), Short Integer (SINT), Unsigned Short Integer (USINT), BYTE, Integer (INT), Unsigned Integer (UNIT), WORD, Double Integer (UDINT), Double Word (DWORD), Long Integer (LINT), Unsigned Long Integer (ULINT), REAL, Long Real (LREAL), Timer (TIME), DATE,

**STRING.** Type also includes the standard AADvance structures and user defined structures.

[ ]: If Type is a string, this property defines the string length (maximum 255 characters).

**Init Value:** Initial variable value, numeric or textual.

**Dimensions:** The size (number of elements) of an Array.

- Example: [1..4, 1..7].

**Group:** Group name, or none.

**Attribute:** The following attribute values can be set:

- Read: Read-only variable with an initial value.
- Write: Write only variable with an initial value.
- Free: Variable that can be used for reading or writing.

**Scope:** Global or local to a program or function.

**Direction:** Sets the Variable and/or device direction:

- Input: Variable connected to an input device (refreshed by the system).
- Output: Variable connected to an output device.
- Internal: Internal variable updated by the programs or communication.

When you set the direction of the variable, you have to apply these rules to the attribute setting:

- An input variable can only have a read attribute setting.
- An output variable can only have a write or free attribute setting.
- An internal variable can have any attribute setting.

**Retain Variable Attribute:** Yes or No. The current value of a retain variable is saved by the controller at each application cycle (if the battery is fitted and is healthy), and the value stored is restored if the controller stops and restarts.

**Address:** Not used.

**Comment:** User comments. Free format.

**Message FALSE:** Message defined for the FALSE value messages. This applies to Boolean variables and can be used for Sequence of Events or OPC to provide an option for a user defined message.

- Example: 'Pump off'.

**Message TRUE:** Message defined for the TRUE value messages. This applies to Boolean variables and can be used for SOE or OPC to provide an option for a user defined message.

- Example: 'Pump on'.

**Modbus:** MODBUS address of the variable, range 1 to 65,536.

**SOE:** Defines how Sequence of Events (SOE) is enabled for a Boolean variable. SOE can trigger on a falling edge or rising edge or both.

**Write Access:** Indicates that a variable can be written to an external client (True). The default value is False.


---

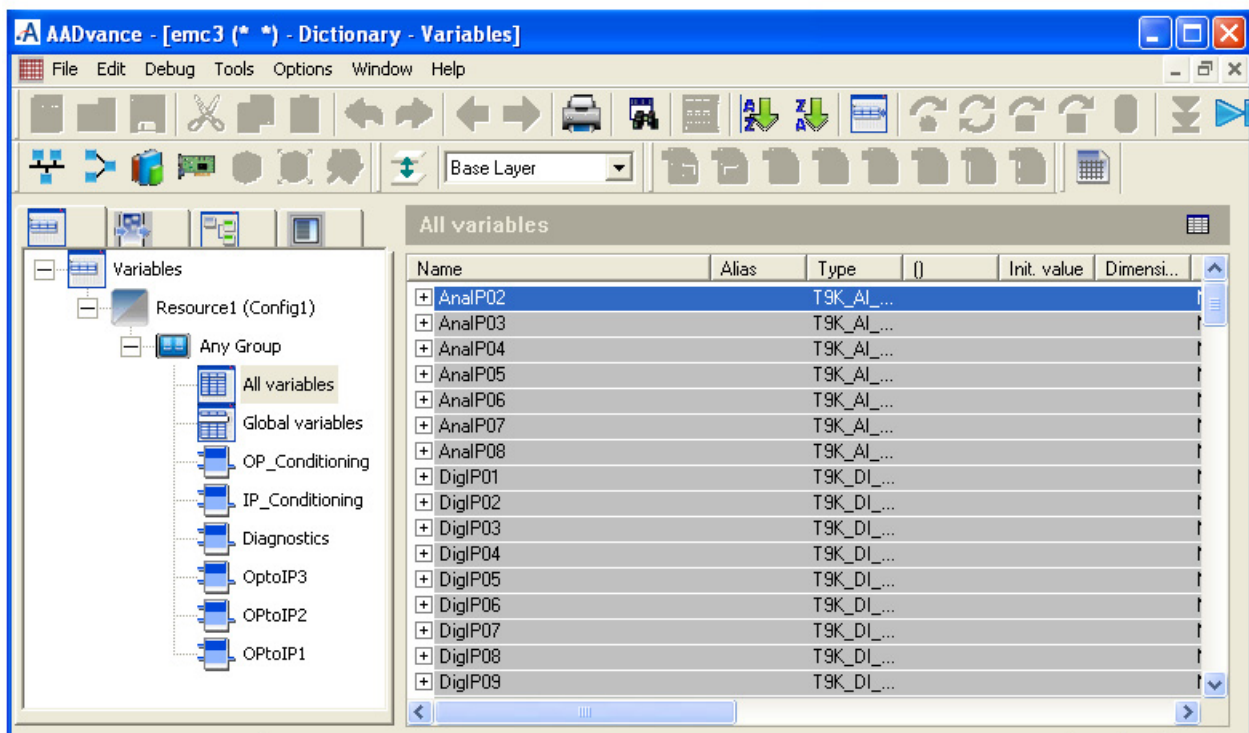
**IMPORTANT** You must now set the Write Access parameter to TRUE if you need to write to that attribute.

---

**CIP:** This indicates that a variable uses CIP or not. Double-clicking on the cell opens a dialog box with a **CIP** tab that is used to set the CIP attributes. It is set to either a producer or a consumer. When it is blank for non CIP variables and a shaded cell indicates that CIP is not available to that variable.

## Create or Modify Variables in the Dictionary

To open the Dictionary view select the dictionary button . The Variable Grid dialog box opens.




Create new variables or modify the properties of variables using this interface. You can edit the contents of individual cells or complete rows.

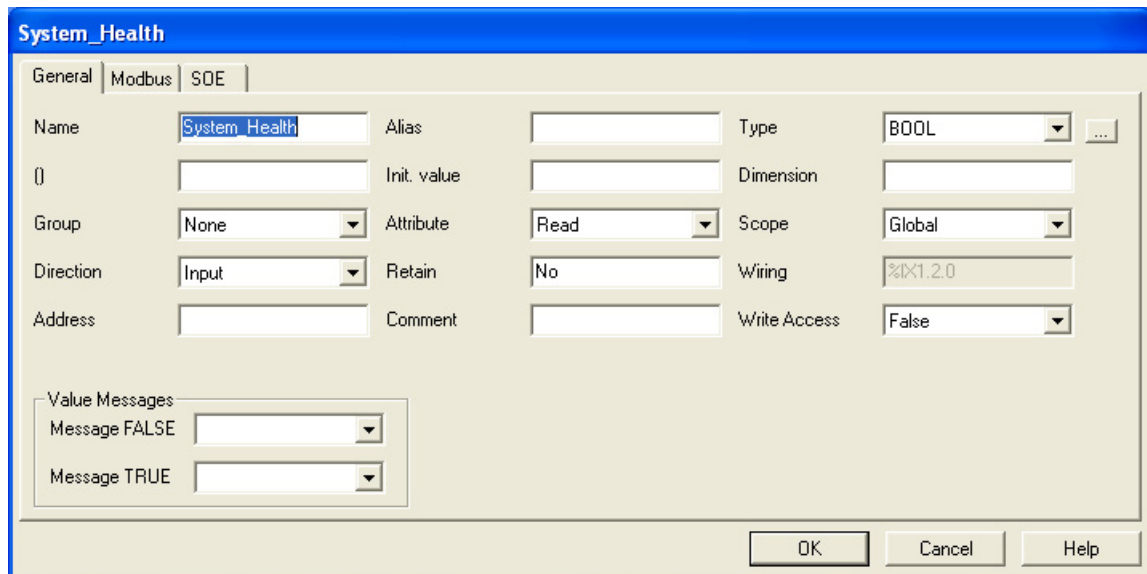
- When defining variables, you need to specify the general properties and, if required, the Modbus and SOE properties.
- For Boolean variables, you can also specify custom values for the FALSE and TRUE messages.

- For complex variables the wired elements set to read or write must have the ATTRIBUTE property set to FREE and the DIRECTION defined as INPUT, OUTPUT or INTERNAL.

## Edit the Contents of a Cell in the Dictionary

Depending on the variable type there are three different variable dialog boxes as shown below. To edit the contents of a cell in the Dictionary do the following:

1. Select grid mode by clicking the  button.
  - A blue line will highlight the first element of the first row.
2. Select the element you want to change within a row, then double-click on the element.
  - The variable dialog box opens.
3. Enter or make the necessary changes to the variable properties in the fields shown the following dialog boxes:



The image shows a dialog box titled "System\_Health" with three tabs: "General", "Modbus", and "SOE". The "General" tab is selected. The dialog box contains various fields for configuring a variable. The "Name" field is "System Health", "Alias" is empty, "Type" is "BOOL", "Init. value" is empty, "Dimension" is empty, "Group" is "None", "Attribute" is "Read", "Scope" is "Global", "Direction" is "Input", "Retain" is "No", "Wiring" is "%IX1.2.0", "Address" is empty, "Comment" is empty, and "Write Access" is "False". There is a "Value Messages" section with "Message FALSE" and "Message TRUE" dropdowns. At the bottom are "OK", "Cancel", and "Help" buttons.

Field	Value
Name	System Health
Alias	
Type	BOOL
Init. value	
Dimension	
Group	None
Attribute	Read
Scope	Global
Direction	Input
Retain	No
Wiring	%IX1.2.0
Address	
Comment	
Write Access	False

Value Messages

Message	Value
Message FALSE	
Message TRUE	

OK Cancel Help



**Slot10\_STA\_Chan1**

General | Modbus

Name	Slot10_STA_Chan1	Alias		Type	UINT	...
()		Init. value		Dimension		
Group	None	Attribute	Free	Scope	Global	
Direction	Internal	Retain	No	Wiring		
Address		Comment		Write Access	False	

Value Messages

Message FALSE

Message TRUE

OK Cancel Help

**Slot10\_Chan1**

General | CIP

Name	Slot10_Chan1	Alias		Type	T9K_AI_FULL	...
()		Init. value		Dimension		
Group	None	Attribute	Read	Scope	Global	
Direction	Input	Retain	No	Wiring	%I110.1.0,%I110.2	
Address		Comment		Write Access	6(False)	

Value Messages

Message FALSE

Message TRUE

OK Cancel Help

## Edit the Contents of a Row in the Dictionary

The screenshot shows a dialog box titled "System\_Health". It has three tabs: "General", "Modbus", and "SOE". The "General" tab is selected. The fields are as follows:


Name	System Health	Alias		Type	BOOL
()		Init. value		Dimension	
Group	None	Attribute	Read	Scope	Global
Direction	Input	Retain	No	Wiring	%IX1.2.0
Address		Comment		Write Access	False


Below the fields, there is a section for "Value Messages":

- Message FALSE: [dropdown]
- Message TRUE: [dropdown]

At the bottom right, there are buttons for "OK", "Cancel", and "Help".

To edit the contents of a row in the Dictionary do the following:

1. Select line mode by clicking the  button.
2. Double-click on an empty line to create a new variable; double-click on an existing line to modify a variable.
  - A variable properties dialog box opens.
3. Make the necessary changes to the variable fields.

**TIP** For the Type field you can also select the Select Data Types browser by clicking .

4. To define the MODBUS service properties for the variable, select the **Modbus** tab, then define the required fields.
5. To define the SOE service properties, select the **SOE** tab, then define the required fields.

## SOE Service Parameters

The AADvance Workbench supports a Sequence of Events (SOE) service for Boolean variables. When a Sequence of Events service is defined, use the **SOE** tab of the Variable dialog box to define the SOE properties:

The screenshot shows the 'DigIP09.DI' Variable dialog box with the 'SOE' tab selected. The 'General' tab is also visible. The 'SOE' tab contains the following controls:

- ☐ Falling-Edge: Severity level [dropdown]
- ☐ Rising-Edge: Severity level [dropdown]
- Filter time: [text box]
- Reference variable: [text box] [Select] [Remove]

At the bottom of the dialog are the 'OK', 'Cancel', and 'Help' buttons.

- **Falling-edge:** the indication of whether the service detects a fall from TRUE to FALSE. When this box is checked, the falling edge will be detected; when it is not checked, the transition will be ignored.
- **Rising-edge:** the indication of whether the service detects a rising edge. When this box is checked, the rising edge will be detected; when it is not checked, the transition will be ignored.
- **Filter time:** the minimum time lapse between two events. Permitted values range from 0 to 65,535 ms; the default value is 0 ms.
- **Reference Variable:** the variable whose value is displayed for a rising-edge or falling-edge event. The supported data types for this variable.



## Configuring the Controller I/O

This chapter describes the configuration process for defining the controller I/O hardware in the AADvance Workbench.

### About Configuring I/O Modules

I/O modules are configured by selecting an I/O bus in the Equipment tree view and then assigning a module to an empty I/O slot. You can configure single modules or two/three modules to form a redundant group to match the arrangement of your hardware.

If you choose to insert one module it will be allocated to the slot you have selected. If you choose to insert more than one module, they will automatically be allocated to adjacent slots. To change the configuration you can clear a slot or move a module to another slot.

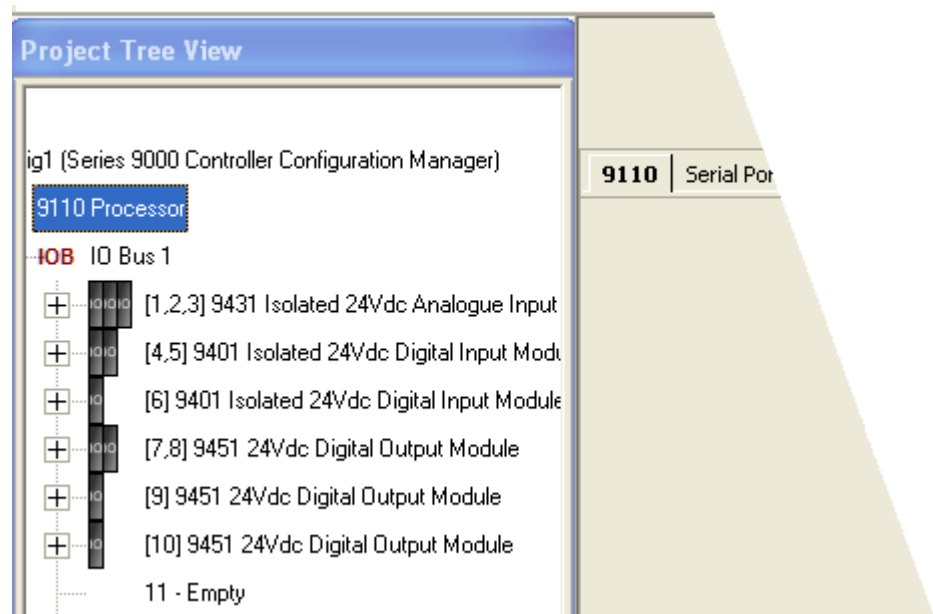
Use this process to configure I/O modules:

- Assign I/O modules to the IO Bus 1 or IO Bus 2 slots.
- Set the process safety time for the I/O modules.
- Configure the I/O module status variables.
- Configure the I/O module channel variables.

The procedures assume that you have already set up all the variables in the Dictionary. If you create a new variable during this process, you will be prompted to store it in the Dictionary.

### Example I/O Slot Configuration

In the example illustrated, the modules have been configured as follows:



- A redundant group of analogue input modules has been configured in the first three slots.
- Two digital input modules are in the next two slots.
- There is a single digital input module.
- There is a group of two digital output modules.
- Two single digital output modules.

---

**IMPORTANT** For Release 1.3 you can change the I/O module configuration with an on-line update. For a simplex configuration this will interrupt the data flow. If you are running an earlier version than R1.3 then you cannot use the on-line update functionality.

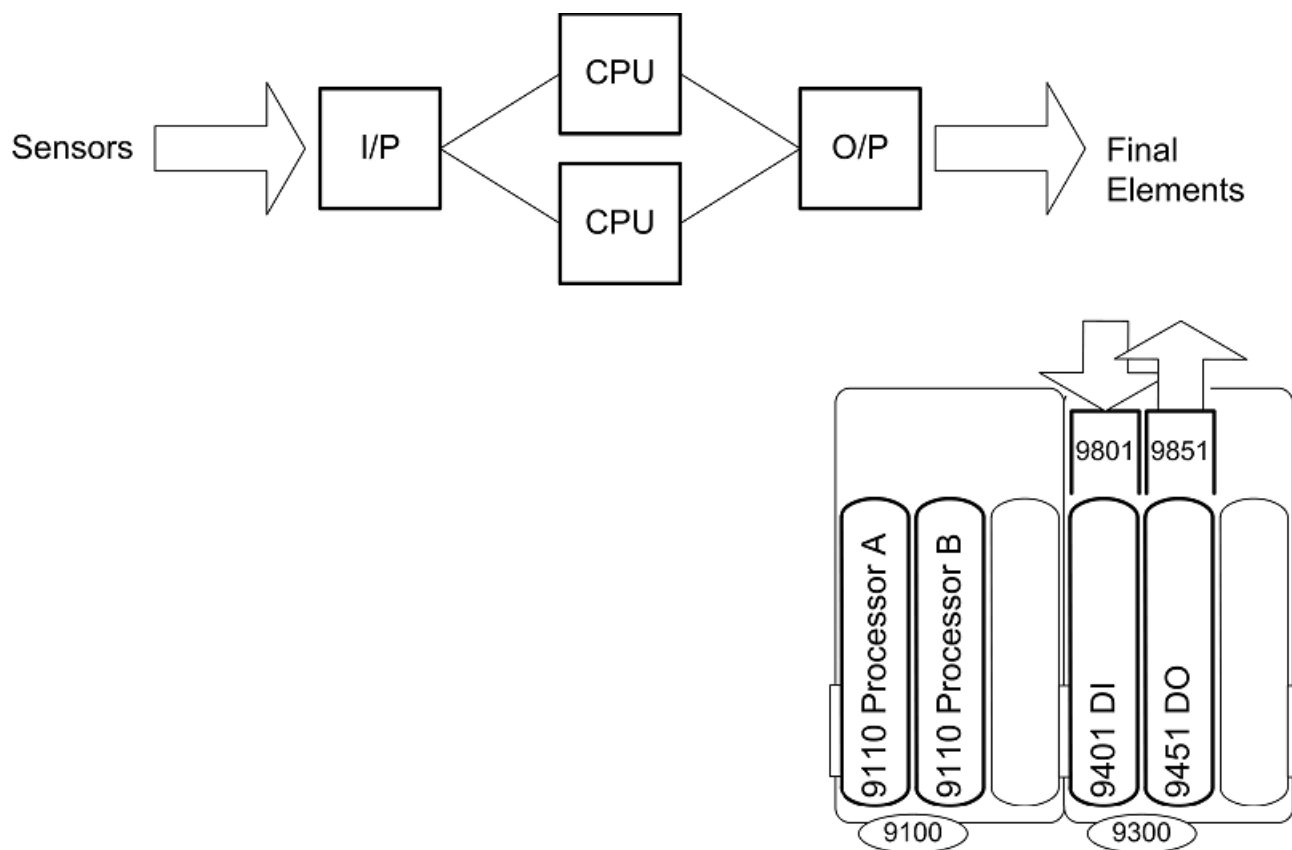
---

### Defining the I/O Hardware Architecture

The I/O hardware architecture is the physical arrangement of the I/O modules in the AADvance controller. To define the I/O hardware architecture in the AADvance Workbench you assign the modules to empty slot numbers on the processor I/O bus. Use the Project Tree View to do this. If desired, you can clear an I/O slot or move an assigned module to a different slot.

### Example Controller Configuration

This example controller has two 9110 processor modules and supports eight digital inputs and eight digital outputs.

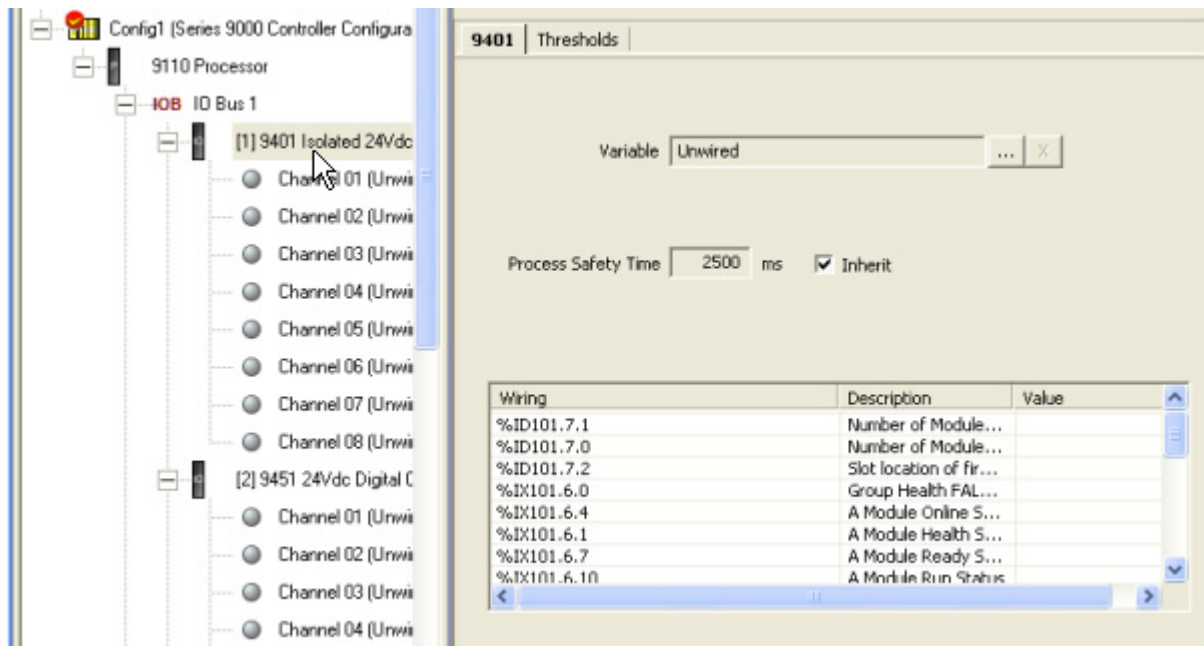
**Figure 6 - Dual Processor System**

The '9801' and '9851' illustrated are simplex termination assemblies for the I/O modules and provide the connections for the field elements.

This controller has the following physical layout: the two I/O modules are installed to the right of the processor base unit, which is IO Bus 1. The 9401 is installed in the first I/O base unit connector, which is slot 1; the 9451 is installed adjacent to the 9401 in the next connector on the I/O base unit, which is slot 2.

You now have to configure the same arrangement in the project tree, connect variables to monitor module status information and I/O data.

Use the Project Tree View within the AADvance Workbench to assign the I/O modules to empty slot numbers on the processor IOB IO Bus 1 or IO Bus 2.



The slot and bus numbers must be the same as the actual physical position of the installed modules. Therefore, for this example you would allocate I/O modules as follows:

- A 9401 module to the empty slot 1 on IO Bus 1
- A 9451 module to the empty slot 2 on IO Bus 1.

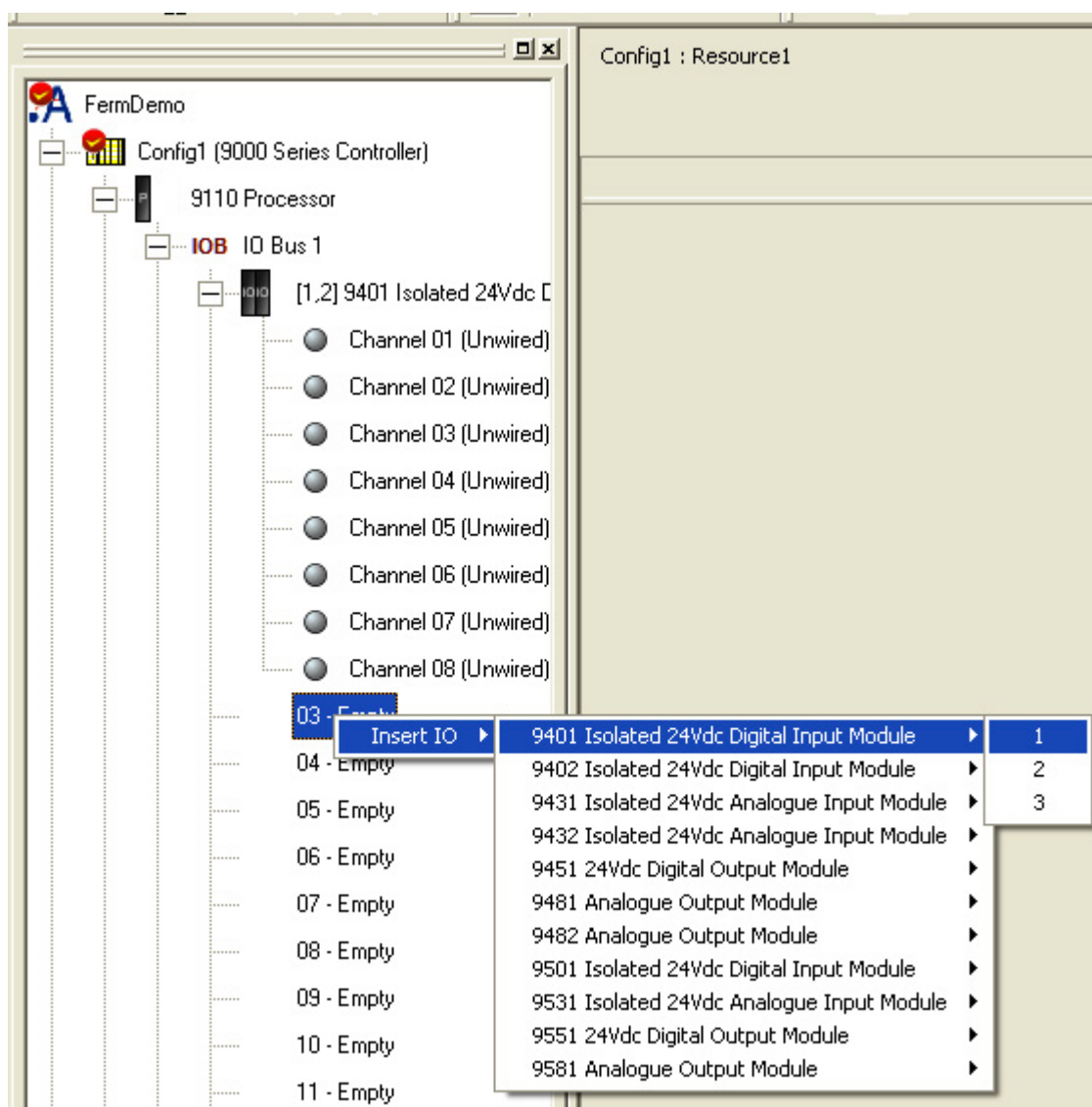
**Assign I/O Modules to I/O Bus Slots**

If you are assigning a single module you can assign the module to any empty I/O bus slot. If you are creating a redundant group you have to find two or three consecutive empty slots and assign the module to the first empty slot in the group.

**TIP** If required, you can use the AADvance Workbench to move configured modules to other slots to create a series of adjacent empty slots. Remember to move the actual modules in the controller to the changed slots.



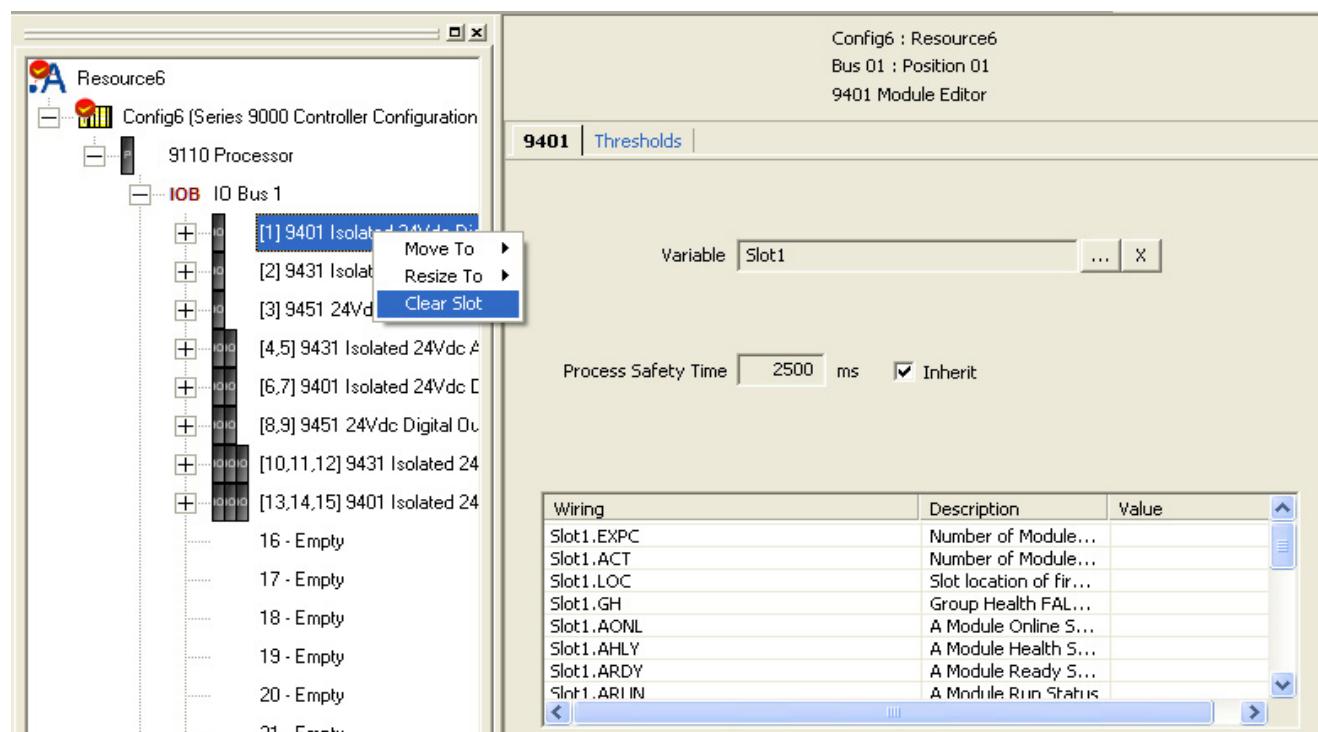
To assign an I/O module, do the following:



1. Select the **Equipment** tab.
2. Expand the IO Bus 1.
3. At an empty slot right-click to select **Insert IO**. Move the cursor to the right to select from the choice of available modules (empty slots position 16 - 24 shown in bold).
4. Select the required module, then move the cursor to the right and select the number of modules you require.
5. Repeat for each I/O Module as required.

## Clear an I/O Bus Slot

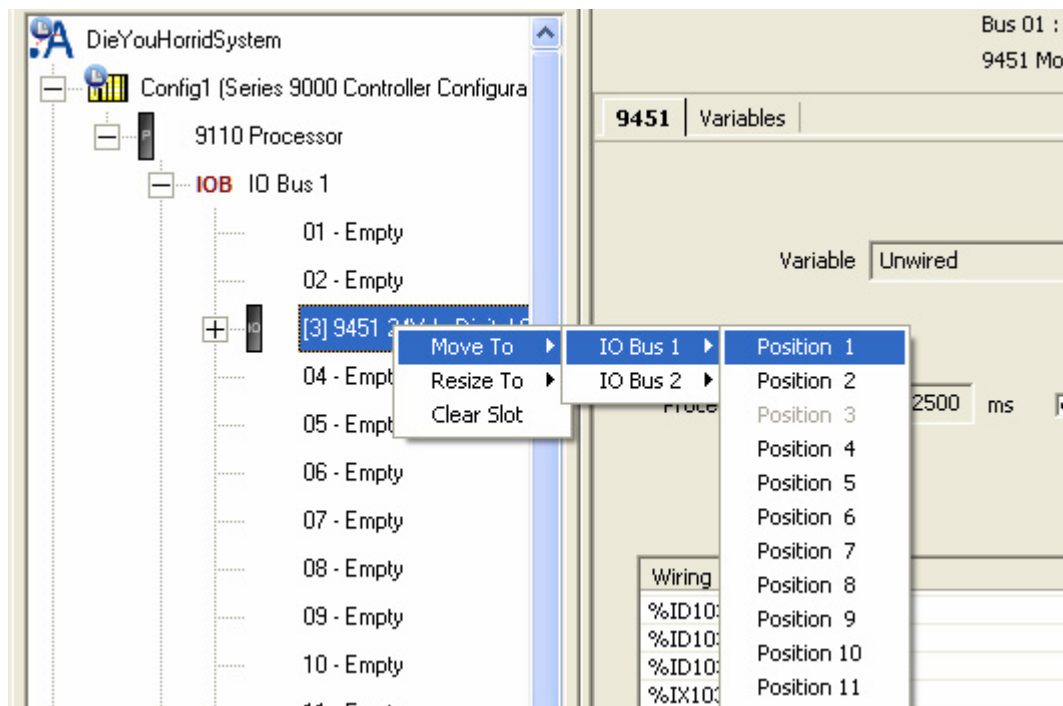
To clear an I/O bus slot do the following:



1. Select the **Equipment** tab.
2. Right-click to select **Clear Slot**.
  - The module will be removed from the slot.
  - The slot will now display as Empty ready to be re-assigned a module.
  - The channel can now be re-assigned, if required.

## Move a Module to a Different Slot

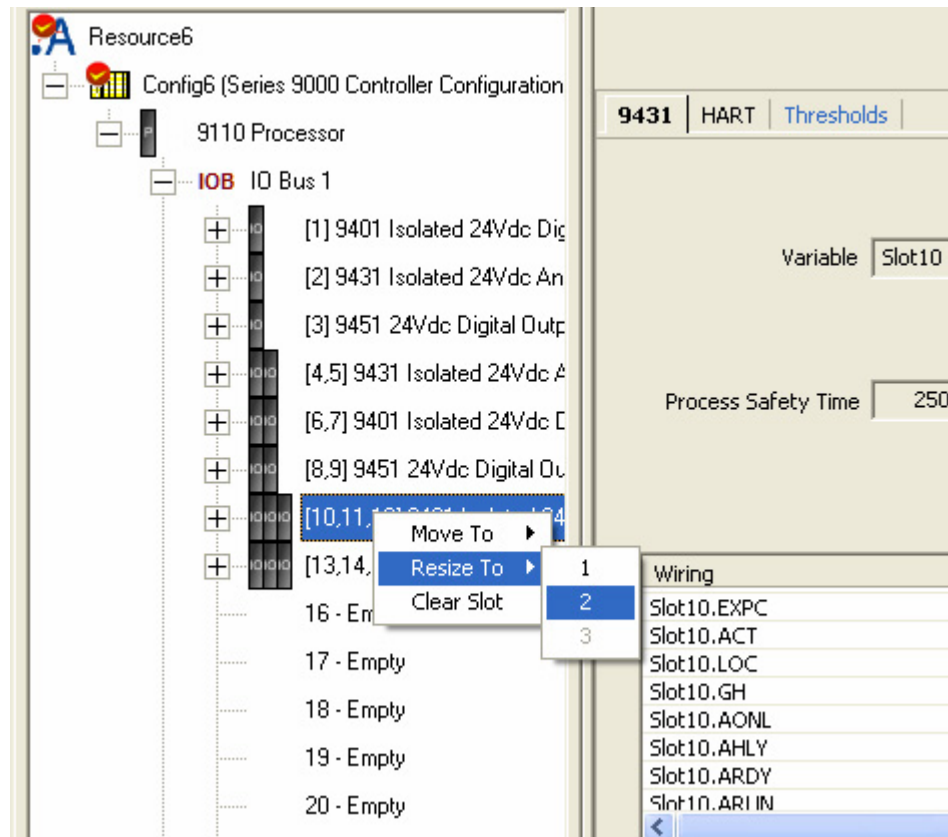
To move an assigned module to a different slot do the following:



1. Select the **Equipment** tab.
2. Right-click to select the **Move To** option.
  - Move the cursor to the right and select the IO Bus 1 position you want to move to.
  - The module is automatically re-assigned to the selected slot.
  - The channel variable wiring will move with the module to the new slot.

## Resize a Group of Modules

You can reduce the size of a redundant group of modules from a triple to a double then a double to a single module. To reduce the size do the following:



1. Select the **Equipment** view.
2. Right-click the required channel and select **Resize To**.
3. Select the number of modules you want to reduce or increase the size to.
  - One module will be removed from a group to become a single module slot or added to become a group from a single slot.
4. Repeat as required.

## Change the I/O Configuration with an On-line Update

You can change the hardware configuration by adding or removing a module(s) from the controller, then changing the configuration by doing an on-line update. You can replace an existing module with a different type by deleting the existing module from the configuration and adding the new module. You can

also remove a module from a redundant group without interrupting the data flow from the field elements.

---

**IMPORTANT** It is recommended that you change the module TA type before changing the module type using an on-line update. To change the module type first do an update to remove the module type then an update to add the new module type.

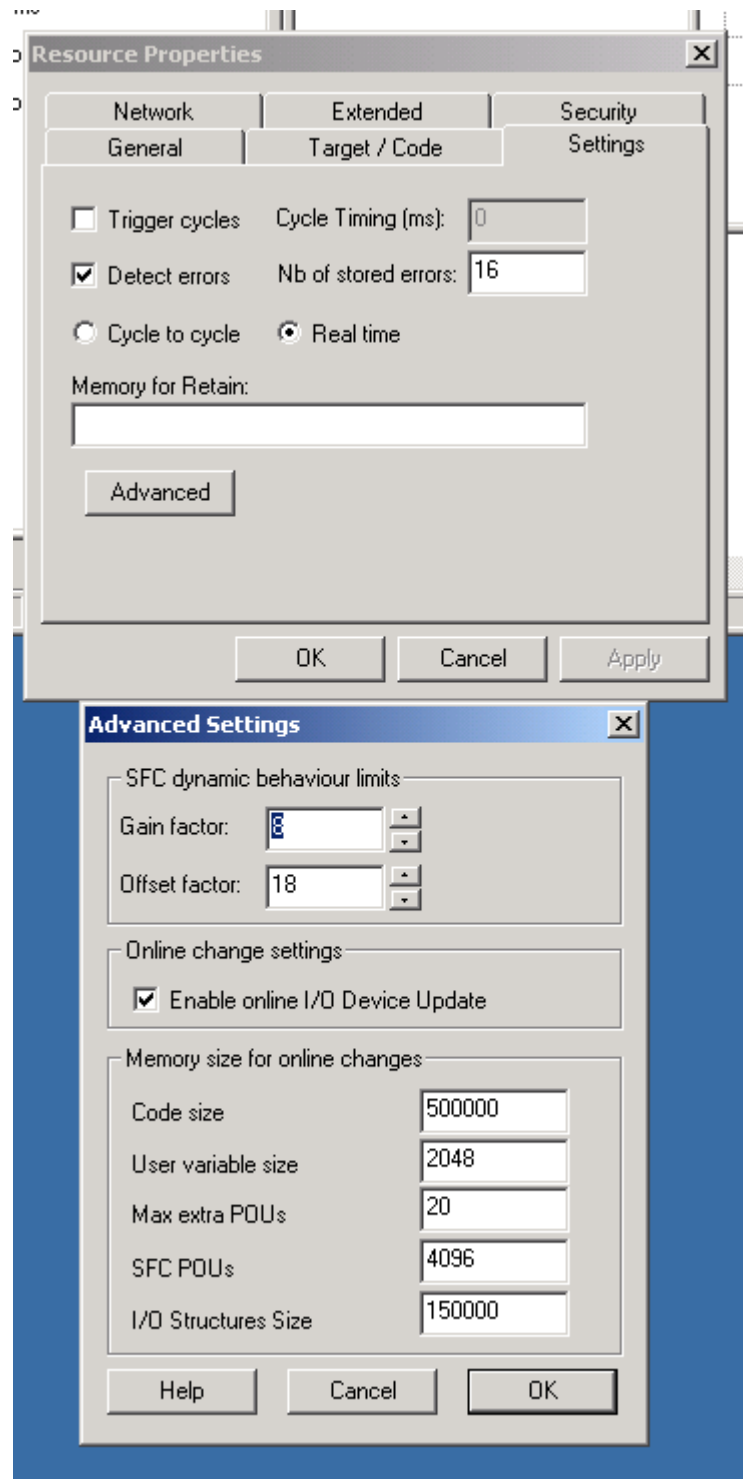
If you remove a module from a simplex configuration you will interrupt the data flow.

---

### **Activate the On-line Update Option**

1. Select a resource and right click on the Resource Header.
2. Select the **Properties** option.
3. Select **Settings** then **Advanced**.

4. Select the Enable online I/O Device Update check box.




#### Add or remove the hardware from the controller.

1. Select the resource and the Equipment tree view.
2. Select the slot and either add or remove the I/O module.

3. Save the change.

4. Select the Build Project Library button  to compile the application.

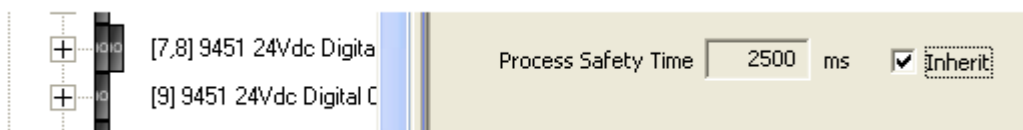
5. Select the On-line Change: Download button  to do an on-line update.

- The application is now downloaded to confirm the configuration change.

## Configure the I/O Module Process Safety Time

When you configure the process safety time for an I/O module, you can choose to inherit the top-level value set for the processor or specify a value for the I/O module.

To define the I/O module process safety time do the following:

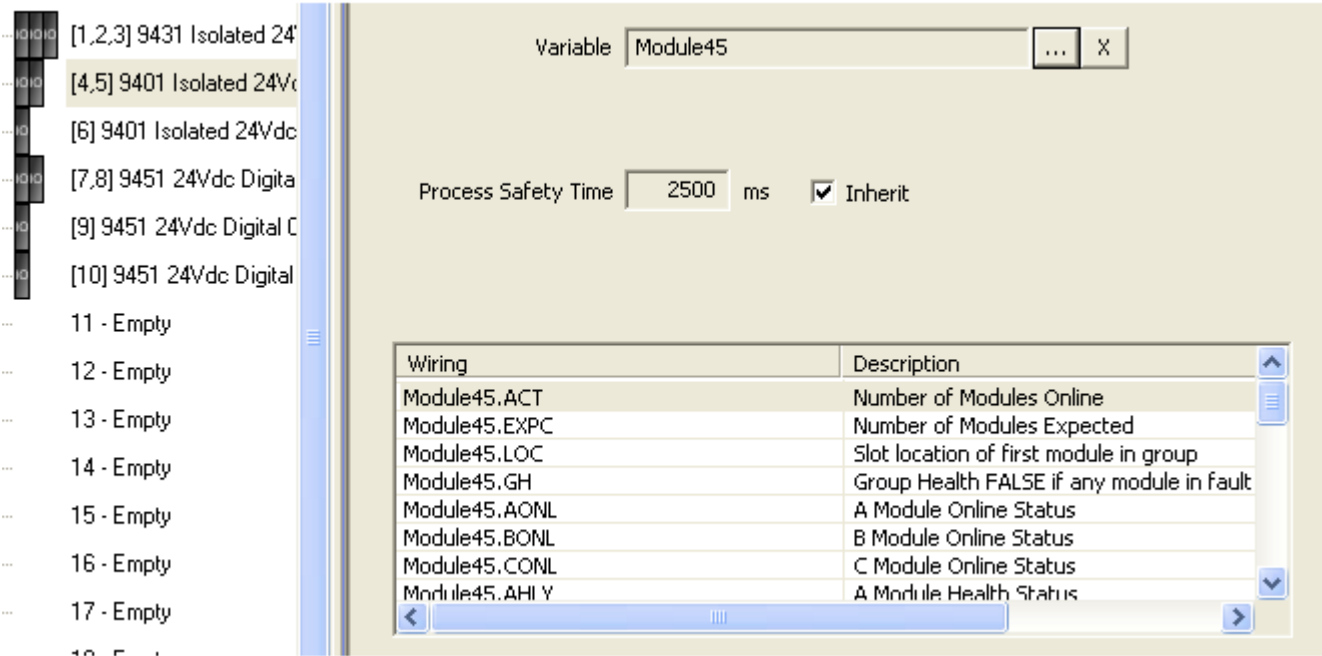


1. Select a module from the I/O Bus.
2. Put a tick in the Inherit box to inherit the top-level Process Safety Time; or
3. De-select the Inherit box and enter a value into the Process Safety Time field.

## Wire Status Variables to I/O Modules

You can wire a variable to an I/O module so the application can receive status information from the module. The AADvance Workbench provides a

structure (T9K\_TA\_GROUP\_STATUS) for module status information. To wire a status variable to an I/O module do the following:



1. Declare a variable in Dictionary. Use the type T9K\_TA\_GROUP\_STATUS and make sure that the direction is set to input.
2. Select a module from the I/O bus.
  - The unwired term appears in the Variable field.
3. Click the [...] button adjacent to the Variable field.
  - The variables dialog box opens.
4. Select the variable you declared in the Dictionary.
  - The variable is displayed in the Variable field.
  - The status variables are automatically assigned and appear in the Wiring column with the description in the Description column.
5. Repeat steps 1 to 4 for other I/O modules.

**T9K\_TA\_GROUP\_STATUS (I/O Module Status Information)**

The data structure for module status information (T9K\_TA\_GROUP\_STATUS) provides the elements detailed in the table.

The controller interrogates an I/O module (designated 'X' in the table) according to the physical arrangement of the module and its position in a group. A simplex module is designated as module A; a duplex module as A or B and a triplicated module as A, B or C.



**Table 7 - Structure for I/O Module Status Data**

Identifier	Type	Description	Remarks
<tagname>.EXPC	INT	Modules expected	Reports the number of modules that are defined in the configuration for the group (1, 2 or 3)
<tagname>.ACT	INT	Modules on-line	Reports the number of modules in a group that are installed, powered, locked and communicating over the I/O bus (1, 2 or 3)
<tagname>.LOC	INT	Slot location	Reports the slot number of the left-most module position for a group, irrespective of whether a module is physically located in a slot (1 to 24) (†)
<tagname>.GH	BOOL	Group health	Reports the general health status of all modules in a group <ul style="list-style-type: none"> <li>• TRUE: all modules are healthy</li> <li>• FALSE: one or more modules in the group is on-line and reporting a fault</li> </ul>
<tagname>.XONL	BOOL	On-line status	Reports the on-line status of module X TRUE: the module is installed, powered, locked and is communicating over the I/O bus, otherwise FALSE
<tagname>.XHLY	BOOL	Health status	Reports the general health of module X TRUE: the module is on-line and has no faults, otherwise FALSE
<tagname>.XRDY	BOOL	Ready status	Reports the ready status of module X TRUE: the module is on-line and ready to report channel values, otherwise FALSE
<tagname>.XRUN	BOOL	Run status	Reports the run status of module X TRUE: the module is on-line and reporting channel values, or requires manual intervention (pressing the <b>Fault Reset</b> button) before values can be reported, otherwise FALSE
<tagname>.XSDN	BOOL	Shutdown status	Reports that module X requires manual intervention (pressing the <b>Fault Reset</b> button) before values can be reported TRUE: the module needs manual intervention
<tagname>.XPOS	INT	Position	Reports the slot number of module X (1 to 24) (†)

(†) Slots are numbered 1 to 24 on each bus; the location (.LOC) and position (XPOS) do not identify the bus.

## About Configuring I/O Channels

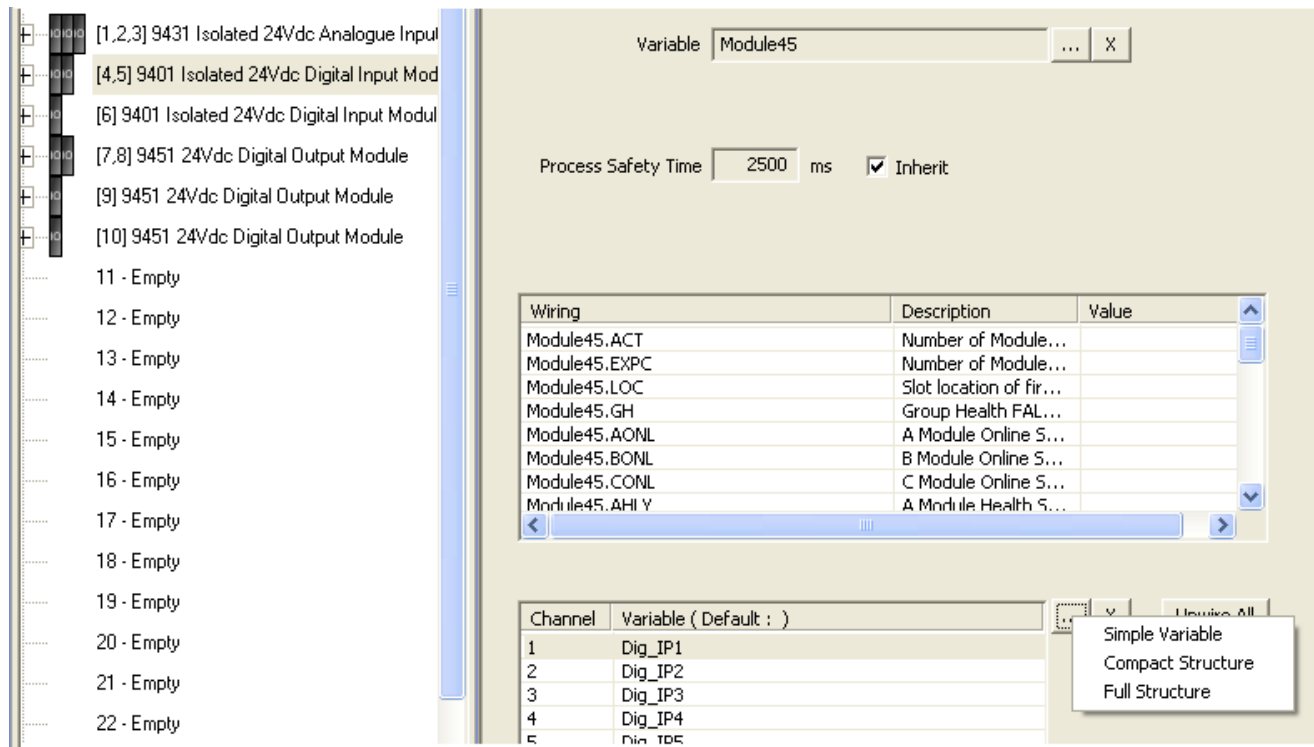
The AADvance Workbench provides a set of variable structures to wire to I/O channels. When you declare the I/O channels variables using your own tagnames you can declare one of two types of structure (compact and full); alternatively, the primary variable can be assigned directly to the base variable type.


The AADvance Workbench will automatically generate a set of variable elements with the same tagname; thus depending on the chosen structure, the system automatically wires a set of I/O variables to the channels.

The syntax for a structure variable is <tagname>.XX where XX represents the reporting element of the variable; for example, <tagname>.DI is a Boolean that reports the digital input state for a channel.

## Wire Variables to Digital Input Channels

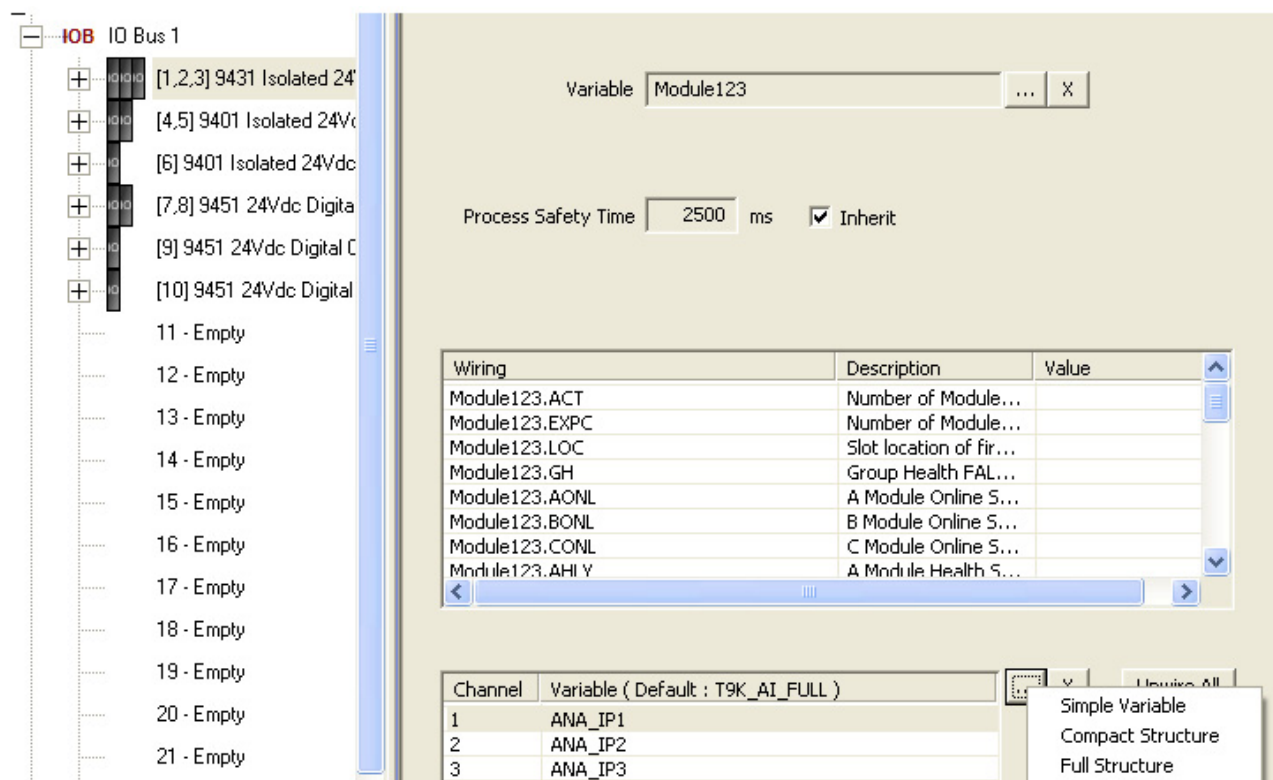
To wire variables to digital input channels do the following:




1. Select a digital input module on the I/O Bus.
  - The module status variable <tagname> that you assigned will appear in the Variable field.
2. Select the channel that you want to wire to a variable.
3. Click the  button adjacent to the Channel Variable fields.
4. Choose a data structure from the three options displayed: Simple, Compact, Full.
  - The Select Variables dialog box opens.
5. Select a named structure, click **OK**.
6. Repeat steps 2 to 5 for each channel you want to wire.

## Wire Variables to Analogue Input Channels

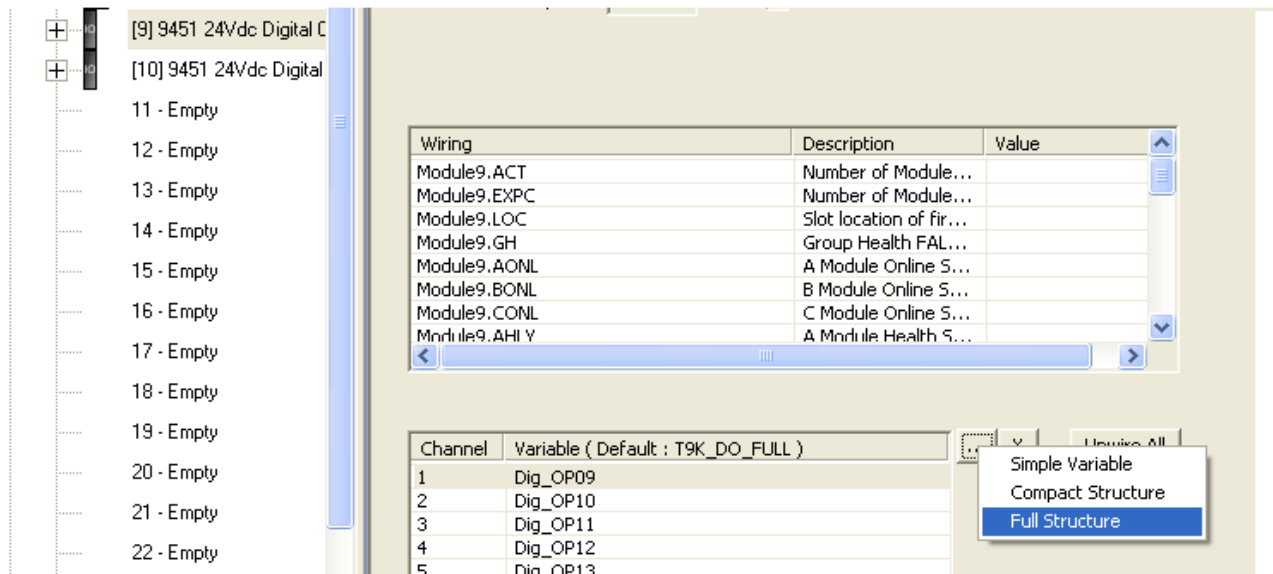
To wire variables to analogue input channels do the following:




1. Select an analogue input module on the I/O bus.
  - The module status variable <tagname> that you assigned will appear in the Variable field.
2. Select the channel that you want to wire to a variable.
3. Click the  button next to the channel variable fields.
4. Choose a data structure from the three options displayed: Simple, Compact, Full.
  - The Select Variables dialog box is displayed.
5. Select a named structure, click **OK**.
6. Repeat steps 2 to 5 for each channel.

## Wire Variables to Digital Output Channels

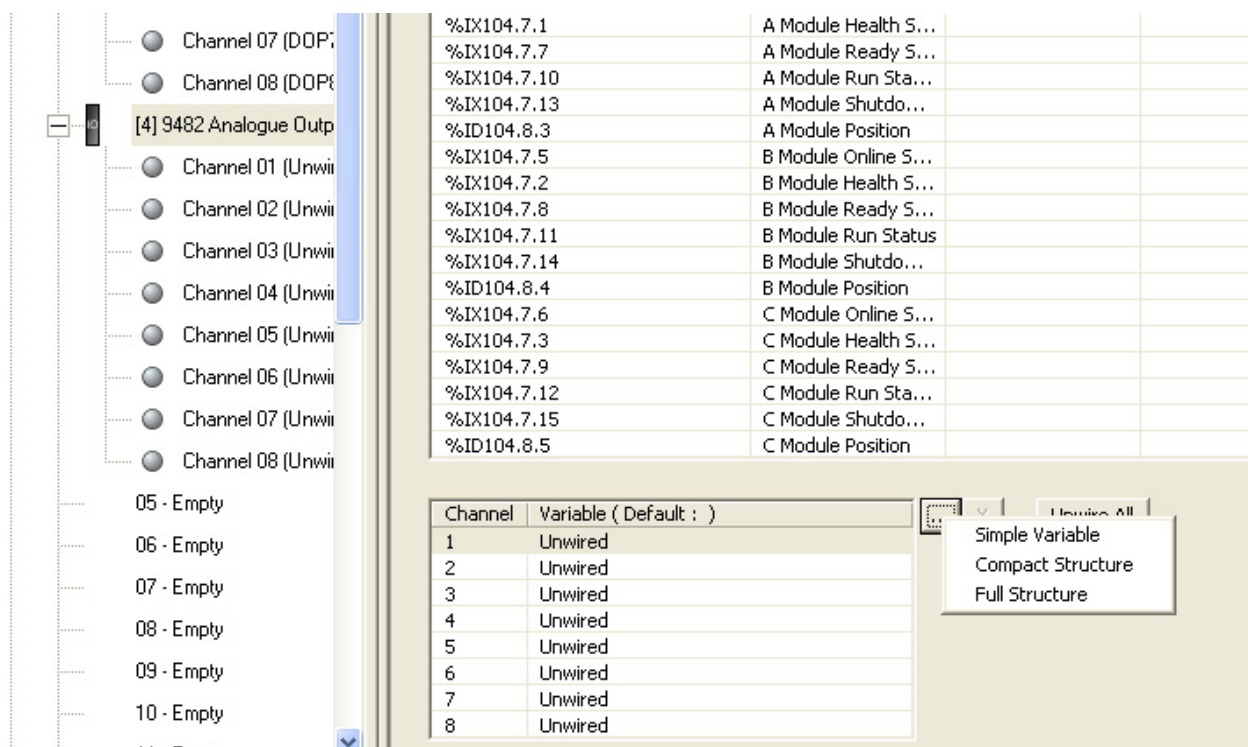
To wire variables to digital output channels do the following:



1. Select an digital output module on the I/O bus.
  - The module status variable <tagname> that you assigned will appear in the Variable field.
2. Select the channel that you want to wire to a variable.
3. Click the  button next to the channel variable fields.
4. Choose a data structure from the three options displayed: Simple, Compact, Full.
  - The Select Variables dialog box is displayed.
5. Select a named structure, click **OK**.
6. Repeat steps 2 to 5 for each channel.

## Wire Variables to Analogue Output Channels

To wire variables to analogue output channels do the following:



1. Select an analogue output module on the I/O bus.
  - The module status variable <tagname> that you assigned will appear in the Variable field.
2. Select the channel that you want to wire to a variable.
3. Click the button next to the channel variable fields.
4. Choose a data structure from the three options displayed: Simple, Compact, Full.
  - The Select Variables dialog box is displayed.
5. Select a named structure, click **OK**.
6. Repeat steps 2 to 5 for each channel.

## Configuring Digital Inputs

You can wire digital input channels to the following variable types and data structures:

- BOOL (the <variable\_name> gives the input state)
- T9K\_DI\_Compact (provides three elements)
- T9K\_DI\_Full (six elements).

The structures contain additional information about the input, such as line fault status and discrepancy status. You can also specify custom thresholds for digital inputs.

## T9K\_DI\_COMPACT and T9K\_DI\_FULL (Digital Inputs)

You wire the channel variables so the controller can receive the reported input values for the channels. The elements for both data structures for digital input channels (T9K\_DI\_COMPACT and T9K\_DI\_FULL) are indicated in the following tables.

**Table 8 - T9K\_DI\_COMPACT Structure for Digital Inputs**

Identifier	Type	Description	Remarks
<tagname>.DI	BOOL	Input state	<ul style="list-style-type: none"> <li>TRUE: input voltage above threshold Tn</li> <li>FALSE: input voltage below threshold T5</li> </ul>
<tagname>.LF	BOOL	Line fault	<ul style="list-style-type: none"> <li>TRUE: input voltage above threshold T8; between T5 and T4; or below T1</li> <li>FALSE: input voltage between thresholds T2 and T3; or between T6 and T7</li> </ul>
<tagname>.DIS	BOOL	Discrepancy	TRUE: there is a discrepancy in voltage larger than 20 % exists between the channels of two or three modules in a redundant configuration (†)

**Table 9 - T9K\_DI\_FULL Structure for Digital Inputs**

Identifier	Type	Description	Remarks
<tagname>.DI	BOOL	Input state	<ul style="list-style-type: none"> <li>TRUE: input voltage above threshold T6</li> <li>FALSE: input voltage below threshold T5</li> </ul>
<tagname>.LF	BOOL	Line fault	<ul style="list-style-type: none"> <li>TRUE: input voltage above threshold T8; between T5 and T4; or below T1</li> <li>FALSE: input voltage between thresholds T2 and T3; or between T6 and T7</li> </ul>
<tagname>.DIS	BOOL	Discrepancy	TRUE: a discrepancy in voltage larger than 8 % (of 24 V) exists between the channels of two or three modules in a redundant configuration (†)
<tagname>.CF	BOOL	Channel fault	TRUE: module diagnostics found a fault in the channel electronics or firmware (state = 7)
<tagname>.V	UINT	Voltage	Reports the channel voltage in units of millivolts with an accuracy of $\pm 500$ mV (††)
<tagname>.STA	USINT	State	Reports a state value for the channel: <ul style="list-style-type: none"> <li>1 = open circuit</li> <li>2 = de-energized</li> <li>3 = indeterminate</li> <li>4 = energized</li> <li>5 = short-circuit</li> <li>6 = over voltage</li> <li>7 = faulted</li> </ul>

(†) Discrepancy can only be reported TRUE when two or three modules are active in a group.

(††) The voltage element cannot report values below 0 mV.

## Faulted State for Digital Inputs

A digital input channel is faulted (the state reports a value of 7) when the channel is not able to report a voltage in a safety accuracy specification of 10 % of the full scale measurement of the 24 Vdc supply (2.4V).

When the state reports the value 7, then the following 'safe' values are reported by the other variables:



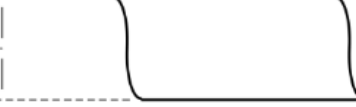

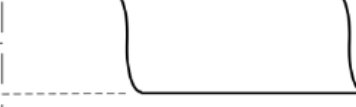


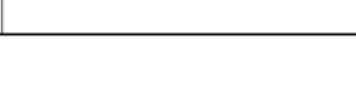
- Input State = FALSE
- Line Fault = TRUE
- Discrepancy = TRUE
- Channel Fault = TRUE
- Voltage = 0 mV.

## Threshold Values for Digital Inputs

The module determines the channel state and the line fault status by comparing the channel input voltage with a set of threshold values. You can specify your own threshold values or use the default values. The values you choose for the module are inherited by each channel; you can define different thresholds for individual channels later.

There is an indeterminate region between the closed and open status to allow for marginal faults in the external wiring or sensor. The AADvance controller provides hysteresis on the thresholds for increasing and decreasing values to prevent chatter. The AADvance Workbench updates the reporting values during each application cycle.

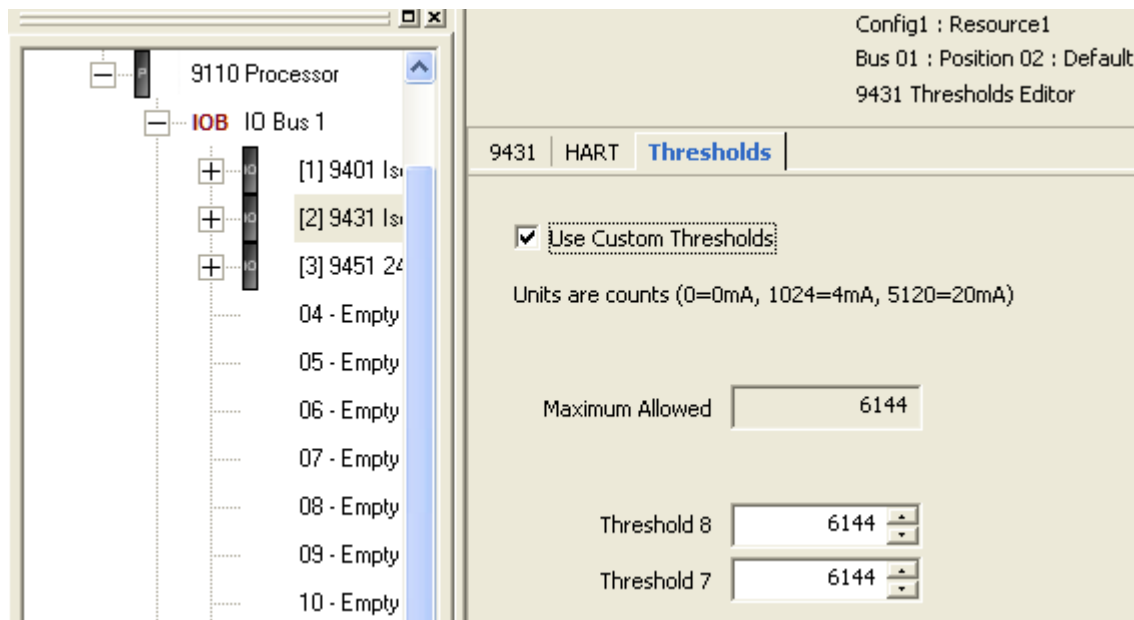
When the system is operational you should change these values only through an on-line update.

	Typical Voltage Threshold (mV)		State Value (STA)	DI Status	Line Fault Status
Over Voltage	Tmax 32000		6	False	True
Short Circuit	T8 30001		5		
On	T7 29502		4 or 5	False or True	False or True
	T6 14992		4	True	False
	T5 14491		3 or 4	False or True	False or True
Indeterminate	T4 5509		3	False	False
Off	T3 4990		2 or 3		
	T2 0		2		
Open Circuit	T1 -259		1 or 2		False or True
			1		True



## Define Thresholds for a Digital Input Module

To define your own threshold values do the following:

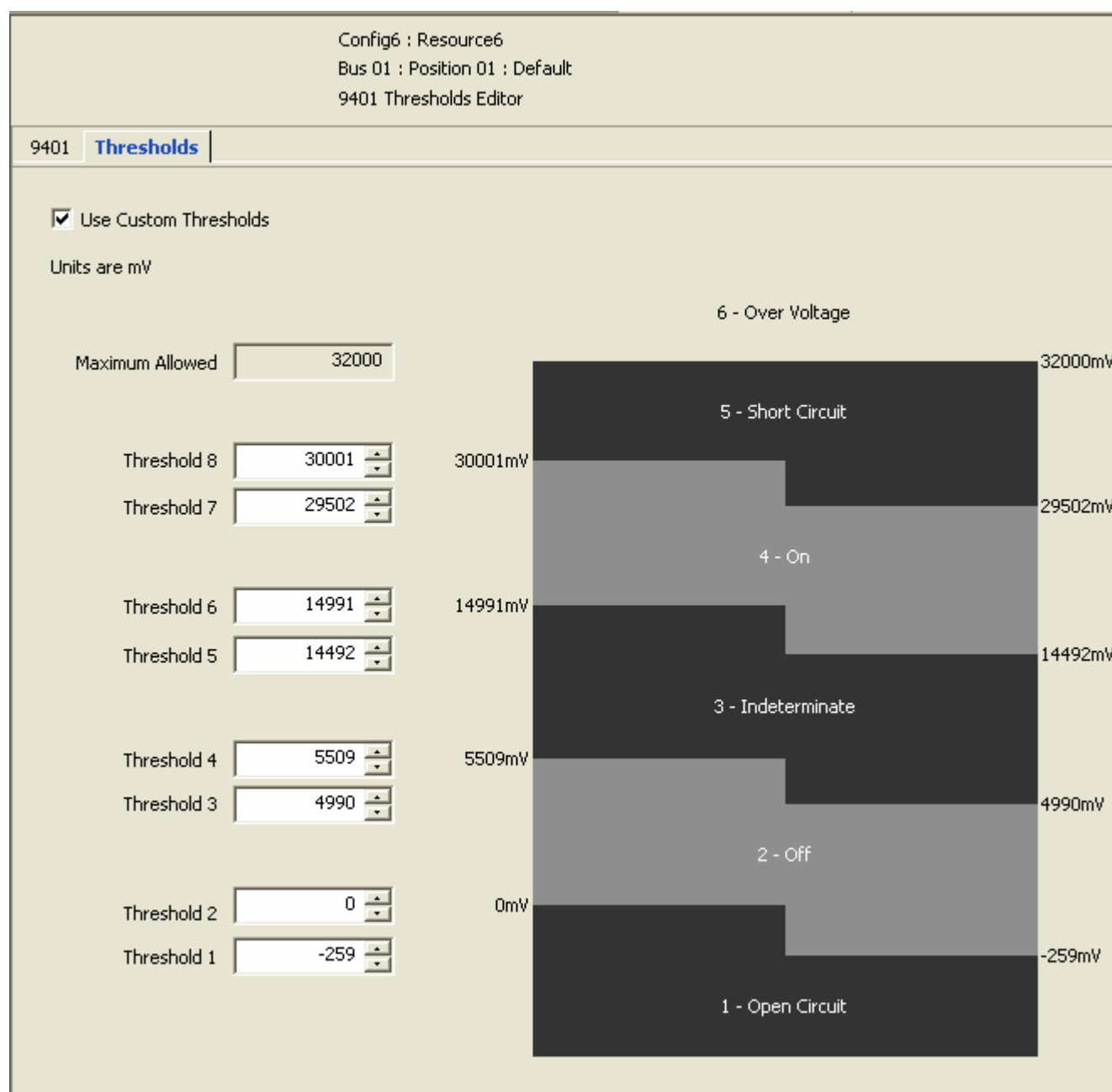


1. Select the **Thresholds** tab on the module editor.
  - A set of default values is shown in the threshold fields.
2. To enter your own values select the Use Custom Thresholds box, enter your own values in the threshold fields, click **Apply**.
3. To restore the default values, Click Default then de-select the Use Custom Thresholds, click **Apply**.

## Default Thresholds for Digital Inputs

The default threshold values for digital inputs are for a standard (non-line monitored) 24 Vdc digital input channel.

The default values are as follows:



## Configuring Analogue Inputs

You can wire analogue input channels to the following variable types and data structures:

- REAL (the <variable\_name> gives a floating-point value representing 4 mA to 20 mA)
- T9K\_AI\_Compact (provides three elements)
- T9K\_AI\_Full (six elements).

The structures contain additional information about the input, such as discrepancy status. You can also set up analogue inputs to operate with HART devices, and specify custom thresholds.

## T9K\_AI\_COMPACT and T9K\_AI\_FULL (Analogue Inputs)

The elements for both data structures for analogue inputs (T9K\_AI\_COMPACT and T9K\_AI\_FULL) are indicated in the following tables.

**Table 10 - T9K\_AI\_COMPACT Structure for Analogue Inputs**

Identifier	Type	Description	Remarks
<tagname>.PV	REAL	PV	Process Value. A scaled, floating-point value representing the analogue loop current. Default scaling factor is 0 to 100 % representing 4 to 20 mA.
<tagname>.CNT	INT	Raw count	A count representing the current on the channel in units of 1/256 mA <ul style="list-style-type: none"> <li>0 represents 0 mA</li> <li>5,120 represents 20 mA</li> <li>Accurate to within <math>\pm 13</math> counts, equivalent to <math>\pm 0.05</math> mA</li> </ul>
<tagname>.DIS	BOOL	Discrepancy	TRUE: there is a discrepancy in current larger than 2 % exists between the channels of two or three modules in a redundant configuration (†)

**Table 11 - T9K\_AI\_FULL Structure for Analogue Inputs**

Identifier	Type	Description	Remarks
<tagname>.PV	REAL	PV	Process Value. A scaled, floating-point value representing the analogue loop current. Default scaling factor is 0 to 100 % representing 4 to 20 mA
<tagname>.CNT	INT	Raw count	A count representing the current on the channel in units of 1/256 mA <ul style="list-style-type: none"> <li>0 represents 0 mA; 5,120 represents 20 mA</li> <li>Accurate to within <math>\pm 13</math> counts, equivalent to <math>\pm 0.05</math> mA</li> </ul>
<tagname>.LF	BOOL	Line fault	<ul style="list-style-type: none"> <li>TRUE: state (.STA) is 1, 5, 6 or 7</li> <li>FALSE: state (.STA) is 2, 3 or 4</li> </ul>
<tagname>.DIS	BOOL	Discrepancy	TRUE: there is a discrepancy in current larger than 2 % exists between the channels of two or three modules in a redundant configuration (†)
<tagname>.CF	BOOL	Channel fault	TRUE: module diagnostics detect a fault in the channel electronics or firmware (state = 7)
<tagname>.STA	USINT	State	Reports a state value for the channel: <ul style="list-style-type: none"> <li>1 = open circuit</li> <li>2 = transmitter fault (low)</li> <li>3 = normal</li> <li>4 = transmitter fault (high)</li> <li>5 = short-circuit</li> <li>6 = over range</li> <li>7 = faulted</li> </ul>

(†) Discrepancy can only be reported TRUE when two or three modules are active in a group.

## Faulted State for Analogue Inputs

An analogue input channel is faulted (the state reports a value of 7) when the channel cannot report a count in a safety accuracy specification of 1 % of the full scale measurement range of 5,120 (51 counts, 0.2 mA).

When the state reports the value 7 then the following 'safe' values are reported by the other variables:

- Process Value = a calculated value based on a Count value of 0
- Line Fault = TRUE
- Discrepancy = TRUE
- Channel Fault = TRUE
- Count = 0.

## Threshold Values for Analogue Inputs

The module determines the channel state and the line fault status by comparing the channel input current with a set of threshold values. You can specify your own threshold values or use the default values. The values you choose for the module are inherited by each channel; you can define different thresholds for individual channels later.





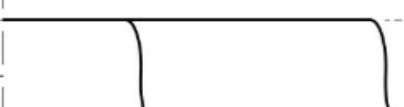


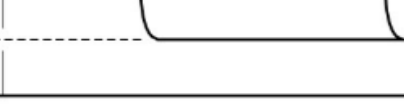
Thresholds are specified in counts, with 0 (zero) being 0 mA, 1,024 being 4 mA, and 5,120 being 20 mA.

---

**IMPORTANT** When the system is operational you should change these values using an on-line update.

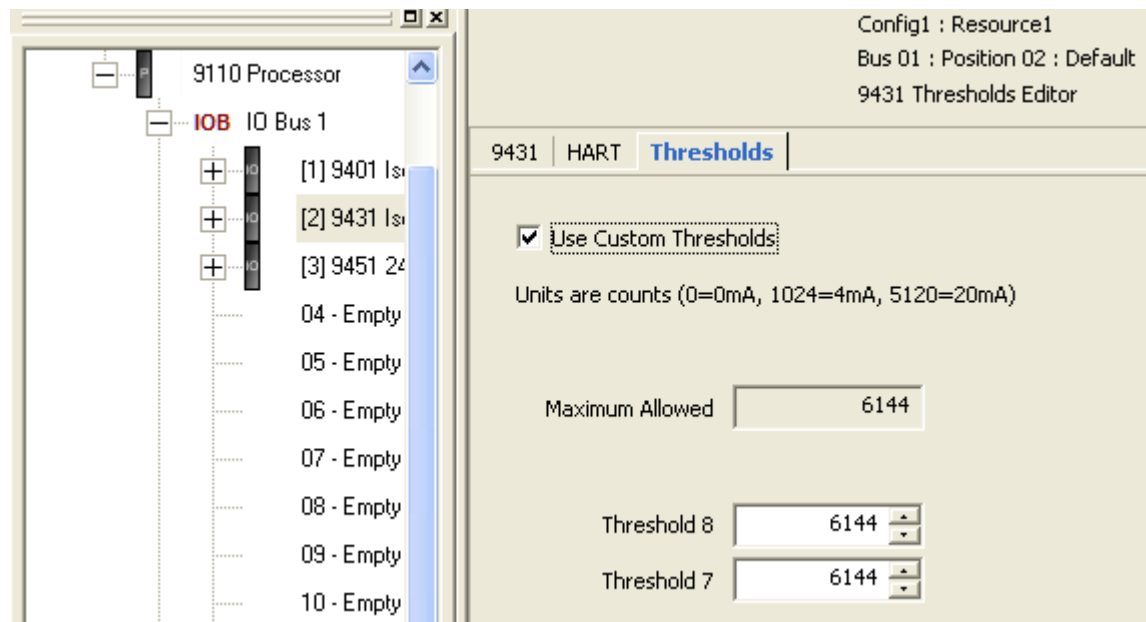
---

The AADvance controller provides hysteresis on the thresholds for increasing and decreasing values to prevent chatter. The AADvance Workbench updates the reporting values during every application cycle.

	Typical Threshold (Count)		State Value (STA)	Line Fault Status
Over Range	Imax 5632		6	True
Short Circuit	T8 5632		5	False
Transmitter Fault	T7 5632		4 or 5	
	T6 5175		4	
	T5 5120		3 or 4	
Normal	T4 1024		3	
	T3 973		2 or 3	
Transmitter Fault	T2 435		2	
	T1 384		1 or 2	True
Open Circuit	T1 384		1	

## Define Thresholds for an Analogue Input Module

To define your own threshold values do the following:

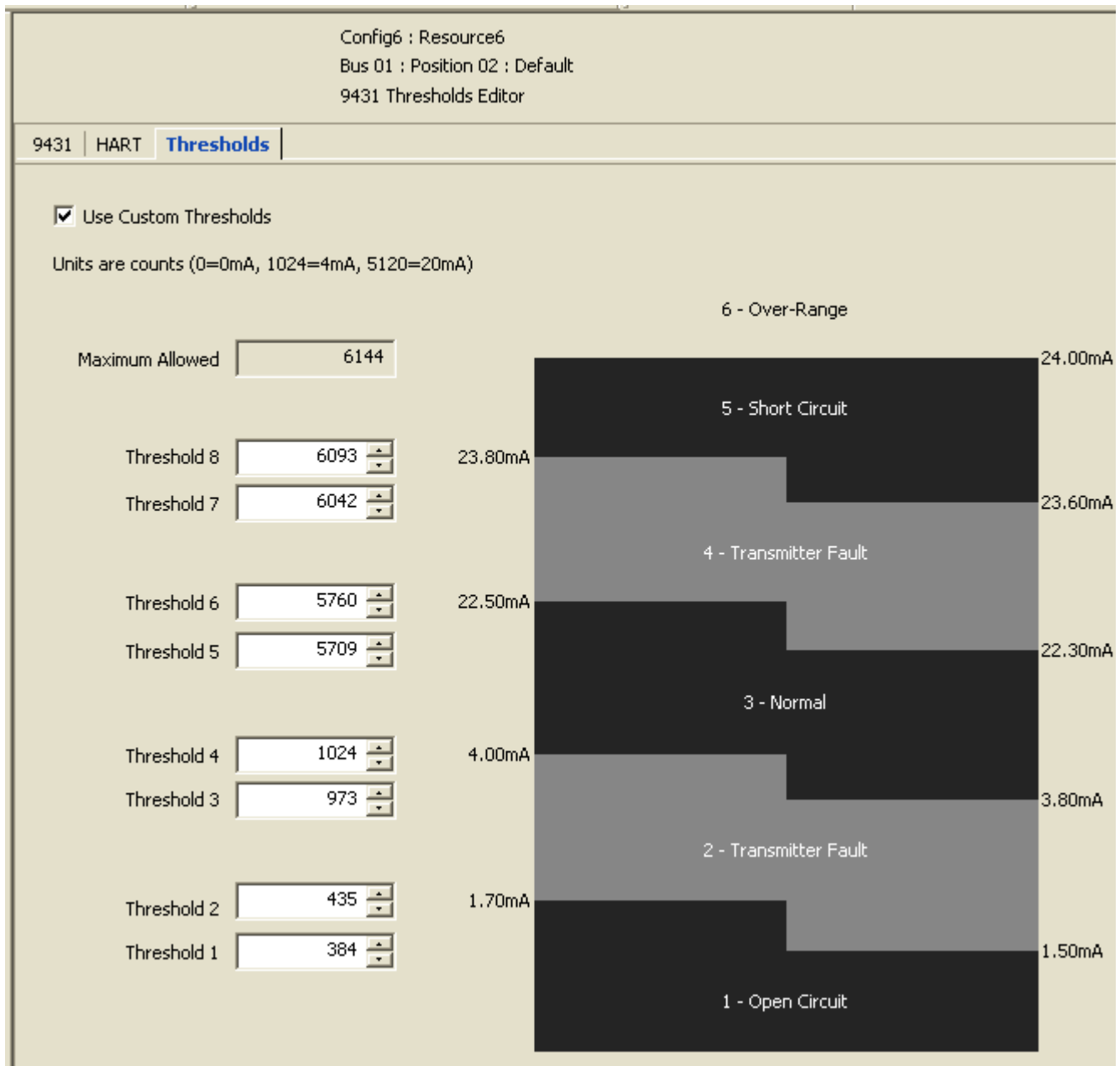


1. Select the **Thresholds** tab on the module editor.
  - A set of default values is shown in the threshold fields.
2. To enter your own values select the Use Custom Thresholds box, enter your own values in the threshold fields, click **Apply**. You can enter the values in counts (the default units) or in milliamps. To specify a value in milliamps, append 'mA' to the value; the AADvance Workbench will convert it into counts.
3. To restore the default values, Click **Default** then de-select the Use Custom Thresholds box, click **Apply**.

## Default Thresholds for Analogue Inputs

The default threshold values for analogue inputs are for a standard (non-line monitored) 24 Vdc analogue input channel. The values comply with the standard NAMUR NE43.

The default values are given in the illustration.



## Configuring Digital Outputs **Settings**

You can configure the following settings for Output Modules:

- **Process Safety Time**  
Use the procedure described earlier "Configure the I/O Module Process Safety Time" and either choose the default value or enter a custom value for the module.

- **Module Status Variables**  
Use the procedure described earlier in the topic "Wire a Status Variable to an I/O Module" to wire the Status Booleans and the Field Power Status integers.
- **Channel Advanced Settings**  
Set the Advanced options.

### Wire Output Channels

You can wire digital output channels to the following variable type and data structures:

- **BOOL** ( <variable\_name> gives the commanded state)
- **T9K\_DO\_Compact** (provides three elements)
- **T9K\_DO\_Full** (seven elements).

The structures provide data about the output such as line fault status and discrepancy status.

---

**IMPORTANT** The controller writes its digital outputs one time in each application cycle and the digital output variables are updated one time in each application cycle.

---

### T9K\_DO\_COMPACT and T9K\_DO\_FULL (Digital Outputs)

The data structures for digital inputs (T9K\_DI\_COMPACT and T9K\_DI\_FULL) provide the elements detailed in the tables.

**Table 12 - T9K\_DO\_COMPACT Structure for Digital Outputs**

Identifier	Type	Description	Remarks
<tagname>.DOP	BOOL	Output demand	The commanded state to be passed to the output channel Set to TRUE to energize Set to FALSE to de-energize
<tagname>.LF	BOOL	Line fault	TRUE: no field supply is present, no load is connected, or a short circuit is detected
<tagname>.DIS	BOOL	Discrepancy	TRUE: there is a discrepancy in current greater than 1 % between the channels of two modules in a redundant configuration (†)

**Table 13 - T9K\_DO\_FULL Structure for Digital Outputs**

Identifier	Type	Description	Remarks
<tagname>.DOP	BOOL	Output demand	The commanded state to be passed to the output channel Set to TRUE to energize Set to FALSE to de-energize
<tagname>.LF	BOOL	Line fault	TRUE: no field supply is present, no load is connected, or a short circuit is detected
<tagname>.DIS	BOOL	Discrepancy	TRUE: there is a discrepancy in current greater than 1 % between the channels of two modules in a redundant configuration (†)



Identifier	Type	Description	Remarks
<tagname>.CF	BOOL	Channel fault	TRUE: module diagnostics detect a fault in the channel electronics or firmware (state = 7)
<tagname>.V	UINT	Voltage	Reports the channel voltage at the output terminals, in units of millivolts and with an accuracy of $\pm 500$ mV (††)
<tagname>.I	INT	Current	Reports the current for the channel in milliamps and with an accuracy of $\pm 2$ mA and $\pm 10$ % of measurement
<tagname>.STA	USINT	Channel state	Reports a state value for the channel: <ul style="list-style-type: none"> <li>• 1 = no vfield</li> <li>• 2 = de-energized</li> <li>• 3 = no load</li> <li>• 4 = energized</li> <li>• 5 = short-circuit</li> <li>• 6 = field fault</li> <li>• 7 = faulted</li> </ul>

(†) Discrepancy can only be reported TRUE when two modules are active in a group.

(††) The voltage element cannot report values below 0 mV.

## The State Variable for Digital Outputs

The state variable for a digital output is an unsigned integer with a value from 1 to 7 representing the following:

- 1 = no-vfield: the field supply voltage is at or below 18 Vdc for that channel.

---

**IMPORTANT** When the state variable is 1, the field voltage (<tagname.V>) is reported as 0 mV.

---

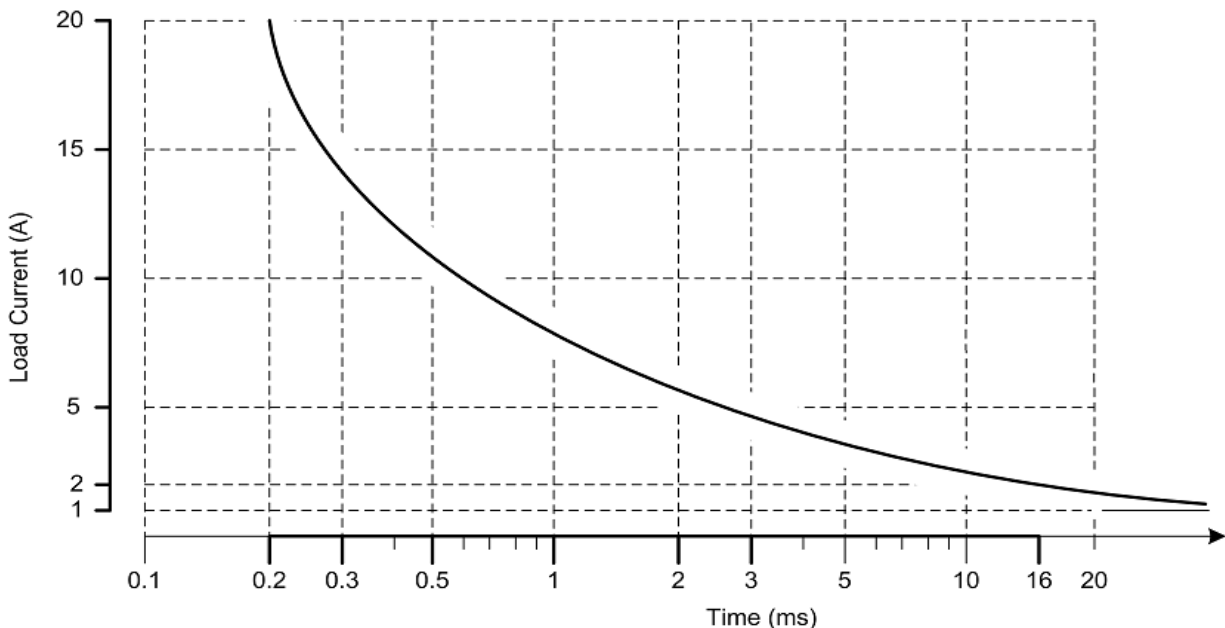
- 2 = de-energized: the commanded state is FALSE and the channel is de-energized.
- 3 = no-load: the controller cannot detect a load connected to the channel field wiring, or the load is below the minimum required channel load of 10 mA when commanded TRUE.
- 4 = on: the commanded state is TRUE and the channel is energized.
- 5 = short-circuit: the controller has detected a short-circuit condition, irrespective of the channel drive state.
- 6 = field fault: an external source is driving the channel to an energized state or a voltage greater than 18 Vdc, irrespective of the channel drive state.
- 7 = faulted.

## Overcurrent Protection for Digital Outputs

The AADvance controller has three mechanisms to protect its digital output channels:

- Inrush current protection
- Short circuit protection for energized channels
- Short circuit protection for de-energized channels

The controller tolerates inrush currents so that its digital outputs can energize capacitive loads without causing the controller to report a short circuit. The illustration shows the characteristics of the maximum load currents that the controller will tolerate when a digital output is commanded on. If the load current enters the region above the curve on the graph, the controller applies its inrush current protection.



After allowing for inrush, the controller engages its short circuit protection for an energized channel when the loop current reaches 2A.

- Short circuit detection on an energized channel is immediate and the channel is de-energized. The controller reports the condition until the short circuit is cleared.
- When the short circuit is removed, the channel will re-energize. The short circuit report is then cleared by pressing the **Fault Reset** button on the 9110 processor module or by setting the commanded state is set to FALSE.

The controller checks de-energized digital output channels for potential short circuits. Periodically, the controller partially turns on each de-energized output in turn and measures the loop current. If the loop current shows a loop resistance of less than approximately 10  $\Omega$ , the controller reports a short circuit.

## Faulted State for Digital Outputs

A digital output channel is faulted (the state reports a value of 7) when normal operation or diagnostics tests have identified a specific fault condition. A single identified fault condition thus results in a state value of 7.

When the state reports the value of 7 then the following 'safe' values are reported by the other variables:

- Line Fault = TRUE
- Discrepancy = TRUE
- Channel Fault = TRUE
- Voltage = 0 mV
- Current = 0 mA..

---

**NOTE** For details of how the AADvance digital output module detects field faults, see article [605753](#) on the Rockwell Knowledgebase website.

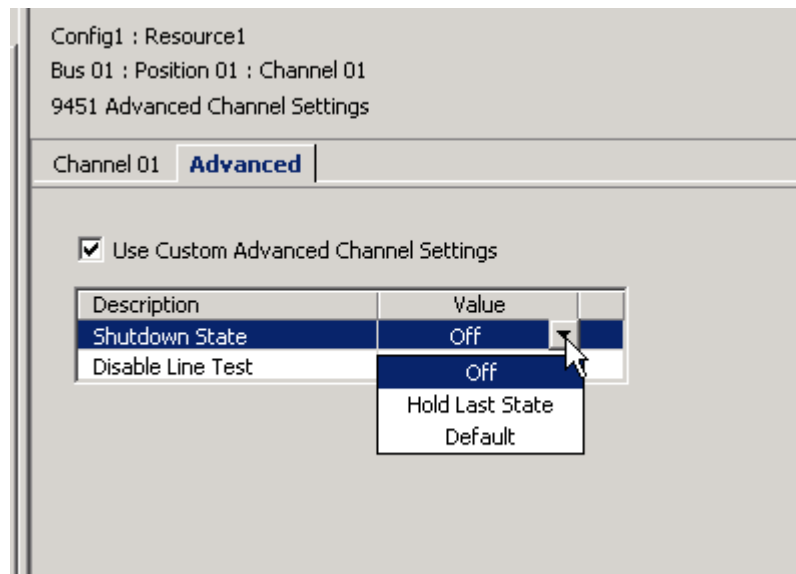
---

## Configure Advanced Channel Settings for Digital Outputs

The AADvance Workbench provides advanced settings for individual digital output channels:

- You can specify a shutdown state for an output; this defines how the output will behave when its parent 9451 digital output module is in a shutdown mode.
- You can disable the line test feature for an output; this disables detection of a no-load condition.

- To configure the advanced channel settings do the following:



1. Select a slot with a digital output module.
  - The module status variable name <tagname> that you assigned appears in the Variable field.
2. Select **Channel 01** and click the **Advanced** tab.
  - The 9451 Advanced Channel Settings dialog box opens.
3. Put a tick in the box labelled Use Custom Advanced Channel Settings.
4. Select the Shutdown State row and then select an option from the pull down list
  - Off will disable the output channel when the module is shutdown due to a loss of communications with the processor or failure of the module.
  - Hold Last State will force the output to remain in its last commanded state during a module shutdown.



**CAUTION:** When channels go to Hold Last State other compensating measures need to be put in place during the failure and the communications must be restored within the MTTR to maintain the safety rating of the system.

- Default is not used.
5. Select the Disable Line Test and then an option from the two choices of Yes or No.
    - Yes disables reporting of the status variable (STA) state 3. In a no-load condition the Channel LED will go Amber.
    - No enables the line test.

### Disabling the Line Test for a Digital Output

The 9451 digital output module does a test for a no-load condition on each output. The test is called the 'Line Test'. A no-load condition occurs when the controller cannot find a load connected to the field wiring, or the load current is below 20 mA when the output is commanded TRUE. You will disable the line test if you want to connect a low load to an output, or if the output is unused and you do not want to fit a dummy load.

When the line test is enabled, the module reports a no-load condition by setting the state variable (<tagname>.STA) to the value 3, and by setting the channel LED to amber. After you disable the line test, then assuming there are no other faults present, the state variable will show 2 or 4 (depending on the commanded value) instead of 3, and the channel LED will show off or green instead of amber.

## Configuring Analogue Outputs

The controller writes its analogue outputs one time in each application cycle; the analogue output variables are also updated one time in each application cycle. You can wire analogue output channels to the following variable type and data structures:

- REAL (the <variable\_name> gives the commanded state)
- T9K\_AO\_Compact (provides three elements)
- T9K\_AO\_Full (seven elements)

The structures provide data about the output, such as line fault status and discrepancy status.

### The State Variable for Analogue Outputs

The state variable for an analogue output is an unsigned integer with a value from 1 to 7 showing the following:

- 1 = no-vfield: the field supply voltage is at or below 18 Vdc and the commanded current is less than 0.4 mA for that channel.

---

**IMPORTANT** When the state variable is 1, the field voltage (<tagname.V>) is shown as 0 mV.

---

- 2 = off: the raw count value is less than 102 (0.4 mA).
- 3 = no-load: the controller cannot detect a load connected to the channel field wiring, or the loop voltage cannot be detected. This happens when the commanded current is greater than 0.4 mA, the raw count value is less than 51 (0.2 mA) and the measured voltage is less than 1000 mV.
- 4 = on: the raw count value is 102 or greater ( $\geq 0.4$  mA).

- 5 = compliance fault: there is not enough loop voltage available to sustain the commanded output current to within the safety accuracy specification (i.e. 1 % full scale or 0.2 mA) and the commanded current is greater than 102 Counts (0.4 mA). The output will supply as much current as is available and will not start a fail-safe action or report a channel fault.
- 6 = reverse-polarity: the measured voltage is less than -1000 mV.
- 7 = faulted.

## T9K\_AO\_COMPACT and T9K\_AO\_FULL (Analogue Outputs)

The data structures for analogue outputs (T9K\_AO\_COMPACT and T9K\_AO\_FULL) have the elements detailed in the tables.

**Table 14 - T9K\_AO\_COMPACT Structure for Analogue Outputs**

Identifier	Type	Description	Remarks
<tagname>.CV	REAL	Command Value	Demanded current. A scaled, floating-point value showing the analogue loop current. Default scaling factor is 0 to 100 % representing 4 to 20 mA
<tagname>.LF	BOOL	Line fault	TRUE: no field supply is present, no load is connected, the commanded output current cannot be met, the wiring polarity is reversed, or channel fault (states 1,3,5,6,7)
<tagname>.DIS	BOOL	Discrepancy	TRUE: measured current and commanded current differ by more than the fail-safe guard band

**Table 15 - T9K\_AO\_FULL Structure for Analogue Outputs**

Identifier	Type	Description	Remarks
<tagname>.CV	REAL	Command Value	Demanded current. A scaled, floating-point value showing the analogue loop current. Default scaling factor is 0 to 100 % representing 4 to 20 mA
<tagname>.LF	BOOL	Line fault	TRUE: no field supply is present, no load is connected, the commanded output current cannot be met, the wiring polarity is reversed, or channel fault (states 1,3,5,6,7)
<tagname>.DIS	BOOL	Discrepancy	TRUE: measured current and commanded current differ by more than the fail-safe guard band
<tagname>.CF	BOOL	Channel fault	TRUE: module diagnostics detect a fault in the channel electronics or firmware (state = 7)
<tagname>.V	INT	Voltage	Reports the channel voltage at the output terminals, in units of millivolts and with an accuracy of $\pm 500$ mV.
<tagname>.CNT	INT	Raw count	Reports the current for the channel in raw units scaled 256 per mA (from 0 mA = 0 to 24 mA = 6,144)
<tagname>.STA	USINT	Channel state	Reports a state value for the channel: <ul style="list-style-type: none"> <li>• 1 = no vfield (&lt; 18 Vdc)</li> <li>• 2 = off (demand &lt; 0.4 mA)</li> <li>• 3 = no load/open circuit</li> <li>• 4 = on (demand &gt; 0.4 mA)</li> <li>• 5 = compliance fault (demand cannot be met)</li> <li>• 6 = reverse polarity (&lt; -1Vdc)</li> <li>• 7 = faulted</li> </ul>

## Faulted State for Analogue Outputs

An analogue output channel is faulted (the state reports a value of 7) when normal operation or diagnostics tests have identified a specific fault condition. A single identified fault condition thus results in a state value of 7.

When the state reports the value of 7 then the following 'safe' values are reported by the other variables:

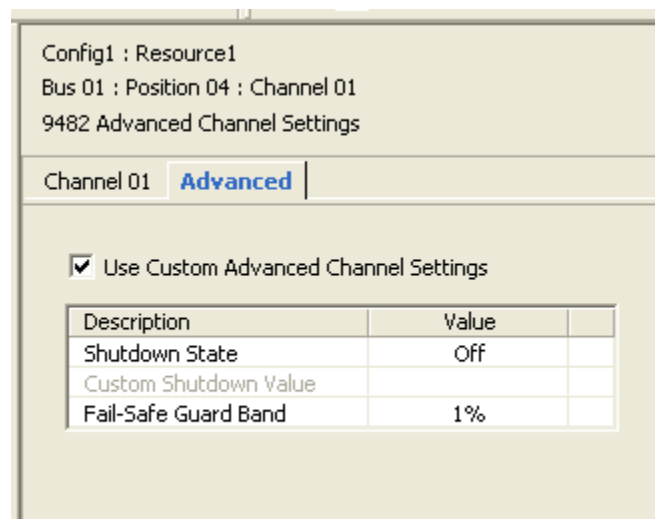
- Line Fault = TRUE
- Discrepancy = TRUE
- Channel Fault = TRUE
- Voltage = 0 mV
- Raw Count = 0 mA.

## Configure Advanced Channel Settings for Analogue Output Channels

The AADvance Workbench provides advanced settings for individual analogue output channels:

- Specify a shutdown state for an output; this defines how the output will behave when its parent 9481 or 9482 analogue output module is in a shutdown mode. The options are Off, Hold Last State and Custom.
- Specify a value for the current output in the shutdown mode, using the same scaling as the Command Value.
- Specify a threshold for reporting of Discrepancy in parameter Fail-Safe Guard Band. This is a percentage of 0-20 mA demand.

To configure the advanced channel settings do the following:



1. Select the slot with the analogue output module.

- The module status variable name <tagname> that you assigned appears in the Variable field.
2. Select the channel to configure and click the **Advanced** tab.
    - The 9481/9482 Advanced Channel Settings dialog box opens.
  3. Put a tick in the box labelled Use Custom Advanced Channel Settings.
  4. Choose the advanced channel settings from the options.

### Analogue Output Advanced Channel Settings

Each output channel from the 9481/9482 analogue output module supports the set of control parameters detailed in the table.

**Table 16 - Analogue Output Control Parameters**

Description	Value(s)	Default	Remarks
Shutdown State	Off, Hold Last State, Custom	Off	'Off' de-energizes the output during a shutdown 'Hold Last State' forces the output to remain in its last commanded current, during a shutdown 'Custom' forces the output to go to the value set in 'Custom Shutdown Value', during a shutdown
Custom Shutdown Value	As scaled range of command value (CV)	-25	Output set when 'Shutdown State' is set to 'Custom', during a shutdown
Fail-Safe Guard Band	0-100 % of full scale 20 mA (1 % = 0.2 mA)	1 %	Threshold for discrepancy alarm between command value (CV) and Count (CNT). A discrepancy alarm is reported on parameter DIS.

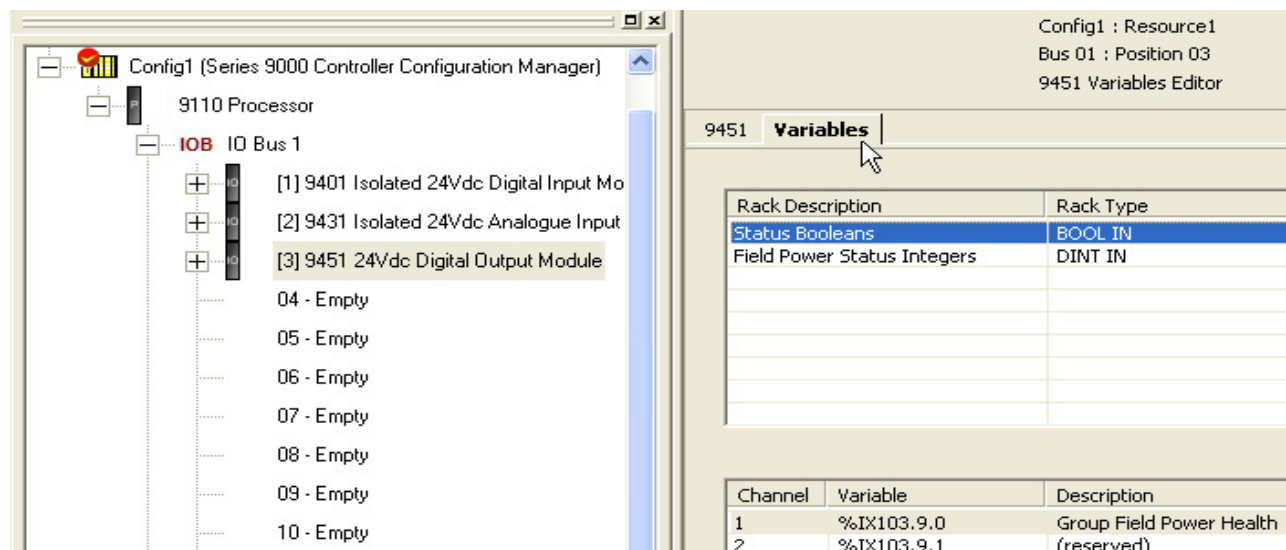
## Status Variables for Digital Output Modules

The 9451 digital output module provides a number of status variables that are available to the application. The 9451 Variables Editor presents the variables in two collections, which it calls 'racks': Status Booleans, and Power Status Integers.



## Wire Status Variables to a Digital Output Module

To wire a status variable to a digital output module do the following:




1. Navigate to the digital output module in the equipment tree view.
2. Select the **Variables** tab of the 9451 Module Editor.
  - The 9451 Variables Editor dialog box opens.
3. Select a rack. The editor displays a list of associated variables.

---

**IMPORTANT** The status variables are for modules, not channels. The column headed 'Channel' shows an index for the variables; it does not relate to individual digital outputs.

---

4. Select a variable.
5. Click the  button. The Select Variable dialog box opens.
6. From the list select an application variable to wire to the status variable, click **OK**.
7. Repeat for each subsequent variable to be wired.
8. Return to the 9451 Variables Editor and click **Apply**. The variable will now be wired.

## Unwire Status Variables from a Digital Output Module

To disconnect a status variable from a digital output module do the following:

1. Select the **Variables** tab of the 9451 Module Editor.
  - The 9451 Variables Editor dialog box opens.

2. Select the relevant rack.
  - The editor displays a list of associated variables.
3. Select the variable to be unwired, click the **X** button.
4. Click **Apply**.
  - The variable will be unwired.

**TIP** Select the **Unwire All** button and click Apply to disconnect all of the wired variables in the rack.

## Status Booleans

The Status Booleans supply data to the application about the field power supplies to a group of digital output modules.

### *Group Field Power Health*

**Direction:** input to application from controller.

**Type:** Boolean.

**Values:**

- TRUE = all field power supplies for all active digital output modules in the group are in the range 18 V to 32 Vdc inclusive.
- FALSE = one or more field power supplies to an active module is less than 18 Vdc or more than 32 Vdc.

**Description:**

Provides a top level indication of the health of field power supplies to active digital output modules.

---

**IMPORTANT** The controller incorporates a 0.5 V hysteresis on these thresholds to prevent chatter. The controller declares a fault when a supply falls below 18 V, but does not clear the fault until the supply rises to 18.5 V. Similarly the controller declares a fault when a supply exceeds 32 V, but does not clear the fault until the supply falls below 31.5 V.

---

### *Group Field Power Health*

**Direction:** input to application from controller.

**Type:** Boolean.

**Values:**

- TRUE = all field power supplies for all active digital output modules in the group are in the range 18 V to 32 Vdc inclusive.

- FALSE = one or more field power supplies to an active module is less than 18 Vdc or greater than 32 Vdc.

**Description:**

Provides a top level indication of the health of field power supplies to active digital output modules.

---

**IMPORTANT** The controller incorporates a 0.5 V hysteresis on these thresholds to prevent chatter. The controller declares a fault when a supply falls below 18 V, but does not clear the fault until the supply rises to 18.5 V. Similarly the controller declares a fault when a supply exceeds 32V, but does not clear the fault until the supply falls below 31.5 V.

---

## Field Power Status Integers

The field power status integers (all DINT) supply data to the application, about the field power supplies to a group of digital output modules.

### *Group Field Power Current*

**Direction:** input to application from controller.

**Type:** DINT.

**Values:**

- 0 to 8,000 mA or greater (limited by capacity of DINT variable).

**Description:**

Reports the total current that all the active digital output modules in a group are drawing from the field power supply. Accuracy is  $\pm 10\%$ .

### *A Module Field Power Voltage 1*

**Direction:** input to application from controller.

**Type:** DINT.

**Values:**

- 0 to 48,000 mV or larger (limited by capacity of DINT variable).

**Description:**

Reports the voltage from the field power supply for the specified module and field power input. Accuracy is  $\pm 500$  mV.

### *A Module Field Power Voltage 2*

**Direction:** input to application from controller

**Type:** DINT.

**Values:**

- 0 to 48,000 mV or larger (limited by capacity of DINT variable).

**Description:**

Reports the voltage from the field power supply for the specified module and field power input. Accuracy is  $\pm 500$  mV.

*B Module Field Power Voltage 1*

**Direction:** input to application from controller.

**Type:** DINT.

**Values:**

- 0 to 48,000 mV or larger (limited by capacity of DINT variable).

**Description:**

Reports the voltage from the field power supply for the specified module and field power input. Accuracy is  $\pm 500$  mV.

*B Module Field Power Voltage 2*

**Direction:** input to application from controller.

**Type:** DINT.

**Values:**

- 0 to 48,000 mV or larger (limited by capacity of DINT variable).

**Description:**

Reports the voltage from the field power supply for the specified module and field power input. Accuracy is  $\pm 500$  mV.

## HART

Highway Addressable Remote Transducer (HART) is an open protocol for process control instrumentation. It combines digital signals with analogue signals to provide control and status data for field devices.

The AADvance controller supports the use of HART on each analogue input and output channel. The application can use HART data to monitor and respond to device conditions and to provide diagnostics information such as data comparison and error reporting. This can provide a useful increase in the level of SIF diagnostics.

## HART Features

The support for HART in the AADvance controller has these features:

- HART support on every analogue input and output channel.
- Variables for each analogue input and output channel to monitor HART device information.

## HART

HART variables can be configured on each analogue input and output channel to monitor the HART field device.

Make sure that your HART field devices support HART command 0 ('read unique ID') and HART command 3 ('read current and four dynamic variables'), the AADvance controller uses these commands to communicate with the HART devices.

The AADvance analogue input and output modules use HART command #03 to collect data from the field device as defined by Revision 5 of the HART specification. The extra data available from HART enabled field devices is reported to the application in custom data structures, T9K\_AI\_HART and T9K\_AI\_HART\_FULL.

The structures provide the following data:

- Loop current in milliamps
- Process measurement in engineering units
- Errors on HART communication seen by the field device
- Status of the field device
- Time in milliseconds since the last update.

You can use the loop current variable for diagnostic checks in the application, to compare the value of the variable with the value on the 4 to 20 mA loop and react if there is a discrepancy. You can also monitor the status of the field device and use this to report diagnostic errors and manual configuration changes.

---

<b>IMPORTANT</b>	The update rate for HART data from field devices is slower than the update rate for the 4 mA to 20 mA analogue signal itself. HART data can take up to 4 seconds to update, depending on the device type and configuration.
------------------	---

---

## Precautions for HART in a Safety System



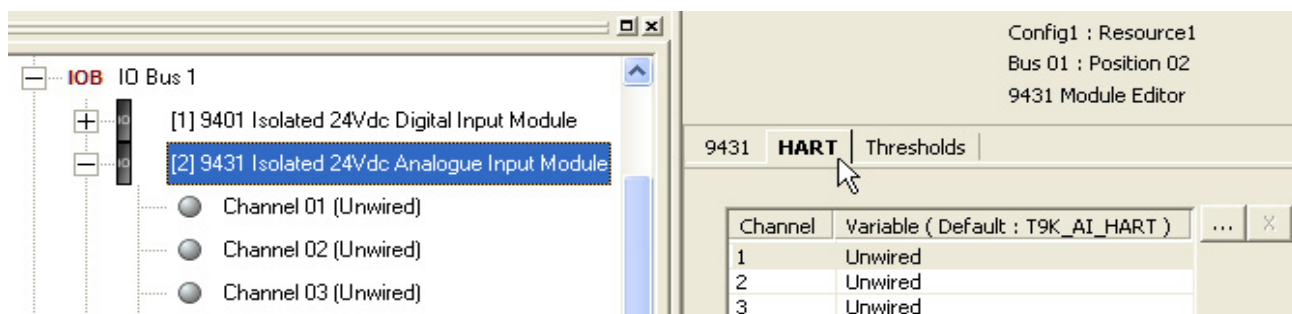
**CAUTION:** If you use HART in a safety system, take these precautions:

- Do not use HART variables as the primary initiator for a Safety Instrumented Function. The HART protocol does not meet the applicable integrity levels for Safety Instrumented Functions.
- Make sure that HART is disabled for field devices that do not have a locked configuration. This will prevent the use of HART to change a device configuration.
- Make sure that the custom data for the device (this is the data provided in response to HART command #03) is used in accordance with the device manufacturers published recommendations.


## Configure HART for Field Device Monitoring

To configure an analogue channel (input or output) to use HART to monitor a field device do the following:

1. Create a HART variable for the channel and set the type to T9K\_AI\_HART.
2. Select the module in the equipment view and then select the channel that you want to configure for field device monitoring.
3. Go to the Equipment Tree View and select the analogue module, click on the **HART** tab.
  - The example in the illustration below is an analogue input module.





4. Select a Channel, then click
5. Choose the variable you created at step 1 and wire this variable to the channel, click **OK**.
6. Return to the Equipment Tree View, select a channel.
7. Select the **HART** tab; put a tick in the box labeled Enable HART on this Channel, click **Apply**.

8. Click  and choose the T9K\_AI\_HART variable.

Config1 : Resource1  
 Bus 01 : Position 01 : Channel 02  
 9432 HART Channel Editor

Channel 02 **HART** | Thresholds

☒ Enable HART on this Channel

Variable   

Wiring	Description	Value	Physical Value	Locked
AIM_HART_FULL_02.I	Current			
AIM_HART_FULL_02.V1	Primary Variable			
AIM_HART_FULL_02.U1	Primary Variable U...			
AIM_HART_FULL_02.V2	Second Variable			
AIM_HART_FULL_02.U2	Second Variable U...			
AIM_HART_FULL_02.V3	Third Variable			
AIM_HART_FULL_02.U3	Third Variable Unit...			
AIM_HART_FULL_02.V4	Fourth Variable			
AIM_HART_FULL_02.U4	Fourth Variable U...			
AIM_HART_FULL_02.COMMS	Communication St...			
AIM_HART_FULL_02.DEVICE	Device Status			
AIM_HART_FULL_02.ELAPSED	Elapsed Time Sinc...			
AIM_HART_FULL_02.PASSTHROUGH	Passthrough Com...			

9. Repeat this procedure for the other channels that you want to monitor.

## T9K\_AI\_HART and T9K\_AI\_HART\_FULL

The two data structures for HART variables (T9K\_AI\_HART and T9K\_AI\_HART\_FULL) have the elements in the following tables.

**Table 17 - T9K\_AI\_HART Data Structure**

Identifier	Type	Description	Remarks
<tagname>.I	REAL	Current	Loop current in mA
<tagname>.V1	REAL	Variable	Primary loop current variable
<tagname>.U1	BYTE	Variable Units	Primary loop current variable units code
<tagname>.V2	REAL	Variable	Second loop current variable
<tagname>.U2	BYTE	Variable Units	Second loop current variable units code

Identifier	Type	Description	Remarks
<tagname>.V3	REAL	Variable	Third loop current variable
<tagname>.U3	BYTE	Variable Units	Third loop current variable units code
<tagname>.V4	REAL	Variable	Fourth loop current variable
<tagname>.U4	BYTE	Variable Units	Third loop current variable units code
<tagname>.COMMS	BOOL	Communication Status	HART Communication status <ul style="list-style-type: none"> <li>• TRUE: Communication OK</li> <li>• FALSE: Communication Stopped</li> </ul>
<tagname>.DEVICE	BYTE	Device Status	Field Device status: <ul style="list-style-type: none"> <li>• Bit 7: field device malfunction</li> <li>• Bit 6: configuration changed</li> <li>• Bit 5: cold start</li> <li>• Bit 4: more status available</li> <li>• Bit 3: analogue output current fixed</li> <li>• Bit 2: analogue output saturated</li> <li>• Bit 1: non-primary variable out of limits</li> <li>• Bit 0: primary variable out of limitsBit</li> </ul>

The HART\_FULL data structure is used for HART Pass-Through communication in WB2.0. This data structure has two additional elements - Elapsed Time data and the Pass-Through communication status.

**Table 18 - T9K\_AI\_HART\_FULL Data Structure**

Identifier	Type	Description	Remarks
<tagname>.I	REAL	Current	Loop current in mA
<tagname>.V1	REAL	Variable	Primary loop current variable
<tagname>.U1	BYTE	Variable Units	Primary loop current variable units code
<tagname>.V2	REAL	Variable	Second loop current variable
<tagname>.U2	BYTE	Variable Units	Second loop current variable units code
<tagname>.V3	REAL	Variable	Third loop current variable
<tagname>.U3	BYTE	Variable Units	Third loop current variable units code
<tagname>.V4	REAL	Variable	Fourth loop current variable
<tagname>.U4	BYTE	Variable Units	Third loop current variable units code
<tagname>.COMMS	BOOL	Communication Status	HART Communication status <ul style="list-style-type: none"> <li>• TRUE: Communication OK</li> <li>• FALSE: Communication Stopped</li> </ul>
<tagname>.DEVICE	BYTE	Device Status	Field Device status: <ul style="list-style-type: none"> <li>• Bit 7: field device malfunction</li> <li>• Bit 6: configuration changed</li> <li>• Bit 5: cold start</li> <li>• Bit 4: more status available</li> <li>• Bit 3: analogue output current fixed</li> <li>• Bit 2: analogue output saturated</li> <li>• Bit 1: non-primary variable out of limits</li> <li>• Bit 0: primary variable out of limits</li> </ul>
<tagname>.TIME	DINT	Time in ms	Elapsed time since last non-Pass-Through communication. This parameter is reset to zero when data is received.
<tagname>.TIME	BOOL	Communication Status	Pass-Through communication status: <ul style="list-style-type: none"> <li>• TRUE: Communication OK</li> <li>• FALSE: Communication stopped</li> </ul>



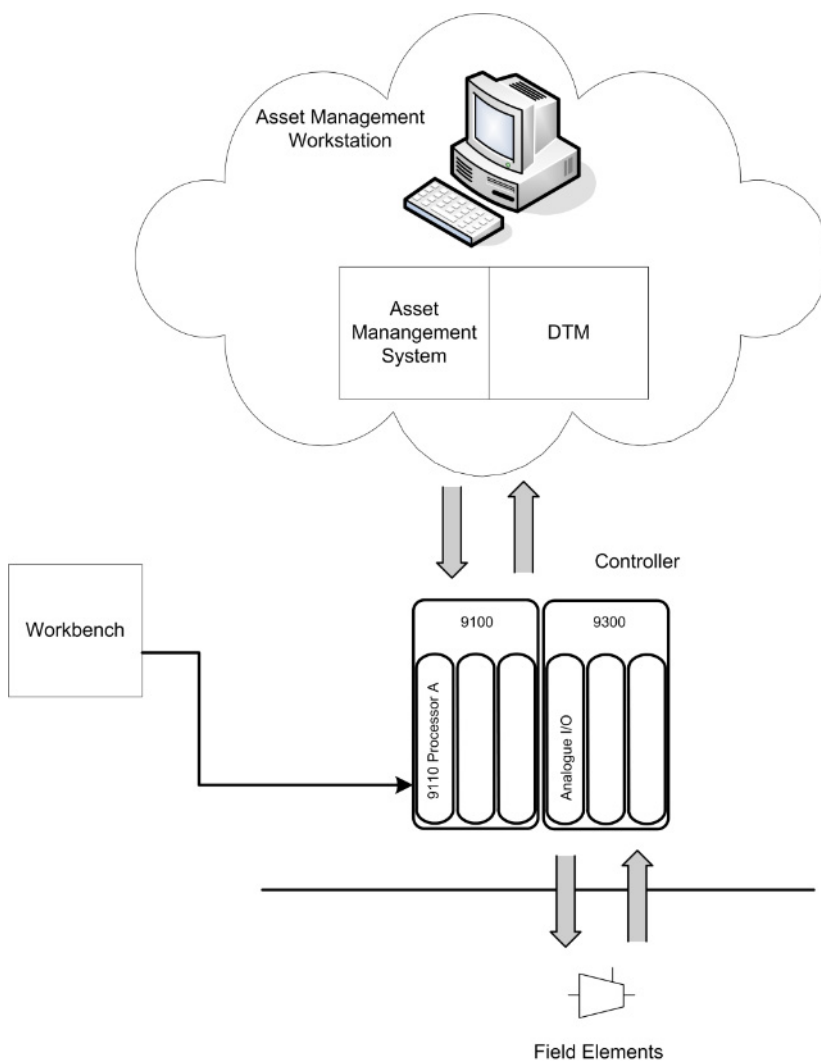
## HART Pass-Through

The HART Pass-Through feature enables using an external asset management system to manage HART compatible field devices connected to an AADvance controller.

HART Pass-Through uses the device type manager (DTM) standard, defined by the HART Communication Foundation, to enable using any asset management system which using the generic 'Frame' standard. Examples of compatible tools are the Fieldcare application by Endress+Hauser and the FactoryTalk Asset Centre by Rockwell Automation.

### Using HART Pass-Through

HART variables can be configured on each analogue input and output channel to monitor HART Pass-Through data. A typical arrangement of a system using HART Pass-Through is shown in the illustration.



To use HART Pass-Through, you have to install the AADvance 9033 DTM software on the computer which is running the asset management system. You

can then enable and disable the HART Pass-Through capability of the controller.

You can also set up the application to get status data for individual analogue channels. There are two elements in the T9K\_AI\_HART\_FULL data structure which provide the data:

- <tagname>.ELAPSED (DINT), which shows the time in milliseconds since the last valid non-pass-through communication. The value of this element resets to 0 (zero) when the application passes new HART data on the channel.
- <taskname>.PASSTHROUGH (BOOL), which shows when the channel is carrying HART Pass-Through data.

The application can use these elements, for example to let it decide when to allow HART Pass-Through communications.

## HART Pass-Through Features

The support for HART Pass-Through in the AADvance controller has these features:

- Pass-Through support for HART standards 5, 6 and 7.
- Dedicated Ethernet port for HART Pass-Through communication.
- Supports the AADvance DTM provided by Rockwell Automation.

## Precautions for HART Pass-Through in a Safety System



**CAUTION:** If you use HART Pass-Through in a safety system, take these precautions:

- Make sure that HART Pass-Through is enabled only under control of the application.
  - Make sure that HART Pass-Through is enabled only when necessary.
  - Configure the application to generate an alarm if HART Pass-Through is enabled on any safety-critical channel of any module.
- 

## Install the 9033 DTM

The 9033 DTM is the device type manager for the HART Pass-Through feature in an AADvance system. It must be installed on the Windows computer which is running the asset management system. To install the DTM, do the following:

1. Locate the file named AADvance DTM 1.xxx Setup.exe and run it.
  - The setup wizard opens.

2. Click **Next**.
3. Accept the terms of the license agreement, click **Next**.
4. Wait for the setup wizard to install the DTM.
5. The installation of the DTM is now complete.

## Enable HART Pass-Through in the Controller

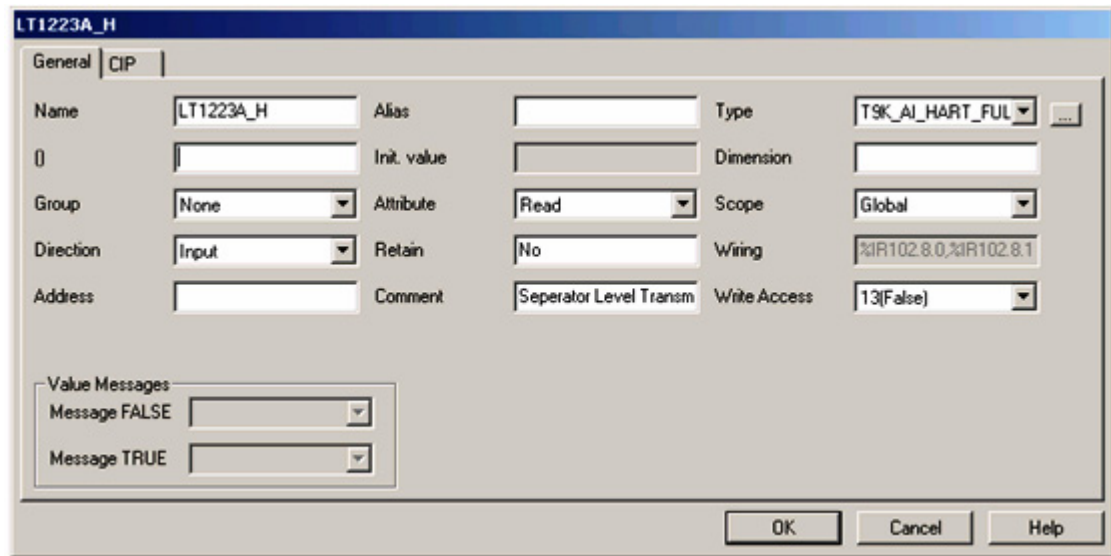
**IMPORTANT** To use HART Pass-Through at least one CIP Produce variable must be defined. If there are no CIP produce/consume variables the CIP stack does not activate. Refer to the CIP Chapter in this manual for instructions on setting up CIP Produce and Consume variables.

1. In the Dictionary create a variable named HART\_CONTROL which the application uses to enable and disable HART Pass-Through. The variable must be a BOOL type with its direction set to Output.

The screenshot shows the 'Variable Properties' dialog box for a variable named 'HART\_CONTROL'. The 'General' tab is selected. The 'Name' field contains 'HART\_CONTROL'. The 'Type' is set to 'BOOL'. The 'Direction' is set to 'Output'. The 'Scope' is set to 'Global'. The 'Write Access' is set to 'False'. The 'Value Messages' section shows 'Message FALSE' and 'Message TRUE' with empty text boxes. The 'OK', 'Cancel', and 'Help' buttons are at the bottom right.

Property	Value
Name	HART_CONTROL
Alias	
Type	BOOL
Init. value	
Dimension	
Group	None
Attribute	Free
Scope	Global
Direction	Output
Retain	No
Wiring	
Address	
Comment	
Write Access	False
Message FALSE	
Message TRUE	

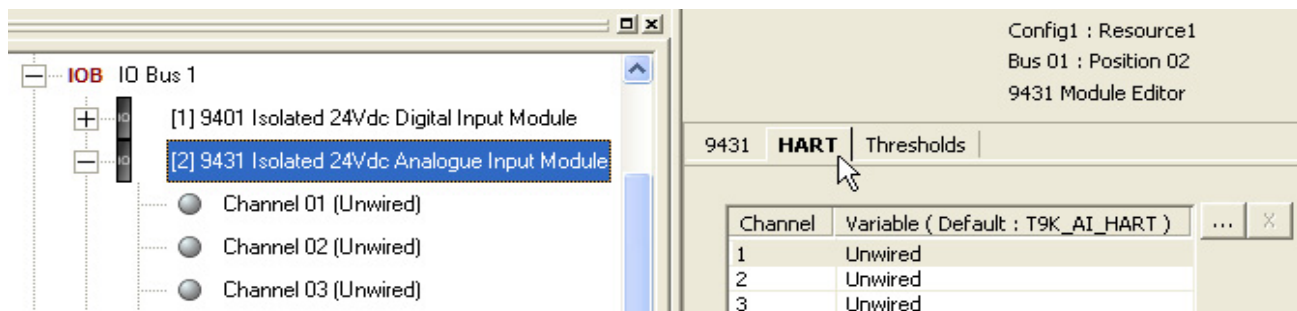
2. Select the HART Pass-Through variable selector button.



- The Variable Selector window opens.
3. Select the HART\_CONTROL variable to wire it to the processor control variable.

## Configure HART for Pass-Through Communication Monitoring

To configure an analogue input channel to provide HART and use pass-through communication monitoring:

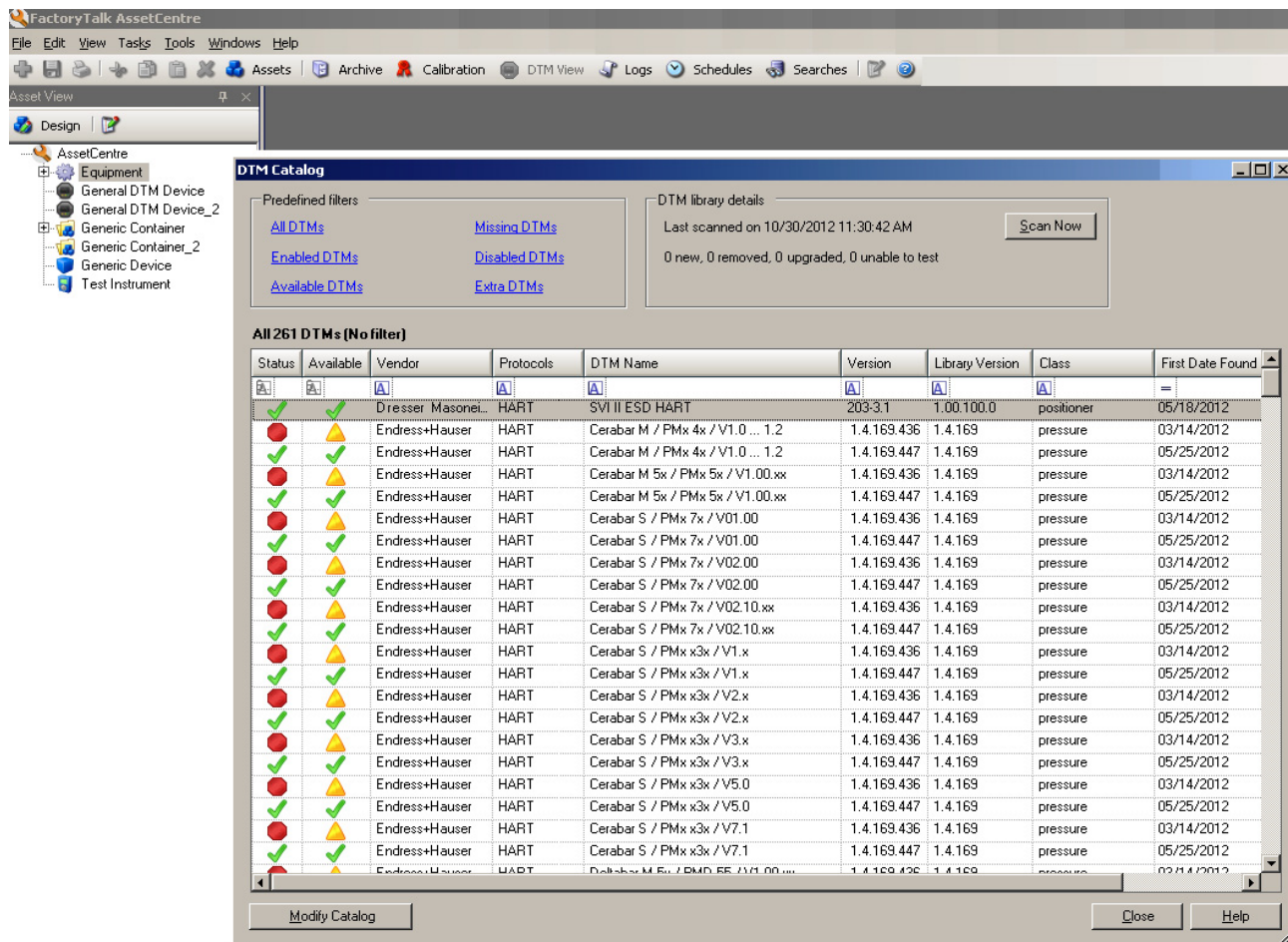


1. Create a HART variable for the channel and set the type to T9K\_AI\_HART\_FULL.
2. Select the module in the equipment view and then select the channel that you want to configure for field device monitoring.
3. Select the **HART** tab.
4. Click the button beside the HART Variable field. The Variable Selector dialog box opens.
5. Select the variable from the list, click **OK**.

- Repeat this procedure for other channels that will use HART enabled devices.

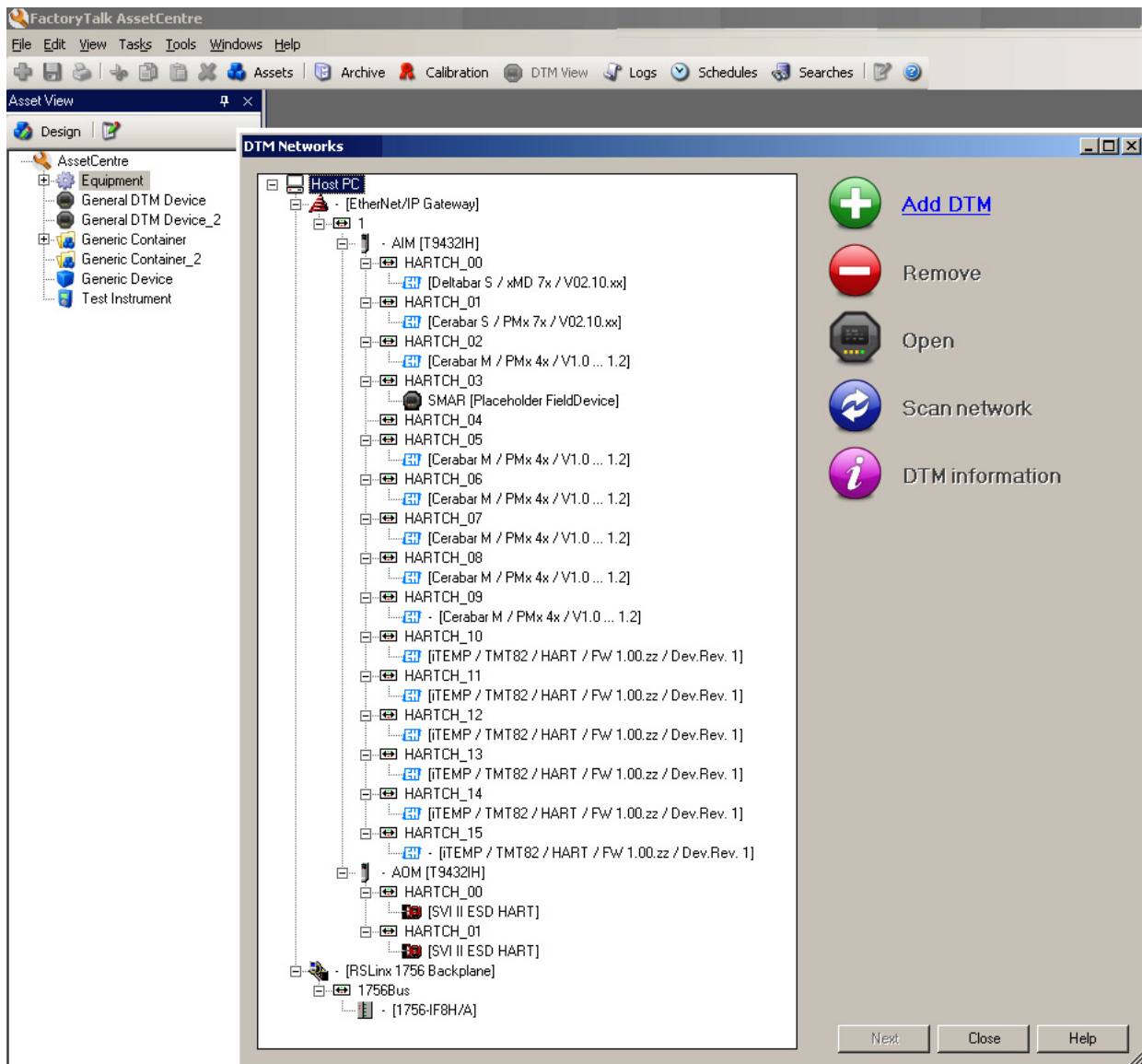
## Use FactoryTalk Asset Center with an Analogue Input Module

This example shows how to configure the FactoryTalk Asset Center to let it use the AADvance HART Pass-Through feature with an analogue input module. Do the following:



- Open the FactoryTalk Asset Centre and select **Tools** ➔ **DTM Catalog**.
- Click **Scan Now**.

- All the DTMs are scanned and automatically updated.



3. Right-click on the Host PC item at the top of the tree, click **Add DTM**.
4. Select EtherNet/IP Gateway, click **OK**.
  - This will appear in the tree structure under Host PC.
5. Right-click on 1 under Ethernet/IP Gateway and click **Add DTM**.
6. Select T9432IH.
  - It will appear in the tree structure under Ethernet/IP Gateway.
7. Name it AIM [T9432IH].
8. Right-click on AIM [T9432IH] and click **Configuration**.
9. Enter the IP address of the AADvance controller and the slot number for the analogue input module, click **Next**.
10. Right-click on AIM [T9432] and click **Scan network**.

11. Select all the channels in the Select Communication Channel window and click **OK**.
  - All the input channels on the AADvance controller are now set up for HART Pass-Through functionality.





# Using CIP over EtherNet/IP

This chapter introduces CIP over EtherNet/IP, and shows how to configure the protocol in the AADvance Workbench. It explains how to make a CIP network, and then configure the transfer of data.

## CIP over EtherNet/IP

The Common Industrial Protocol (CIP) over EtherNet/IP protocol enables AADvance controllers to exchange data with ControlLogix controllers programmed by RSLogix 5000. The exchange of data uses the produce/consume tag method currently used for sharing data between Logix-based controllers; this mechanism is similar to the variable bindings mechanism used by the AADvance controller.

The AADvance controller supports produce and consume communications to redundancy systems. The support for produce/consume variables is non-interfering; a failure of the EtherNet/IP stack will not interfere with the safe operation of the controller.

To use CIP over EtherNet/IP you have to first define a CIP network. Then you configure the exchange of data by defining a produce variable (or structure) for the AADvance controller and a corresponding consume variable (or structure) for the ControlLogix controller. At runtime, the controller with the consume variable pulls data from the controller with the produce variable.

IMPORTANT

The CIP Protocol is intended to allow AADvance users to exchange data between AADvance controllers and the Allen Bradley Logix family controllers, using produce/consume messaging. Produce/Consume messaging does not support downloading to or for monitoring AADvance controllers. It is not recommended to use the CIP network to exchange data between AADvance controllers unless this is exclusively for non-safety data. The SNCP network should be used for Safety related data exchange between AADvance controllers (see SNCP and variable Bindings in this publication).

## Connection Requirements of a Produced or Consumed Tag

Produced and consumed tags, all require connections. Each connection uses memory and processing resources. The demands are as follows:

Type of Tag	Number of Connections Used
Produced Tag	number of consumers + 1
Consumed Tag	1

**Example:**

Connection Requirements of a Produced or Consumed Tag:

- An AADvance controller producing 4 tags for 1 controller uses 8 connections
- Each tag uses 2 connections (1 consumer + 1 = 2)
- 2 connections per tag x 4 tags = 8 connections
- Consuming 4 tags from a controller uses 4 connections (1 connection per tag x 4 tags = 4 connections).

The number of connections rather than size of the data transferred over the connection is the significant factor.

It is far more efficient to include multiple items in a single connections that have individual connections for each one. (e.g. send an array of 10 DINTs rather than 10 individual DINTs )

An individual AADvance processor slice has a finite capability to process the amount of communications traffic that passes through its associated Ethernet ports.

Memory available to CIP is available per controller rather than per slice. (see memory requirements below)

A connection defines a single Producer or Consumer that can be as small as a DINT or as large as a structure up to 500 bytes.

A consumer or producer is associated with a specific AADvance Ethernet interface. This association is formed by virtue of the network path that a consumer is using to access a producer or the network interface through which a consume connection from another controller arrives.

Each AADvance interface, when used for CIP Produce/consume, must be assigned a unique subnet.

The number of consumer connections per slice is **C**

The number of producer connections per slice is **P**

'Load' **L** Per Slice = ( **C**+2**P** )

**L** must be <= 32

RPIs can be calculated as a function of **L**

**L** <= 16 Minimum RPI for any producer or consumer is 200 ms

16 < **L** < 24 Minimum RPI for any producer or consumer is 300 ms

**L** > 24 Minimum RPI for any producer or consumer is 500 ms

**Memory Considerations.**

CIP Produce/Consume variables utilize memory from a pool that is shared with other communications features of the AADvance controller.

Specifically:-

- MODBUS
- Sequence Of Events

- Variable Bindings

The total amount of space available is 131072 bytes

A Produced variable of 120 DINTS will consume 1552 bytes of this space

A Consumed variable of 120 DINTS will consume 1120 bytes of this space.

---

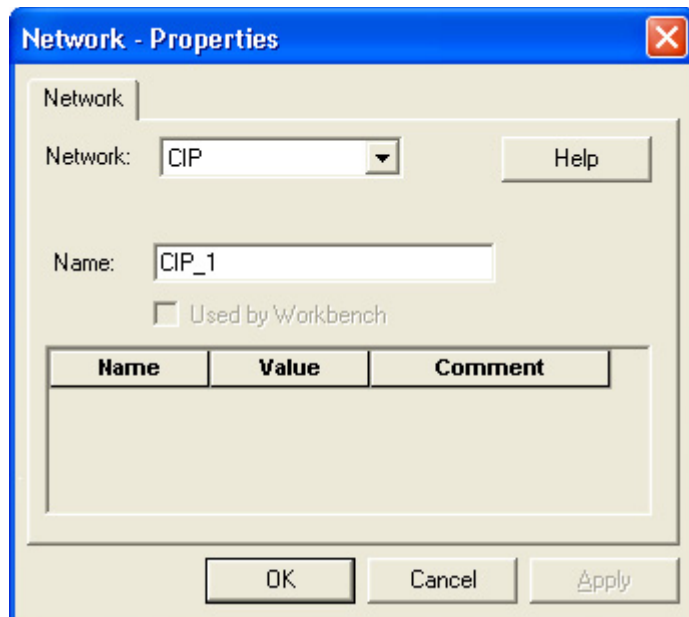
**NOTE** This memory is used by every slice e.g. 48 producers (each slice producing the maximum of 16 producers) will consume  $48 * 1152 = 74496$  bytes.

---

## Define a CIP Network

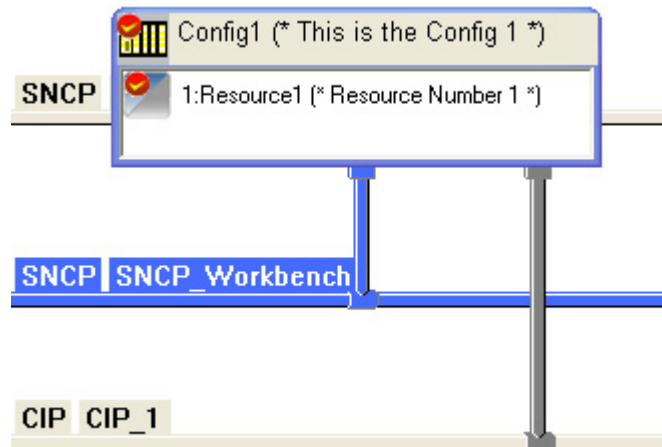
Before you use CIP over Ethernet/IP, you have to define a CIP network. You will specify a single IP address for each connection between a Configuration and a CIP network. Do the following:

1. Select the Hardware Architecture view.
2. Go to the main menu and choose **Insert → Network**.
  - The Network – Properties dialog opens.



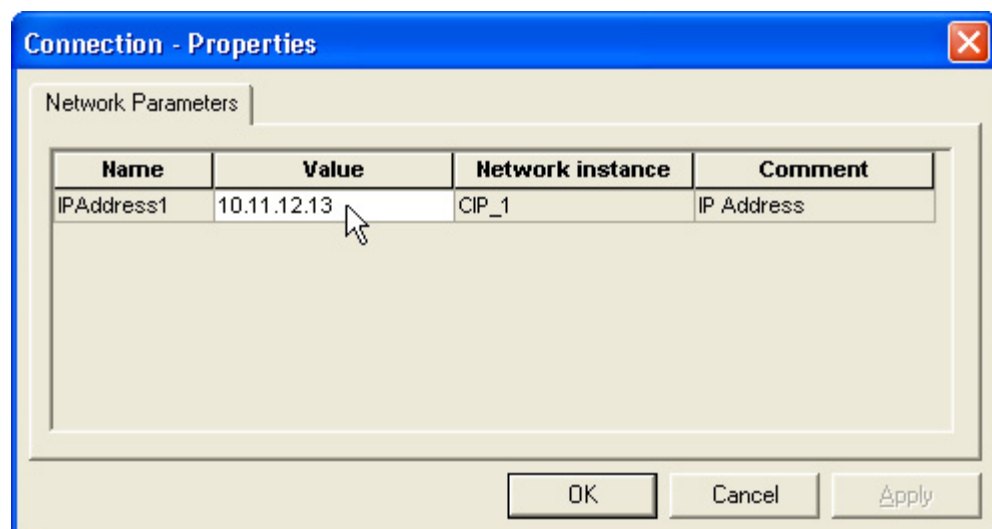
3. Set the Network to CIP, click **OK**.
  - The AADvance Workbench inserts a network into the Hardware Architecture view.

4. Create a connection between a Configuration and the newly inserted CIP network



**NOTE** You can only connect Series 9000 and Series 8000 configurations to a CIP network. If you try to connect a CIP network to a Configuration that does not support CIP (for example, a T6210 system) then the AADvance Workbench displays a warning, and the connection will not be made.

5. Select the CIP connection link then select the properties
  - The Connection – Properties dialog opens.



6. Enter the IP Address in the allocated to the AADvance controller for the CIP network

**NOTE** The default IP Address for a connection between a Configuration and a CIP Network is blank. If you do not specify a valid IP Address here, the AADvance Workbench will report an error at build time, 'Network CIP: invalid connection properties for ... (the particular configuration).'

The CIP communications network on the AADvance controller is now defined.

7. Now set up the RSLogix controller.

## Data Types for CIP

You can use the following IEC 61131-3 data types (or a compound type constructed from these) as producers or consumers with CIP over Ethernet/IP, as long as the data type is of size DINT (4 bytes / 32 bits) or larger (maximum is 500 bytes).

- BOOL
- SINT
- INT
- DINT
- LINT
- REAL
- LREAL.

You can use array or structure variables, but not the member elements of an array or structure. There are some limitations:

- The **CIP** tab is available for array or structure variables, but not for the member elements of an array or structure.
- You cannot use the STRING data type with CIP over Ethernet/IP in an AADvance system.

To share data types smaller than four bytes, you have to declare a user-defined data type and pad it accordingly. This is like the restriction defined within RSLogix 5000; the instruction there for sending items smaller than four bytes is to create a user-defined data type which will be padded to four bytes. The last member of the data type of the variable being produced/consumed must be either a structure or a 4 byte type. e.g. DINT, REAL.

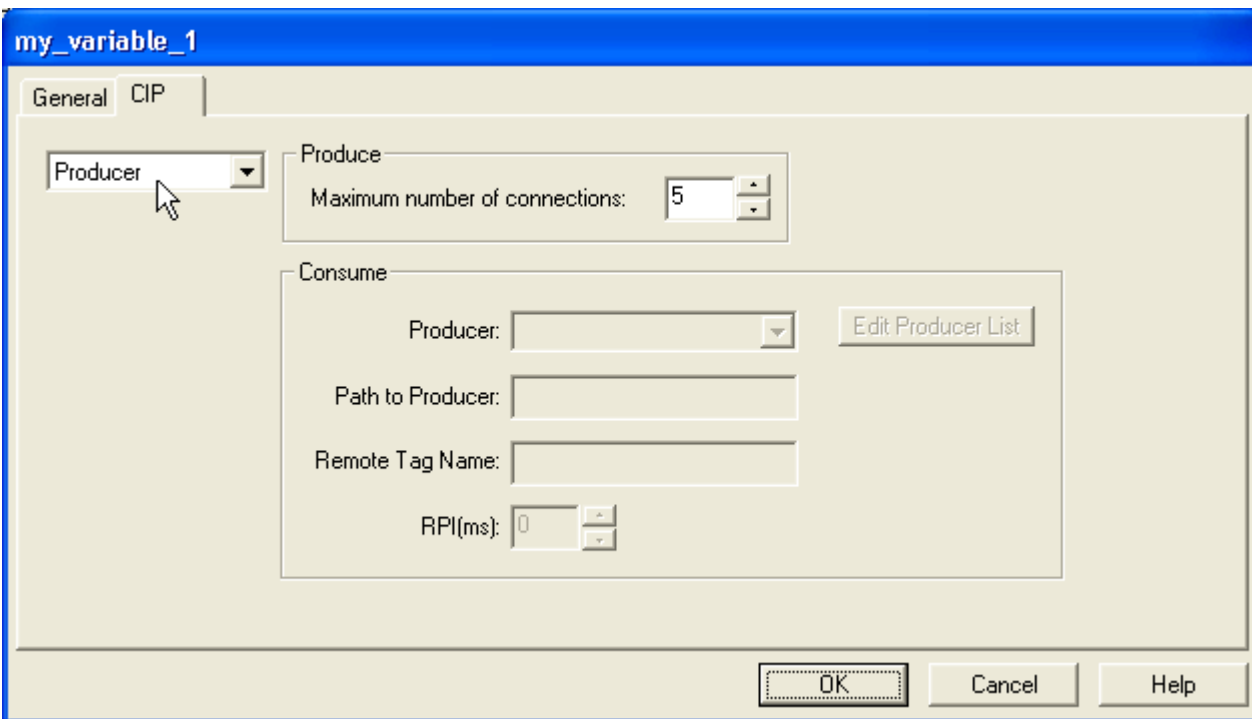
Arrays must be 4 byte aligned either by placing them inside a structure type or by ensuring they are preceded by a 4 byte type.

Structures are always 4 byte aligned unless they contain LREAL/LINT values which must be 8 byte aligned. The size of a structure containing LINT/LREAL members must also be a multiple of 8 bytes in size.

## Configure an AADvance Variable as a Producer

When you configure a variable as a producer for CIP over Ethernet/IP do the following:

1. Go to the Dictionary and select a CIP produce variable that you have previously defined in the dictionary.



2. Make sure that the variable is a type that supports CIP over Ethernet/IP.
3. Open the variable properties dialog box and select the **CIP** tab.
4. Select **Producer** from the drop-down list.
5. Set the Maximum number of connections to the maximum number of simultaneous consumers allowable for this variable.
6. Click **OK**.

The configuration of the variable as a producer is now complete.

## Defining an AADvance Controller as a CIP Producer

Within the RSLogix5000 CLX Configuration it is necessary to configure an AADvance controller as a CIP provider. Presently this is performed in the I/O Configuration Tree, do the following:

1. Expand the relevant CLX Ethernet I/P Module to display the Ethernet Network Tree.
2. Insert a new module.
3. In the Select Module dialog box select **Ethernet Module**.

4. Open the properties for the selected module and enter the following details:
  - Enter a Name
  - Enter a Description
  - Enter the IP Address of the AADvance Ethernet port that is configured for CIP communications.
  - In the Comm Format drop-down selector, choose None.

## Using the Dictionary with CIP

The variable properties dialog box in the Dictionary has a tab for CIP; use this tab for all CIP-related variables. You can configure a variable as a producer or as a consumer, but not both. The default is for a variable to not use CIP, you have to select CIP for a variable. The CIP tab is only available for global variables within a configuration that supports a connection to a CIP Network.

---

**IMPORTANT** Produce and consume with status tags are not supported prior to AADvance firmware R1.4

---

## CIP in the Dictionary View

The Dictionary View has a CIP column that shows whether a variable uses CIP, and it can be set to a Producer or a Consumer. The default entry in the CIP column is blank.

The entry in the CIP column is shaded for members of an array or structure variable types to show that the column is not applicable.

When the Dictionary View is in grid mode, double-clicking on a box in the CIP column opens the CIP tab of the variable properties dialog for the variable.

## Parameters for CIP Producer and Consumer Variables

Each production variable and consumption variable for CIP over Ethernet/IP communications has a set of parameters as detailed in the tables.

**Table 19 - Parameters for Production Variables**

Description	Value(s)	Default	Remarks
Maximum number of connections	1 to 10	5	The maximum number of simultaneous consumers of the particular variable (†)

Table 20 - Parameters for Consumption Variables

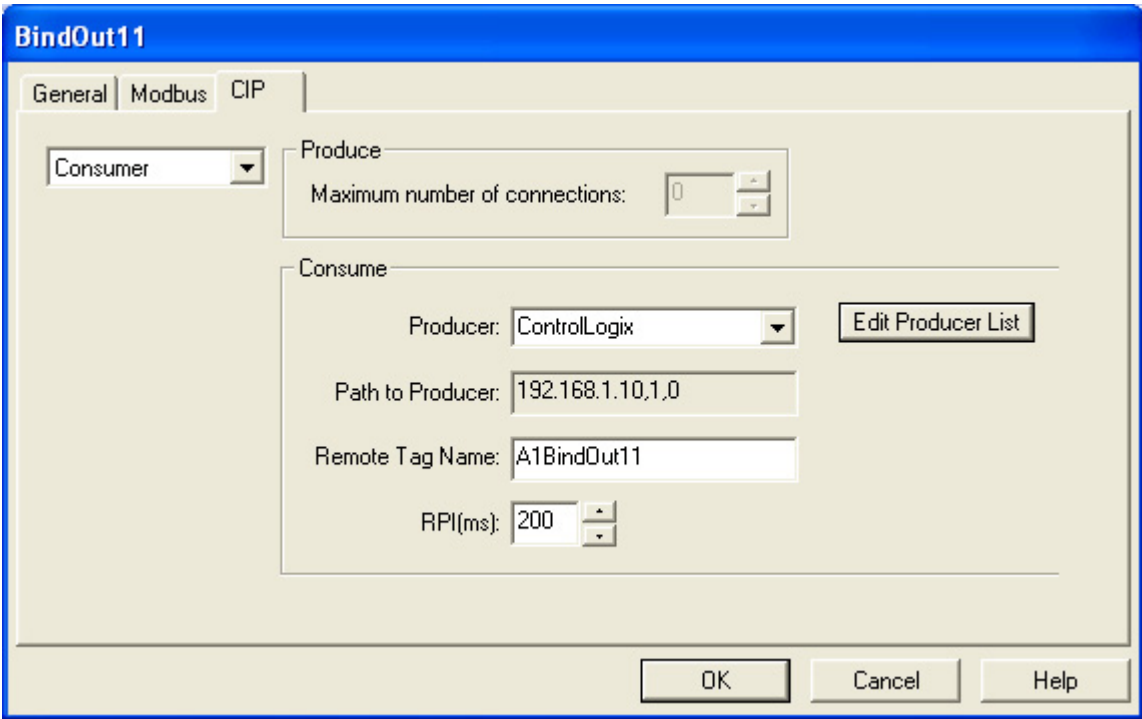
Description	Value(s)	Default	Remarks
Producer	A unique string Can include spaces and other punctuation characters	None	A user-defined name for the producing device labelled 'Device Name' in Producer List.
Path to Producer	A valid address for a Logix controller	None	Labelled 'CIP Path to Device' in Producer List. (refer to Configure an AADvance Variable as a consumer)
Remote Tag Name	A string	None	Specifies the name of the variable as it is known to the Logix controller.
RPI (ms)	2 to 1,000	20 (#)	Requested packet interval defines how frequently the Logix controller should offer the variable to the AADvance controller.

**NOTE** (#) The RPI should not be set below 200 ms at the current release.

Configure an AADvance Variable as a Consumer

When you configure a variable as a consumer for CIP over Ethernet/IP, you have to identify the producing Logix controller (it needs a name and an address), provide the name of the corresponding production variable (the remote tag name), and specify the requested packet interval. You can manage a list of producing controllers (called 'devices') from the CIP tab. This facility includes the ability to create a record of a device (giving it a name and specifying its CIP path), and to modify or delete records of existing devices.

Do the following:





1. Open the variable properties dialog and select the **CIP** tab.

**TIP** If the CIP tab is not visible, return to the Dictionary and make sure that the variable is a type that supports CIP over Ethernet/IP.

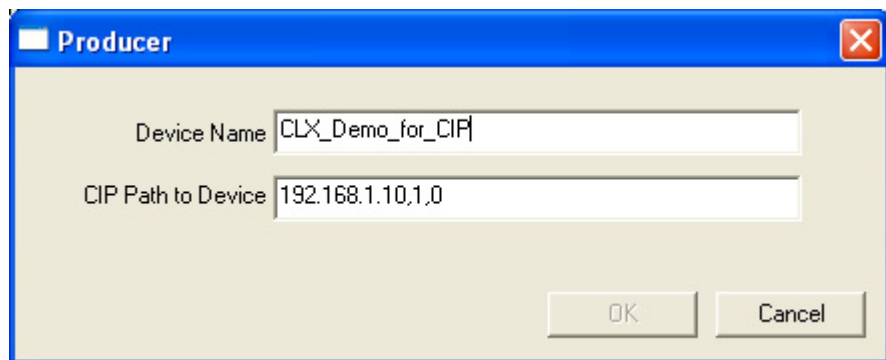
2. Select **Consumer** from the drop-down list.
3. If this is the first consumer variable in the AADvance Workbench project for the desired producer, you have to create a record for the producer. Do the following:
  - Click **Edit Producer List**. The Producer List opens; the dialog shows a list of records for producers.
  - Click **Add** and create a record to identify the producer. Specify the Device Name (this is a name of your choice).
  - Specify the CIP Path to Device (its address). This is in the format: <IP address>,<port>,<slot>

The IP address is the address of the ControlLogix Ethernet Module in the ControlLogix system.

The port is the communications route from ControlLogix. The value for the port should always be 1, which specifies the ControlLogix backplane. Other numbers specify different communications routes from ControlLogix.

The value for the slot is the slot number in the ControlLogix chassis where the CPU module is installed. These slots start from 0 and are numbered from left to right in the ControlLogix chassis.

- Click **OK**.



4. Click **OK** again to close the Producer List and return to the variable properties dialog.

You can create multiple records for a single producer; this may be useful if you expect the role of the producer controller to be divided between multiple controllers in the future. Each record will have a unique name, and will share the same path.

**TIP** If you make a mistake, you can edit a record to change the device name or its path, but you cannot delete a record. If a record is no longer needed, change its name to an obvious description.

5. Choose the producer for the variable from the drop-down list.

- The Path to Producer field shows the address of the producer as defined in the Producer List.
6. Set the Remote Tag Name to the name of the variable as it is known to the Logix controller.
  7. Set the RPI field to the requested packet interval, or accept the default. This value is specified in milliseconds. Do not set the RPI to less than 200 ms at the current release.
  8. Click **OK**.
  9. The configuration of the variable as a consumer is now complete.

## Obtaining the Connection Status for a Consumed Variable

---

**NOTE** Only available in release 1.4 and later

---

Your application can obtain the connection status for a consumed variable. To do this, make sure that the variable is a member of a structure data type that has two elements: a first element of the type `CONNECTION_STATUS` and a second element to match the data type of the variable. `CONNECTION_STATUS` is a pre-defined type.

The `CONNECTION_STATUS` type is itself a structure comprising two `BOOL` elements, `RunMode` and `ConnectionFaulted`:

- When the structure data type for a variable includes `CONNECTION_STATUS` as its first element then these fields will be automatically filled in by the AADvance controller, ready for your application to read them.
- In AADvance WB1.4, the `RunMode` of a consumed variable with a status indicates the Run Mode of the Logix controller producing the value being consumed.
- In AADvance WB1.4, the `RunMode` status variable can be set by the user's application to `True/False`. AADvance can only produce/consume when applications are running (unlike a Logix controller) so this flag can be used to indicate some other boolean condition in the AADvance controller application - e.g. is the security dongle fitted.

If the connection fails, the value of the consumption variable remains at its last received value; you cannot define an 'error' value to revert to if the communications link fails.

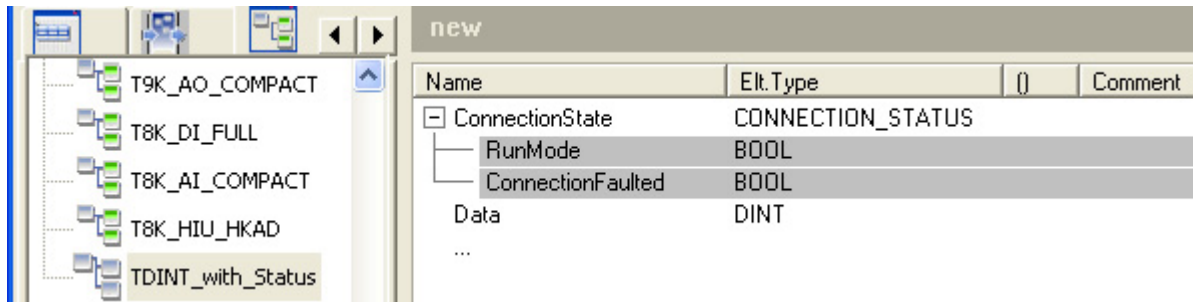
### Example

As an example, consider the consumer of a remote variable named 'T' that is a `DINT`. The variable will be named 'T' to match.

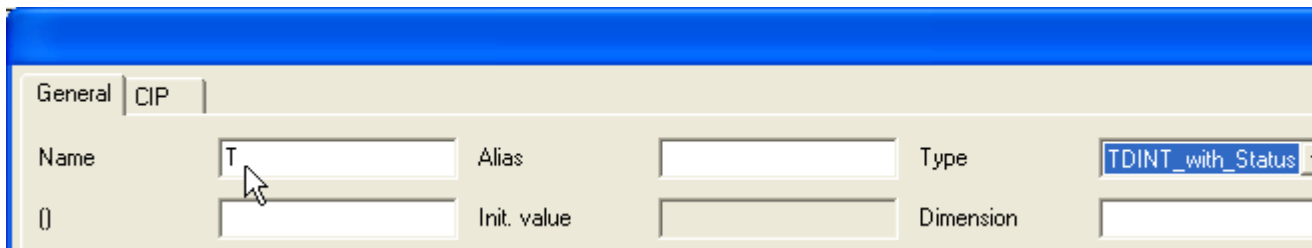
You have to define a structure data type of the form:

- `CONNECTION_STATUS` (the connection status information)
- `DINT` (the value of the data).

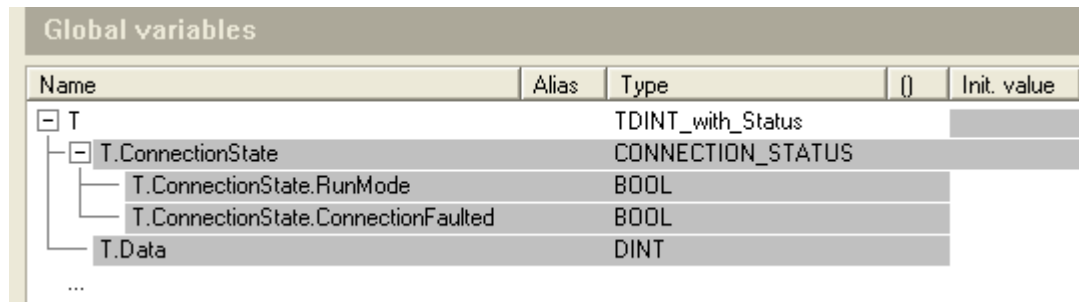
Give the structure data type a name, suitable for re-use within the project, such as TDINT.



When you create the variable 'T' in the Dictionary, declare its type to be TDINT\_with\_Status:



You can now examine the structure of 'T' within the Dictionary:



The AADvance controller sets the values of the elements T.ConnectionState.RunMode and T.ConnectionState.ConnectionFaulted based upon the run mode of the remote Logix controller and the presence of a communications fault.

## AADvance Producer RunMode

In AADvance WB1.4, the RunMode of a Produced with Status variable can be set by the user's application to True/False. AADvance can only produce/consume when applications are running so this flag can be used to indicate some Boolean condition in the AADvance controller application -e.g. is the security dongle fitted.

## CIP within the Application Scan Cycle

The AADvance controller updates variables configured for CIP production with new values at the end of each application scan cycle, after the logic has been executed. Variables configured for CIP production honor the Requested Packet Interval (RPI) expected by the Logix controller by providing the most recent value of the variable each time a packet is sent. Variables configured for CIP consumption are updated with the last received value at the beginning of each application cycle, before the logic is executed.

## About the RSLogix 5000 Configuration

Within the RSLogix 5000 Configuration it is necessary to configure the AADvance controller as a CIP provider. Presently this is done in the I/O Configuration Tree, by expanding the CLX Ethernet I/P Module to display the Ethernet Network tree, then insert a new module.

You can then proceed to configure production and consumption ('produce' and 'consume') variables normally. When you configure a variable for consumption, select the producing AADvance controller by name as defined in the I/O Config Tree.

## Set the RSLogix UNICAST Configuration

The AADvance controller uses the Unicast mode for CIP. You will need to configure the Unicast option in the RSLogix CLX Configuration.

### Produced Variable

1. Select a CLX Configuration Produced CIP variable.
2. Check the Allow Unicast Consumer Connections option.

### Consumed variable

For a Consumed Variable select a Consumed CIP variable.

1. Set the RPI value to 500 ms (the default value is 20.0 ms).
2. Check the Use Unicast Connection over Ethernet IP option.

## Further Information on CIP over Ethernet/IP

CIP over Ethernet/IP was created by Rockwell Automation for the Logix controller family. For further details of the protocol, such as details of Produce/Consume Tags, check the online help for RSLogix 5000.

## Configuring MODBUS Master

This chapter describes the process to configure the AADvance controller for MODBUS Master.

### MODBUS Master

The AADvance controller can be used as a MODBUS Master to one or more MODBUS Slave devices. Slave devices can include programmable logic controllers, remote devices (typically with little or no processing capability) and, more rarely, other functional safety controllers (Trusted or AADvance).

The controller supports the MODBUS RTU and MODBUS TCP protocols, and a subset of MODBUS commands. You can use MODBUS RTU with point-to-point and multi-drop serial links, and MODBUS TCP with Ethernet.

---

**NOTE** The AADvance controller does not support the MODBUS ASCII protocol

---

You can set up an individual list of messages (commands) for each Slave device. MODBUS read commands cause data to read from the Slave device to the MODBUS Master, while MODBUS write commands cause data to be copied from the MODBUS Master to the Slave device. You can also define a sequence of broadcast write commands, which a MODBUS Master can send to multiple MODBUS RTU Slaves without requiring an acknowledgment. The AADvance controller can control and monitor individual MODBUS Master objects and their Slave links.

The MODBUS Master functionality has a safety integrity level of zero (SIL 0) and should only be used for non-safety applications.

### MODBUS Standards

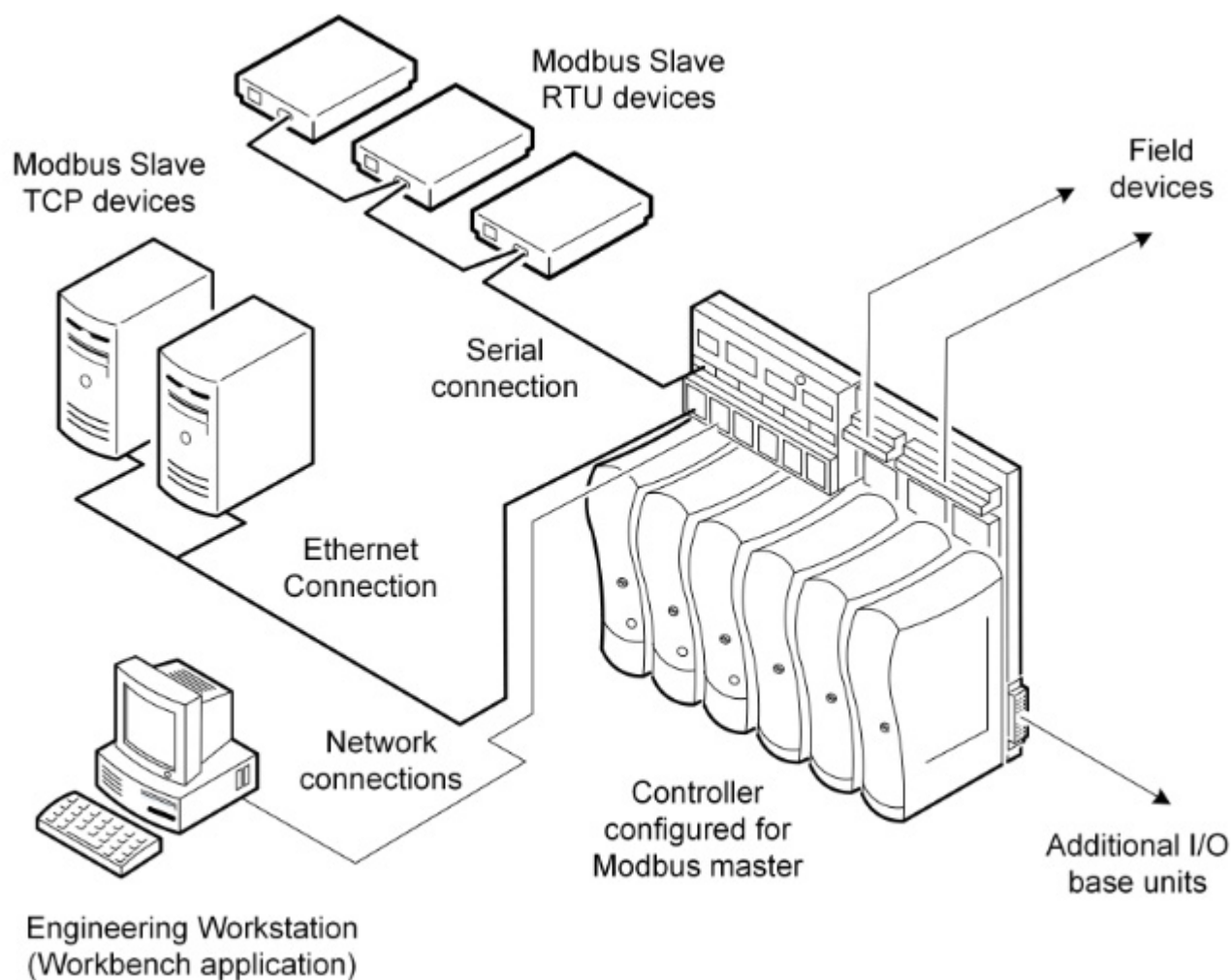
The MODBUS functionality implemented by the AADvance controller meets the following standard:

MODBUS Application Protocol Specification, version 1.1b. December 2006. The Modbus Organization.

## MODBUS Master Hardware and Physical Connections

The MODBUS Master functionality is built into the T9110 Processor Module; the physical communication ports are found on the T9100 Processor Base Unit. You do not need to add any extra hardware to the AADvance controller except to make the physical connections to the processor base unit. The illustration shows some possible arrangements of MODBUS Master connections.

**Figure 6 - Typical MODBUS Master Network**



The MODBUS RTU Slave devices are connected to one or more of the serial ports on the controller; a usual arrangement will use a multi-drop (RS-485) arrangement. The engineering workstation and the MODBUS TCP devices are shown connected to the Ethernet ports on different networks; alternatively these can be combined onto one network.

## MODBUS Master Command Set

The AADvance MODBUS Master supports the subset of MODBUS commands listed in the table.

**Table 21 - AADvance MODBUS Master Commands**

Command Code	Command	Remarks
01	Read coil status	Maximum 512 coils per message
02	Read input status	Maximum 512 inputs per message
03	Read holding registers	Maximum 125 registers per message
04	Read input registers	Maximum 125 registers per message
05	Force single coil	
06	Preset single register	
15	Write multiple coils	Maximum 512 coils per message
16	Write holding registers	Maximum 123 registers per message
08	Diagnostics	Subfunction 00, query used (RTU only)

## MODBUS Data Types and Addressing

The MODBUS Master supports the MODBUS data types listed in the table.

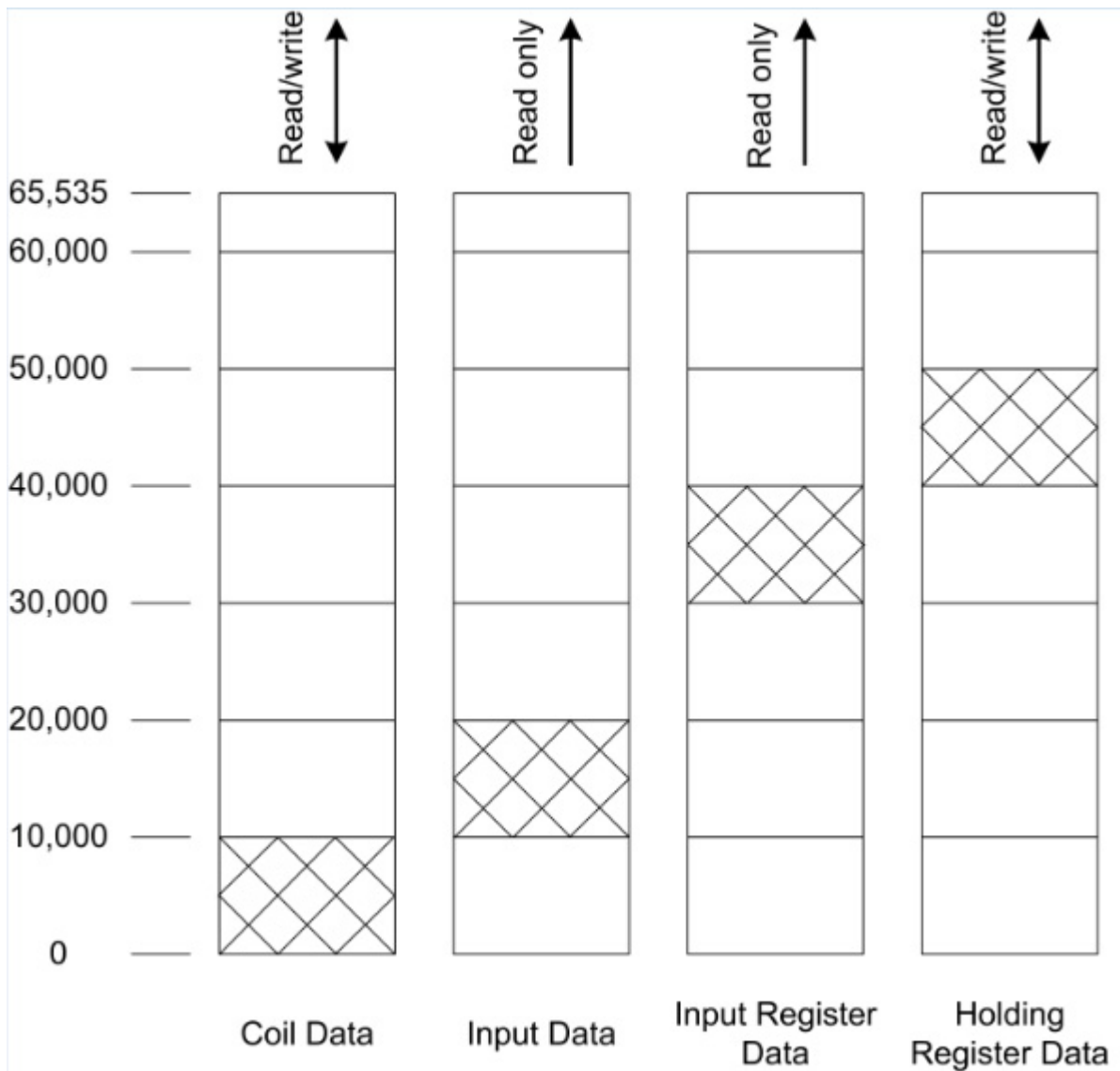
**Table 22 - MODBUS Data Types**

Data Type	Address Range
Coils	1 to 65,535
Inputs	1 to 65,535
Input registers	1 to 65,535
Holding registers	1 to 65,535

MODBUS uses a numeric addressing scheme to pass data between devices. AADvance controllers allocate a dedicated area for each of the four MODBUS data types and follow the model defined in the MODBUS Application Protocol Specification.

The original MODBUS standard defines the address field as a four-digit field, but with a prefix which related to the data type. The areas shown hatched in

the illustration show how original-style, five-digit MODBUS addresses (for example, a holding register at 40,001) relate to the AADvance memory map.



The addresses used for MODBUS data transfer listings start at one; the first AADvance Workbench variable network address is 1 and the first coil is 00001. An address of 0 for either field is illegal.

### MODBUS Message Scheduling

In operation, all of the MODBUS Master objects defined for a project (be they MODBUS RTU or MODBUS TCP) function independently of each other. Each Master polls its associated Slave devices, to send the messages scheduled for the Slaves.

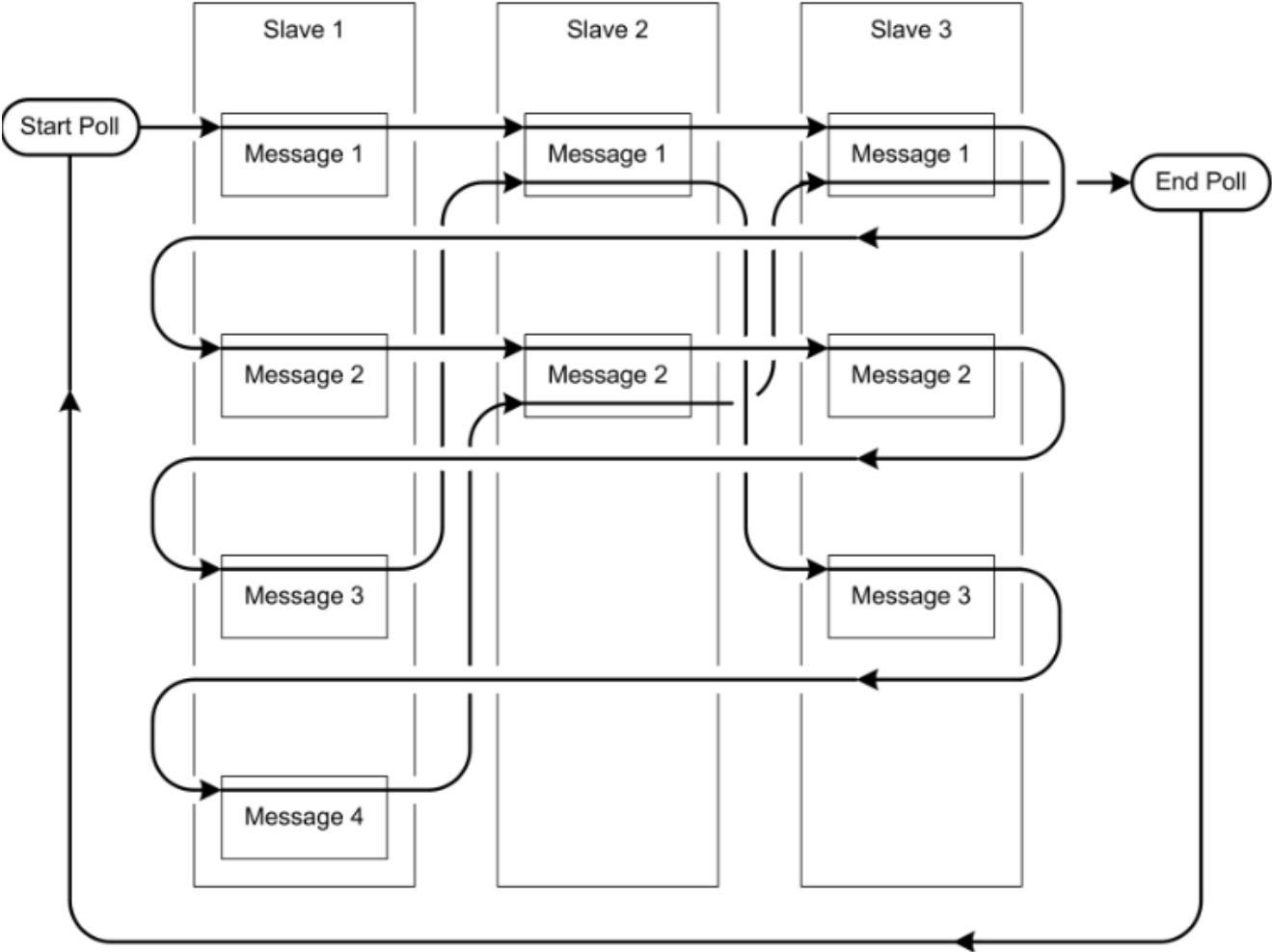
For MODBUS RTU, a Master object polls its Slave devices one at a time, executing one message for each Slave in turn. Different Slaves attached to the same Master will usually need different numbers of messages. To accommodate this, the Master proceeds to work through the messages listed for the Slaves,



one message for each pass, returning to the first message for a Slave after it has sent the last one. This means that some messages (for Slaves with fewer scheduled messages) will be sent more than once during a poll cycle. The poll cycle completes when the MODBUS Master has made an equal number of polls to each Slave and has sent every message; the cycle then starts again.

**Example**

The illustration shows the message scheduling for a Master with three Slaves needing four, two and three messages.



The total number of messages that the Master has to send in a complete poll cycle is equal to the largest number of messages scheduled for any one Slave, multiplied by the total number of Slaves. So, in this example the total number of messages is  $4 \times 3 = 12$ . There are nine messages defined for the three Slaves; the message scheduling mechanism sends three of these messages twice to make the total of twelve. The sequence of messages seen on the serial communications link in this example is listed in the table.

**Table 23 - Message Sequence for Example MODBUS Master**

Sequence Number	Slave	Message	Remarks
1	Slave 1	Message 1	
2	Slave 2	Message 1	
3	Slave 3	Message 1	End of first pass through all Slaves
4	Slave 1	Message 2	
5	Slave 2	Message 2	End of pass through Slave 2
6	Slave 3	Message 2	
7	Slave 1	Message 3	
8	Slave 2	Message 1	Repeat message
9	Slave 3	Message 3	End of pass through Slave 3
10	Slave 1	Message 4	End of pass through Slave 1
11	Slave 2	Message 2	Repeat message
12	Slave 3	Message 1	Repeat message; end of poll

## Handling MODBUS Communication Errors

If a MODBUS Master encounters communication errors to a Slave device, it suspends polling of that Slave for a period of time. This allows the polling of other Slaves on the same Master to continue without pausing for the communications time-out on each cycle through the poll. At regular intervals, the controller can ping the non-communicating Slave (this is a configurable setting); once the non-communicating Slave responds to a ping, then polling of the Slave will recommence on the next cycle.

---

**NOTE** If the controller is receiving error messages in the replies from a Slave, the Slave remains in the polling cycle; the controller does not deem this to be a communication failure.

---

## MODBUS Statistics

The AADvance controller maintains a cycle timer and counters to generate statistics for each Slave. The statistics are available to the application, so the application can react accordingly or report how well a link is operating; the statistics are also used by diagnostic commands. The application can read the statistics as variables (you specify the address of each variable) and can also reset the statistics to zero. Statistics are available for all point-to-point message transfers, but not for broadcast messages.

Three kinds of statistics are available:

- Last rate, which reports the length of the most recent scan time
- Maximum rate, which reports the longest scan time
- Average rate, which reports the average scan time

The statistics are reported in hundredths of a second (not milliseconds).

In operation, the statistics are updated on the first message of each Slave and on the first Slave link of each Master to produce the Slave and Master cycle times. The software also checks the application interface to see if a statistics reset is required.

Meanwhile, the MODBUS Master measures the overall scan time; this is from the first Slave start, to round to first Slave start again. Again, the statistics are updated on the first message of each Slave and the first Slave of each Master to produce the Slave and Master cycle times, and the application interface is checked to see if a statistics reset is required.

## MODBUS Service Parameters

When a MODBUS service is defined, use the **MODBUS** tab of the Variable dialog box to define the MODBUS service properties:

- **Write protected:** Defines the variable direction, and whether it can be written to. Available for coils and holding registers only.
- **Base Address:** The MODBUS base address of the variable. If an array is used, this field specifies the starting address of the array. For example, if the address is 10 and an array of 10INT is used, the address of the tenth element of the array is 20. The format of this property is hexadecimal; its value ranges from 1 to 65,536 (one-register wide) for BOOL, INT and UINT variables, 1 to 65,535 (two-registers wide) for DINT, UNIT and REAL variables, and 1 to 65,533 (four-registers wide) for LINT, ULINT and LREAL variables.
- **MODBUS Type:** The type of MODBUS variables. For Boolean variables, the available types are discrete input or coil. For INT, UINT, DINT, UDINT and REAL variables, the available types are input registers and holding registers.
- **MODBUS Functions:** The available MODBUS functions for use with the selected MODBUS type.
- **Data Type:** The variable data type.

## Diagnosing MODBUS Communications and Slave Devices

The AADvance Workbench makes diagnostic information available as application variables. The diagnostic information includes the status of a communication link and of the MODBUS Slave devices connected to it; you can define the location of the variables during programming within the Workbench.

## MODBUS Exception Responses

The AADvance controller uses the lack of a response from a Slave device to detect an error on a MODBUS link; also in some cases the Slave can return a code for a specific exception.

The MODBUS protocol allows for these errors by returning an error frame to the Master. The error frame consists of the original requested function code with the high bit set and a data field consisting of the specific error code. The AADvance MODBUS Master does not log these codes explicitly, but it does

recognize them and treat them as a general error to the device. The Slave status variable is set to 'Slave Error' to allow the application to respond to the exception.

## AADvance Objects for MODBUS Master

The AADvance controller implements its MODBUS Master functionality through one or more MODBUS Master objects, which you create within the AADvance Workbench. Each MODBUS Master object can connect to one MODBUS TCP device; or to one or more MODBUS RTU devices.

MODBUS Master objects are allocated to 9110 processor modules; a MODBUS Master object can use one of the two Ethernet ports or one of the two serial ports associated with the processor module. If you need more than one MODBUS Master object, and the controller has two or three processor modules, you can distribute your MODBUS Master objects between the processor modules to make the best use of the ports. This gives flexibility if for example the connections to your MODBUS TCP devices are through more than one network.

Failure of a processor module will cause the failure of the MODBUS communications allocated to that processor module. When a controller has more than one processor module, you can distribute your MODBUS Master objects between two or three of the processor modules. Do this within the AADvance Workbench when you create the MODBUS Master objects. If a processor module fails, Slave communications will continue to operate on the channels associated with the remaining processor modules.

The AADvance Workbench represents each Slave link by a Slave object, one object for each Slave device. You can distribute Slave objects between MODBUS Master objects as desired, subject to a constraint of no more than one Slave link for a Master object configured for MODBUS TCP. Master/Slave relationships are configured under the Master objects.

## MODBUS Master Capacities

MODBUS Master, Slave and message configurations are subject to the limits shown in the table.

**Table 24 - MODBUS Master Capacities per processor module**

Item	Limit
Maximum number of Masters	32
Maximum number of Slaves	5
Maximum number of messages	400

## Planning for MODBUS Master

To use the AADvance controller as a MODBUS Master, you have to complete three activities:

- Define the physical connections from the controller to the Slave devices.
- Configure the characteristics for the serial ports (MODBUS RTU only).
- Set up the project for MODBUS Master, and configure the application.

Each of these activities is independent. Therefore, you can execute these in any order, but you have to complete all three activities.

## Physical Connections for MODBUS RTU

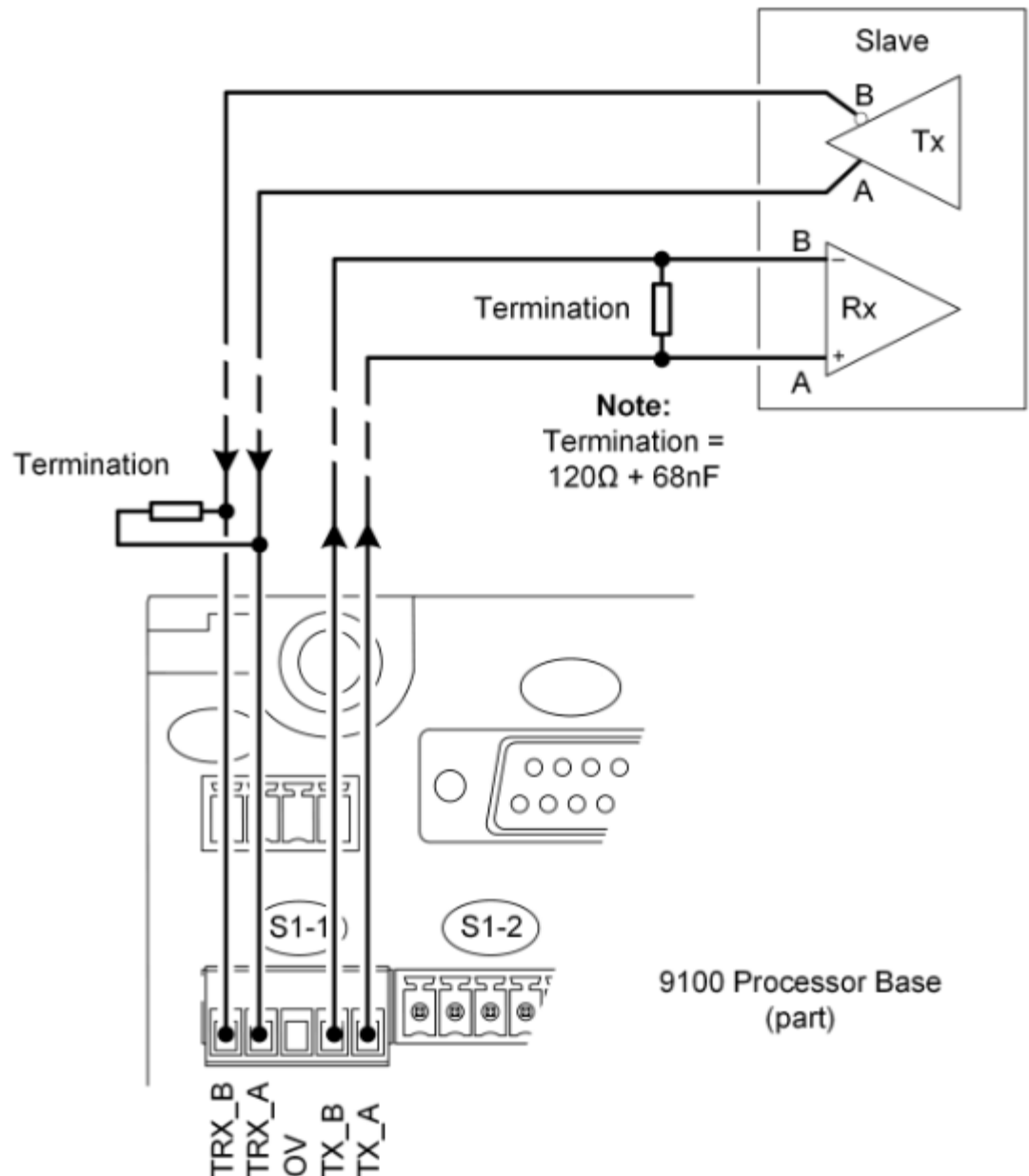
MODBUS Master RTU communications use the serial ports on the 9100 processor base unit. The following fundamental topologies are possible:

- Point to point
- Point to multi-point (multi-drop).

Each of these topologies can use a full-duplex, 4-wire connection or a half-duplex, 2-wire connection. For details refer to the AADvance System Build Manual.

## Connect a Slave Device, Full Duplex

You can use a full duplex serial connection to connect a single MODBUS Slave device to the AADvance controller. To make the physical connection, do the following:



1. Select a suitable cable. We recommend 3-pair, overall shielded cable.
2. Remove the serial port connector from the 9100 processor base unit.
3. Make the connections shown in the illustration. Terminate the twisted pairs with a  $120\Omega$  resistor in series with a  $68\text{ nF}$  capacitor at the receiver ends.

4. Insert the connector into the 9100 processor base unit.

---

**IMPORTANT** Do not connect the signal ground to the AC safety ground.

---

## Connect Multiple Slave Devices, Full Duplex

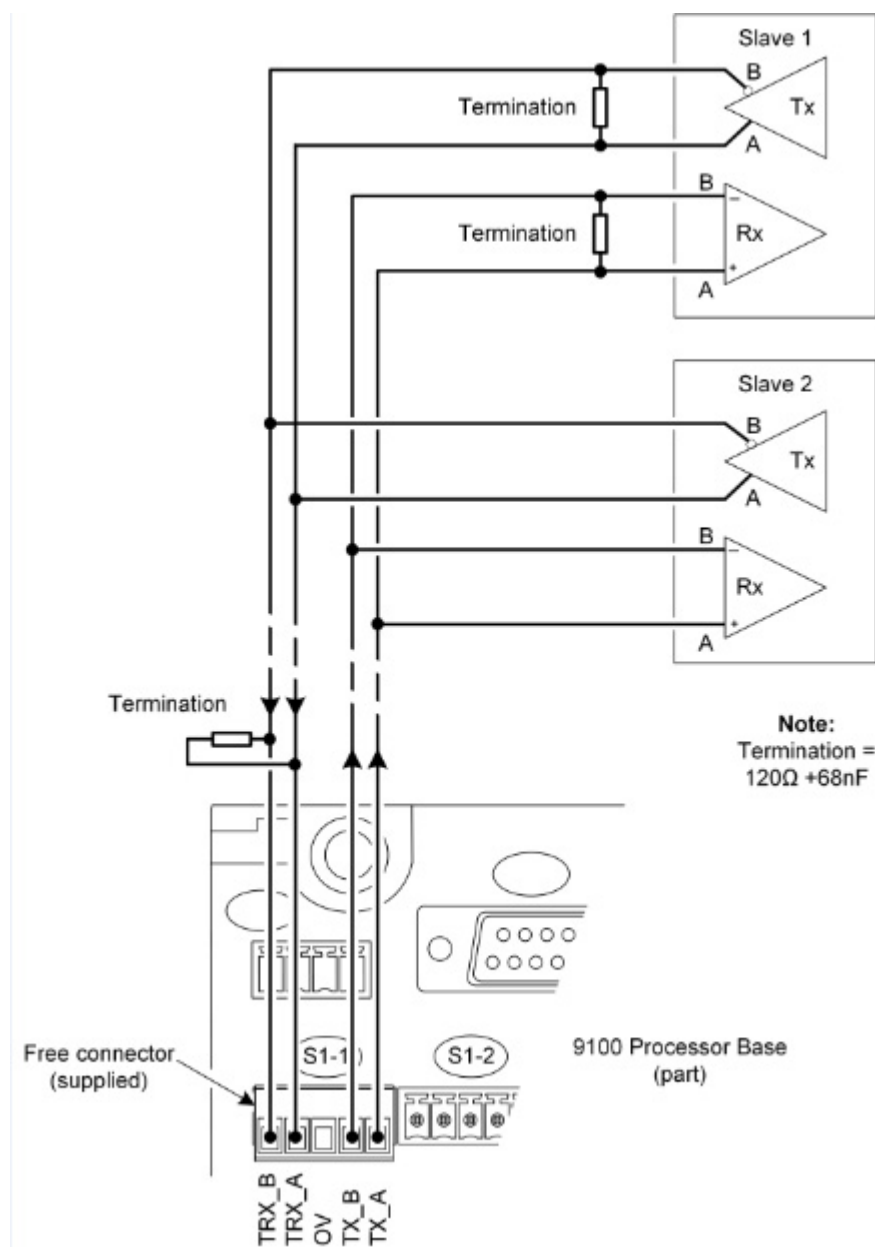
You can use a full duplex serial connection to connect multiple MODBUS Slave devices to the AADvance controller. To make the physical connection, do the following:

1. Select a suitable cable. We recommend 3-pair, overall shielded cable.
2. Remove the serial port connector from the 9100 processor base unit.
3. Make the connections shown in the illustration. Terminate the twisted pairs with a 120 ohm resistor in series with a 68 nF capacitor at the locations shown.
4. Connect the signal ground (not illustrated) from the 0V terminal to each Slave device.
5. Insert the connector into the 9100 processor base unit.

---

**IMPORTANT** Do not connect the signal ground to the AC safety ground.

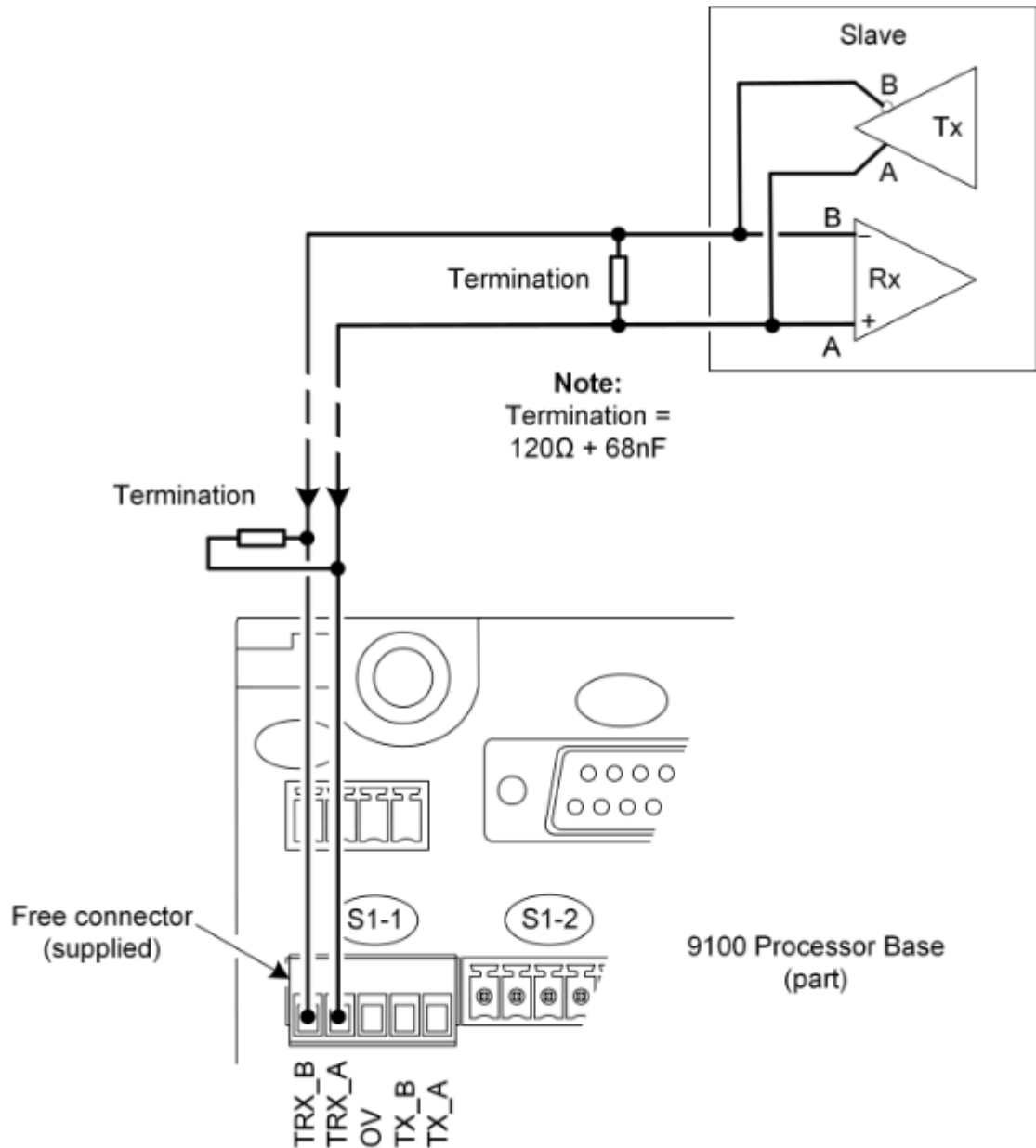
---





## Connect a Slave Device, Half Duplex

You can use a half duplex serial connection to connect a single MODBUS Slave device to the AADvance controller. To make the physical connection, do the following:



1. Select a suitable cable. We recommend 2-pair, overall shielded cable.
2. Remove the serial port connector from the 9100 processor base unit.
3. Make the connections shown in the illustration. Terminate the twisted pair with a 120 ohm resistor in series with a 68 nF capacitor at both ends.

4. Connect the signal ground (not illustrated) from the 0V terminal to the Slave device.
5. Insert the connector into the 9100 processor base unit.

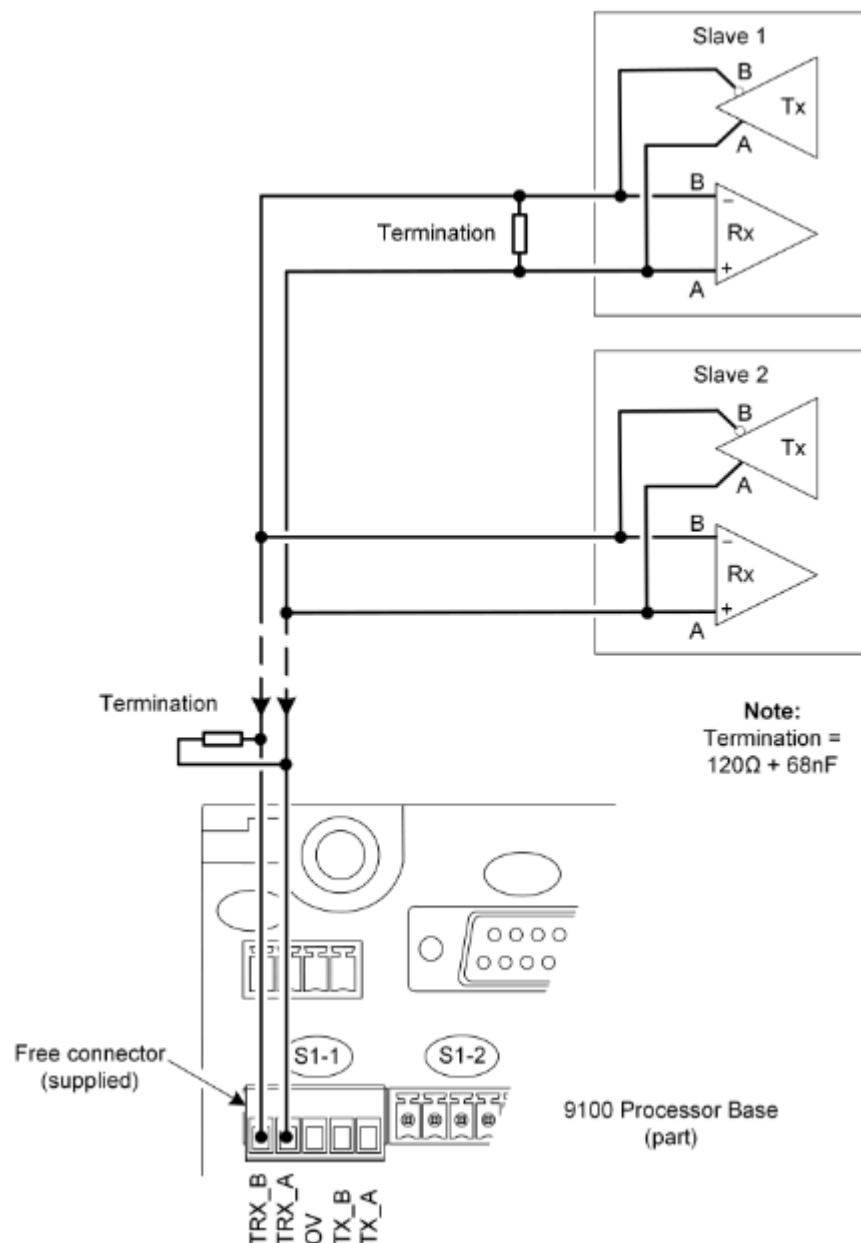
---

**IMPORTANT** Do not connect the signal ground to the AC safety ground.

---

## Connect Multiple Slave Devices, Half Duplex

You can use a half duplex serial connection to connect multiple MODBUS Slave devices to the AADvance controller. To make the physical connection, do the following:



1. Select a suitable cable. We recommend 2-pair, overall shielded cable.
2. Remove the serial port connector from the 9100 processor base unit.
3. Make the connections shown in the illustration. Terminate the twisted pair at both ends with a 120 ohm resistor in series with a 68 nF capacitor.
4. Connect the signal ground (not illustrated) from the 0V terminal to each Slave device.
5. Insert the connector into the 9100 processor base unit.

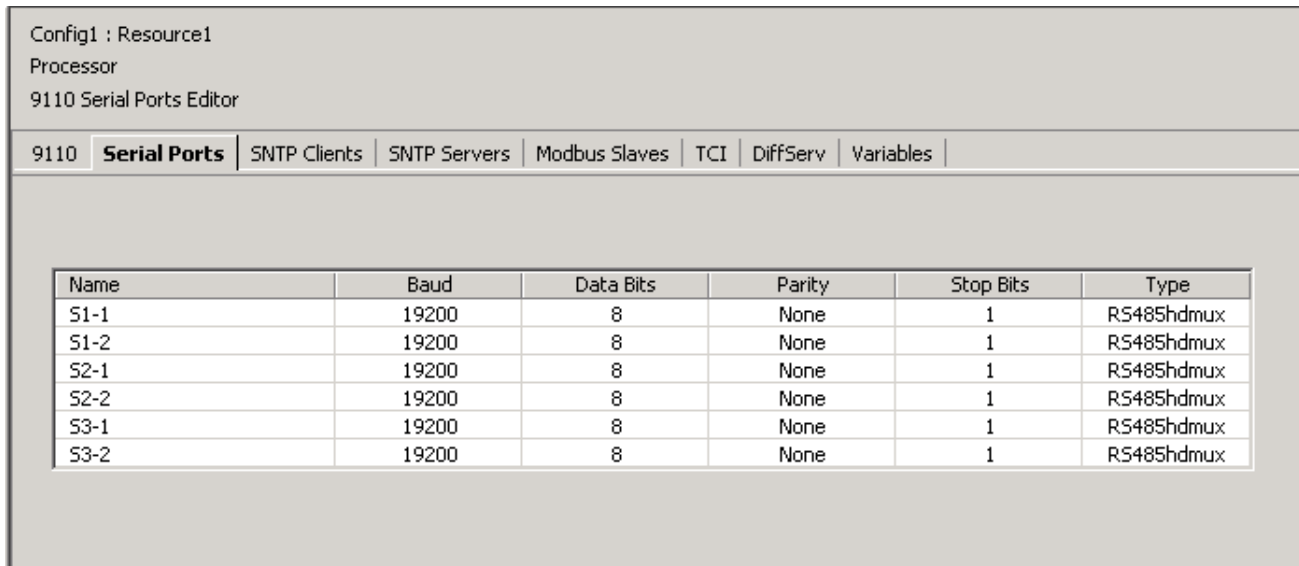
---

**IMPORTANT** Do not connect the signal ground to the AC safety ground.

---

## Configure the Serial Ports for MODBUS Master

When you use a serial port on the AADvance controller for MODBUS Master (MODBUS RTU), you have to configure the serial port. In particular, you have to set the port type to suit the arrangement of the physical connection. To configure the serial port, do the following:



1. Select **9110 Processor** on the Project Tree View to open the 9110 Module Editor.
2. Select the **Serial Ports** tab.
  - The Serial Ports editor dialog box opens.
3. Locate the chosen serial port (S1-1 for example) in the editor and set the Type parameter:
  - If the physical connection is a full duplex (4-wire) point-to-point connection to a single Slave, choose **RS485fd**.
  - If the physical connection is a full duplex (4-wire) connection to multiple Slaves (tri-state outputs on the Slave transmitters), choose **RS485fdmux**.
  - If the physical connection is half duplex (2-wire), choose **RS485hdmux**.

4. Set the remaining communication parameters from the drop down lists, to match those used by the Slave devices; click **Apply**.
5. To restore the default values, click **Default** then **Apply**.

## Serial Port Parameters

Each serial port on the AADvance controller supports the set of control parameters as detailed in the table.

**Table 25 - Controller Serial Port Parameters**

Description	Value(s)	Default	Remarks
Baud	1,200; 2,400; 4,800; 9,600; 19,200; 38,400; 57,600; 76,800 or 115,200	19,200	
Data Bits	5 to 8	8	
Parity	None, Odd or Even	None	
Stop Bits	1 or 2	1	
Type	RS485fd RS485fdmux RS485hdmux	RS485hdmux	'fd' means 'full duplex' 'hd' means 'half duplex'

**TIP** Most systems use two bits after each data byte. The two bits are either a parity bit (odd or even) and one stop bit, or no parity and two stop bits.

## Physical Connections for MODBUS TCP

MODBUS Master TCP communications use the Ethernet ports on the 9100 processor base unit.

## Setting Up the Project for MODBUS Master Operation

The top-level process to set up your project for MODBUS Master operation consists of the following tasks.

1. Insert the MODBUS Master Bus.
2. Create and configure a MODBUS Master object on a 9110 processor module.
3. Create Slave link objects for each MODBUS Slave device.
4. Configure the Slave link objects.
5. Add further MODBUS Master objects and Slave link objects as needed.

Before you begin the process, make sure you have the following information for each MODBUS Slave device:

- The MODBUS protocol to use: MODBUS RTU for serial links or MODBUS TCP for an Ethernet link.
- The identity of the serial or Ethernet port you intend to use.

The AADvance Workbench provides separate editors for the MODBUS Master objects and for their Slave links. In each editor, you will navigate between tabbed pages to reach the various configuration settings. Click **Apply** to save your settings before you leave a tabbed page.

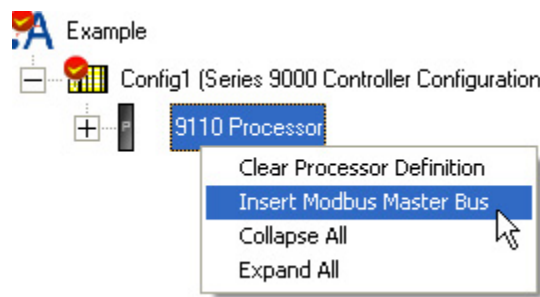
**TIP** If you alter the settings and omit to click **Apply** before you try to navigate away from them, the Workbench prompts you to save your changes. Click **Yes**.

## About the MODBUS Master Bus

The controller accesses its MODBUS Master functionality through a single MODBUS Master Bus. You have to configure the AADvance Workbench to use the MODBUS Master Bus with an application. After you have done this, the AADvance Workbench has a graphical display to show you the relationships between your MODBUS objects.

## Insert the MODBUS Master Bus

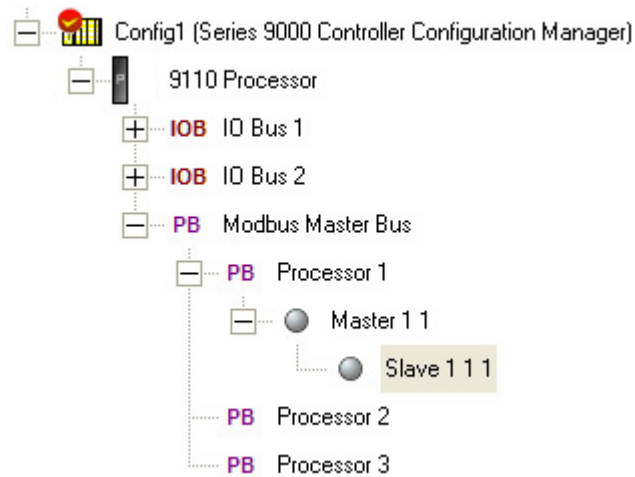
You have to insert the MODBUS Master Bus before you can identify MODBUS Master objects and their MODBUS Slave link objects.



To insert the MODBUS Master Bus do the following:

1. Select the **Equipment** tab on the Project tree view.
2. Right-click on **9110 Processor** and select **Insert Modbus Master Bus**.
  - The Workbench inserts the MODBUS Master Bus below the 9110 Processor.
  - You can now create a MODBUS Master object.

The structure of the Master objects and their Slave link objects in relation to the MODBUS Master Bus looks like this.



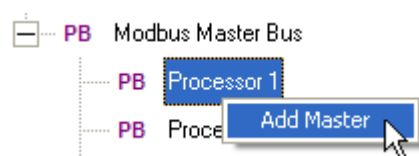
## Create a MODBUS Master Object

---


**IMPORTANT** You have to insert the MODBUS Master Bus before you can do this task.

---

A MODBUS Master object provides the configuration settings you need to add a MODBUS Master capability to the AADvance controller. You have to create a new MODBUS Master object for each group of multi-drop MODBUS RTU Slaves that use the same serial port on the controller, and a new MODBUS Master object for each MODBUS TCP Slave on the network.



Select the **Equipment** tab on the Project tree view and locate the **Modbus Master Bus** item.

1. Click the  symbol beside the Modbus Master Bus.
2. The Modbus Master Bus item expands to show the three potential processor modules in the controller.
  - Choose the processor module that will support the MODBUS Master object. This will be the module associated with the serial port or the Ethernet port that you intend to use.
3. Right-click on the desired **Processor Item** and select **Add Master**.
4. The AADvance Workbench inserts a MODBUS Master object below the chosen processor item.
  - The object is configured for MODBUS RTU; you can change this later to MODBUS TCP.

- The AADvance Workbench allocates a default name to the object; the name includes the number of the processor module, with a serial number for the object. You can change the name when you configure the object.
- You can now configure the MODBUS Master object.

## MODBUS Master Communication and Control Settings

Each MODBUS Master object has a set of communication and control settings as detailed in the table. The AADvance Workbench groups these settings together under the label 'General'.

**Table 26 - MODBUS Master Communication and Control Settings**

Description	Value(s)	Default	Remarks
Protocol	RTU or TCP	RTU	
Port Type	Serial or Ethernet	Serial (†)	
Port Id	S1-1 .. S3-2	Sn-1, where n identifies processor	
Timeout	0 to 60,000 ms	1000 ms	
Control Variable Address	1 to 65,536, or 0 to disable	0	The address of a control register (a holding register). (††) 0 = inactive 1 = standby 2 = active
Status Variable Address	1 to 65,536, or 0 to disable	0	The address of a status register (a holding register). (††) 0 = healthy 1 = initializing 2 = error
Message Wait Interval	0 to 65,535 ms	100 ms	For legacy MODBUS Slave devices with slow communications responses

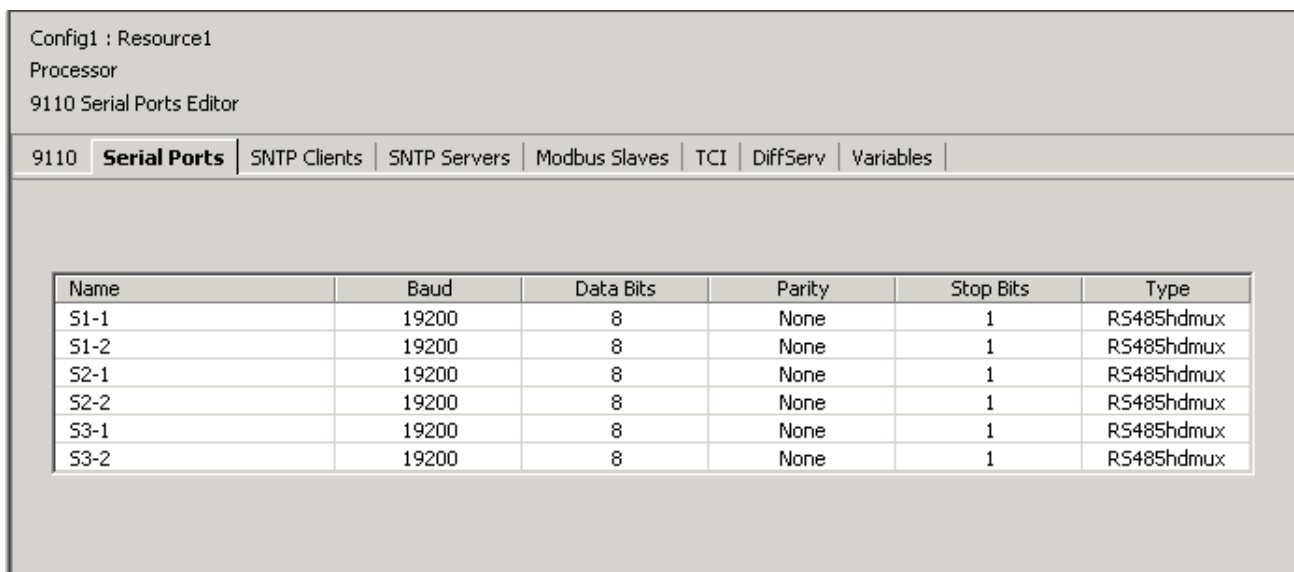
(†) The AADvance Workbench sets the Port Type field automatically after you set the Protocol field. You cannot set the Port Type field manually.

(††) The 'remarks' here summarize the values allowed for the contents of the register. The Control variable may be written to by the application to switch the state of the Master. The Status variable may be read by the application to monitor the Master. These variables are described in Controlling a MODBUS Master Object.

## Configure a MODBUS Master Object for MODBUS RTU

**IMPORTANT** You have to create a MODBUS Master object before you can carry out this task.

You have to configure the communications and control settings for a MODBUS Master object for MODBUS RTU when you want to use the MODBUS RTU protocol. Do the following:



1. Select the **Equipment** tab on the Project tree view and then select the **MODBUS Master object** you want to configure.
  - The editor for the MODBUS Master object opens.
2. Select the **General** tab.
3. Enter a suitable name for the MODBUS Master object, or accept the default. You can use any combination of alphanumeric characters and punctuation, including spaces.
4. Ensure that the Protocol is set to RTU (the default), click **Apply**.
  - The editor has three tabs; the tabs are named General, Broadcast and Statistics.
  - The Workbench sets the Port Type to Serial.
  - The Workbench sets the Port Id to the first serial port on the associated processor module.
5. Set the **Port Id** to match the identity of the **Serial Port** you will use for the link to the MODBUS Slave(s).
6. Set the **Timeout** (in milliseconds) to suit the system, or accept the default value. We recommend the Timeout be set to a value less than 3,000 ms.
7. Set the **Message Wait Interval** (in milliseconds) to suit the Slave devices, or accept the default value.



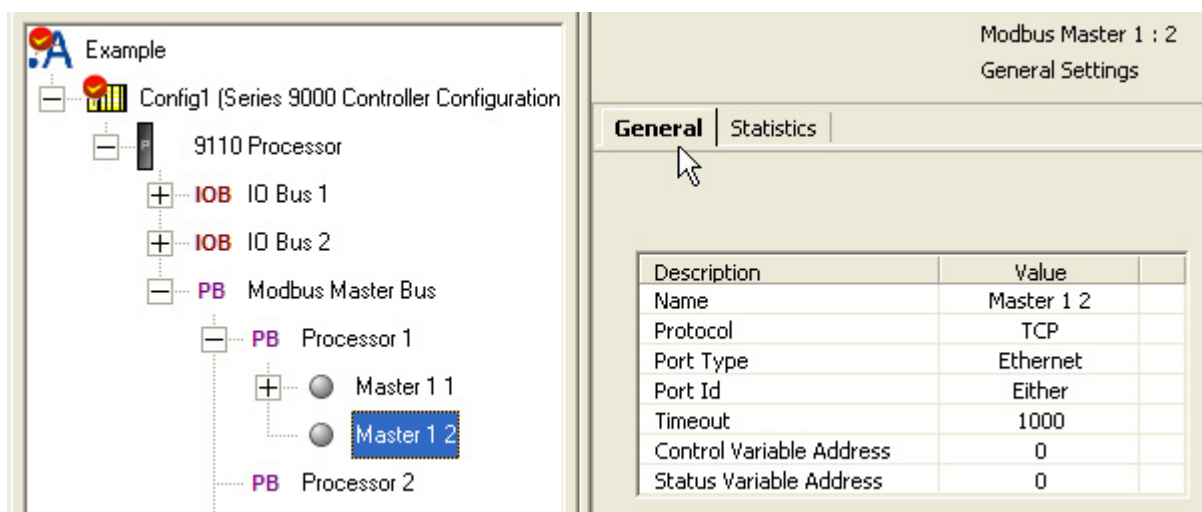
8. If you intend to use the control variable, set the **Control Variable Address** to the **MODBUS address of the holding register**; otherwise accept the default 0 to disable.
9. Similarly set the **Status Variable Address**.
10. Click **Apply**.

## Configure a MODBUS Master Object for MODBUS TCP

**IMPORTANT** You have to create a MODBUS Master object before you can do this task.

You have to configure the communications and control settings for a MODBUS Master object for MODBUS TCP when you want to use the MODBUS TCP protocol. Do the following:

1. Select the **Equipment** tab on the Project tree view and then select the **MODBUS Master** object you want to configure.
  - The editor for the MODBUS Master object opens.
2. Select the **General** tab.
3. Enter a suitable name for the MODBUS Master object, or accept the default. You can use any combination of alphanumeric characters and punctuation, including spaces.
4. Set the Protocol to TCP and click **Apply**.
  - The editor has two tabs; the tabs are named General and Statistics.
  - The AADvance Workbench sets the Port Type to Ethernet.
  - The AADvance Workbench sets the Port Id to Either (this means either physical RJ45 socket).



5. Set the Timeout (in milliseconds) to suit the performance of the network and of the Slave, or accept the default value. We recommend the Timeout be set to a value less than 3,000 ms.

6. If you intend to use the control variable, set the Control Variable Address to the MODBUS address of the holding register; otherwise accept the default 0 to disable.
7. Similarly set the Status Variable Address.
8. Click **Apply**.

### **Choosing Names for MODBUS Objects**

The MODBUS object Name identifies a MODBUS Master object or Slave link within the AADvance Workbench, and is included in printed reports; but it is not used by the application. We recommend that you use a different name for each object. Alternatively, accept the default name.

### **MODBUS Master Timeout**

The MODBUS Master Timeout is common to all Slaves configured to a particular Master. It specifies the period of time the MODBUS Master will wait for a response from its Slave device(s) before retrying or assuming that the Slave is unavailable. The MODBUS Master uses the timeout for each of its configured Slaves to determine whether they are still communicating. If a Slave does not respond within this period (and a specific number of retries), it will be deemed to have failed.

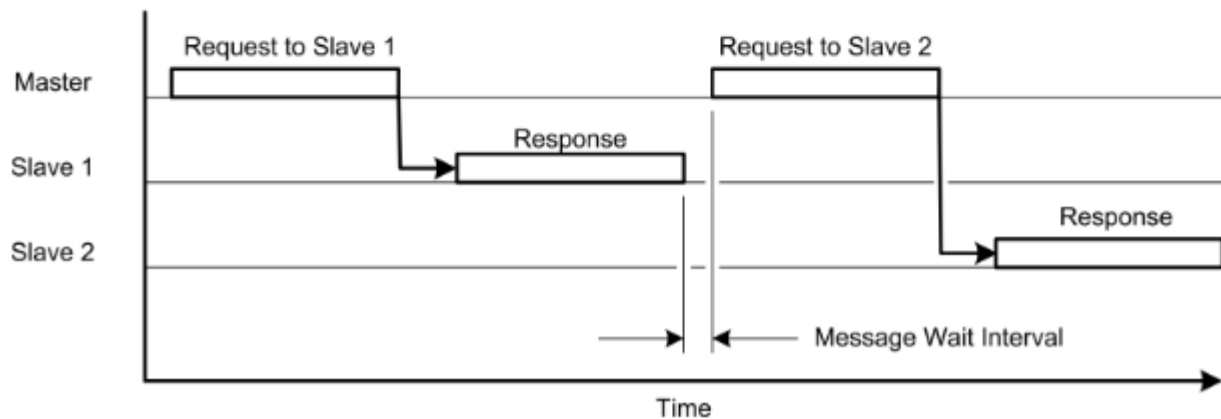
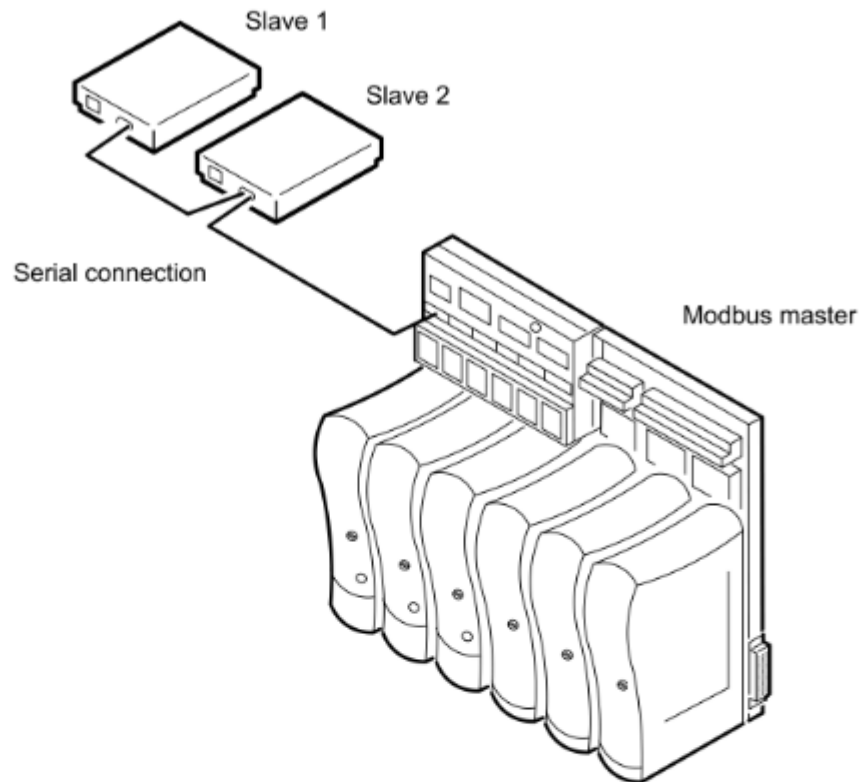
**TIP** Although the value of timeout is common to all Slaves configured to a particular Master, you configure the number of retries independently for each Slave.

### **Message Wait Interval**

The Message Wait Interval enables the MODBUS Master to support legacy MODBUS Slave devices, with slow communications responses, attached to multi-drop serial links. It is used only for serial MODBUS communications using MODBUS RTU.

The interval provides time for a legacy Slave device to establish that a response from another Slave to an earlier request from the MODBUS Master is not a new request targeted to the legacy Slave. The illustration shows a multi-drop

link from the MODBUS Master to two Slave devices, with the request message to Slave 2 delayed by the message wait interval.



The interval applies to all transmissions from the MODBUS Master, including ping messages and broadcast messages as well as regular messages to Slaves, and is common to all Slave links configured to a particular Master.

---

**IMPORTANT** Do not use the Message Wait Interval as a timing control.

---

## Controlling a MODBUS Master Object

Every MODBUS Master object has a pair of registers (holding registers), which allow the application to control the object, and to retrieve status information. The registers are a 'control register' and a 'status register'. Each register is located at a unique address — the Control Variable Address and the Status Variable Address.

The use of the registers is optional. If you want to use a register, you have to declare a variable for it in the application Dictionary and also specify the address in the holding register map.

An application can use the control register to control a specific Master. If you do not specify a control variable address, then the Master is enabled automatically when the controller is started. The variable in the application Dictionary has the functions listed in the table.

**Table 27 - MODBUS Master Control Register**

Value	Meaning	Description
0	Inactive	Disables the MODBUS Master The link is inactive, no communications activity All connected Slaves set inactive
1	Standby	Forces the MODBUS Master to operate in its standby mode Link is active but no data is being transferred
2	Active	Forces the MODBUS Master to operate in its active mode Link is active and transferring data

**TIP** If the application sets the control register to any other value, the MODBUS Master is disabled

The status register is an unsigned integer value (UINT) that is returned by the MODBUS driver to allow the application to monitor and act on faults. If you do not specify a status register address, then the Master does not report its status information to the application. The variable in the application Dictionary has the values listed in the table.

**Table 28 - MODBUS Master Status Register**

Value	Meaning	Description
0	Healthy	The MODBUS Master is operating normally Link is active, no errors reported
1	Initializing	The MODBUS Master is initializing
2	Error	The MODBUS Master has encountered an error and is disabled Unable to establish link

## MODBUS Ping Mode, Interval and Address

The ping mode is specified individually for each Slave; it defines the action the MODBUS Master should take if the Slave fails to respond after the specified number of retries.

The AADvance 40 controller supports a Function Code 08 ping mode for MODBUS RTU Slaves; choose this if the Slave supports it because it is a short

message. The Read Holding Register mode is an alternative; choose this if the Slave does not support Function Code 08. The MODBUS Master will try to read a single register. Read Holding Register is the only method available to ping a MODBUS TCP Slave. You can also set a Do Not Ping setting for both protocols.

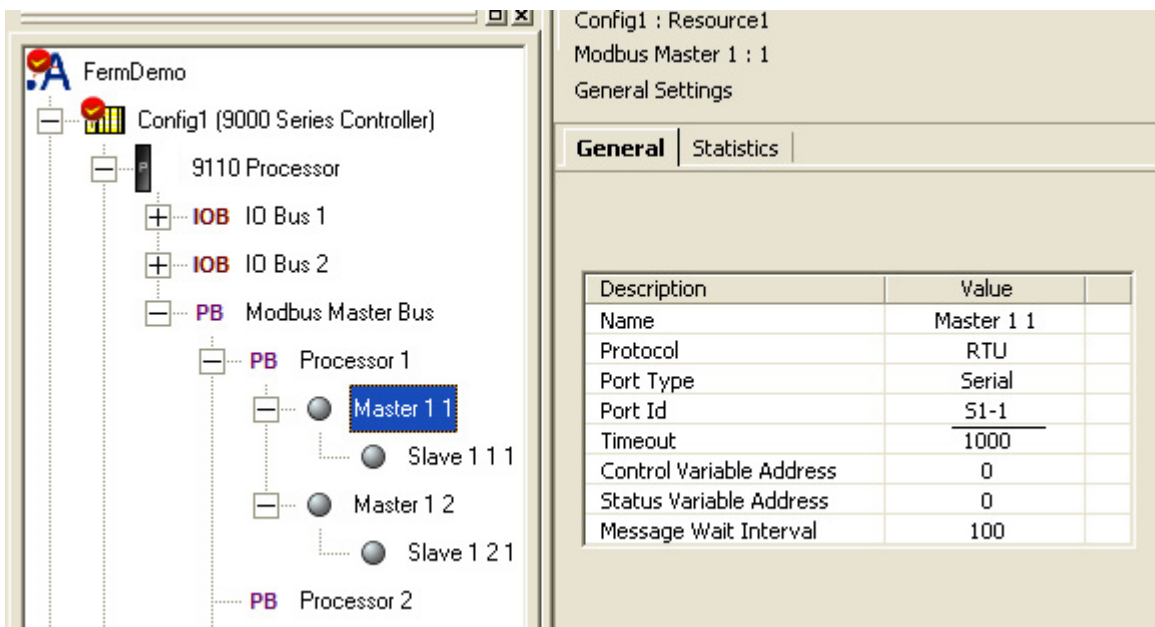
The ping address is for the Read Holding Register mode and is also specified individually for each Slave; it is the MODBUS address of the holding register. It is not used by the Function Code 08 mode.

### *MODBUS Ping Mode Settings - RTU*

To check the settings do the following:

#### **Function Code 08 Ping Mode Setting**

1. Select the **Equipment** tab on the Project tree and then a MODBUS Master object that has been set up to run the RTU protocol.
2. Check this by selecting the **General** tab and that the Protocol row is set to RTU.

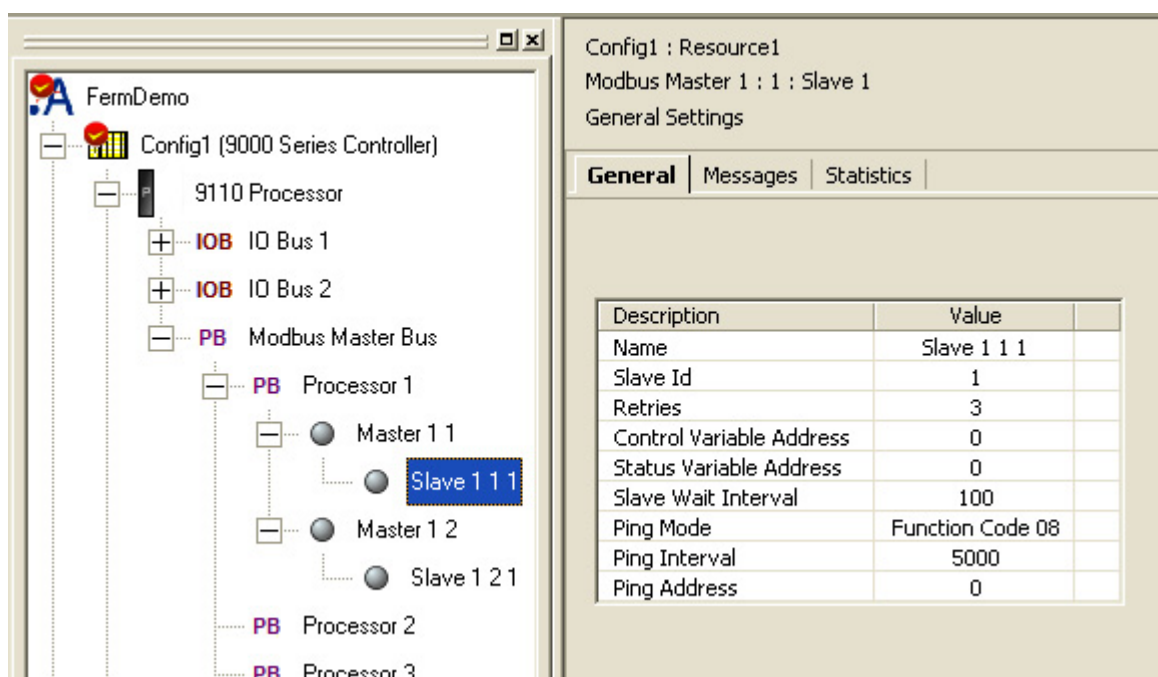


Config1 : Resource1  
Modbus Master 1 : 1  
General Settings

Description	Value
Name	Master 1 1
Protocol	RTU
Port Type	Serial
Port Id	S1-1
Timeout	1000
Control Variable Address	0
Status Variable Address	0
Message Wait Interval	100

3. Select the **Slave Link** associated with the Master object.
  - The editor for the Slave link opens.
4. Select the **General** tab.

5. Check that the Ping Mode has been set for Function Code 08.



The screenshot displays the configuration interface for a 9000 Series Controller. The left pane shows a tree view of the configuration, including the 9110 Processor, IO Bus 1, IO Bus 2, Modbus Master Bus, and Processor 1. The right pane shows the 'General' settings tab for 'Slave 1 1 1'.

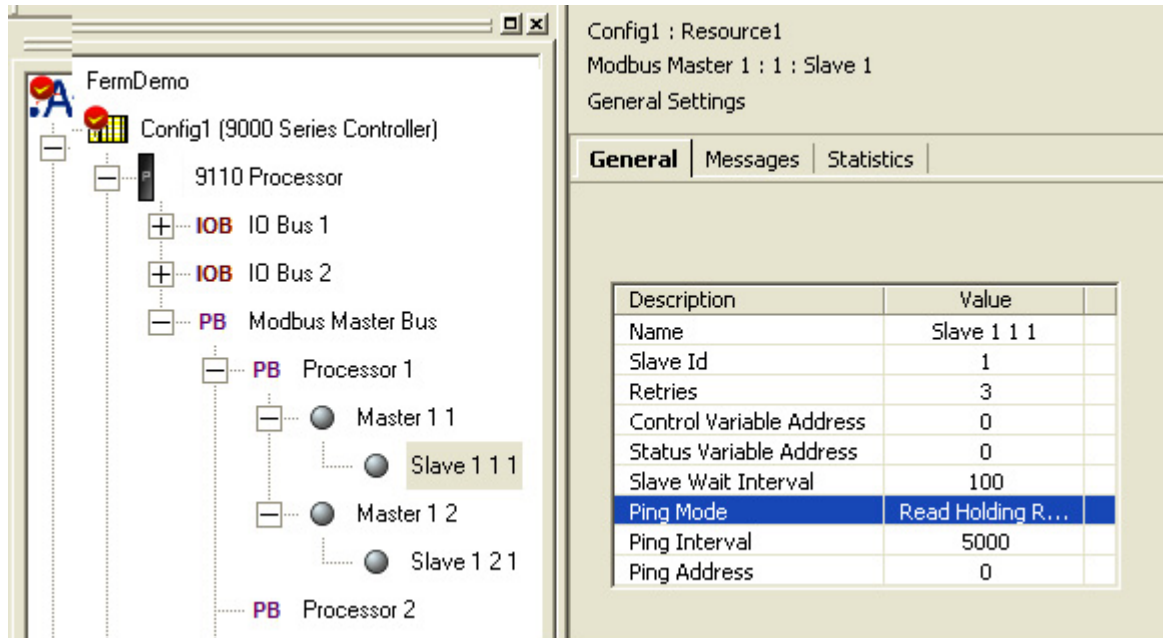
Description	Value
Name	Slave 1 1 1
Slave Id	1
Retries	3
Control Variable Address	0
Status Variable Address	0
Slave Wait Interval	100
Ping Mode	Function Code 08
Ping Interval	5000
Ping Address	0

6. Check that the Ping Interval is suitable for your application.

- The ping interval is specified individually for each Slave; it defines the maximum time between each initiation of a Function Code 08 diagnostic message.

## Read Holding Register Ping Mode Setting

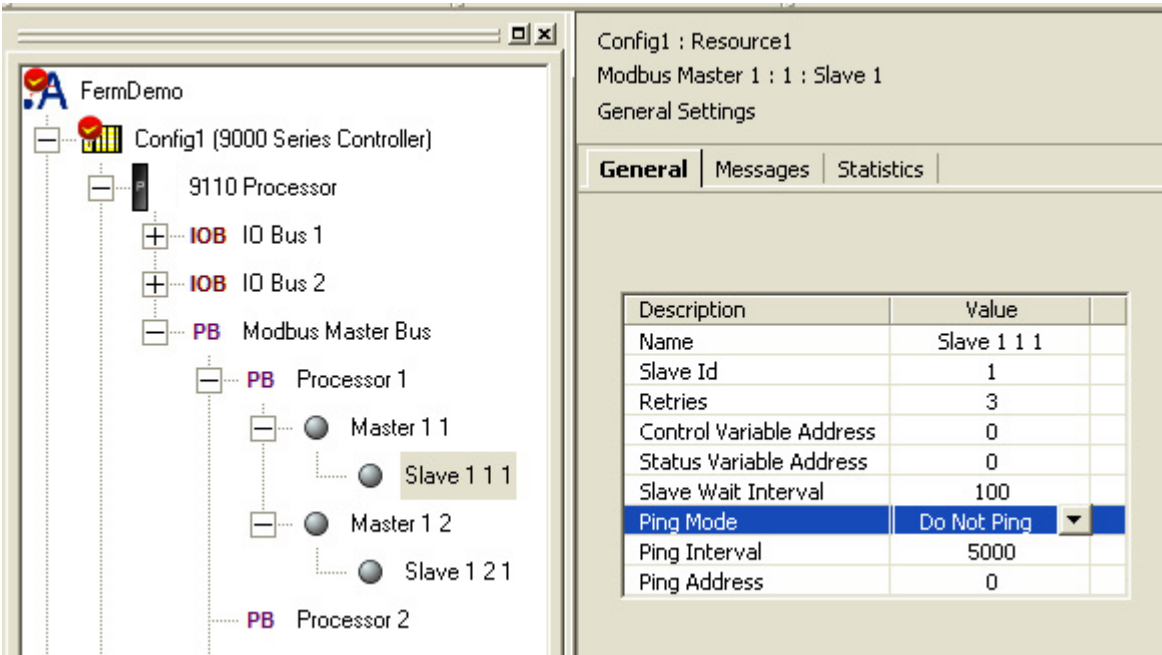
1. Select the **Slave Link** associated with the Master object set to RTU protocol.
  - The editor for the Slave link opens.



2. Select the **General** tab.
3. Check that the Ping Mode has been set for Read Holding Register.
4. Ensure that a Ping Address has been entered.
5. Check that the Ping Interval is suitable for your application.
  - The ping interval is specified individually for each Slave; it defines the maximum time between each initiation of a Read Holding Register diagnostic message.

Do Not Ping Setting

1. Select the **Slave Link** associated with the Master object set to RTU protocol.
  - The editor for the Slave link opens.



2. Select the **General** tab.
3. Check that the Ping Mode has been set to Do Not Ping.

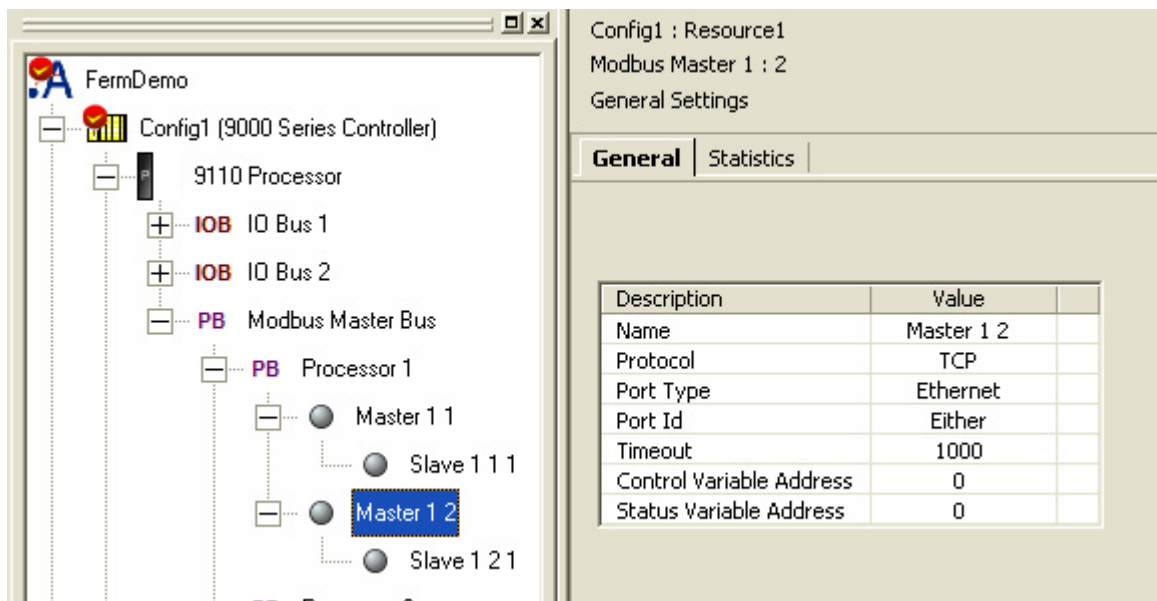
*MODBUS Ping Mode Setting - TCP*

To check the settings do the following:

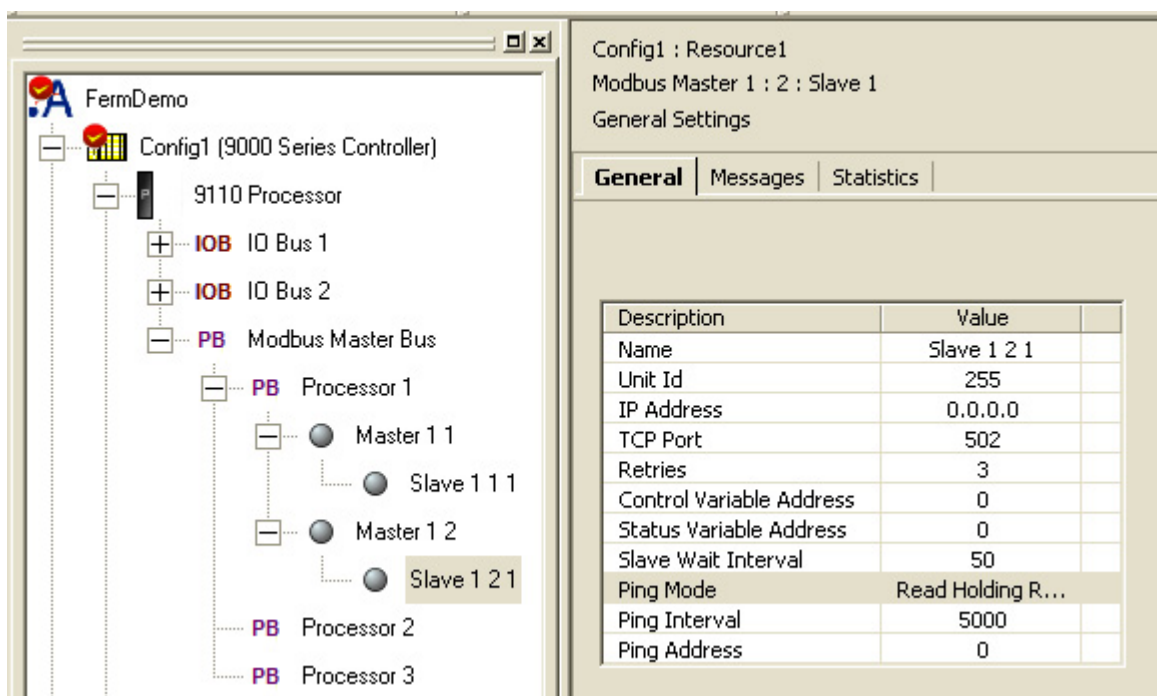


### Read Holding Register Ping Mode Setting

1. Select the **Equipment** tab on the Project tree and then a MODBUS Master object that has been set up to run the TCP protocol.
  - Check this by selecting the **General** tab and that the Protocol row is set to TCP.



2. Select the **Slave Link** associated with the Master object.
  - The editor for the Slave link opens.
3. Select the **General** tab.

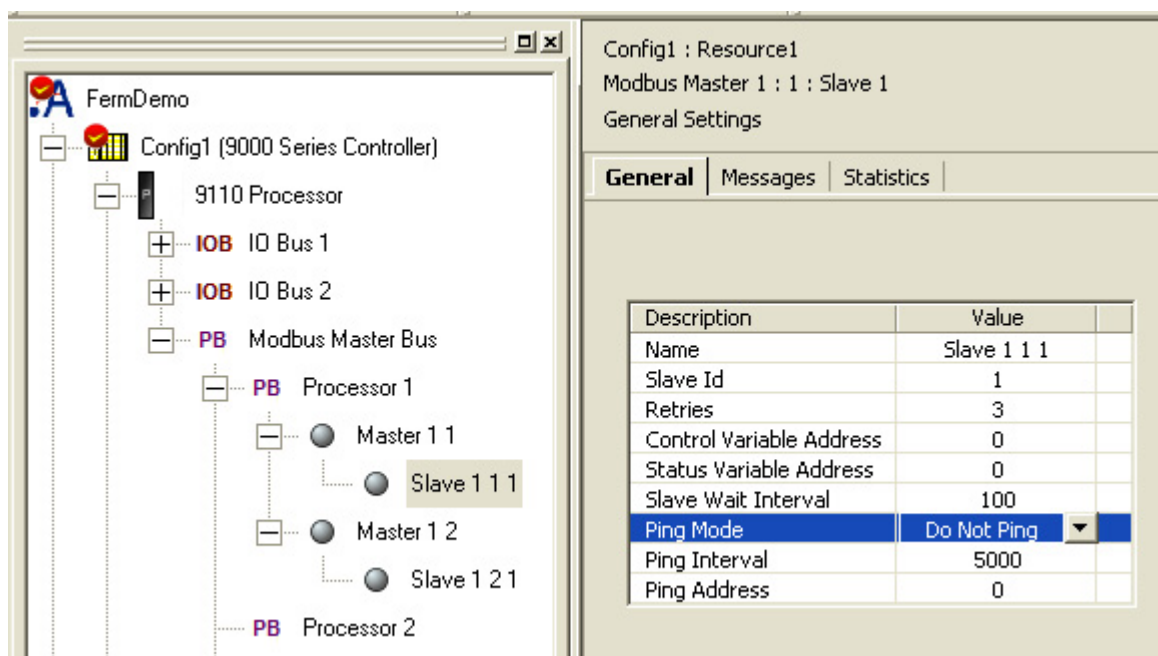


4. Check that the Ping Mode is set to Read Holding Register.

5. Ensure that a Ping Address has been entered.
6. Check that the Ping Interval is suitable for your application.

### Do Not Ping Mode Setting

1. Select the **Equipment** tab on the Project tree and then a MODBUS Master object that has been set up to run the TCP protocol.
  - Check this by selecting the **General** tab and that the Protocol row is set to TCP.
2. Select a **Slave Link** associated with a Master object set to the TCP protocol.



- The editor Slave link opens.
3. Check that the Ping Mode has been set to Do Not Ping.

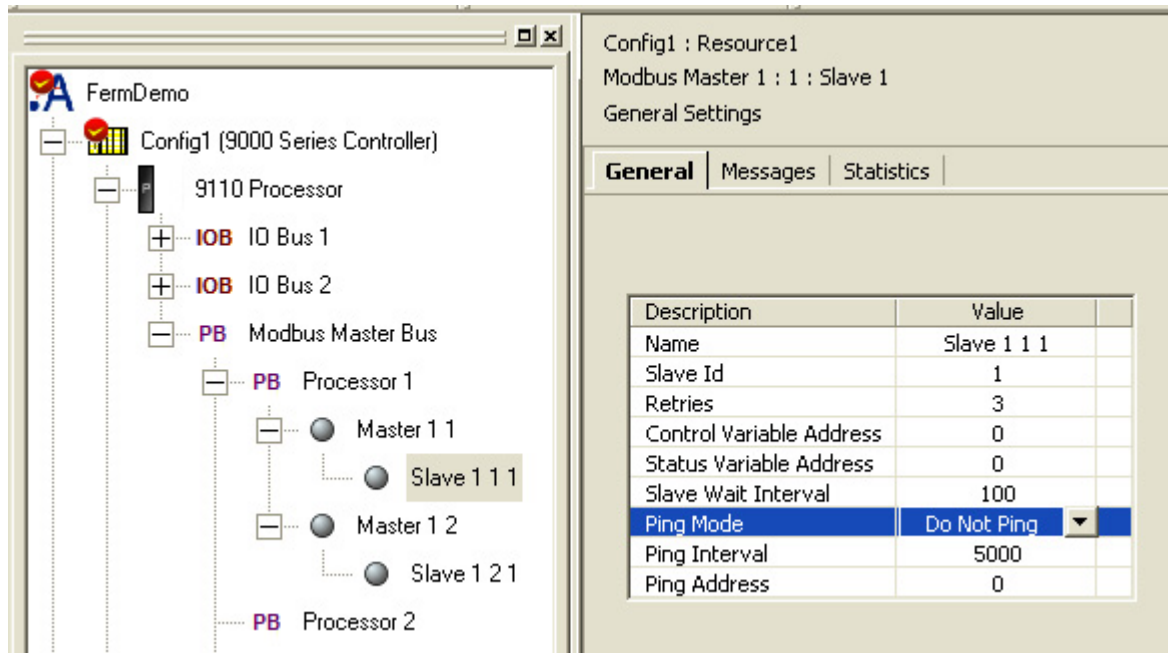
### Restart a Link When No Ping Mode is Set

When a Slave module fails, the Master stops sending messages to it except for the ping message (if one is configured). If the failed Slave module later becomes "Healthy" the Master will automatically detect this change because the module will respond to the ping messages, the Master will then start sending messages to it again.

However, when a Slave module configured with Do Not Ping fails and later becomes "Healthy" the Master cannot detect this change because it is not sending any ping messages to it. To resolve this situation the Slave module must be restarted using either the Slave or the Master control register. This can be done by the application or by using the AADvance Workbench in debug mode.

### Restart a Slave with Ping disabled

1. Configure the Slave with a Control Variable Address.
  - This is the address of the MODBUS holding register for the Slave link control variable.



2. Set the Slave control register variable to a 0 (off).
3. Set the control register variable to 2 (active).
  - The Master will restart communication with the Slave.

### Restart All Active Slaves Associated with a Master

1. Select the **Master Object**.
2. Set a Control Variable Address
  - This is the address of the MODBUS holding register for the control variable.
3. Set the **Master Control Variable** to 0 (off)
4. Set the **Master Control Variable** to 2 (active).

## Configure Statistics for a MODBUS Master Object

MODBUS Master statistics are available to the application. Set the **Statistics parameters** on the Master Object parameter screen.

Each MODBUS message has a series of parameters as detailed in the table.

Table 29 - MODBUS Slave Link Message Parameters

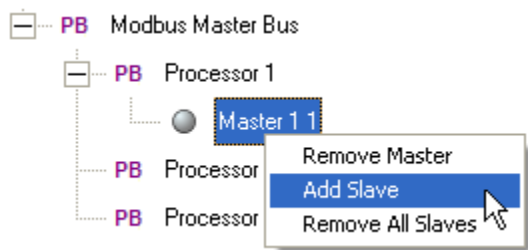
Description	Value(s)	Default	Remarks
Reporting Mode	Disabled; Last Rate; Maximum Rate; Average Rate	Disabled	Sets the application statistic to report
Data Variable Address	1 to 65,536, or 0 to disable	0	Data variable is reported in hundredths of a second
Reset Variable Address	1 to 65,536, or 0 to disable	0	Write any non-zero value to the reset variable to reset the data variable to zero

## Create Links to MODBUS Slaves

A Slave link represents the connection between a MODBUS Master object and a server. The AADvance Workbench represents each Slave link by a Slave object, one object for each server.

**NOTE** If you have configured the Master object for MODBUS TCP (with Ethernet communications), create only one Slave link for the Master object.

To create a Slave link, do the following:



1. Select the **Equipment** tab on the Project tree view and locate the MODBUS Master object to which you will add the Slave link.
2. Right-click on the **MODBUS Master object** and select **Add Slave**.
  - A Slave link object is inserted below the MODBUS Master object.
  - A default name is allocated to the object; the name includes the number of the processor module and the number of the Master, with a serial number for the Slave object. You can change the name when you configure the object.
3. Create additional Slave links as needed, one for each MODBUS Slave device.

**NOTE** If you create multiple Slave links for a MODBUS RTU Master and then choose to reconfigure the Master for MODBUS TCP, you must keep one Slave link and delete all of the others. Right-click on each **unwanted Slave link** in turn and select **Remove Slave**.

## MODBUS Slave Link Identification and Control Settings

Each MODBUS Slave link object has a set of identification and control settings as detailed in the tables.

**NOTE** The AADvance Workbench groups these settings together under the label 'General'.

**Table 30 - MODBUS RTU Slave Link Identification and Control Settings**

Description	Value(s)	Default	Remarks
Slave Id	1 to 247	1	You have to allocate a unique Slave Id for each Slave configured to a particular Master
Retries	0 to 10	3	Used in conjunction with the Master message timeout value
Control Variable Address	1 to 65,536, or 0 to disable	0	The address of a control register (†) 0 = inactive 1 = standby 2 = active
Status Variable Address	1 to 65,536, or 0 to disable	0	The address of a status register (†) 0 = healthy 1 = initializing 2 = communications failure 3 = error
Slave Wait Interval	0 to 65,535 ms	100 ms	
Ping Mode	Do Not Ping; Function Code 08; Read Holding Register	Function Code 08	
Ping Interval	0 to 65,535 ms	5,000 ms	
Ping Address	1 to 65,536, or 0 to disable	0	

**NOTE** (†) The 'remarks' here summarize the values allowed for the contents of the register. The Control variable may be written to by the application to switch the state of the Slave. The Status variable may be read by the application to monitor the Slave. These variables are described in MODBUS Slave Link Control and Status Registers.

**Table 31 - MODBUS TCP Slave Link Identification and Control Settings**

Description	Value(s)	Default	Remarks
Unit Id	0 to 255	255	
IP Address	0.0.0.0 to 255.255.255.255	0.0.0.0	
TCP Port	1 to 65,535	502	TCP port 502 is suggested for MODBUS TCP devices (MODBUS specification version 1.1b)
Retries	0 to 10	3	

Description	Value(s)	Default	Remarks
Control Variable Address	1 to 65,536, or 0 to disable	0	The address of a control register (†) 0 = inactive 1 = standby 2 = active
Status Variable Address	1 to 65,536, or 0 to disable	0	The address of a status register (†) 0 = healthy 1 = initializing 2 = communications failure 3 = error
Slave Wait Interval	0 to 65,535 ms	50 ms	
Ping Mode	Do Not Ping; Read Holding Register	Do Not Ping	
Ping Interval	0 to 65,535 ms	5,000 ms	
Ping Address	1 to 65,536, or 0 to disable	0	

**NOTE** (†) The 'remarks' here summarize the values allowed for the contents of the register.

Configure a MODBUS Slave Link for MODBUS RTU

**IMPORTANT** You have to create a MODBUS Slave link object below a MODBUS Master object configured for MODBUS RTU before you can carry out this task.

You have to configure each MODBUS RTU Slave link to suit its individual Slave device. Do the following:

FermDemo

Config1 (9000 Series Controller)

9110 Processor

IOB IO Bus 1

IOB IO Bus 2

PB Modbus Master Bus

PB Processor 1

Master 1 1

Slave 1 1 1

Master 1 2

Slave 1 2 1

PB Processor 2

PB Processor 3

Config1 : Resource1

Modbus Master 1 : 1 : Slave 1

General Settings

General

Messages

Statistics

Description	Value
Name	Slave 1 1 1
Slave Id	1
Retries	3
Control Variable Address	0
Status Variable Address	0
Slave Wait Interval	100
Ping Mode	Function Code 08
Ping Interval	5000
Ping Address	0

210

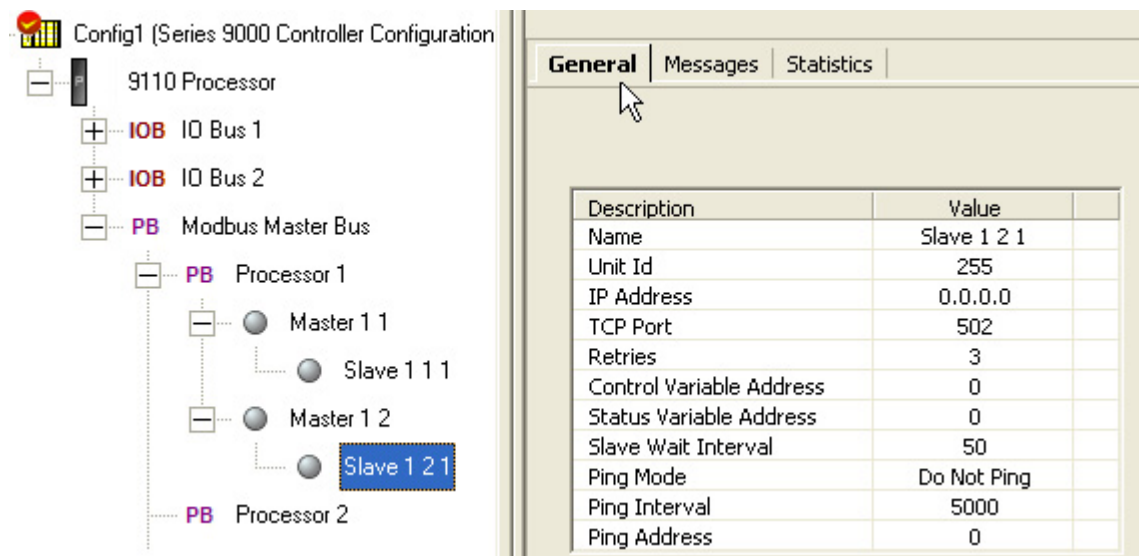
Rockwell Automation Publication ICSTT-RM405H-EN-P - April 2018

1. Select the **Equipment** tab on the Project tree view and then select the **Slave Link** you want to configure.
  - The editor for the Slave link opens.
  - The editor has three tabs; the tabs are named General, Messages and Statistics.
2. Select the **General** tab.
3. Enter a suitable Name for the Slave link, or accept the default. You can use any combination of alphanumeric characters and punctuation, including spaces.
4. Set the Slave Id to a value that is unique amongst the Slaves attached to the same Master.
5. The Retries field is used with the Message Timeout value set for the MODBUS Master. Set the Retries field to the number of times the MODBUS Master should try to retransmit a message to the Slave if it does not receive a response.
6. If you intend to use the control variable, set the Control Variable Address to the MODBUS address of the holding register; otherwise accept the default 0 to disable.
7. Similarly set the Status Variable Address.
8. Specify the Slave Wait Interval (in milliseconds), or accept the default.
9. Set the Ping Mode and Ping Interval. The interval is specified in milliseconds. If you are using the Read Holding Register mode, specify the Ping Address as well; otherwise accept the default 0.
10. Click **Apply**.

## Configure a MODBUS Slave Link for MODBUS TCP

**IMPORTANT** You have to create a single MODBUS Slave link object below a MODBUS Master object configured for MODBUS TCP before you can do this task.

You have to configure the MODBUS TCP Slave link to suit the Slave device. Do the following:



1. Select the **Equipment** tab on the Project tree view and then select the **Slave Link**.
  - The editor for the Slave link opens.
  - The editor has three tabs; the tabs are named General, Messages and Statistics.
2. Select the **General** tab.
3. Enter a suitable Name for the Slave link, or accept the default. You can use any combination of alphanumeric characters and punctuation, including spaces.
4. Accept the default Unit Id (the Unit Id is a part of the MODBUS TCP header and is not normally used).
5. Set the IP Address to the network address used by the Slave device.
6. Set the TCP Port to the TCP port number on which the Slave device is offering services.
7. The Retries field is used with the message timeout value set for the MODBUS Master. Set the Retries field to the number of times the MODBUS Master should try to retransmit a message to the Slave if it does not receive a response.
8. If you intend to use the control variable, set the Control Variable Address to the MODBUS address of the holding register; otherwise accept the default 0 to disable.



9. Set the Status Variable Address.
10. Specify the Slave Wait Interval (in milliseconds), or accept the default.
11. Set the Ping Mode and Ping Interval. The interval is specified in milliseconds. If you are using ping (which uses the Read Holding Register mode), specify the Ping Address as well; otherwise accept the default 0.
12. Click **Apply**.

## Choosing Names for MODBUS Objects

The MODBUS object Name identifies a MODBUS Master object or Slave link within the AADvance Workbench, and is included in printed reports; but it is not used by the application. We recommend that you use a different name for each object. Alternatively, accept the default name.

## MODBUS Slave ID

Each MODBUS RTU Slave link object has a Unit Id. This is the MODBUS communication ID — a unique identifier for the MODBUS Slave device. It must be unique for every Slave link configured for a particular Master.

## MODBUS Slave Wait Interval

The Slave Wait Interval is specified individually for each MODBUS Slave; it is the minimum amount of time the Master will wait from the start of one message to a Slave to the beginning of the next message to the same Slave.

## MODBUS Ping Mode, Interval and Address

The ping mode is specified individually for each Slave; it defines the action the MODBUS Master should take if the Slave fails to respond after the specified number of retries.

The AADvance controller supports a Function Code 08 ping mode for MODBUS RTU Slaves; choose this if the Slave supports it because it is a short message. The Read Holding Register mode is an alternative; choose this if the Slave does not support Function Code 08. The MODBUS Master will try to read a single register. Read Holding Register is the only method available to ping a MODBUS TCP Slave. You can also set a Do Not Ping setting for both protocols.

The ping address is for the Read Holding Register mode and is also specified individually for each Slave; it is the MODBUS address of the holding register. It is not used by the Function Code 08 mode.

## MODBUS Slave Commands

The AADvance controller supports a subset of MODBUS standard command types.

### Serial Port

MODBUS Slave serial port commands are available on only the AADvance controller serial ports.

#### Diagnostics

- Function: Diag
- Function Code = 8
- Address Offset: 0
- Data Type: Boolean
- Fixed Length: 8
- Varbyte length: 0

Purpose: Provides a series of tests for checking the communications system between a client (Master) and a server (Slave).

### Ethernet and Serial Port

MODBUS Slave Ethernet and serial port commands are available on the AADvance controller Ethernet ports and serial ports.

#### Read Coils

- Function: read
- Function Code = 1
- Address Offset: COIL\_STATUS\_ADDRESS
- Data Type: Boolean
- Fixed Length: 8
- Varbyte length: 0

Purpose: Reads from 1 to 2,000 contiguous status of coils in a remote device.

#### Read Discrete Inputs

- Function: read
- Function Code = 2
- Address Offset: INPUT\_STATUS\_ADDR
- Data Type: Boolean
- Fixed Length: 8
- Varbyte length: 0

Purpose: Reads from 1 to 2,000 contiguous status of discrete inputs in a remote device.

**Read Holding Registers**

- Function: read
- Function Code = 3
- Address Offset: HOLDING\_REG\_ADDR
- Data Type: analogue
- Fixed Length: 8
- Varbyte length: 0

Purpose: Reads the contents of a contiguous block of holding registers in a remote device.

**Read Input Registers**

- Function: read
- Function Code = 4
- Address Offset: INPUT\_REG\_ADDR
- Data Type: Boolean
- Fixed Length: 8
- Varbyte length: 0

Purpose: Reads from 1 to 125 contiguous input registers in a remote device.

**Write Single Coil**

- Function: write
- Function Code = 5
- Address Offset: COIL\_STATUS\_ADDR
- Data Type: Boolean
- Fixed Length: 8
- Varbyte length: 0

Purpose: Writes a single output to either ON or OFF in a remote device.

**Write Single Register**

- Function: write
- Function Code = 6
- Address Offset: HOLDING\_REG\_ADDR
- Data Type: analogue
- Fixed Length: 8
- Varbyte length: 0

Purpose: Writes a single holding register in a remote device.

**Write Multiple Coils**

- Function: write
- Function Code = 15
- Address Offset: COIL\_STATUS\_ADDR

- Data Type: Boolean
- Fixed Length: 9
- Varbyte length: 6

Purpose: Forces each coil in a sequence of coils to either ON or OFF in a remote device.

**Write Multiple Registers**

- Function: write
- Function Code = 16
- Address Offset: HOLDING\_REG\_ADDR
- Data Type: analogue
- Fixed Length: 9
- Varbyte length: 6

Purpose: Writes a block of contiguous registers (1 to 123) in a remote device.

**MODBUS Slave Link Control and Status Registers**

Every MODBUS Slave link object has a pair of registers (holding registers), which allow the application to control the link, and to retrieve status information. The registers are a 'control register' and a 'status register'. Each register is located at a unique address — the Control Variable Address and the Status Variable Address.

The use of the registers is optional. If you want to use a register, you have to declare a variable for it in the application Dictionary; and also specify the address in the holding register map.

**MODBUS Slave Link Control Register**

The application can use the control register to control a specific Slave link. If you do not specify a control variable address, then the link is enabled automatically when the controller is started. The variable in the application Dictionary has the functions listed in the table.

**Table 32 - MODBUS Slave Control Register**

Value	Meaning	Description
0	Inactive	Disables the Slave link Slave is neither polled or pinged
1	Standby	Forces the Slave link to operate in its standby mode MODBUS Master continues to ping the Slave to make sure that communications are possible
2	Active	Forces the Slave link to operate in its active mode Slave is being polled for data

**TIP** If the application sets the register to any other value, the Slave link is disabled.

## MODBUS Slave Link Status Register

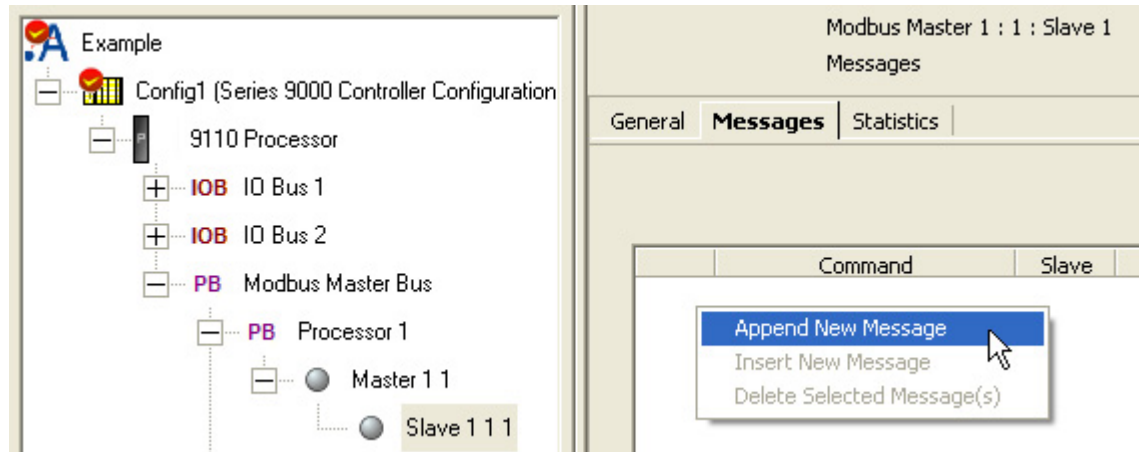
The status register is an unsigned integer value (UINT) that is returned by the MODBUS driver to allow the application to monitor and act on faults. If you do not specify a status register address, then the Slave link does not report its status information to the application. The variable in the application Dictionary has the values listed in the table.

**Table 33 - MODBUS Slave Status Register**

Value	Meaning	Description
0	Healthy	Slave is operating normally Link is active, no errors reported
1	Initializing/ unavailable	The Slave link is initializing This value is also reported if the Slave device has failed to respond to a ping on a link declared as standby; if the MODBUS Master is in error; or if the MODBUS Master is set to inactive
2	Communications failure	Slave is being polled for data and is not responding
3	Error/exception	The last message to the Slave resulted in an exception response

## Add Messages for a MODBUS Slave

The Slave link object uses a list of messages to hold the commands to send to its Slave device. To create the list, do the following:



1. Select the **Equipment** tab on the Project tree view and then select the **Slave Link** you want to configure.
  - The editor for the Slave link opens.
2. Select the **Messages** tab.
  - The editor shows an empty list area, which will hold your sequence of messages.
3. Right-click **anywhere within the list area** and select **Append New Message**.
  - The editor creates a message.

- The default command type is Read Coils.
4. Set the command type, then click the empty area below the list to update the display.
    - The Master Mimic field provides a drop-down list of values for the MODBUS address space on the associated 9110 processor module.
  5. Set the Slave field to the MODBUS address of the coil or holding register.
  6. If you chose Write Coils or Write Holding Registers, set the Master Mimic field to match the MODBUS address space on the **9110** processor module. Dictionary variables may be specified on the MODBUS tab as different MODBUS types. These choose separate variable address spaces. The Master Mimic field defines the MODBUS address space from which the data will be sent.
  7. If you choose a command other than Write Coils or Write Holding Registers, there is no choice of Master Mimic settings; accept the appropriate default.
  8. Set the Master field to the MODBUS address of the Master within the mimic.
  9. Set the Count field to the number of coils or registers to be copied across. The range is defined by the MODBUS protocol.
  10. The Control field defines the address of the coil, the input register or the holding register; it also controls whether the controller broadcasts the message. Set the Control field to a valid address; use 0 (zero) to disable the message.
  11. The Comment field is a free text field for user notes; it is not used by the software. If desired, enter a comment.
  12. Click **Apply**.

Proceed to add and configure additional messages as needed. Select an **existing message** and then right-click at the **left-hand end of the row**; select **Append New Message** to add a message below the current message or **Insert New Message** to add a message above.

**TIP** You can delete one or more messages together. Select the **unwanted messages** from the list; use Shift and Ctrl as modifiers. Right-click **any selected message** and select **Delete Selected Message(s)**.

## MODBUS Slave Link Message Parameters

Each MODBUS Slave link message has a set of parameters as detailed in the table.

**Table 34 - MODBUS Slave Link Message Parameters**

Description	Value(s)	Default	Remarks
Command	Read Inputs; Read Coils; Write Coils; Read Input Registers; Read Holding Registers; Write Holding Registers	Read Coils	Command type specifies the direction of the message
Slave	1 to 65,536	1	Range of values usable depends on the Slave device
Master Mimic	Depends on command see table below	Various	
Master	1 to 65,536	1	
Count	1 to 2,000 for a Write Coils message; 1 to 125 for a Write Holding Registers message		
Control	1 to 65,536, or 0 to disable		The address of a control register (coil) (†) 0 = enabled 1 = disabled

(†) The 'remarks' here summarize the values allowed for the contents of the register.

**Table 35 - Master Mimic Values for MODBUS Slave Link Messages**

Command	Master Mimic Value(s)	Default	Remarks
Read Inputs	Coils	Coils	
Read Coils	Coils	Coils	
Write Coils	Coils; Inputs	Coils	
Read Input Registers	Holding Registers	Holding Registers	
Read Holding Registers	Holding Registers	Holding Registers	
Write Holding Registers	Input Registers; Holding Registers	Holding Registers	

## Controlling a MODBUS Message

Every MODBUS message that originates at a MODBUS Master (broadcast message or Slave link message) has provision for a control (a coil), which allows the application to control whether or not the message should be sent. The coil is located at the address that is specified by the message parameter named 'Control'. You can allocate the same address (and coil) to multiple messages.

The use of a coil is optional. If you want to use a coil, you have to declare a variable for it in the application Dictionary; and also specify the address in the memory map of coil data.

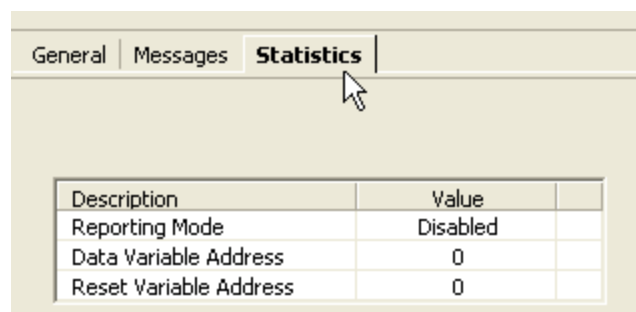
If you do not specify an address then the message is always enabled. The variable in the application Dictionary has the functions listed in the table.

**Table 36 - MODBUS Message Control Register**

Value	Meaning	Description
0	Enabled	The message is enabled and, if the associated MODBUS object is enabled, the message is sent
1	Disabled	The message is disabled and is never sent

## Configure Statistics for a MODBUS Slave Link

Slave link statistics are available to the application. To use the statistics, do the following:



1. Select the **Equipment** tab on the Project tree view and then select the **Slave Link** you want to configure.
  - The editor for the Slave link opens.
2. Select the **Statistics** tab.
  - The editor shows three fields.
  - The default values disable the statistics reporting.
3. Set the **Reporting Mode**.
4. Set the **Data Variable Address** to the MODBUS address of the relevant holding register.
5. Similarly set the **Reset Variable Address**.
6. Click **Apply**.

## MODBUS Statistics Parameters

Each MODBUS message has a series of parameters as detailed in the table.

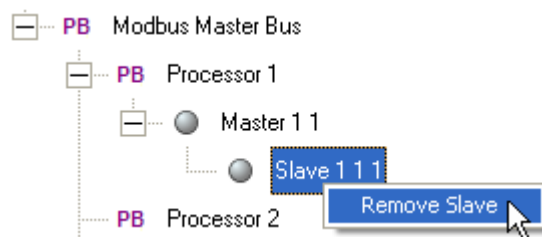


**Table 37 - MODBUS Slave Link Message Parameters**

Description	Value(s)	Default	Remarks
Reporting Mode	Disabled; Last Rate; Maximum Rate; Average Rate	Disabled	Sets the application statistic to report
Data Variable Address	1 to 65,536, or 0 to disable	0	Data variable is reported in hundredths of a second
Reset Variable Address	1 to 65,536, or 0 to disable	0	Write any non-zero value to the reset variable to reset the data variable to zero

## Remove a Slave Link

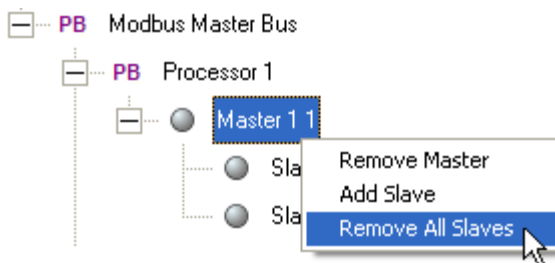
You can remove an unwanted Slave link. Do the following:



1. Select the **Equipment** tab on the Project tree view and locate the unwanted MODBUS Slave link.
2. Right-click on the **Slave link** and select **Remove Slave**.
  - The Slave link is deleted.

## Remove all Slave Links

You can remove all the Slave links associated with a single MODBUS Master object. Do the following:

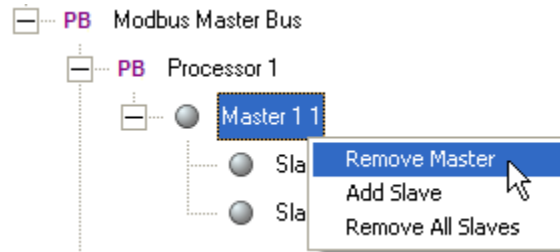


1. Select the **Equipment** tab on the Project tree view and locate the MODBUS Master object.
2. Right-click on the **MODBUS Master object** and select **Remove All Slaves**.
  - All the Slave links associated with the Master are removed.

## Remove a MODBUS Master Object

You can remove an unwanted MODBUS Master object.

To remove the object do the following:



1. Select the **Equipment** tab on the Project tree view and locate the unwanted MODBUS Master object.
2. Right-click on the **MODBUS Master object** and select **Remove Master**.
  - The MODBUS Master object is deleted.
  - All Slave link objects associated with the Master are also deleted.

---

**NOTE** You can remove all the MODBUS Master objects associated with a particular 9110 processor module. Right-click on the **processor item** and select **Remove All Masters**.

---

## SNCP

SNCP (Safety Network Control Protocol) is a SIL 3 certified protocol which supplies a safety layer for the Ethernet network making the network connection a "Black Channel". Data is exchanged by creating a relationship between variables in different AADvance controllers; this is called "Binding Variables". This chapter describes the process for setting up SNCP variable bindings.

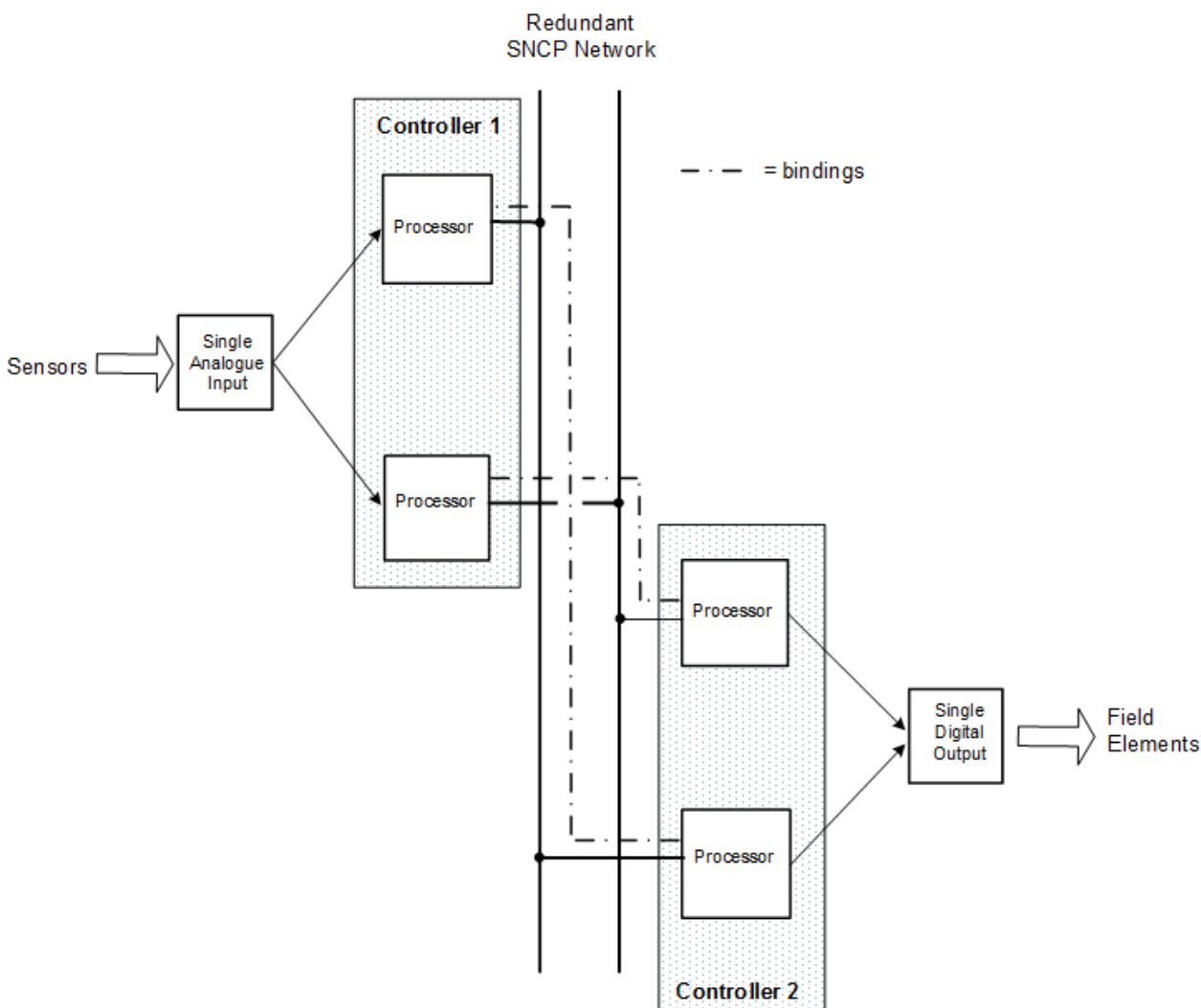
### **Bindings and the SNCP Network**

Bindings are based on a producer/consumer model. The controller consuming the data establishes a binding link with the controller producing the data and manages all of the sending and receiving of data. It schedules the sending and receiving of data, sending the diagnostic data, managing the safety response if faults occur and managing the communications redundancy. A SNCP network is illustrated in the diagram.

First there must be a physical connection between the two controllers. The design of the Ethernet network and the equipment used does not impact the SIL rating of the communications interface, but the design of the network does change the reliability of the network and does impact the spurious trip rate. SNCP Network data can be combined on a common network resulting in safety and non-safety data sharing a common physical network. This does not compromise the SIL rating of the network but again does introduce failure modes and possibly security risks which can increase the spurious trip rate, therefore, careful consideration must be given to the network topology during the applications specification and design phase.

SNCP Networks can be configured as Simplex (Fail Safe) or Redundant (Fault tolerant), the network configuration is dependent on the applications safety and availability requirements. The giving and receiving of data occurs

independently from the physical network configuration as the connection between the controllers is treated as a logical network.



## SNCP Networks

You can set up a single SNCP network to communicate with the target. When two controllers are set up with their IP Addresses on the same subnet, they are also on the same physical network. This single logical network can also be used to configure variable bindings over redundant physical networks.

### Setting up Single Networks

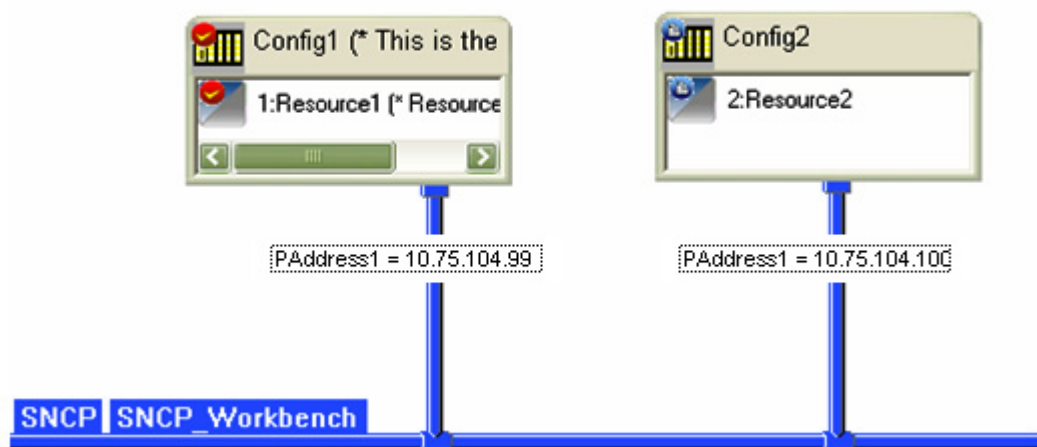
When you create a project it has a default single network configuration.

1. Create a project.
  - This configuration represents a single logical network.
  - The configured IP address is the IP address used by the Workbench to communicate with the target.

The Workbench will always use the first address that you configure for the SNCP\_Workbench network for communications irrespective of any other network instances that have been configured.

**TIP** It is recommended that you declare consumer variables in the Dictionary with the "read-only" attribute to avoid conflicts between binding and the execution of POU's.

2. Set up a second controller on the same logical network.
3. Set up the IP Addresses as follows:
  - Config 1 IP Address 1 = 10.75.104.99 and Config 2 IP Address 1 = 10.75.104.100.



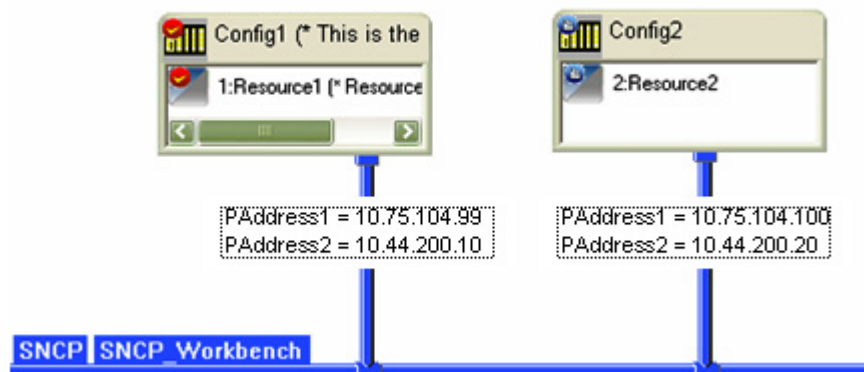
- This figure shows two controllers configured on the same logical network and as they are on the same subnet they will appear on the same physical network (SNCP\_Workbench).
- All bindings will be sent over the SNCP\_Workbench network.

### Redundant Single Networks

The single logical network can be used to configure variable bindings over redundant physical networks. This is done by configuring additional IP addresses for the redundant networks. The IP Addresses shown are for information purposes only.

1. Set up two controllers on a single logical network.
2. Configure additional IP Addresses for each resource on the same subnets as follows:
  - Config 1 set IP Address 1 = 10.75.104.99
  - Config 1 set IP Address 2 = 10.44.200.10
  - Config 2 set IP Address 1 = 10.75.104.100

- Config 2 set IP Address 2 = 10.44.200.20



- When creating bindings for this network the SNCP\_Workbench will be the only network available for selection.
- The SNCP\_Workbench binding driver sends each binding data message over the 10.75.104.xxx and the 10.44.200.xxx networks. Loss of one of these networks will not result in a loss of binding data.

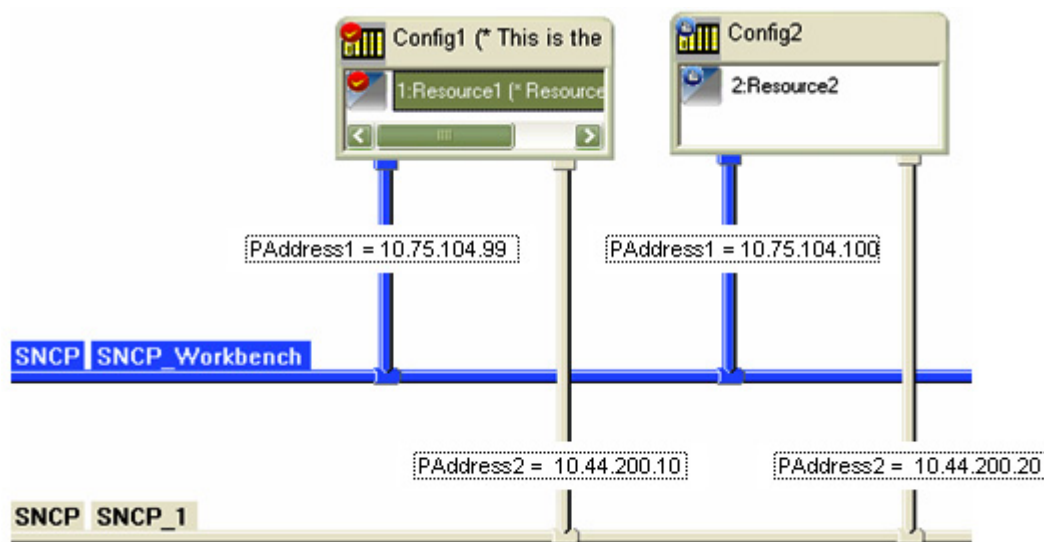
## Set Up Multiple SNCP Networks

### Multiple Logical Networks

The Workbench also supports multiple redundant networks. This type of network configuration provides a dedicated logical network (which can be physically redundant) to bindings only, to prevent Safety Data from being

combined with non Safety Data. The SNCP\_Workbench network is the one used by the Workbench to communicate with the target while additional networks, such as the SNCP\_1 network shown, are used specifically for transporting variables over bindings.

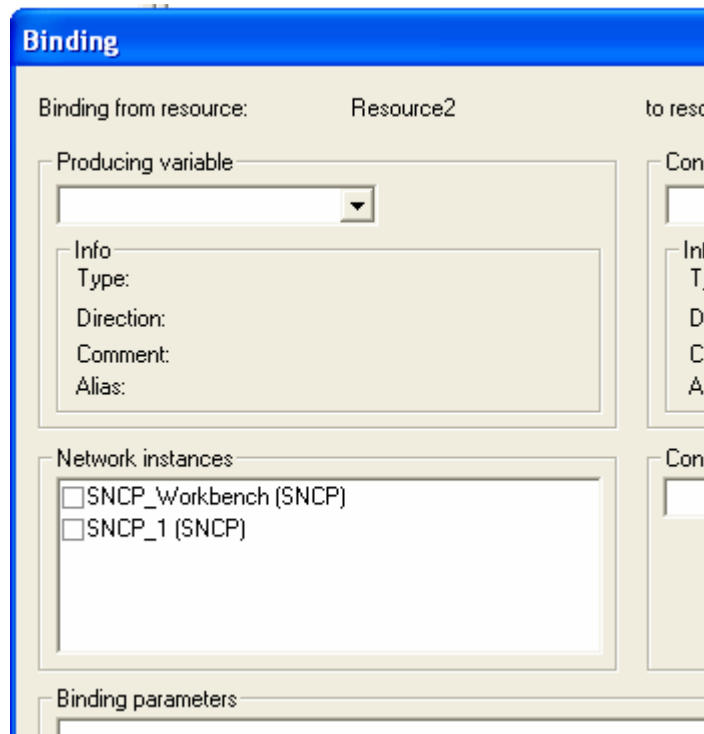
1. Set up a second SNCP network and connect the resources to that network.
2. Enter the IP Addresses using the same subnet for each controller on the same network:
  - SNCP\_Workbench subnet:  
Config 1 set IP Address 1 = 10.75.104.99  
Config 2 set IP Address 1 = 10.75.104.100.
  - SNCP\_1 subnet:  
Config 1 set IP Address 2 = 10.44.200.10  
Config 2 set IP Address 2 = 10.44.200.20.



- This configuration increases the options for the configuration of bindings, two networks are now available when a variable binding is created. The SNCP binding driver will only send the bound variable over the selected network(s). This means if the variable is bound over the SNCP\_1 network only, and that network fails, the data for that variable will not be received by the consumer. However, you can select both networks to configure a redundant dual network.

## Configure a Dual Network

1. To configure a redundant dual network select both networks when creating a bound variable.

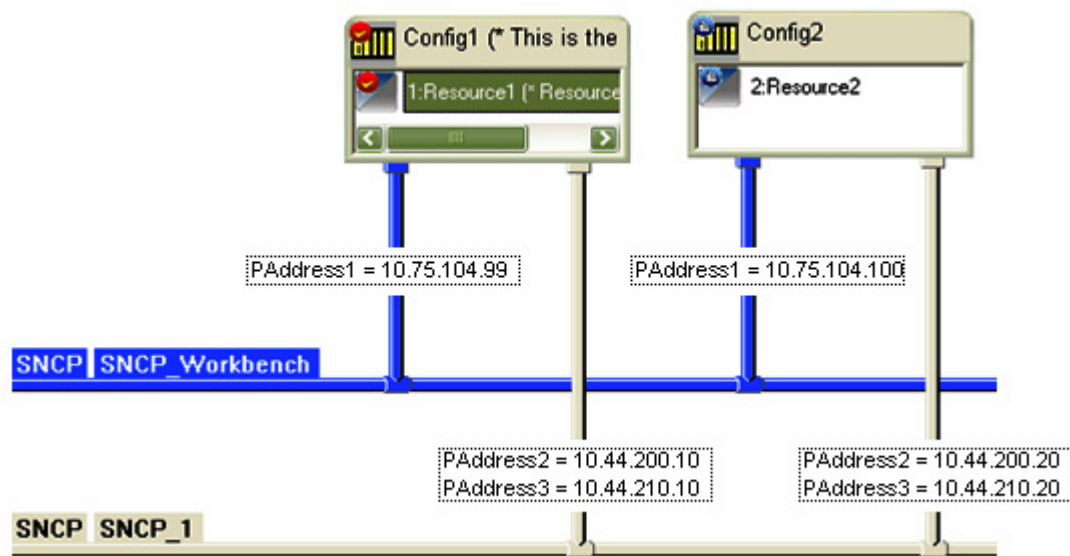


- The variable will be sent over both physical networks, so if one network fails the consumer will still receive the variable.

## Redundant Physical and Logical Networks

1. Set up two networks connected to resources 1 and 2.
2. Configure the IP Addresses as follows:
  - SNCP\_Workbench subnet:  
Config 1 set IP Address 1 = 10.75.104.99  
Config 2 set IP Address 1 = 10.75.104.100
  - SNCP\_1 subnet:  
Config 1 set IP Address 2 = 10.44.200.10  
Config 1 set IP Address 3 = 10.44.210.10;  
Config 2 set IP Address 2 = 10.44.200.20  
Config 2 set IP Address 3 = 10.44.210.20





- This configuration provides two logical networks and three physical networks.
3. Create a bound variable and select **SNCP\_1** for the binding.
    - A bound variable configured to use **SNCP\_1** is sent over both physical networks and the loss of one network will not affect the consumer receiving bound variable over this network.
  4. Create a bound variable and select both networks for the variable.
    - A bound variable configured for both logical networks will be sent over all three physical networks and the loss of two logical or two physical networks can be tolerated and the consumer will still receive the variable.

## SNCP Link Properties

The SNCP key time out values are the MaxAge and BindRespTimeout. These settings govern the number of binding requests a consumer will send without getting a response before considering the bindings link as failed and disconnecting.

### ConnectTimeout

This timeout serves two purposes:

- It is the time the resource waits to connect to a producer when a consumer resource starts up.

- It is used to timeout a producer's response once a consumer has sent a "connect" request to a producer. If the producer does not respond within the timeout, the consumer sends another "connect" request. The consumer continues to send "connect" request at intervals given by this timeout until a connection is established.

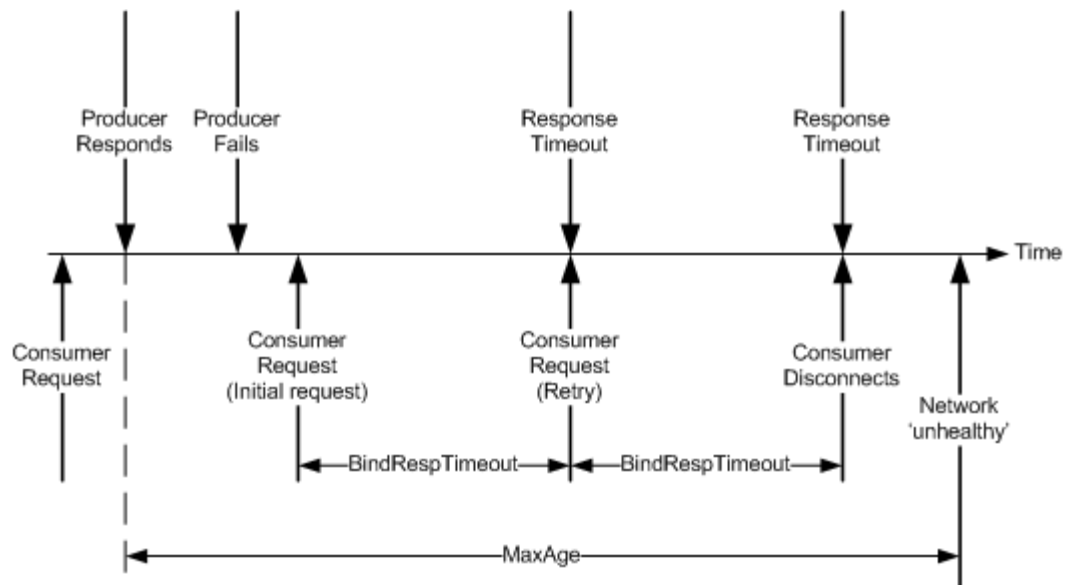
### BindRespTimeout

This timeout is used in a consumer to timeout a binding data response from a producer. Once a consumer has established a binding connection with a producer, it sends a request for binding data; this timeout value timeouts out the producer response.

If no response is received, either another request (retry) is sent, or the consumer disconnects from the producer. The number of retries that are attempted before a consumer disconnects depends on the value of this parameter and MaxAge. The consumer will continue to send requests until MaxAge expires.

In general the number of requests sent before disconnecting is given by:  
 $\text{MaxAge} / \text{BindRespTimeout}$

The following diagram illustrates (not to scale) the timeout behavior when the MaxAge is configured as 2500 ms and BindRespTimeout is 1000 ms. The total number of request sent before disconnecting is 2.



### MaxAge

This timeout is used in a consumer resource, and serves two purposes. Firstly, it is the time during which a consumer must receive a valid binding message from the producer before the physical network is considered 'unhealthy'. Stale, corrupt, or out of sequence messages are not considered valid. Continued reception of such messages can be indicative of underlying network problems,

in which case this will be reflected by an 'unhealthy' network status (as reported by the KVB status function blocks).

Secondly this timeout value is also used to examine the age of binding data message received by the consumer. If a message contains data that is older than this value it is discarded. This can occur if the message is delayed because of problems with the network.

The number of requests is determined by MaxAge/BindRespTimeout. To increase the number of retries you must increase MaxAge or decrease BindRespTimeout.

### **BindReqTimeout**

This timeout is used by the producer. Once a binding connection has been established with a consumer, the producer uses this timeout value to timeout binding data requests from the consumer. If producer does not receive a request for binding data from the consumer within this timeout, the producer disconnects the consumer.

Subsequent requests for data from the consumer are ignored until the consumer establishes a new connection.

---

**NOTE** The Consumers Network bindings parameters (i.e. timeout values) are those located in the Producing Resource.

---

### **UpdateTimeout**

This timeout is used in the consumer and producer resources during an on-line update. During an on-line update all binding connections are closed. The SNCP binding driver then restarts with the potentially new binding configuration. This timeout value is the time in which the consumer must make its binding connections again.

During this time any configured binding error variables will continue to indicate 'healthy', even though there is no connection between the consumer and producer.

Failure to make the connection again within the timeout results in binding error variables becoming 'unhealthy'.

---

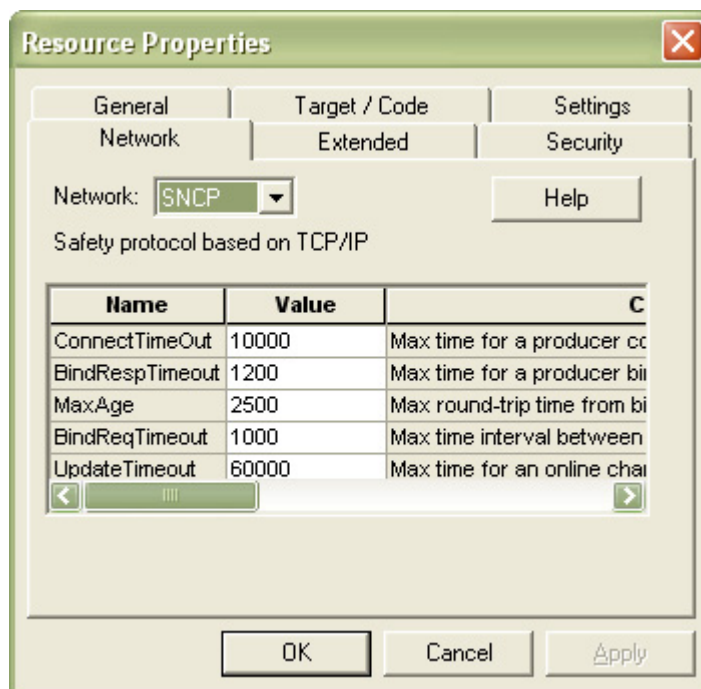
**IMPORTANT** From the Workbench perspective, it is important to do on-line updates for both the producer and consumer resources as quickly as possible, to prevent this timeout from occurring.

---

### **Setting SNCP KVB Driver Timeouts**

The SNCP KVB driver timeouts are configured via the Resource Properties dialog box.

1. Select the **Hardware Architecture View**.
2. Select a **Resource** then right-click and select **Properties**.
  - The Resource Properties dialog box opens.



3. Select the **Network** tab and enter the values; the default recommended values:
  - Connect TimeOut = 1000
  - BindResp Timeout = 1200
  - MaxAge = 2500
  - BindReqTimeout = 1000
  - Update Timeout = 60000

## Configure Bindings

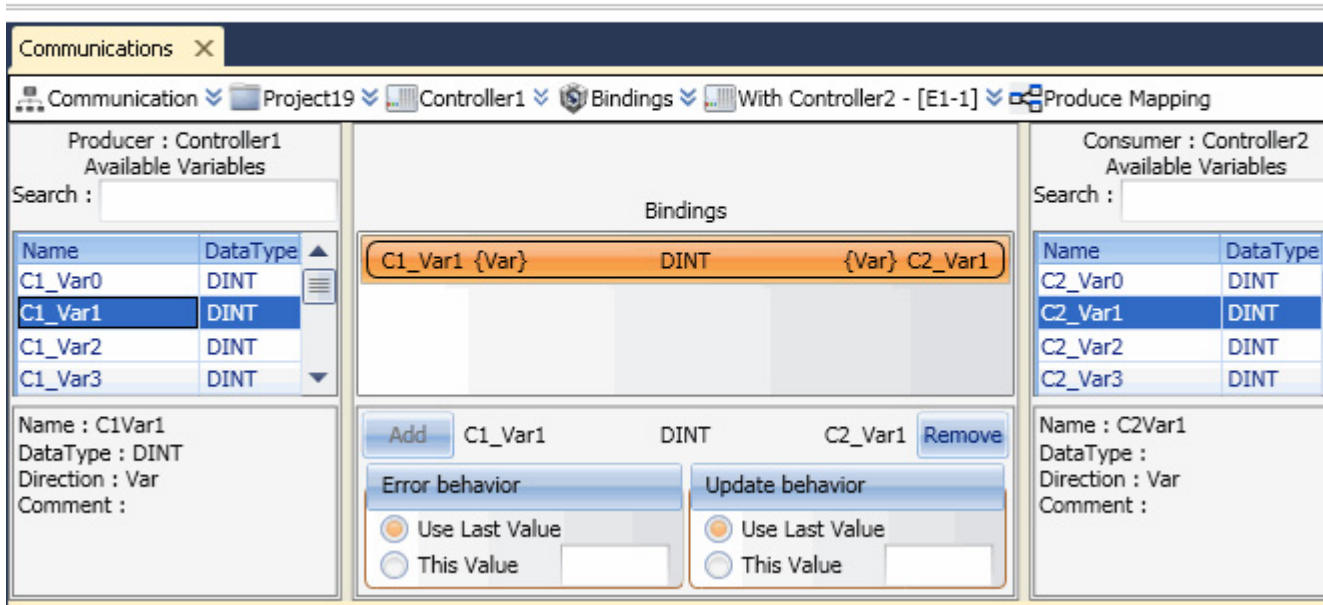
When creating a variable binding you can do it from the Producer perspective to the Consumer perspective or the other way round. The following procedure uses the Producer to Consumer direction:

Using the Controllers linked in the SNTP network set up procedure.

1. Right-click on the **With Config"n" - [En-En]** and select **Open** to display the Produce Mapping and Consume Mapping icons.

2. Select the **Produce Mapping** element. The mapping editor is displayed.

## Variable binding from controller produce mapping perspective



3. The Produce Mapping editor lists:  
the Producer variables: **Config "n" Available Variables** on the left, and,  
the linked Consumer: **Config "n" Available Variables** on the right.
4. Select a **Variable** from the **Producer** list. The details for the variables are displayed below the list and is shown next to the **Add** button.
5. Select a **Variable** from the **Consumer** List. The details are displayed below the list and is shown next to the **Remove** button. Verify that the variables are the correct type (same type).
6. Enter the **Error behavior** and the **Update behavior**. You can select the Last Value or enter a specific value or the variable when an error occurs.
7. Select the **Add** button and the bindings are moved to the top of the Bindings list.

## SNCP Binding Error Variables

To test values of binding error variables you must create the following words in the dictionary for your project:

Error Message	Code	Description
SNCP_KVB_ERR_BINDING_READY	0x00	Normal Steady State Data Exchange
SNCP_KVB_ERR_BINDING_IN_PROCESS	0x01	Producer and consumer are in the process of connecting
SNCP_KVB_ERR_UPD_IN_PROCESS	0x02	Producer and Consumer are in the process of re-connecting following an on-line-change update

Error Message	Code	Description
SNCP_KVB_ERR_UPD_TIMEOUT	0x03	Producer and consumer did not re-connect within "Update Timeout" milliseconds following an on-line change update.
SNCP_KVB_ERR_NO_PRODUCER	0x04	At the beginning of a connection or following "BindRespTimeout" milliseconds during data exchange.
SNCP_KVB_ERR_BAD_CRC	0x05	Producer and consumer binding tables CRC mismatch during connection
SNCP_KVB_ERR_IMPOSSIBLE_TO_BIND	0x06	Conversion of structures not supported on heterogeneous binding link
SNCP_KVB_ERR_IP_DENIED	0x07	Producer only error, indicates that a consumer with an unknown/unexpected IP Address is requesting a bindings link. The link is denied.
SNCP_KVB_ERR_BAD_PRODUCER_ID	0x08	Producer only error, indicates that the producer has received a bindings link request from a consumer and that request does not contain the expected producer identity. The link is denied.
SNCP_KVB_ERR_BAD_GROUP_ID	0x09	Producer only error, indicates that the producer has received a bindings link request from a consumer, and that request does not contain a known bound variable group identity. The link is denied.
SNCP_KVB_ERR_BAD_BUFF_SIZE	0x0A	Producer only error, indicates that either the producer or consumer does not have sufficient buffer space to translate the bound variables so that the link cannot be established.
SNCP_ERR_BAD_CONFIG	0x0B	Producer only error, indicates that the producer is unable to obtain the consumers IP address information from the network configuration file. The link is denied.

## Peer-to-Peer Network

AADvance provides the options to set up a Peer-to-Peer network. The Peer-to-Peer protocol enables you to communicate application data between up to 40 AADvance or Trusted systems for each peer network. Data can be transferred between individual systems or from one to many systems at the same time using multicast network communication.

---

**IMPORTANT** The current AADvance controller only supports two Multicast IP Addresses, one fixed address can be configured for each Ethernet port.

---

### Peer-to-Peer Features

A Peer-to-Peer network consists of one or more Ethernet networks connecting together a series of AADvance and/or Trusted controllers to enable application data to be passed between them. A Trusted controller with four communication interfaces has eight Ethernet ports and Peer-to-Peer network can use all eight physical Ethernet networks (referred to below as subnets) to provide redundant data paths via eight separate physical routes.

Both network interfaces of an AADvance processor module can be used for peer communications at the same time. A TMR controller with three processor modules provides a maximum of six physical connections that can be divided between different peer networks, or assigned to the same network as required.

Network subnets may be assigned to the processor Ethernet ports. Normally, subnets of the same redundant network would use different processor modules.

The information to be transferred over the peer network is defined within the application using input and output boards of the standard form. The boards configure data blocks of 16 or 128 Boolean points, 16 or 128 analogue points, and network status information.

### Peer-to-Peer Configuration Process

Configuring the AADvance controller to communicate with a Trusted Peer-to-Peer network requires the same parameters as a Trusted configuration. The configuration for AADvance is arranged in a tree in the I/O Configuration view. The recommended process for setting up a Peer-to-Peer network is as follows:

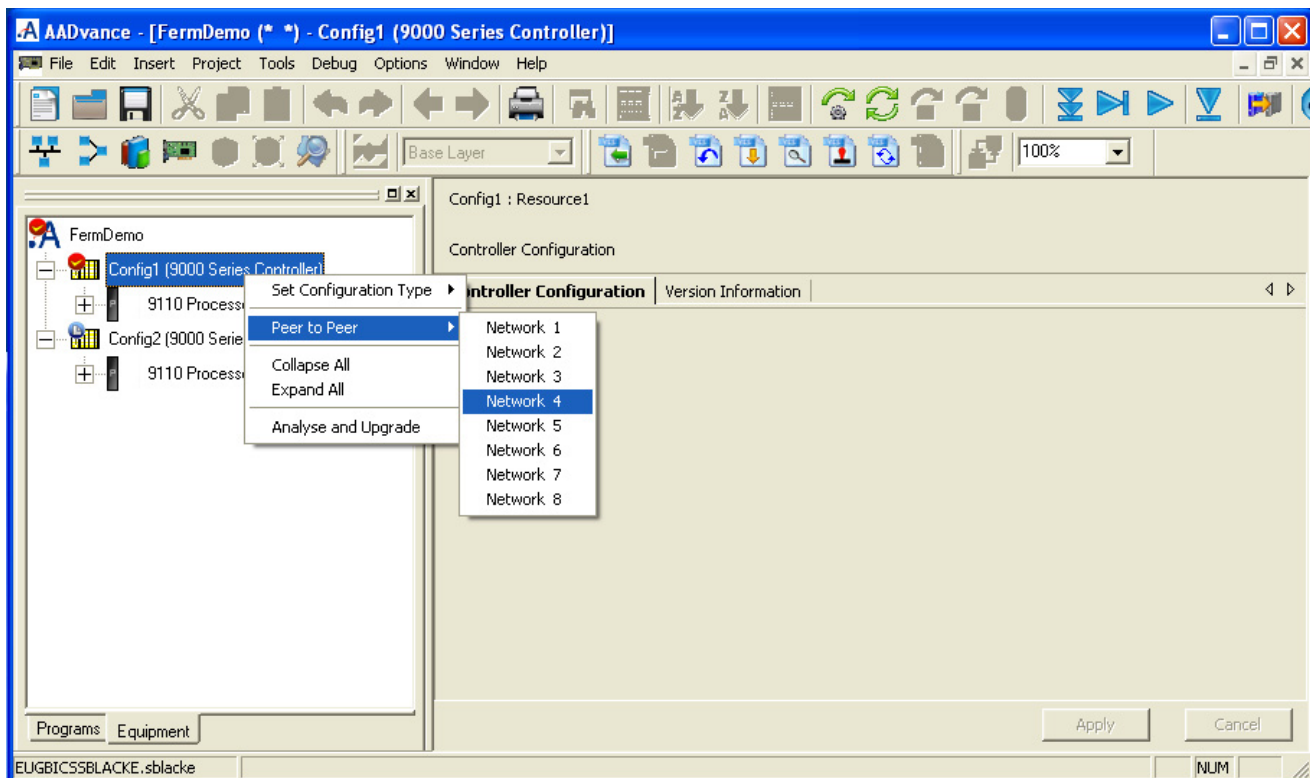
- Create a **Peer-to-Peer network** and give it an **identity**.
- Select a **Peer-to-Peer subnet configuration** and enter the **subnet data**.

- Enter the **peer IP Address** information.
- Configure the **I/O boards**.
- Wire the **I/O board status parameters** to **variables**.
- Wire the **I/O board channel data** to **variables**.

## Create a Peer-to-Peer Network

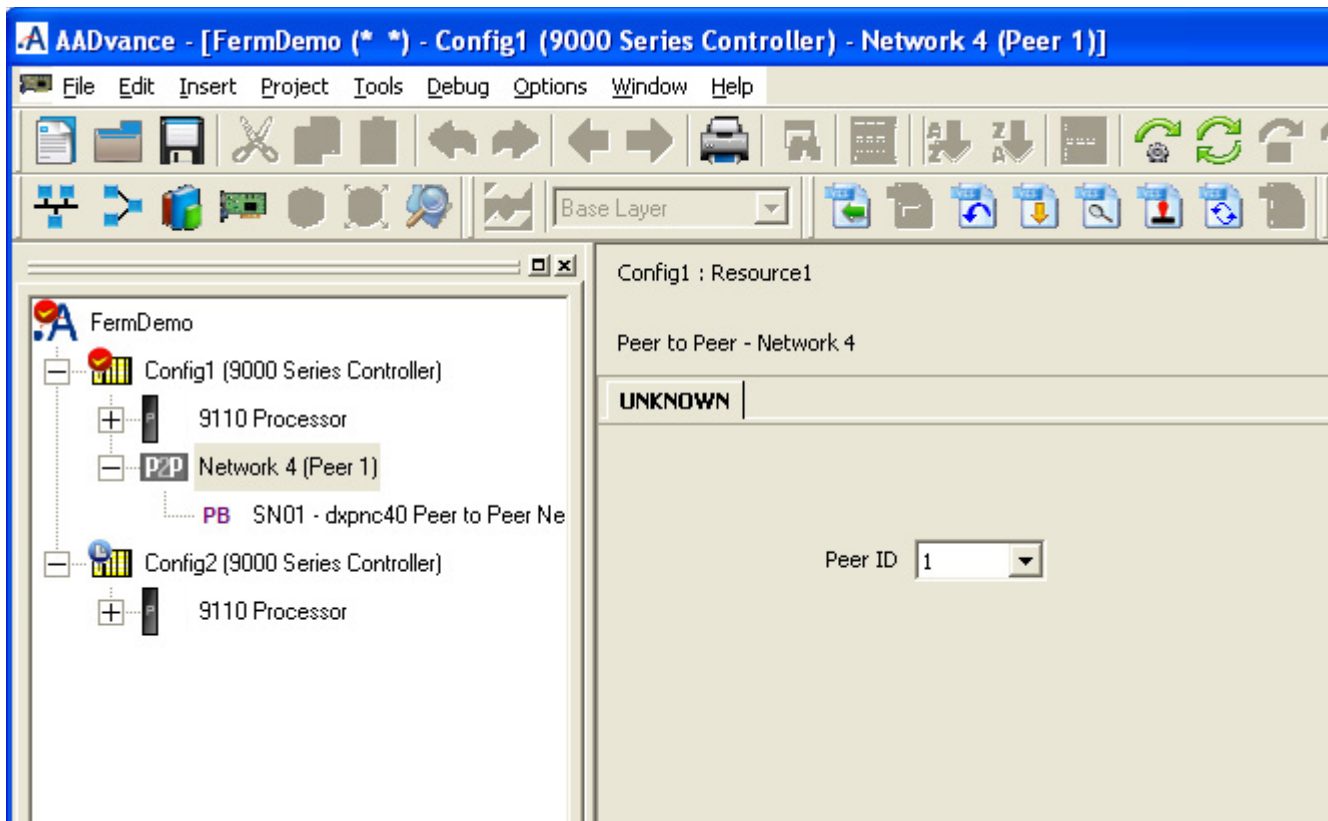
Create a Peer-to-Peer network as follows:

1. Select the **Equipment** tab.
2. Right-click on the **Config (9000 Series Controller)**.
3. Select the **Peer-to-Peer** option and any **network** option, in this case Network 4 has been selected.
  - A peer network Network 4 is set up.





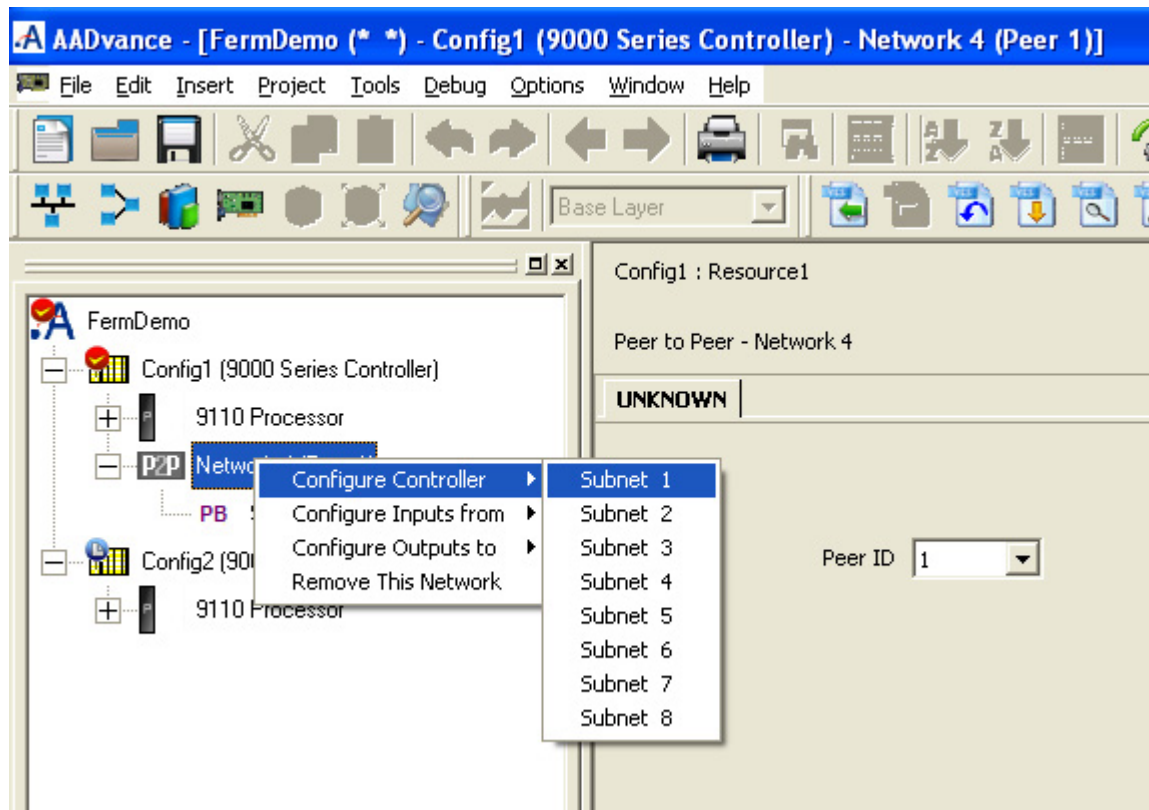
4. Enter the **peer identity number** in the **Peer ID** field.
  - The peer identity gives the position of the controller's IP address in the Peer List.



## Peer-to-Peer Subnet Controller Configuration

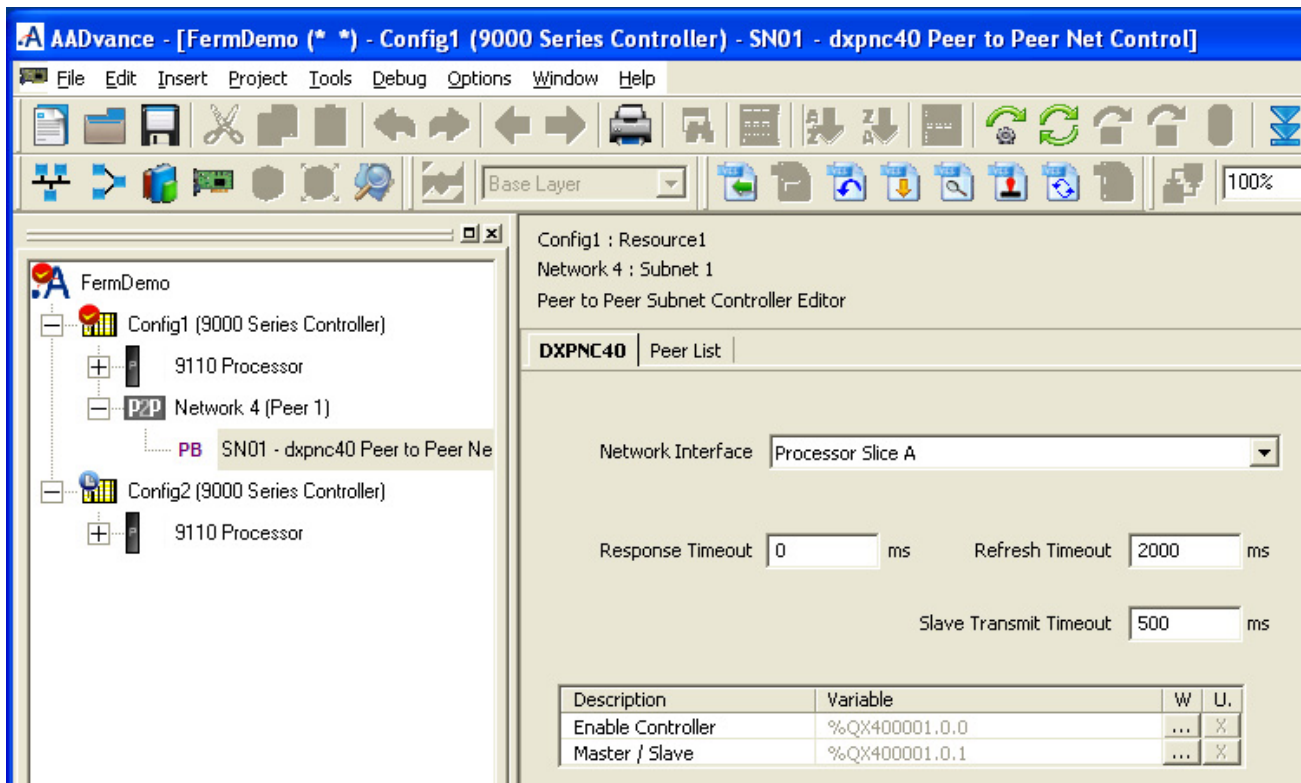
Configure the subnets as follows:

1. Right-click on **Network 4 (Peer 1)**, select **Configure Controller** and select a **Subnet**.



- A SN01 subnet is set up.
2. Select the **SN01 subnet**.

3. Select the **DXPNC40** tab.
  - The subnet DXPNC40 set up screen is displayed.

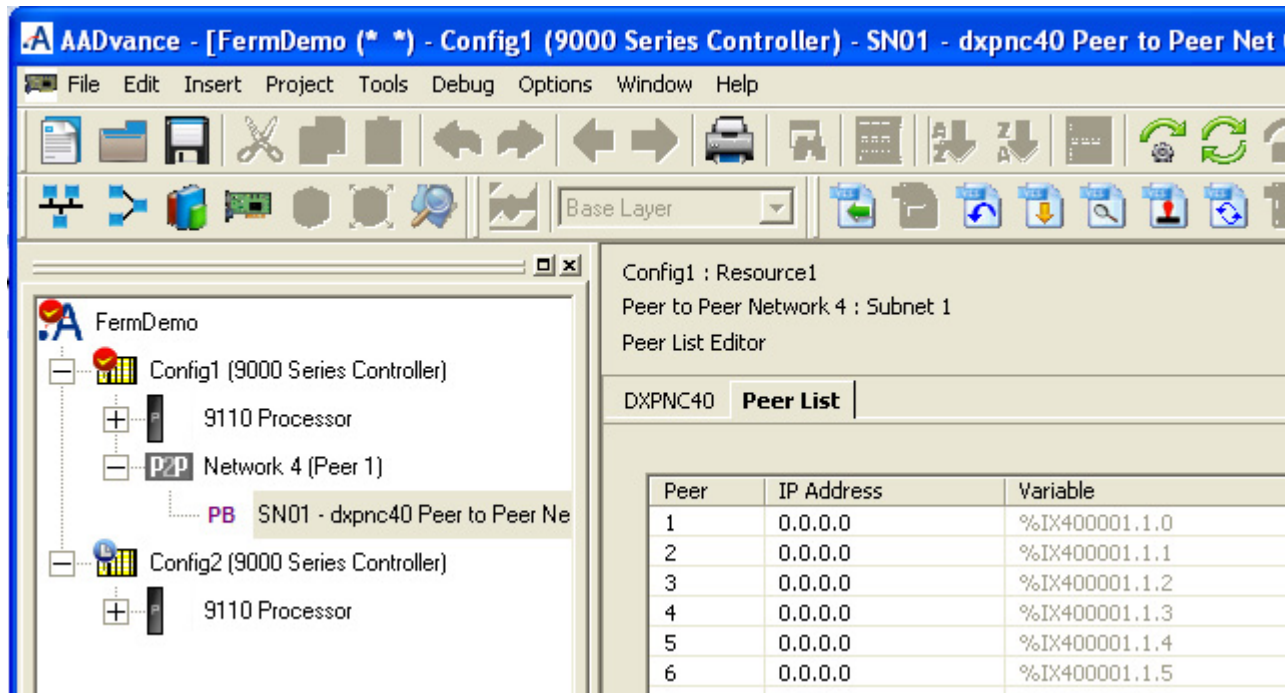


4. Enter the following data in the required fields.
  - Network Interface - enter which processor slice (**Processor A, B or C**) the peer network will be configured on, i.e. which processor module the network is physically connected to.
  - Response Timeout - Enter a **value in the range 0 to 10000 ms**. This value sets the time allowed for the peer to acknowledge a data packet. If the field is set to zero this indicates that no acknowledgment is required. It should only be set to a value above zero to avoid network packet sequence errors in networks where the propagation delay between any two nodes could exceed 1 ms.
  - If the entire Peer-to-Peer network is on a local Ethernet network with no routers or gateways, this setting may be zero. If the network is not a simple local network, set this parameter to exceed the longest expected propagation delay between any two nodes. The 'ping' command may be used to measure delays once the network is built. If the parameter is not zero, an acknowledgment is expected for every packet, to ensure it has been delivered. The Response Timeout acts as a watchdog trip for a lost packet, and it should be the shortest of the timeout settings.
  - Refresh Timeout - Enter a **value in the range 0 to 10000 ms**. This value sets the time that the controller will wait for its turn in the cycle to send its output data. The setting is used by both masters and slaves.

- The Refresh Timeout should be set to the Slave Transmit Timeout (see below) multiplied by the number of peers that will be lost on a worst-case network break.
  - The Slave Transmit Timeout should trip before the Refresh Timeout if the network is broken. This will minimize the loss of data transfer.
  - Slave Transmit Timeout - Enter a **value in the range 1 to 10000 ms**. This value sets the time a network master controller will wait for a slave to complete transmission of its data and return control of the network before declaring the slave absent. This parameter will be ignored during slave mode.
  - The Peer-to-Peer network operates in a cycle. Each controller in turn is asked to send all its configured output data. The Slave Transmit Timeout is only used by the master of a network. It acts as a watchdog for a lost slave, and it has to allow time for the slave to send all its outputs even if the slave is itself waiting for lost packets for its Response Timeout setting. Therefore the Slave Transmit Timeout should be set to:  
$$(\text{The slave's Response Timeout}) \times (\text{The maximum number of output data boards in any controller}) + 16 \text{ ms.}$$
  - If the Response Timeout is 0 ms, use 2 ms in the calculation above. A minimum Slave Transmit Timeout of 64 ms is recommended.
5. Wire Enable Controller and Master/Slave to variables you have set up for this purpose.
- Enable Controller - This variable starts or stops the peer communications using this controller, it should be set it to **TRUE** to enable the controller.
  - Master/Slave - This is a Boolean value that selects the controller's role, it should be set it to **TRUE** to make the controller act as a Master or **FALSE** to act as a Slave.

## Set up the Peer IP Addresses and Status Variable

1. Select the **Peer List** tab.



2. Enter the **IP Address** for each peer.

- You must ensure that when you enter an **IP Address** the **Row Number** you select from the **Peer Column** must correspond to the **Peer Identity Number**; e.g. for Network 4 (peer 1) enter the IP Address in row 1 of the Peer column.
- To configure a Multicast IP Address, enter the following IP Addresses:

**Port 1: 224.1.2.3**

**Port 2: 224.4.5.6**

3. Select a **Boolean input Variable** for the **same Peer number**.

- This variable reports the peer status and is TRUE when the peer is active and working and is FALSE when the peer is inactive or faulty. Except for multicast peers which are always set FALSE even when the multicast peer is active or inactive.

## Peer-to-Peer Data Boards

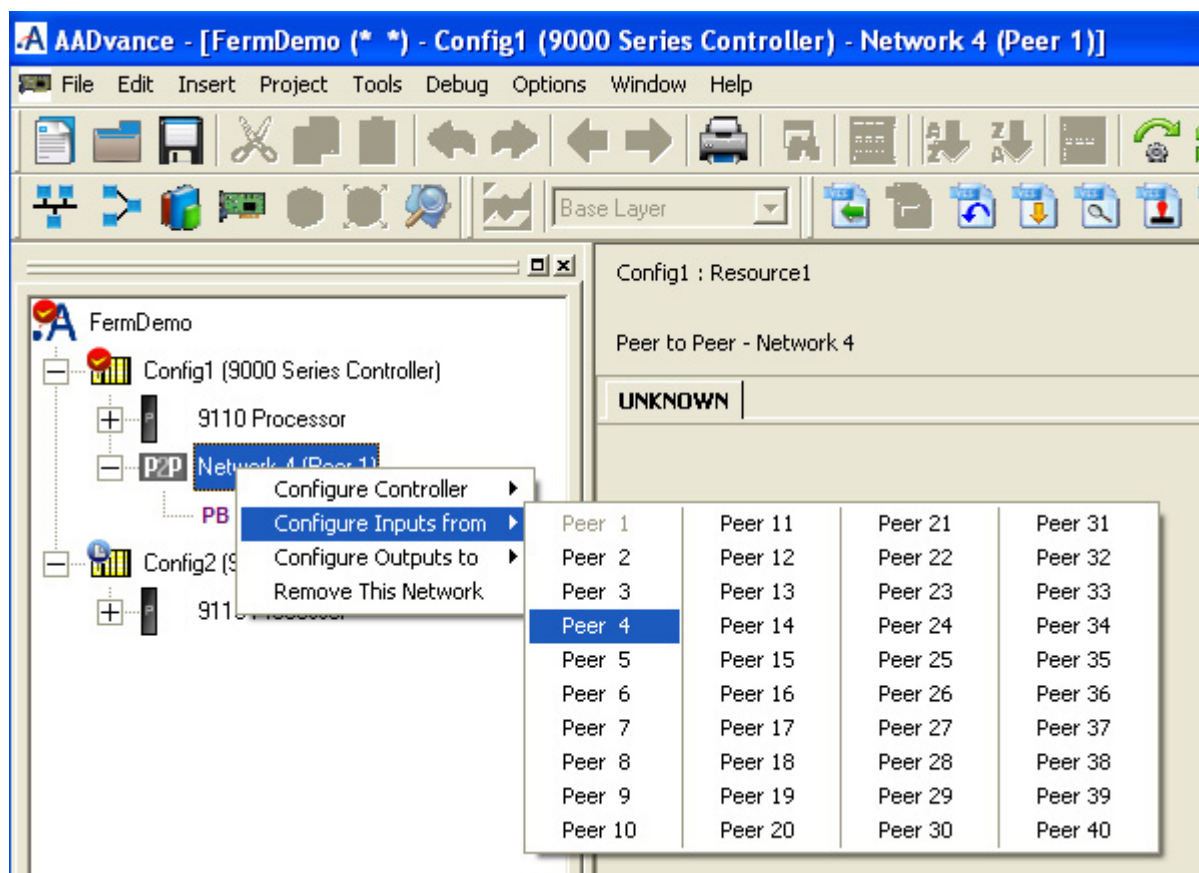
You can configure two types of input and output boards - Analogue or Digital - and they can be large or small versions. Select the type you require to ensure the optimum communication packet size can be used for your application. Each input board has a corresponding output board that must be of the same type and channel capacity.

Each output board delivers data to one or more input boards across one peer network. The subnet of the peer network used to send the data is transparent

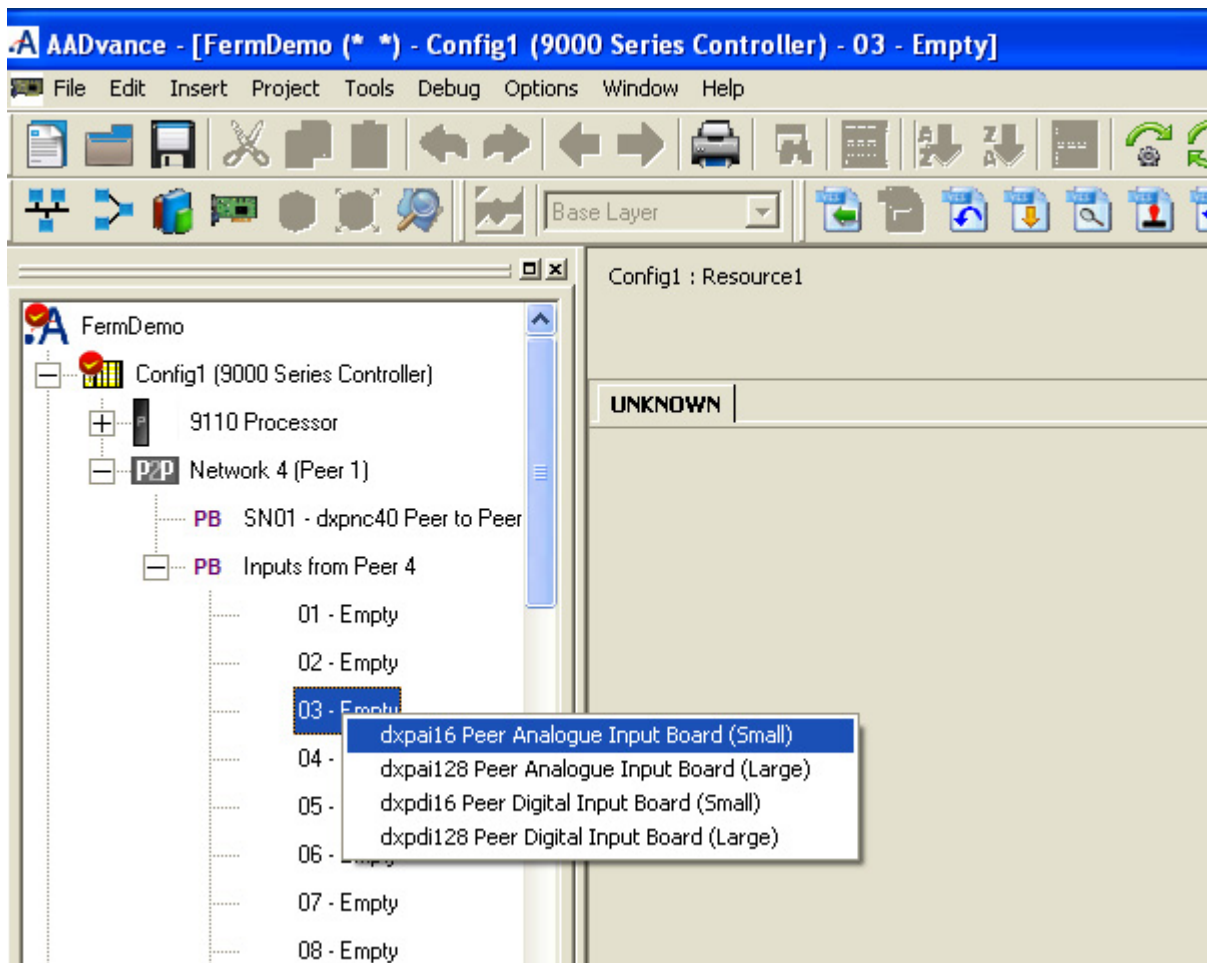
to the input and output boards and more than one subnet can be defined to provide redundant communications.

## Configure Input Boards

1. Select the **Input Source**.



## 2. Add an **Input Board**.



## Configure Analogue Input Boards

1. Select an **Analogue Input board**.
2. Select the **DXPAI16** tab.
3. Enter the following data
  - Refresh Timeout - Enter a **value in the range 1 to 10000 ms**. This is the maximum number of milliseconds allowed between successive refreshes of input data before the data is declared invalid. Should the time be exceed and the data declared invalid the input data will either retain the last received value or revert to a fail-safe condition according to the Hold Last Valid Value setting.



- The Refresh Timeout should be set to the longest delay before a fail-safe action should be taken. This will be the Process Safety Time for the input data. The corresponding Output board has a Refresh Interval parameter; this controls how often data is sent to the Input board. The Refresh Interval should be calculated to deliver at least two updates within the Refresh Timeout, under worst case conditions. A Refresh Timeout which is too short for the network size and complexity will make it impossible to deliver fresh data often enough.
- Value in Failed State - Enter a **value in the range-  $9.999999e+38$  to  $+9.999999e+38$** . This sets the control value to be adopted by the inputs when the input has not been refreshed within Refresh Timeout and is therefore declared invalid. Where the input corresponds to an integer the fractional part if truncated. The value is always adopted at application start-up, though it will not be used while the Hold Last Valid Value variable is TRUE.

The screen view below shows the configuration of a 16 channel analogue input board. The board being configured is for Peer Network 4, source peer is Peer 4 and the data block identity Board 3.

The screenshot shows the AADvance software interface for configuring a 9000 Series Controller. The project tree on the left shows the hierarchy: FermDemo > Config1 (9000 Series Controller) > 9110 Processor > Network 4 (Peer 1) > Inputs from Peer 4 > 03 - dxpai16 Peer Anal.

The main configuration area displays the following settings:

- Config1 : Resource1
- Network 4 : Peer 4 : Board 3
- Peer to Peer Analogue Input Control Editor
- DXPAI16 | Data Variables
- Refresh Timeout: 5000 ms
- Value in Failed State: -1024.0

The table below lists the data variables for the DXPAI16 board:

Description	Variable	W	U
Input Data is Valid	%IX404103.2.0	...	X
Refreshed by Subnet 1	%IX404103.2.1	...	X
Refreshed by Subnet 2	%IX404103.2.2	...	X
Refreshed by Subnet 3	%IX404103.2.3	...	X
Refreshed by Subnet 4	%IX404103.2.4	...	X
Refreshed by Subnet 5	%IX404103.2.5	...	X
Refreshed by Subnet 6	%IX404103.2.6	...	X
Refreshed by Subnet 7	%IX404103.2.7	...	X
Refreshed by Subnet 8	%IX404103.2.8	...	X
Hold Last Valid Value	%QX404103.3.0	...	X



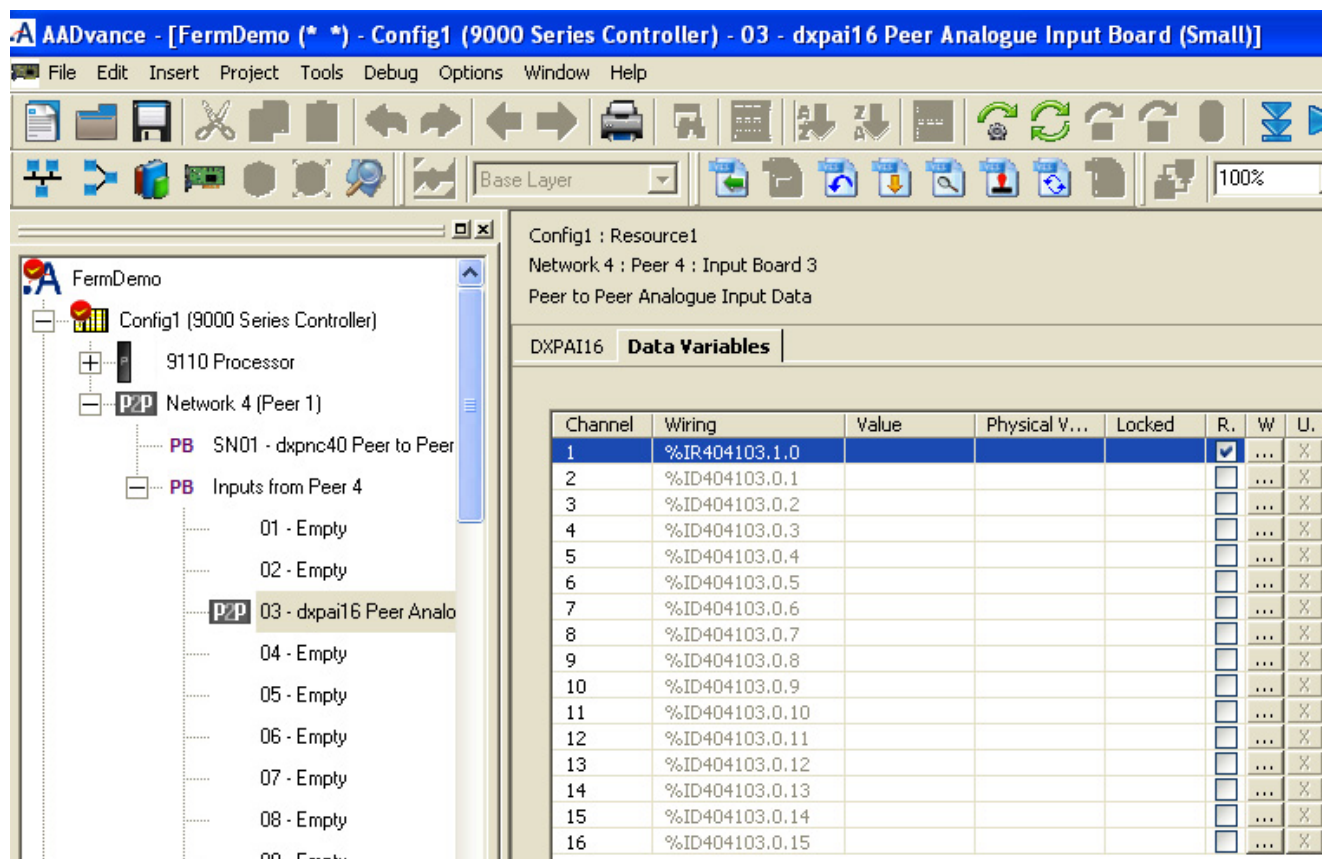
### Wire the Analogue Input Status Variables

1. Select the **Required Row** from the list shown.
  - Select the **W** button to connect the status parameter to a variable.
  - Input Data is Valid - this status variable is used to report that the input data is refreshed within the Refresh Timeout value.
  - Refreshed by Subnet 1 to 8 - Each of these variables can be used to indicate that the data has been refreshed by subnet 1 to 8 within the Refresh Timeout value. This status should be used to detect latent faults within a redundant network. The data is delivered over all available subnets simultaneously. If any variable goes FALSE for a programmed subnet, then the data has failed to arrive on that subnet within the Refresh Timeout value. The variables for programmed subnets may be combined through an AND gate to provide an indication of full redundancy on that particular data path.
2. Hold Last Valid Value - When it is set to FALSE it will force data to the fail safe state when the input data is invalid. When it is TRUE it will allow previous data to persist when the input data is invalid.
3. If you need to disconnect a variable select the **U** button.

### Wire Analogue Input Channel Data Variables

Analogue values are received from the corresponding output of the selected output board in the sending system. The values are 32 bit and will assume either a 32 bit signed integer format or a 32 bit real format depending on the

variable to which it is connected. Both the specific input channel and the corresponding output channel must be connected to the same variable type.



1. Select the **Data Variables** tab.
2. Select a **Channel** row.
3. Check or leave the **R** box.
  - check box to set the variable type to REAL
  - leave unchecked to select DINT.
4. Select the **W** button to connect to a variable.
5. If you need to disconnect a variable select the **U** button.

The 128 input peer board supports 128 analogue inputs instead of 16 but is otherwise identical.

---

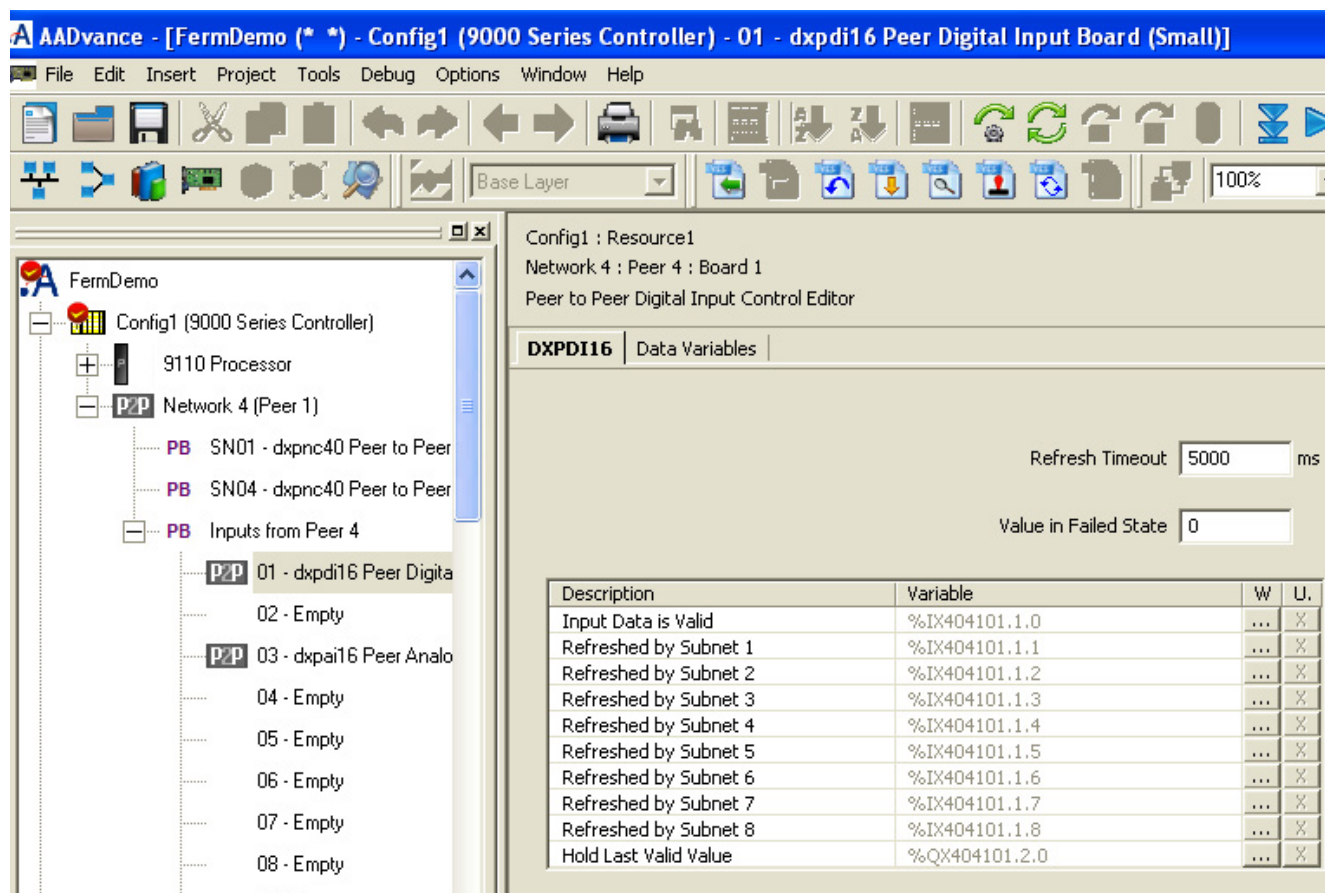
**IMPORTANT** The safety related data using 128 analogue channel blocks must be sent by two different input/output block pairs and compared at the receiving input end to ensure safety integrity. Alternatively it may be broken into 16 channel blocks.

---

## Configure Digital Input Boards

1. Select the **Digital Input Board**
  - The Peer-to-Peer Digital Input board editor dialog box is displayed.
2. Select the **DXDI16** tab
3. Enter the **Data** in the **Required Fields**.
  - Refresh Timeout - Enter a **value in the range 1 ms to 10000 ms**. This is the maximum number of milliseconds allowed between successive refreshes of input data before the data is declared invalid. Should the time be exceed and the data declared invalid the input data will either retain the last received value or revert to a fail-safe condition according to the Hold Last Valid Value setting.
  - The Refresh Timeout should be set to the longest delay before a fail-safe action should be taken. This will be the Process Safety Time for the input data. The corresponding Output board has a Refresh Interval parameter; this controls how often data is sent to the Input board. The Refresh Interval should be calculated to deliver at least two updates within the Refresh Timeout, under worst case conditions. A Refresh Timeout which is too short for the network size and complexity will make it impossible to deliver fresh data often enough.

- Value in Failed State - Enter a **value in the range  $-9.999999e+38$  to  $+9.999999e+38$** . This sets the control value to be adopted by the inputs when the input has not been refreshed within Refresh Timeout and is therefore declared invalid. Where the input corresponds to an integer the fractional part is truncated. The value is always adopted at application start-up, though it will not be used while the Hold Last Valid Value variable is set TRUE.



### Wire Digital Input Board Status Variables

1. Select the **Required Row** from the list shown.
2. Select the **W** button to connect the status parameter to the required variables.
  - Input Data is Valid - this status variable is used to report that the input data is refreshed within the Refresh Timeout value.

- Refreshed by Subnet 1 to 8 - Each of these variables can be used to indicate that the data has been refreshed by subnet 1 to 8 within the Refresh Timeout value. This status should be used to detect latent faults within a redundant network. The data is delivered over all available subnets simultaneously. If any variable goes FALSE for a programmed subnet, then the data has failed to arrive on that subnet within the Refresh Timeout value. The variables for programmed subnets may be combined through an AND gate to provide an indication of full redundancy on that particular data path.
- Hold Last Valid Value - When this variable is set to FALSE it will force data to the fail safe state when data is invalid. When it is TRUE it will allow previous data to persist when data is invalid.

3. If you need to disconnect a variable select the **U** button.

### Wire Digital Input Channel Data Variables

Digital input values are received from the corresponding output of the selected output board in the sending system; the values are Boolean values. Both the specific input channel and the corresponding output channel must be connected to the same variable type.

1. Select the **Data Variables** tab.

The screenshot shows the AADvance software interface for a 9000 Series Controller. The project tree on the left shows the hierarchy: Config1 (9000 Series Controller) > Network 4 (Peer 1) > Inputs from Peer 4 > 01 - dxpdi16 Peer Digital Input Board (Small). The main workspace displays the 'Data Variables' tab for the DXPD16 board. The table below shows the configuration for 16 channels.

Channel	Wiring	Value	Physical V...	Locked	R.	W	U.
1	%IX404101.0.0					...	X
2	%IX404101.0.1					...	X
3	%IX404101.0.2					...	X
4	%IX404101.0.3					...	X
5	%IX404101.0.4					...	X
6	%IX404101.0.5					...	X
7	%IX404101.0.6					...	X
8	%IX404101.0.7					...	X
9	%IX404101.0.8					...	X
10	%IX404101.0.9					...	X
11	%IX404101.0.10					...	X
12	%IX404101.0.11					...	X
13	%IX404101.0.12					...	X
14	%IX404101.0.13					...	X
15	%IX404101.0.14					...	X
16	%IX404101.0.15					...	X

2. Select a **Channel Row**.

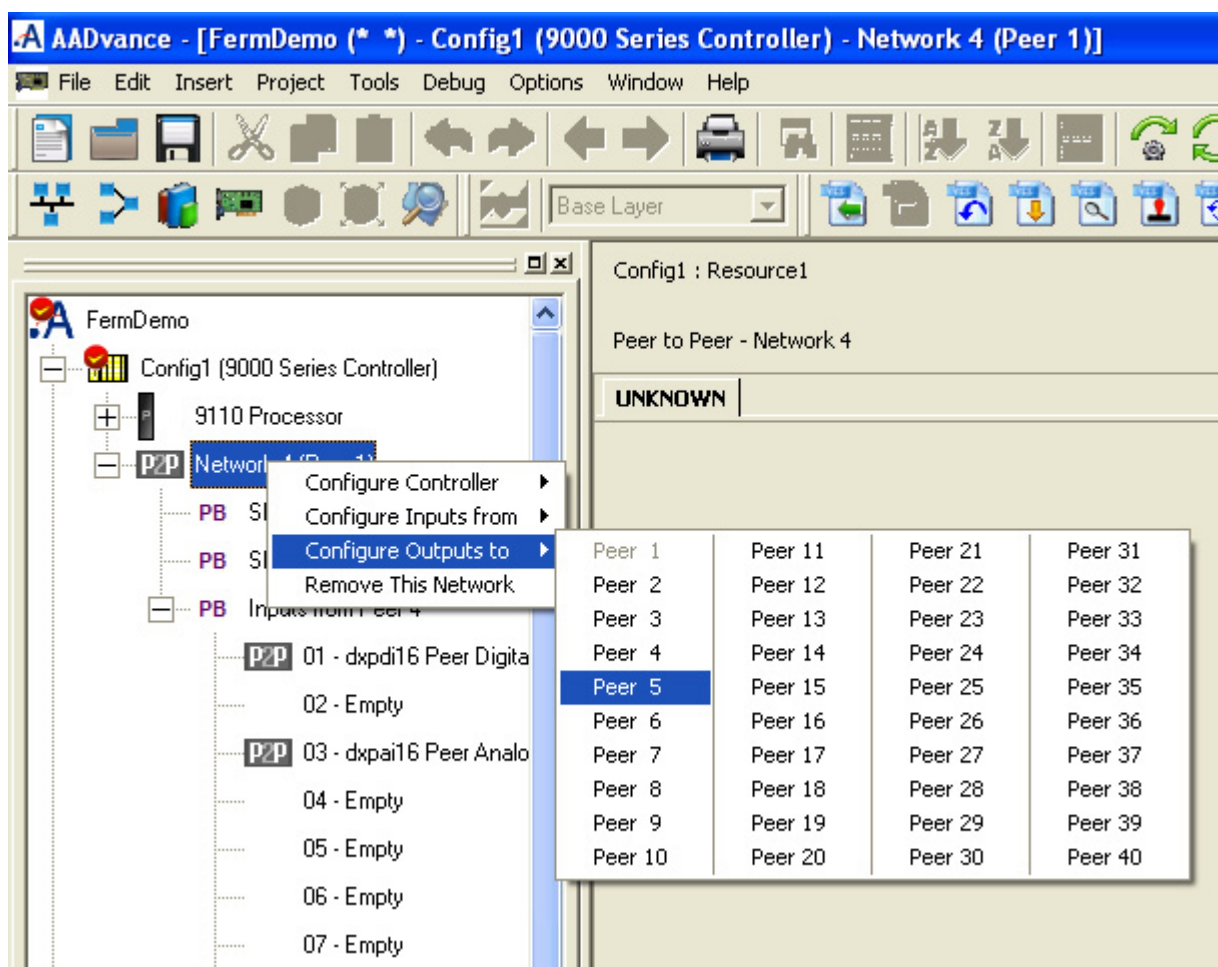
3. Select the **W** button to connect to a variable.
4. If you need to disconnect a variable select the **U** button.

The 128 input peer board supports 128 analogue inputs instead of 16 but is otherwise identical.

## Configure Output Boards

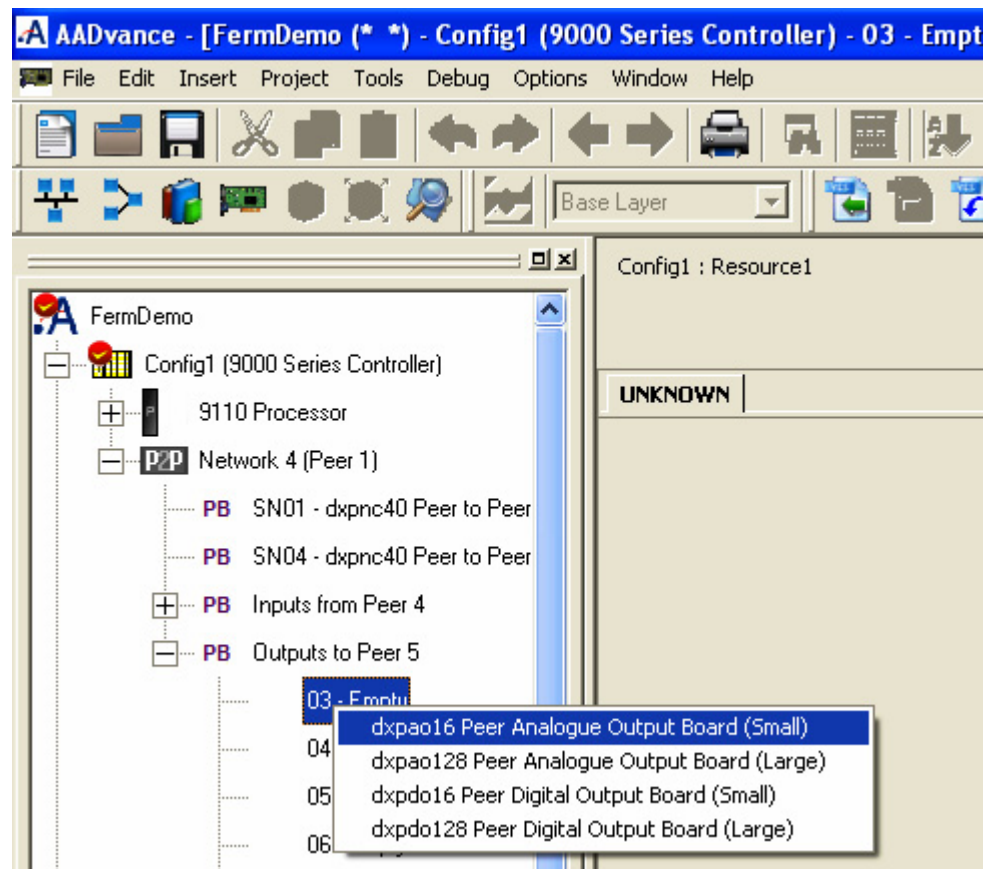
To configure an output for a peer network do the following:

1. Right click on **Network 4 (Peer 1)**
2. Select a **Peer Number** which is the target for the output data.





3. Add **Output Boards** as required.

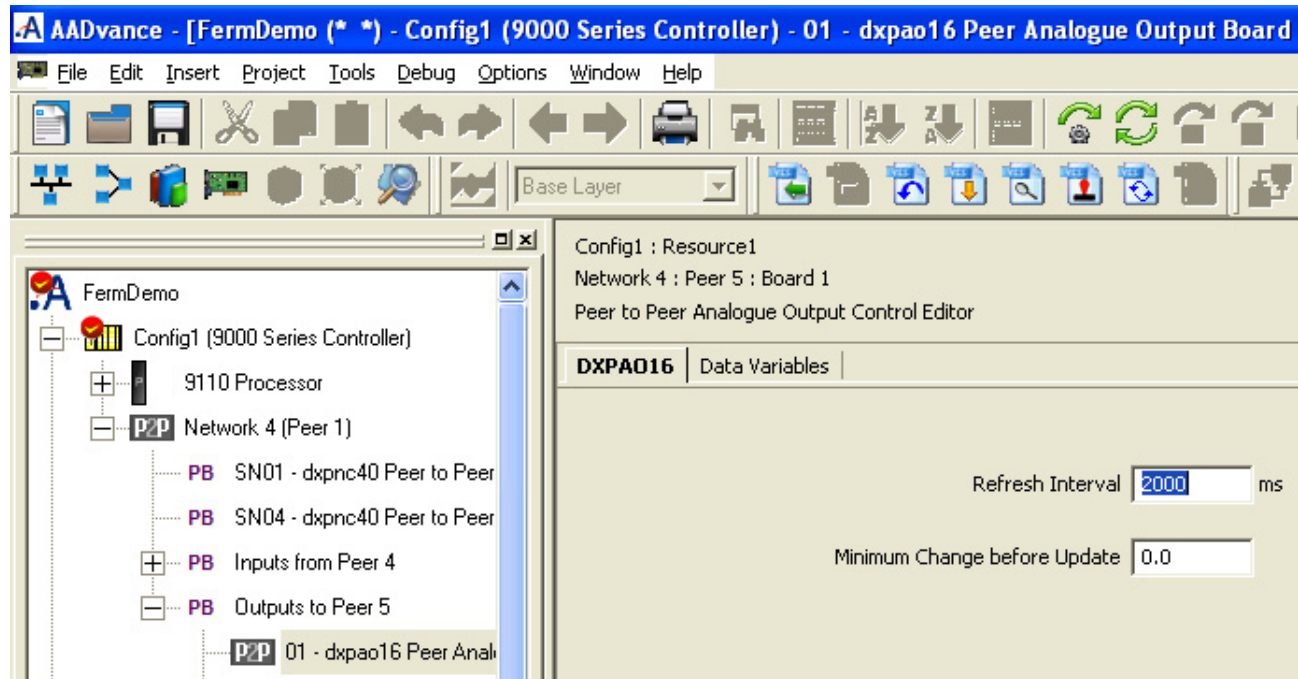


## Configure Analogue Output Boards

1. Select the **Board**.
2. Select the **DXPAO16** tab.
3. Enter the following data.
  - Refresh Interval - Enter a **value in the range 0 – 10000 ms**. This is the maximum time allowed between transmissions of the output data.
  - Data will be sent immediately following any change of output state. If a value of zero is specified in this field then data will be refreshed every application scan regardless of output state change.
  - The Refresh Interval should be calculated to deliver at least two updates within the Refresh Timeout of the corresponding Input board, under worst case conditions. A Refresh Timeout which is too short for the network size and complexity will make it impossible to deliver fresh data often enough.
  - The setting for the Refresh Interval should be less than:
 
$$(\text{Input board Refresh Timeout} - \text{Worst case Delivery Delay}) / 2$$
  - The Worst Case Delivery Delay is:

Refresh Timeout (on DXPN40 control board) + Scan time of this controller + Scan time of receiving controller + 50 ms

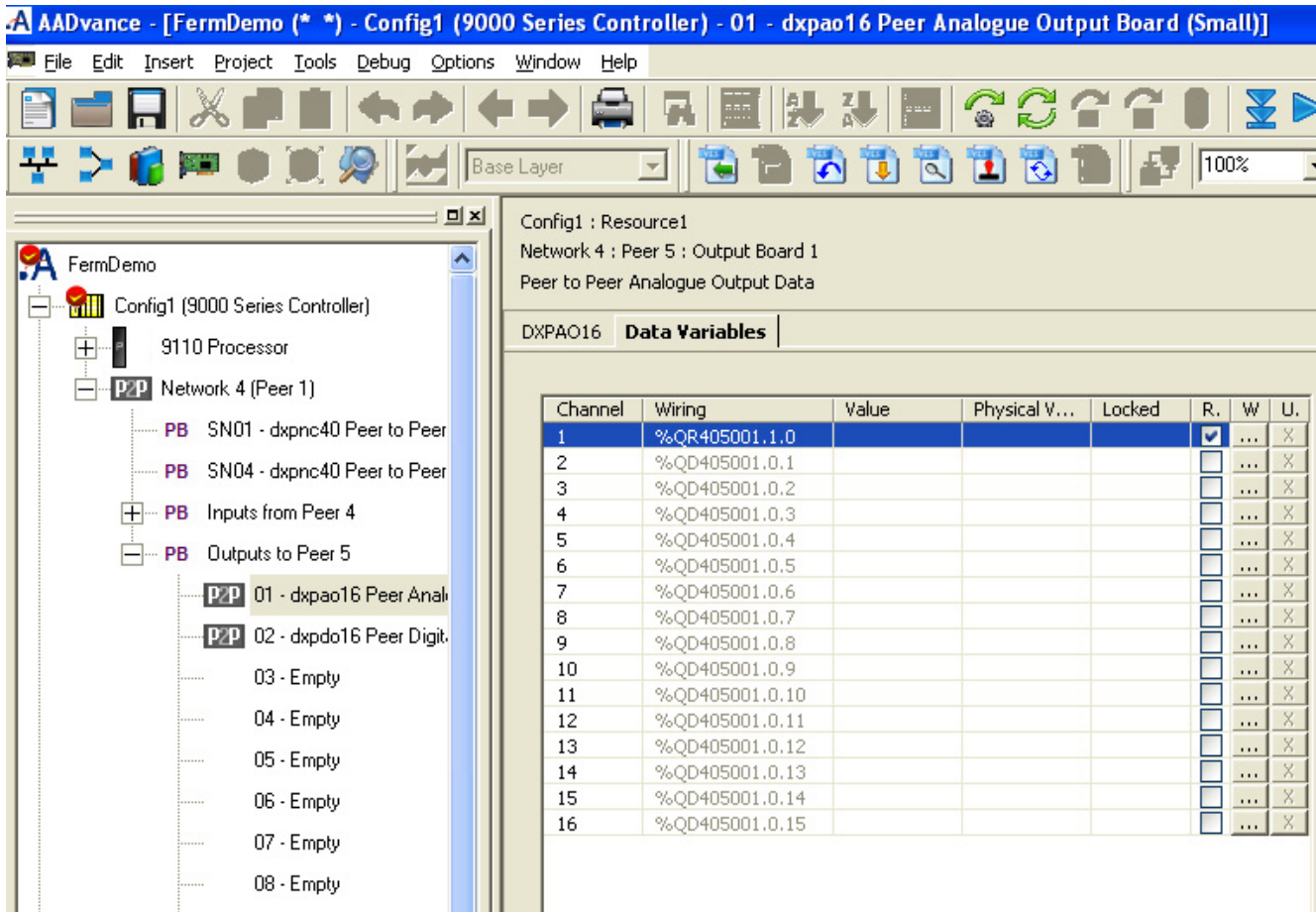
- Minimum Change before Update - Enter a **value in the range 0 to 9.999999e+038**. This value sets the minimum change in any output variable before the update is sent to the Peer input board (excluding any refresh interval). When applied to integers the fractional part is truncated.





## Wire the Analogue Output Channel Variables

Analogue output channels send analogue data to the corresponding analogue input channels on the peer network.

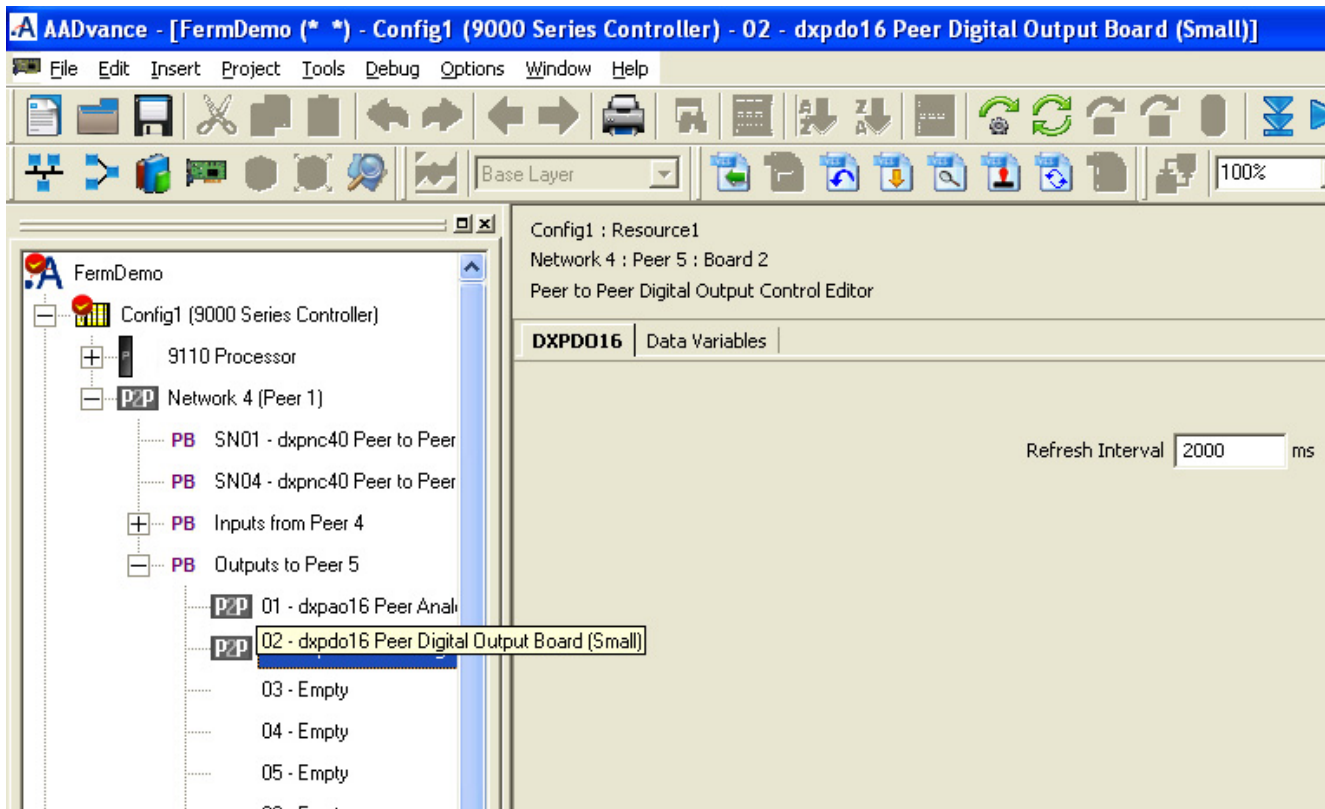


1. Select the **Data Variables** tab.
2. Select a **Channel Row**.
3. Select the **R** check box to set the variable data type,
  - not checked = DINT
  - checked = REAL.
4. Select the **W** button to connect to a variable.
5. If you need to disconnect a variable select **U** button.

## Configure Digital Output Boards

1. Select the **Board**.

2. Select the **DXPDO16** tab.



3. Enter the following data.

- Refresh Interval - Enter a **value in the range 0 – 10000 ms**. This is the maximum time allowed between transmissions of the output data.
- Data will be sent immediately following any change of output state. If a value of zero is specified in this field then data will be refreshed every application scan regardless of output state change.
- The Refresh Interval should be calculated to deliver at least two updates within the Refresh Timeout of the corresponding Input board, under worst case conditions. A Refresh Timeout which is too short for the network size and complexity will make it impossible to deliver fresh data often enough.
- The setting for the Refresh Interval should be less than:  

$$(\text{Input board Refresh Timeout} - \text{Worst case Delivery Delay}) / 2$$
- The Worst Case Delivery Delay is:  

$$\text{Refresh Timeout (on DXPN40 control board)} + \text{Scan time of this controller} + \text{Scan time of receiving controller} + 50 \text{ ms.}$$

### Wire the Digital Output Channel Data Variables

The digital output boards send digital data to the corresponding digital input boards on the peer network.

1. Select the **Data Variables** tab.

2. Select a **Channel Row**.
3. Select a **W** button to connect to a variable.
  - The outputs are Boolean values.
4. To disconnect a variable select the **U** button.

The screenshot shows the AADvance software interface. The title bar reads: "AADvance - [FermDemo (\* \*) - Config1 (9000 Series Controller) - 02 - dxpdo16 Peer Digital Output Board (Small)]". The menu bar includes: File, Edit, Insert, Project, Tools, Debug, Options, Window, Help. The toolbar contains various icons for file operations, navigation, and configuration. The left pane shows a tree view with the following structure:

- FermDemo
  - Config1 (9000 Series Controller)
    - 9110 Processor
    - Network 4 (Peer 1)
      - SN01 - dxpnc40 Peer to Peer
      - SN04 - dxpnc40 Peer to Peer
      - Inputs from Peer 4
      - Outputs to Peer 5
        - 01 - dxpao16 Peer Anal
        - 02 - dxpdo16 Peer Digit.
        - 03 - Empty
        - 04 - Empty
        - 05 - Empty
        - 06 - Empty
        - 07 - Empty
        - 08 - Empty

The right pane shows the "Data Variables" table for "DXPDO16". The table has the following columns: Channel, Wiring, Value, Physical V..., Locked, R., W, U.

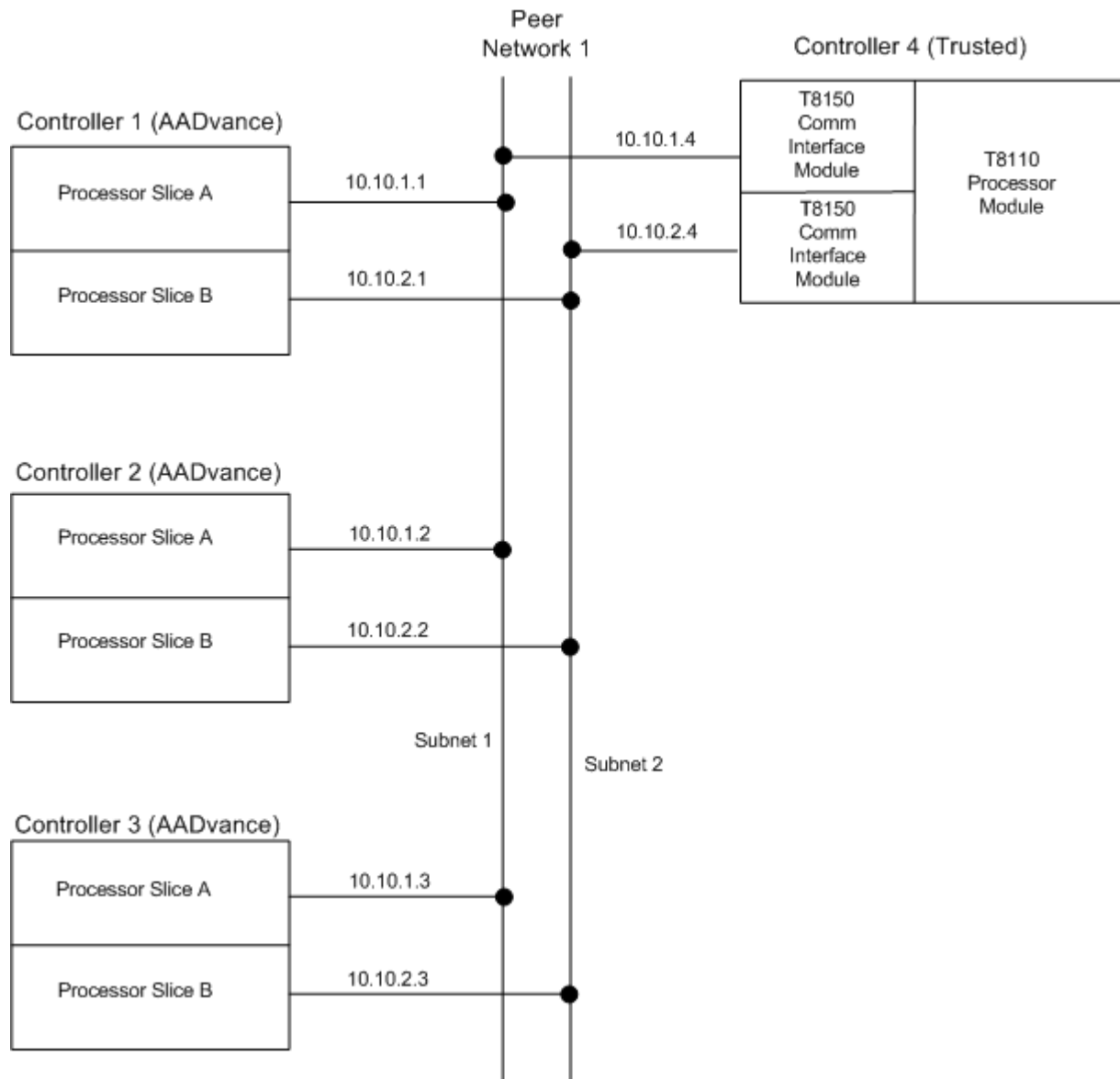
Channel	Wiring	Value	Physical V...	Locked	R.	W	U
1	%QX405002.0.0					...	X
2	%QX405002.0.1					...	X
3	%QX405002.0.2					...	X
4	%QX405002.0.3					...	X
5	%QX405002.0.4					...	X
6	%QX405002.0.5					...	X
7	%QX405002.0.6					...	X
8	%QX405002.0.7					...	X
9	%QX405002.0.8					...	X
10	%QX405002.0.9					...	X
11	%QX405002.0.10					...	X
12	%QX405002.0.11					...	X
13	%QX405002.0.12					...	X
14	%QX405002.0.13					...	X
15	%QX405002.0.14					...	X
16	%QX405002.0.15					...	X

## Peer-to-Peer Configuration Example 1

This example uses 4 controllers connected to two subnets forming one Peer-to-Peer network.

- Controllers 1, 2 and 3 are dual processor AADvance controllers and Controller 4 is Trusted.

- Controller 4 is designated the "Master" of Network 1 Subnet 1 and Controller 1 the "Master" of Network 1 Subnet 2.



## Peer-to-Peer Controller Setting Summary

You should follow the process described in this chapter.

The network configuration is as follows:

- 16 analogue outputs are sent from Controller 1 to Controller 2
- 16 analogue values are sent from Controller 1 to Controller 3
- 16 digital values are sent from Controller 2 to Controller 4

**Table 38 - Controller 1 - Dual Peer-to-Peer Net Control, Network 1, Subnet 1**

	Value	Comment
<b>DXPNC40</b>	x	
Network Interface	A	Network connected to Processor A
Network Identity	1	Network 1
Subnet Identity	1	Subnet 1
Peer ID	1	Identity of this controller
Response Timeout	0	Default
Refresh Timeout	2000	Default
Slave Transmit timeout	500	Default
Enable Controller	TRUE	Enable Peer on this subnet
Master/Slave	FALSE	This is a slave connection
<b>Peer List:</b>		
Peer 1	10.10.1.1	Controller 1, Network 1, Subnet 1
Peer 2	10.10.1.2	Controller 2, Network 1, Subnet 1
Peer 3	10.10.1.3	Controller 3, Network 1, Subnet 1
Peer 4	10.10.1.4	Controller 4, Network 1, Subnet 1
Peer 5		
Peer 6		
Peer 7		

**Table 39 - Controller 1 - Dual Peer-to-Peer Net Control, Network 1 Subnet 2**

	Value	Comment
<b>DXPNC40</b>		
Network Interface	B	Network connected to Processor B
Network Identity	1	Network 1
Subnet Identity	2	Subnet 1
Peer ID	1	Identity of this controller
Response Timeout	0	Default
Refresh Timeout	2000	Default
Slave Transmit timeout	500	Default
Enable Controller	TRUE	Enable Peer on this subnet
Master/Slave	TRUE	This is a Master connection
<b>Peer List:</b>		
Peer 1	10.10.2.1	Controller 1, Network 1, Subnet 2
Peer 2	10.10.2.2	Controller 2, Network 1, Subnet 2
Peer 3	10.10.2.3	Controller 3, Network 1, Subnet 2
Peer 4	10.10.2.4	Controller 4, Network 1, Subnet 2
Peer 5		
Peer 6		
Peer 7		

### Network Set Up - Network 1, Subnet 1

The following are the values that should be entered during the set up process to configure an example Peer-to-Peer network.

**Table 40 - Controllers 1, 2 and 3**

	Controller 1	Controller 2	Controller 3
<b>dxpnc40</b>			
Network interface	A	A	A
Network Identity	1	1	1
Subnet Identity	1	1	1
Peer ID	1	2	3
Response Timeout	0	0	0
Refresh Timeout	2000	2000	2000
Slave Transmit Timeout	500	500	500
Enable Controller	TRUE	TRUE	TRUE
Master Slave	FALSE	FALSE	FALSE
<b>Peer List:</b>			
Peer 1	10.10.1.1	10.10.1.1	10.10.1.1
Peer 2	10.10.1.2	10.10.1.2	10.10.1.2
Peer 3	10.10.1.3	10.10.1.3	10.10.1.3
Peer 4	10.10.1.4	10.10.1.4	10.10.1.4
Peer 5			

**Table 41 - Controller 4**

	Controller 4
<b>dxpnc40 - Control Rack</b>	
Chassis	1
Slot	7
Network Id	1
Subnet_ID	1
Peer ID	4
Response_TMO	0
Refresh_TMO	2000
TX_DATA_TMO	500
Variable 1	TRUE
Variable 2	TRUE
<b>Peers_1 rack:</b>	
Peer_IP_01	10.10.1.1
Peer_IP_02	10.10.1.2

	Controller 4
Peer_IP_03	10.10.1.3
Peer_IP_04	10.10.1.4
Peer_IP_05	

### Dual Peer-to-Peer Net Control Network Set Up - Network 1, Subnet 2

**Table 42 - Controllers 1, 2 & 3**

	Controller 1	Controller 2	Controller 3
<b>dxpnc40</b>			
Network interface	B	B	B
Network Identity	1	1	1
Subnet Identity	2	2	2
Peer ID	1	2	3
Response Timeout	0	0	0
Refresh Timeout	2000	2000	2000
Slave Transmit Timeout	500	500	500
Enable Controller	TRUE	TRUE	TRUE
Master Slave	TRUE	FALSE	FALSE
<b>Peer List:</b>			
Peer 1	10.10.2.1	10.10.2.1	10.10.2.1
Peer 2	10.10.2.2	10.10.2.2	10.10.2.2
Peer 3	10.10.2.3	10.10.2.3	10.10.2.3
Peer 4	10.10.2.4	10.10.2.4	10.10.2.4
Peer 5			

**Table 43 - Controller 4**

	Controller 4
<b>dxpnc40 - Control Rack</b>	
Chassis	1
Slot	8
Network Id	1
Subnet_ID	1
Peer ID	4
Response_TMO	0
Refresh_TMO	2000
TX_DATA_TMO	500
Variable 1	TRUE
Variable 2	FALSE
<b>Peers_1 rack:</b>	
Peer_IP_01	10.10.2.1
Peer_IP_02	10.10.2.2

	Controller 4
Peer_IP_03	10.10.2.3
Peer_IP_04	10.10.2.4
Peer_IP_05	

## Peer-to-Peer Data Summary

### Output Data

**Table 44 - Controller 1 and 2**

	Controller 1		Controller 2
Board Type	dxpao	dxpao	dxpdo
Network Identity	1	1	1
Target Peer Identity	2	3	4
Source Data Identity	1	2	1
Refresh Timeout	2000	2000	2000
Minimum Change before update	20	20	
Variable 1 -16	Analogue Data Output	Analogue Data Output	Digital Data Output

### Input Data

**Table 45 - Controller 2 and 3**

	Controller 2	Controller 3
	dxpai	dxpai
Network Identity	1	1
Source Peer Identity	1	1
Source Data Identity	1	2
Refresh Timeout	5000	5000
Value in Failed State	-1024	-1024
Input Data is Valid	Input data valid	Input data valid
Refreshed by Subnet 1...8	Refreshed on subnet 1...8	Refreshed on subnet 1...8
Host Last Valid Value	Failure Action	Failure action

#### Data Variables Tab

Variable 1 - 16	Analogue Data Input	Analogue Data Input
-----------------	---------------------	---------------------

**Table 46 - Controller 4**

	Controller 4
DATA RACK	dxpdi
Network Identity	1
Source Peer Identity	2
Source data Identity	1



	<b>Controller 4</b>
Refresh Timeout	5000
Value in Failed State	FALSE
<b>STATUS RACK</b>	
Variable 1	Boolean Data Input
Input Data is Valid	Input Data Valid
Refreshed by Subnet 1...9	Refreshed on Subnet 1...8
<b>CONTROL RACK</b>	
Variable 1	Failure Action



## Additional Resources

### Associated AADvance Publications

For more information about the AADvance system refer to the associated Rockwell Automation technical manuals shown in table below.

Resource	Document Number
Safety Manual	<a href="#">ICSTT - RM446</a>
System Build Manual	<a href="#">ICSTT - RM448</a>
* Configuration Guide	<a href="#">ICSTT - RM405</a>
* Configuration Guide	<a href="#">ICSTT - RM458</a>
OPC Portal Server User Manual	<a href="#">ICSTT - RM407</a>
PFH and PFD <sub>avg</sub> Data Manual	<a href="#">ICSTT - RM449</a>
Solutions Handbook	<a href="#">ICSTT - RM447</a>
Troubleshooting and Maintenance Manual	<a href="#">ICSTT - RM406</a>

\* Actual configuration guide applicable is dependent upon version of AADvance Workbench used.

Publication	Purpose and Scope
Safety Manual	This technical manual defines how to safely apply AADvance controllers for a Safety Instrumented Function. It sets out standards (which are mandatory) and makes recommendations to make sure that installations satisfy and maintain their required safety integrity level.
Solutions Handbook	This technical manual describes the features, performance and functionality of the AADvance controller and systems. It gives guidance on how to design a system to satisfy your application requirements.
System Build Manual	This technical manual describes how to assemble a system, switch on and validate the operation of your system.
Configuration Guide	This software technical manual defines how to configure an AADvance controller using the AADvance Workbench to satisfy your system operation and application requirements.
Troubleshooting and Maintenance Manual	This technical manual describes how to maintain, troubleshoot and repair an AADvance Controller.
OPC Portal Server User Manual	This manual describes how to install, configure and use the OPC Server for an AADvance Controller.
PFH and PFD <sub>avg</sub> Data	This document contains the PFH and PFD <sub>avg</sub> Data for the AADvance Controller. It includes examples on how to calculate the final figures for different controller configurations.

## Regional Offices

Rockwell Automation 4325 West Sam Houston Parkway North, Suite 100 Houston Texas 77043-1219 USA	Rockwell Automation Hall Road Maldon Essex CM9 4LA England	Rockwell Automation Millennium House Campus 1 Aberdeen Science & Technology Park Balgownie Road, Bridge of Don Scotland
Tel: +1 713 353 2400 Fax: +1 713 353 2401	Tel: +44 1621 854444 Fax: +44 1621 851531	Tel: +44 1224 227780
Rockwell Automation No. 2 Corporation Road #04-01 to 03 Corporation Place Singapore 618494	Abu Dhabi: 903, Bin Hamoodah Building 9th Floor Khalifa Street Abu Dhabi, UAE	Dubai: Silvertech Middle East FZCO PO Box 17910 Jebel Ali Free Zone Dubai, UAE
Tel: +65 6622 4888 Fax: +65 6622 4884	Tel: +971 2 627 6763	Tel: +971 4 883 7070

Internet: <http://www.rockwellautomation.com/icstriplex>

Technical support: [icstsupport@ra.rockwell.com](mailto:icstsupport@ra.rockwell.com)

Sales enquiries: [sales@icstriplex.com](mailto:sales@icstriplex.com)

## Glossary of Terms

### A

**accuracy** The degree of conformity of a measure to a standard or a true value. See also 'resolution'.

**achievable safe state** A safe state that is achievable.

---

**NOTE** Sometimes, a safe state cannot be achieved. An example is a non-recoverable fault such as a voting element with a shorted switch and no means to bypass the effect of the short.

---

**actuator** A device which cause an electrical, mechanical or pneumatic action to occur when required within a plant component. Examples are valves and pumps.

**AITA** Analogue input termination assembly.

**alarms and events (AE)** An OPC data type that provides time stamped alarm and event notifications.

**allotted process safety time** The portion of the total process safety time allotted to a sub function of that process.

**application software** Software specific to the user application, typically using logic sequences, limits and expressions to read inputs, make decisions and control outputs to suit the requirements of the system for functional safety.

**architecture** Organizational structure of a computing system which describes the functional relationship between board level, device level and system level components.

**asynchronous** A data communications term describing a serial transmission protocol. A start signal is sent before each byte or character and a stop signal is sent after each byte or character. An example is ASCII over RS-232-C. See also 'RS-232-C, RS-422, RS-485'.

**availability** The probability that a system will be able to carry out its designated function when required for use — normally expressed as a percentage.

**B**

- backplane clip** A sprung, plastic device to hold together two adjacent AADvance base units. Part number 9904. Used in pairs.
- base unit** One of two designs which form the supporting parts of an AADvance controller. See 'I/O base unit' and 'processor base unit'.
- bindings** Bindings describe a "relationship" between variables in different AADvance controllers. Once a variable is "bound" to another variable, a unique and strong relationships is created between the two variables and the SIL 3 Certified SNCP protocol is used to ensure that the consuming variable is updated with the data from the producing variable.
- black channel** A communication path whose layer (i.e. cabling, connections, media converters, routers/switches and associated firmware/software, etc.) has no requirement to maintain the integrity of safety critical data transferred over it. Measures to detect and compensate for any errors introduced into the black channel must be implemented by the safety critical sender and receiver (by software and/or hardware means) to make sure the data retains its integrity.
- blanking cover** A plastic moulding to hide an unused slot in an AADvance base unit.
- boolean** A type of variable that can accept only the values 'true' and 'false'.
- BPCS** Basic process control system. A system which responds to input signals and generates output signals causing a process and associated equipment to operate in a desired manner, but which does not perform any safety instrumented functions with a claimed safety integrity level of 1 or higher.
- Refer to IEC 61511 or to ANSI/ISA—84.00.01—2004 Part 1 (IEC 61511-1 Mod) for a formal definition.
- Equivalent to the Process Control System (PCS) defined by IEC 61508.
- breakdown voltage** The maximum voltage (AC or DC) that can be continuously applied between isolated circuits without a breakdown occurring.
- BS EN 54** A standard for fire detection and fire alarm systems.
- BS EN 60204** A standard for the electrical equipment of machines, which promotes the safety of persons and property, consistency of control response and ease of maintenance.
- bus** A group of conductors which carry related data. Typically allocated to address, data and control functions in a microprocessor-based system.
- bus arbitration** A mechanism for deciding which device has control of a bus.

---

## C

- CIP** Common Industrial Protocol. A communications protocol, formally known as 'CIP over Ethernet/IP', created by Rockwell Automation for the Logix controller family, and which is also supported by the AADvance controller. AADvance controllers use the protocol to exchange data with Logix controllers. The data exchange uses a consumer/producer model.
- clearance** The shortest distance in air between two conductive parts.
- coding peg** A polarization key, fitted to the 9100 processor base unit and to each termination assembly, which ensures only a module of the correct type may be fitted in a particular slot. Part number 9903.
- coil** In IEC 61131-3, a graphical component of a Ladder Diagram program, which represents the assignment of an output variable. In Modbus language, a discrete output value.
- Compiler Verification Tool (CVT)** The Compiler Verification Tool (CVT) is an automatic software utility that validates the output of the application compilation process. This process, in conjunction with the validated execution code produced by the AADvance Workbench, ensures a high degree of confidence that there are no errors introduced by the Workbench or the compiler during the compilation of the application.
- configuration** A grouping of all the application software and settings for a particular AADvance controller. The grouping must have a 'target', but for an AADvance controller it can have only one 'resource'.
- consumer** The consuming controller requests the tag from the producing controller.
- contact** A graphical component of a Ladder Diagram program, which represents the status of an input variable.
- continuous mode** Where the Safety Instrumented Function in the Safety System is continually maintaining the process in a safe state.
- controller** A logic solver; the combination of application execution engine and I/O hardware.
- controller system** One or more controllers, their power sources, communications networks and workstations.
- coverage** The percentage of faults that will be detected by automated diagnostics. See also 'SFF'.
- creepage distance** The shortest distance along the surface of an insulating material between two conductive parts.

**cross reference** Information calculated by the AADvance Workbench relating to the dictionary of variables and where those variables are used in a project.

## D

**data access (DA)** An OPC data type that provides real-time data from AADvance controllers to OPC clients.

**de-energize to action** A safety instrumented function circuit where the devices are energized under normal operation. Removal of power de-activates the field devices.

**dictionary** The set of internal input and output variables and defined words used in a program.

**discrepancy** A condition that exists if one or more of the elements disagree.

**DITA** Digital input termination assembly.

**DOTA** Digital output termination assembly.

## E

**element** A set of input conditioning, application processing and output conditioning.

**energize to action** A safety instrumented function circuit where the outputs and devices are de-energized under normal operation. Application of power activates the field device.

**EUC** Equipment Under Control. The machinery, apparatus or plant used for manufacturing, process, transportation, medical or other activities.

**expansion cable assembly** A flexible interconnection carrying bus signals and power supplies between AADvance base units, available in a variety of lengths. Used in conjunction with a cable socket assembly (at the left hand side of a base unit) and a cable plug assembly (at the right hand side of a base unit).



---

## F

<b>fail operational state</b>	A state in which the fault has been masked. See 'fault tolerant'.
<b>fail safe</b>	The capability to go to a pre-determined safe state in the event of a specific malfunction.
<b>fault reset button</b>	The momentary action push switch located on the front panel of the 9110 processor module.
<b>fault tolerance</b>	Built-in capability of a system to provide continued correct execution of its assigned function in the presence of a limited number of hardware and software faults.
<b>fault tolerant</b>	The capability to accept the effect of a single arbitrary fault and continue correct operation.
<b>fault warning receiving station</b>	A centre from which the necessary corrective measures can be initiated.
<b>fault warning routing equipment</b>	Intermediate equipment which routes a fault warning signal from the control and indicating equipment to a fault warning receiving station.
<b>field device</b>	Item of equipment connected to the field side of the I/O terminals. Such equipment includes field wiring, sensors, final control elements and those operator interface devices hard-wired to I/O terminals.
<b>fire alarm device</b>	A component of a fire alarm system, not incorporated in the control and indicating equipment which is used to give a warning of fire — for example a sounder or visual indicator.
<b>fire alarm receiving station</b>	A centre from which the necessary fire protection or fire fighting measures can be initiated at any time.
<b>fire alarm routing equipment</b>	Intermediate equipment which routes an alarm signal from control and indicating equipment to a fire alarm receiving station.
<b>function block diagram</b>	An IEC 61131 language that describes a function between input variables and output variables. Input and output variables are connected to blocks by connection lines. See 'limited variability language'.
<b>functional safety</b>	The ability of a system to carry out the actions necessary to achieve or to maintain a safe state for the process and its associated equipment.

## G

**group** A collection of two or three input modules (or two output modules), arranged together to provide enhanced availability for their respective input or output channels.

## H

**hand-held equipment** Equipment which is intended to be held in one hand while being operated with the other hand.

**HART** HART (Highway Addressable Remote Transducer) is an open protocol for process control instrumentation. It combines digital signals with analogue signals to provide field device control and status information. The HART protocol also provides diagnostic data. (For more details of HART devices refer to the HART Application Guide, created by the HART Communication Foundation, and their detailed HART specifications. You can download documents from [www.hartcomm.org](http://www.hartcomm.org).)

**high demand mode** Where the Safety Instrumented Function in the Safety System only performs its designed function on a demand, and the frequency of demands is greater than one per year.

**hot swap** See live insertion.

## I

**I/O base unit** A backplane assembly which holds up to three I/O modules and their associated termination assembly or assemblies in an AADvance controller. Part number 9300. See 'I/O module' and 'termination assembly'.

**I/O module** A collation of interfaces for field sensors (inputs) or final elements (outputs), arranged in a self-contained and standardized physical form factor.

**IEC 61000** A series of international standards giving test and measurement techniques for electromagnetic compatibility.

**IEC 61131** An international standard defining programming languages, electrical parameters and environmental conditions for programmable logic controllers. Part 3, which is entitled 'Programming Languages', defines several limited variability languages.

**IEC 61508** An international standard for functional safety, encompassing electrical, electronic and programmable electronic systems; hardware and software aspects.

- IEC 61511** An international standard for functional safety and safety instrumented systems (SIS) for the process industry, encompassing electrical, electronic and programmable electronic systems, hardware and software aspects.
- indicator** A device which can change its state to give information.
- input (Workbench variable)** In the context of an AADvance Workbench variable, this term describes a quantity passed to the Workbench from a controller.
- instruction list** An IEC 61131 language, similar to the simple textual language of PLCs. See 'limited variability language'.
- integer** A variable type defined by the IEC 61131 standard.
- IXL** IXL stands for ISaGRAF eXchange Layer. This is the communication protocol between ISaGRAF based components.

## K

- key connector** The receptacle on the AADvance controller for the program enable key. A 9-way 'D' type socket, located on the 9100 processor base unit.

## L

- ladder diagram** An IEC 61131 language composed of contact symbols representing logical equations and simple actions. The main function is to control outputs based on input conditions. See 'limited variability language'.
- LAN** Local area network. A computer network covering a small physical area, characterised by a limited geographic range and lack of a need for leased telecommunication lines.
- live insertion** The removal and then reinsertion of an electronic module into a system while the system remains powered. The assumption is that removal of the module and reinsertion will cause no electrical harm to the system. Also referred to as 'hot swap'.
- low demand mode** Where the Safety Instrumented Function only performs its designed function on demand, and the frequency of demands is no greater than one per year.

## M

- manual call point** A component of a fire detection and fire alarm system which is used for the manual initiation of an alarm.
- Modbus** An industry standard communications protocol developed by Modicon. Used to communicate with external devices such as distributed control systems or operator interfaces.
- Modbus object** A representation of the configuration settings for a Modbus master or for its associated slave links, within the AADvance Workbench. The settings include communication settings and messages.
- module locking screw** The AADvance latch mechanism seen on the front panel of each module and operated by a broad, flat-blade screwdriver. Uses a cam action to lock to the processor base unit or I/O base unit.

## N

- NFPA 85** The Boiler and Combustion Systems Hazards Code. Applies to certain boilers, stokers, fuel systems, and steam generators. The purpose of this code is to contribute to operating safety and to prevent uncontrolled fires, explosions and implosions.
- NFPA 86** A standard for Ovens and Furnaces. Provides the requirements for the prevention of fire and explosion hazards in associated with heat processing of materials in ovens, furnaces and related equipment.
- NFPA 87** The code for recommended practice for fluid heaters. This code provides safety guidance in order to minimize fire and explosion hazards in Type F, Type G and Type H fluid heating systems including the control system and related equipment.

## O

- on-line** The state of a controller that is executing the application software.
- OPC** A series of standards specifications which support open connectivity in industrial automation.
- output (Workbench variable)** In the context of an AADvance Workbench variable, this term describes a quantity passed from the Workbench to a controller.

---

## P

- peer to peer** A Peer to Peer network consists of one or more Ethernet networks connecting together a series of AADvance and/or Trusted controllers to enable application data to be passed between them.
- pinging** In Modbus communications, sending the diagnostic Query Data command over a link and by receiving a reply ensuring that the link is healthy and the controller is able to communicate with the master. No process data is transferred or modified. In the case of slave devices that will not support pinging then the Standby command will default to Inactive state, but no error will be returned.
- portable equipment** Enclosed equipment that is moved while in operation or which can easily be moved from one place to another while connected to the supply. Examples are programming and debugging tools and test equipment.
- process safety time (PST)** For equipment under control this represents the period of time a dangerous condition can exist without the protection of a safety instrumented system before a hazardous event occurs.
- processor base unit** A backplane assembly which holds all of the processor modules in an AADvance controller. Part number 9100. See also 'processor module'.
- processor module** The application execution engine of the AADvance controller, housed in a self-contained and standardized physical form factor.
- producer** A controller producing a tag to one or more consumers, at the request of the consumers.
- program enable key** A security device that protects the application from unauthorized access and change, in the form factor of a 9-way 'D' type plug. Part number 9906. Supplied with the processor base unit. See also 'key connector'.
- project** A collection of configurations and the definition of the linking between them. See 'configuration'.
- proof test** A test performed at a predetermined frequency which functionally tests all of the components that comprise a Safety Instrumented Function, designed specifically to reveal any undetected failures that may exist so that they can be repaired to ensure that the Safety Instrumented Function continues to meet its designed performance criteria over the entire safety life cycle.
- protocol** A set of rules that is used by devices (such as AADvance controllers, serial devices and engineering workstations) to communicate with each other. The rules encompass electrical parameters, data representation, signalling, authentication, and error detection. Examples include Modbus, TCP and IP.

**PST** Process Safety Time. The process safety time for the equipment under control (denoted PST-EUC) is the period a dangerous condition can exist before a hazardous event occurs without a safety system as a protection.

## R

**real** A class of analogue variable stored in a floating, single-precision 32-bit format.

**redundancy** The use of two or more devices, each carrying out the same function, to improve reliability or availability.

**resolution** The smallest interval measurable by an instrument; the level of detail which may be represented. For example, 12 bits can distinguish between 4096 values.

**RS-232-C, RS-422, RS-485** Standard interfaces introduced by the Electronic Industries Alliance covering the electrical connection between data communication equipment. RS-232-C is the most commonly used interface; RS-422 and RS-485 allow for higher transmission rates over increased distances.

**RTC** Real-time clock.

**RTU** Remote terminal unit. The Modbus protocol supported by the AADvance controller for Modbus communications over serial links, with the ability to multi-drop to multiple slave devices.

## S

**safe state** A state which enables the execution of a process demand. Usually entered after the detection of a fault condition; it makes sure the effect of the fault is to enable rather than disable a process demand.

**safety accuracy** The accuracy of a signal within which the signal is guaranteed to be free of dangerous faults. If the signal drifts outside of this range, it is declared faulty.

**safety-critical state** A faulted state which prevents the execution of a process demand.

**sensor** A device or combination of devices that measure a process condition. Examples are transmitters, transducers, process switches and position switches.

**sequential function chart** An IEC 61131 language that divides the process cycle into a number of well-defined steps separated by transitions. See 'limited variability language'.

**SFF** Safe Failure Fraction. Given by (the sum of the rate of safe failures plus the rate of detected dangerous failures) divided by (the sum of the rate of safe failures plus the rate of detected and undetected dangerous failures).

- SIF** Safety Instrumented Function. A form of process control that performs specified functions to achieve or maintain a safe state of a process when unacceptable or dangerous process conditions are detected.
- SIL** Safety Integrity Level. One of four possible discrete levels, defined in IEC 61508 and IEC 61511, for specifying the safety integrity requirements of the safety functions to be allocated to a safety-related system. SIL4 has the highest level of safety integrity; SIL1 has the lowest.

The whole of an installation (of which the AADvance system forms a part) must meet these requirements in order to achieve an overall SIL rating.

- SNCP** SNCP (Safety Network Control Protocol) is the Safety Protocol that allows elements of an AADvance System to exchange data. SNCP is a SIL 3 certified protocol which provides a safety layer for the Ethernet network making it a "Black Channel".
- SNTP** Simple Network Time Protocol. Used for synchronizing the clocks of computer systems over packet-switched, variable-latency data networks.
- structured text** A high level IEC 61131-3 language with syntax similar to Pascal. Used mainly to implement complex procedures that cannot be expressed easily with graphical languages.
- synchronous** A data communications term describing a serial transmission protocol. A pre-arranged number of bits is expected to be sent across a line per second. To synchronise the sending and receiving machines, a clocking signal is sent by the transmitting computer. There are no start or stop bits.

## T

- TA** See 'termination assembly'.
- target** An attribute of a 'configuration' which describes characteristics of the AADvance controller on which the configuration will run. Includes characteristics such as the memory model and the sizes of variable types for the controller.
- TCP** Transmission control protocol. One of the core protocols of the Internet Protocol suite. It provides reliable, ordered delivery of a stream of bytes from a program on one computer to another program on another computer. Common applications include the World Wide Web, e-mail and file transfer and, for an AADvance controller, Modbus communications over Ethernet.
- termination assembly** A printed circuit board which connects field wiring to an input or output module. The circuit includes fuses for field circuits. The board carries screw terminals to connect field wiring to the controller, and the whole assembly clips onto the 9300 I/O base unit.

**TMR** Triple modular redundant. A fault tolerant arrangement in which three systems carry out a process and their result is processed by a voting system to produce a single output.

**TÜV certification** Independent third party certification against a defined range of international standards including IEC 61508.

## U

**U** Rack unit. A unit of measure used to describe the height of equipment intended for mounting in a standard rack. Equivalent to 44.45mm (1-<sup>3</sup>/<sub>4</sub> inches).

## V

**validation** In quality assurance, confirmation that the product does what the user requires.

**verification** In quality assurance, confirmation that the product conforms to the specifications.

**voting system** A redundant system (m out of n) which requires at least m of the n channels to be in agreement before the system can take action.

## W

**withstand voltage** The maximum voltage level that can be applied between circuits or components without causing a breakdown.



## Rockwell Automation Support

Use the following resources to access support information.

<b>Technical Support Center</b>	Knowledgebase Articles, How-to Videos, FAQs, Chat, User Forums, and Product Notification Updates.	<a href="https://rockwellautomation.custhelp.com/">https://rockwellautomation.custhelp.com/</a>
<b>Local Technical Support Phone Numbers</b>	Locate the phone number for your country.	<a href="http://www.rockwellautomation.com/global/support/get-support-now.page">http://www.rockwellautomation.com/global/support/get-support-now.page</a>
<b>Direct Dial Codes</b>	Find the Direct Dial Code for your product. Use the code to route your call directly to a technical support engineer.	<a href="http://www.rockwellautomation.com/global/support/direct-dial.page">http://www.rockwellautomation.com/global/support/direct-dial.page</a>
<b>Literature Library</b>	Installation Instructions, Manuals, Brochures, and Technical Data.	<a href="http://www.rockwellautomation.com/global/literature-library/overview.page">http://www.rockwellautomation.com/global/literature-library/overview.page</a>
<b>Product Compatibility and Download Center (PCDC)</b>	Get help determining how products interact, check features and capabilities, and find associated firmware.	<a href="http://www.rockwellautomation.com/global/support/pcdc.page">http://www.rockwellautomation.com/global/support/pcdc.page</a>

## Documentation Feedback

Your comments will help us serve your documentation needs better. If you have any suggestions on how to improve this document, complete the How Are We Doing? form at [http://literature.rockwellautomation.com/idc/groups/literature/documents/du/ra-du002\\_-en-e.pdf](http://literature.rockwellautomation.com/idc/groups/literature/documents/du/ra-du002_-en-e.pdf).

Rockwell Automation maintains current product environmental information on its website at <http://www.rockwellautomation.com/rockwellautomation/about-us/sustainability-ethics/product-environmental-compliance.page>.

Allen-Bradley, Rockwell Software, and Rockwell Automation are trademarks of Rockwell Automation, Inc.

Trademarks not belonging to Rockwell Automation are property of their respective companies.

Rockwell Otomasyon Ticaret A.Ş., Kar Plaza İş Merkezi E Blok Kat:6 34752 İçerenköy, İstanbul, Tel: +90 (216) 5698400

**[www.rockwellautomation.com](http://www.rockwellautomation.com)**

### Power, Control and Information Solutions Headquarters

Americas: Rockwell Automation, 1201 South Second Street, Milwaukee, WI 53204-2496 USA, Tel: (1) 414.382.2000, Fax: (1) 414.382.4444

Europe/Middle East/Africa: Rockwell Automation NV, Pegasus Park, De Kleetlaan 12a, 1831 Diegem, Belgium, Tel: (32) 2 663 0600, Fax: (32) 2 663 0640

Asia Pacific: Rockwell Automation, Level 14, Core F, Cyberport 3, 100 Cyberport Road, Hong Kong, Tel: (852) 2887 4788, Fax: (852) 2508 1846

Publication ICSTT-RM405H-EN-P - April 2018

Supersedes Publication XXXX-X.XX - Month Year

XXXXXX-XX

Copyright © 2018 Rockwell Automation, Inc. All rights reserved. Printed in the U.S.A.