

TriStation 1131™ Developer's Workbench

TriStation 1131 v4.1

Assembly No. 9750002-001

Information in this document is subject to change without notice. Companies, names and data used in examples herein are fictitious unless otherwise noted. No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of Triconex.

© 2004 Invensys Systems, Inc. All Rights Reserved.

Triconex, Tricon, Trident, TriStation 1131, TriStation MSW, and CEMPLE are trademarks of Invensys plc, its subsidiaries and affiliates. All other brands may be trademarks of their respective owners.

Acknowledgements

Triconex would like to thank the following people for their contributions to this project: Gary Wilkinson, Parrish Lee, Rhonda Seamonds, Doris Alex, Susan Mathewson, Les Powers, Richard Roach, and Pamela Pescara.

Contents

Acknowledgements	ii
Preface.....	xvii
Course Objectives	xvii
TriStation 1131 Installation.....	xvii
Installing the TriStation 1131 Software.....	xviii
Uninstalling the TriStation 1131 Software	xviii
Verifying the TriStation 1131 Installation	xix
Product and Training Information.....	xx
Technical Support.....	xx
Telephone.....	xx
Fax.....	xx
E-mail.....	xx
Chapter 1 Introduction.....	1
Overview	2
TriStation 1131 Developer's Workbench.....	3
New Features in TriStation 1131 v4.1	3
Programmable Logic Controller (PLC)	4
Triconex Controllers.....	5
IEC Standards.....	5
TriStation 1131 Projects.....	7
Parts of an Application	7
Programs	8
Multiple Programs within a Project	9
Function Blocks	10
Functions.....	10
Variables.....	11
Tagnames	11
Data Types	11
Elementary Data Types.....	12
Generic Data Types	13
TriStation Libraries.....	13
Configuration	13
TriStation Programming Languages	14
Function Block Diagram Language.....	14
FBD Language Editor	15

Ladder Diagram Language	15
LD Editor.....	15
Structured Text Language	16
ST Editor.....	16
Cause and Effect Matrix Programming Language (CEMPLE)	17
To Begin the Course	18
Open Existing Project.....	18
TriStation Tools.....	19
Main Menu and Project Toolbar	19
Workspaces.....	19
Logic Sheet.....	20
Application Workspace	21
Declaration Tree.....	22
Implementation Tree.....	22
Controller Workspace	23
Controller Tree.....	23
Configuration Tree	24
Controller and Emulator Panels.....	24
The Project Workspace	25
Help System	26
Lab 1	27
Lab 2	30
Tutorial: Creating a Timer.....	30

Chapter 2 Project Administration49

Developing a Control Strategy	50
Determining Application Type.....	50
Safety Application	50
Control Application.....	50
Safety and Control Application	50
Determining the Number of Programs	52
When to Create Function Blocks	52
When to Create Functions	52
Project Administration	53
Lesson 1: Creating a TriStation 1131 Project.....	54
Specifying User Access	56
Lesson 2: Adding a New User	57
Lesson 3: Changing the Security Level for Privileges	59
Lesson 4: Changing Security Level Names.....	60
Lesson 5: Adding a Project Description.....	61
Specifying Project Options	62
Lesson 6: Specifying Language Options.....	63
Specifying Annotations	64
Annotations Tab	65

General	65
Style.....	66
Default Macros or Text	66
Lesson 7: Specifying Annotation Properties	68
Lesson 8: Specifying Monitor Colors for BOOL Values	69
Setting TriStation 1131 Options	71
Lesson 9: Specifying Directories	72
Lesson 10: Specifying CEM Editor Options	73
Lesson 11: Specifying Drawing Colors	75
Lesson 12: FBD Editor Options	78
Lesson 13: LD Editor Settings	80
Library Documents.....	82
Lesson 14: Creating Libraries	83
Lesson 15: Managing Libraries	86
Lesson 16: Adding Libraries.....	87
Lesson 17: Updating Libraries	88
Lesson 18: Deleting Libraries	89
Lesson 19: Verifying a Library Version	90
Lab 3	91
Chapter 3 Writing Function Block Logic	93
Function Block Diagrams	94
Lesson 20: Creating a User Document.....	96
Lesson 21: Selecting the Sheet Template	100
Lesson 22: Specifying Sheet Title Properties.....	101
Lesson 23: Placing a Function on the Logic Sheet	103
Lesson 24: Specifying Function Properties.....	105
Lesson 25: Placing Variables on the Logic Sheet	108
Declaring Variables	110
Lesson 26: Declaring Program Variables for OR Function	111
Adding Comments	115
Comment Tab.....	115
Style Tab.....	116
Pickup/Drop Tab.....	117
Edit Fields Tab.....	117
Lesson 27: Adding a Comment.....	118
Lesson 28: Using Macros with Comments	120
Lesson 29: Inserting a Network Divider.....	122
Lesson 30: Compiling the Program	123
Lesson 31: Printing Logic Sheets.....	125
Lab 4	127

Chapter 4	Writing Ladder Diagram Logic	133
	Ladder Diagram Logic.....	134
	Contacts.....	135
	Coils	135
	Links	136
	Power Rails.....	136
	Lesson 32: Creating a User Document.....	138
	Lesson 33: Placing Contacts and Coils on the Logic Sheet	140
	Lesson 34: Declaring Contacts and Coils	142
	Lab 5	145
Chapter 5	Writing Structured Text Logic	149
	Structured Text Logic.....	150
	Structured Text Conventions.....	150
	Structured Text Example	151
	Structured Text Constructs	151
	Expressions.....	151
	Operands and Operators	152
	Order of Evaluation.....	153
	Statements.....	153
	Assignment Statements	154
	Conditional Statements.....	154
	Function and Function Block Control Statements	154
	Selection Statements	154
	CASE Statements	155
	Iteration Statements.....	155
	ST Editor	156
	Lesson 35: Creating a User Document.....	157
	Lesson 36: Entering Structured Text Logic	158
	Lesson 37: Compiling the Program.....	161
Chapter 6	Writing Cemple Logic	163
	Writing CEMPLE Logic	164
	Matrix Planning.....	165
	Restrictions and Limitations	165
	Matrix Evaluation Options.....	165
	How a Matrix is Evaluated.....	165
	Using the CEM Editor.....	167
	Matrix	168
	FBD Network	169
	IVariable Detail Table.....	169
	Selecting and Editing Cells in a CEM Program	170
	Displaying and Sizing Cells from the Matrix	171
	Specifying CEM Editor Options	172

Specifying Monitor Colors and Names	173
Specifying CEM Element Options	174
User-Defined Functions and Application-Defined States	175
User-Defined Functions	175
Application States	175
Enabling User-Defining Functions and Application-Defined States	176
Specifying Local Variables, Tagnames, and Constants in a CEM Program.....	176
Specifying Properties in the Variable Detail Table	177
Lesson 38: Creating a CEMPLE Project	178
Lesson 39: Creating a Function	181
Lesson 40: Creating a Function Block	184
Lesson 41: Creating a Simple Matrix.....	186
Lesson 42: Creating a Matrix with Intersection Functions.....	191
Lesson 43: Creating a Matrix with Cause Header Functions	195
Lesson 44: Creating a Matrix with Effect Header Functions	197
Lesson 45: Monitoring an Instance View.....	199
Chapter 7 Tricon and Trident Controller Configuration	203
Tricon Controller Configuration	204
Overview.....	204
Modules.....	204
Chassis	204
Field Wiring Connections	204
Programmer's Workstation	204
System Configuration.....	205
Operating Parameters	206
Setting Operating Parameters	208
Setting the Password	208
Disabling Stop on the Keyswitch.....	208
Disabling Remote Changes to Outputs	209
Allow Disabling of Points.....	209
Lesson 46: Setting Tricon Operation Parameters	210
Allocating Memory	211
Lesson 47: Allocating Memory For Points	212
Allocating Hardware	214
Chassis Power Usage	214
Lesson 48: Determining Chassis Power Usage.....	215
Chassis Window	218
Types of Tricon Modules.....	221
Tricon Main Processor Modules (MP)	221
Tricon Selectable Modules.....	221
Digital Input Modules	221
Digital Output Modules.....	222
Analog Input Modules.....	222

Analog Output Modules.....	222
Pulse Input Modules	222
Pulse Totalizer Input Modules	222
Thermocouple Input Modules	222
Enhanced Intelligent Communication Module (EICM).....	223
Network Communication Module (NCM)	223
Safety Manager Module (SMM)	223
Hiway Interface Module (HCM)	223
Advanced Communication Module (ACM).....	223
Lesson 49: Inserting Tricon Modules	224
Lesson 50: Replacing the Tricon MP Model.....	229
Lesson 51: Configuring a Tricon Pulse Input Module	230
Lesson 52: Specifying a Tricon Thermocouple Module	232
Configuring Tricon Communication with External Devices	234
Lesson 53: Specifying the Tricon Default Connection.....	235
Lesson 54: Configuring Tricon ACM Ports.....	237
Lesson 55: Configuring Tricon EICM Ports	239
Lesson 56: Configuring Tricon HIM Ports	241
Lesson 57: Configuring Tricon NCM Ports.....	242
Lesson 58: Configuring Tricon SMM Ports.....	243
Tricon Time Synchronization	244
Lesson 59: Using Tricon ACM to Synchronize Time.....	245
Lesson 60: Using a Tricon Network Communication Module to Synchronize Time	246
Lesson 61: Using a Tricon NCMG to Synchronize Time	247
Lesson 62: Using a Tricon SMM to Synchronize Time	249
Trident Controller Configuration	250
Overview	250
Modules.....	250
Assembly.....	250
Field Wiring Connections.....	250
Programmer's Workstation	250
System Configuration	251
Operating Parameters.....	252
Lesson 63: Setting Trident Operating Parameters	254
Lesson 64: Allocating Memory for Points	256
Lesson 65: Specifying Trident MP Module Properties.....	257
Lesson 66: Displaying Trident MP Attribute Properties.....	258
Lesson 67: Inserting Trident Modules	259
Lesson 68: Removing Trident Modules.....	260
Lesson 69: Configuring a Trident PI Module	261
Configuring Trident Communication with External Devices	262
Lesson 70: Specifying the Trident Default Connection	263
Lesson 71: Configuring Trident MP Network Ports.....	265

Lesson 72: Configuring Trident MP Serial Ports	266
Lesson 73: Configuring Trident CM Network Ports.....	268
Lesson 74: Configuring Trident CM Serial Ports	270
Lesson 75: Configuring Trident CM Routing	272
Trident Time Synchronization.....	273
Lesson 76: Using a Trident CM to Synchronize Time	274
Lesson 77: Specifying an Alias Number for a Trident Attribute	275
Chapter 8 TriStation 1131 Communication	277
TriStation 1131 Communication.....	278
TCM Connection.....	278
Tricon EICM Connection.....	279
Lesson 78: Setting Tricon EICM Switches for the TriStation 1131 Port.....	280
Lesson 79: Connecting a Tricon EICM to a TriStation 1131 PC.....	281
Lesson 80: Configuring a Tricon EICM Connection	282
Tricon ACM or NCM Connection.....	284
Lesson 81: Installing a NIC Card in a TriStation 1131 PC.....	285
Lesson 82: Installing TCP/IP Protocol on a TriStation 1131 PC	286
Lesson 83: Using Tricon ACM Switches to Set the Node Number.....	287
Lesson 84: Using Tricon NCM Switches to Set the Node Number	289
Changing Tricon ACM or NCM Node Numbers	291
Lesson 85: Directly Connecting a Tricon ACM or NCM to a TriStation 1131 PC 292	
Lesson 86: Connecting a Tricon ACM or NCM Using a Media Converter ..	293
Lesson 87: Configuring a Tricon ACM or NCM Connection	294
Trident MP Connection	296
Lesson 88: Installing an NIC Card in a TriStation PC	297
Lesson 89: Installing DLC Protocol on a TriStation 1131 PC.....	298
Lesson 90: Setting a Trident Node Number with an Address Plug	299
Lesson 91: Directly Connecting a Trident MP to a TriStation 1131 PC.....	300
Lesson 92: Connecting a Trident MP to a TriStation 1131 PC Using a Hub.	301
Lesson 93: Configuring a Trident MP Connection to a TriStation 1131 PC .	302
Trident CM Connection.....	304
Lesson 94: Installing an NIC Card in a TriStation PC	305
Lesson 95: Installing TCP/IP Protocol on a TriStation 1131 PC	306
Lesson 96: Setting a Trident Node Number with an Address Plug	307
Directly Connecting a Trident CM to a TriStation 1131 PC.....	308
Lesson 97: Connecting a Trident CM to a TriStation 1131 PC Using a Hub	309
Lesson 98: Configuring a Trident CM Connection	310
Tricon Printing	313
Effect of Printing on Scan Time	313
Lesson 99: Connecting a Tricon EICM Port to a Printer.....	314
Lesson 100: Configuring a Tricon EICM Port for Printing	315
About Function Blocks for Tricon Printing.....	316

Trident Printing	317
Affect of Printing on Scan Time	317
Devices for Trident Printing	317
Lesson 101: Directly Connecting a Trident CM to Printing Devices.....	319
Lesson 102: Connecting a Trident CM to Printing Devices Using a Hub	320
Lesson 103: Configuring a Trident CM for Printing Devices	321
IP Addresses.....	323
Lesson 104: Using the Default IP Address for TriStation Communication..	324
Lesson 105: Setting an IP Address Using a RARP Server	325
Lesson 106: Setting a Tricon IP Address Using an EICM Connection.....	326
Lesson 107: Setting a Trident IP Address Using an MP Connection	327
Lesson 108: Setting a Trident IP Address Using a CM Connection	328
Lesson 109: Specifying a Trident CM Default Gateway	330
Lesson 110: Specifying a Trident CM for Network Routing	331
Lesson 111: Testing an Ethernet Connection	332

Chapter 9 Application Building and Implementation 333

Application Building and Implementation	334
Application Building.....	334
Implementation.....	334
Emulator Testing.....	334
Controller Testing.....	334
Lesson 112: Building an Application	335
Build Application.....	335
Scan Time	335
Lesson 113: Emulator Testing	338
Lesson 114: Monitoring the Program Execution	339
Controller Testing.....	341
Download All.....	341
Download Change.....	341
Lesson 115: Performing a Downloading All to the Controller.....	342
Lesson 116: Monitoring Variables on the Controller.....	344
Lesson 117: Monitoring the Program Execution	345
Lesson 118: Adding Annotation for Variables	346
Determining the Scan Surplus	346
Positive Scan Surplus	347
Negative Scan Surplus	347
Process Safety Time Requirements	347
Lesson 119: Displaying Hardware Allocation Exceptions.....	348
Maintenance	349
Steps for a Download Change	349
Planning and Controlling Changes.....	350
Making Changes to a Downloaded Application	350
Effects of a Download Change.....	350

IP Address Change	351
I/O Module Changes	351
Effects of a Download All	351
Lesson 120: Disabling (Forcing) Points	353
Lesson 121: Downloading Changes to a Downloaded (Running) Application. 354	
Lesson 122: Downloading a Changed Application to the Controller	355
Chapter 10 Writing Custom Function Blocks	357
Writing Custom Function Blocks	358
Lesson 123: Creating a TriStation 1131 Project	360
Lesson 124: Creating an UPDOWN Counter	361
Lesson 125: Creating a Function Block Using the ST Editor	373
Lesson 126: Creating a Move Function	376
Lesson 127: Creating a Rising Trigger Function Block	378
Lesson 128: Creating a Timer	380
Lesson 129: Invoking an UPDOWN Function Block	384
Lesson 130: Invoking an AVERAGE Function Block	386
Lesson 131: Building the Application	390
Lesson 132: Configuring the Project	391
Lesson 133: Memory Allocation	392
Lesson 134: Allocating Hardware	393
Lesson 135: Running Program Logic	394
Chapter 11 TriStation 1131 Advanced Programming Lessons	397
Lesson 136: Tank Farm	398
Lesson 137: Tank Farm	400
Lesson 138: Tank Farm	402
Appendix A Diagnostic Monitor	405
Steps for Diagnostic Monitoring	406
Installing and Starting the Diagnostic Monitor	407
Setting Up a Network Configuration	408
Types of Network Configurations	408
Creating or Changing a Network Configuration	409
Connecting a Node to a Network	411
Monitoring Controller Hardware	412
Tricon Overview and Chassis Windows	413
Tricon Module Status Screen	414
Trident Overview and IOP Windows	415
Trident Module Status Screen	417
Changing the View of an IOP Window	418
Module Indicator Behavior	418
Understanding External Faults	419

Locating and Correcting External Faults	419
Understanding Internal Faults	420
Locating and Correcting Internal Faults	421
Clearing Faults on All Modules.....	422
Monitoring Output Voter Diagnostics (OVD).....	423
Displaying Firmware Versions	424
Monitoring Controller Status.....	425
Viewing Controller Status	426
Monitoring and Changing the Scan Time	427
Monitoring and Changing the Memory Allocation	427
Refreshing Controller Status	428
Viewing Program Execution Times	428
Collecting System Diagnostic Events	428
Understanding the Diagnostic Message Window	429
Appendix B Triconex OPC Server.....	431
Overview	432
Configuring the OPC Server.....	433
Redundant Configuration	435
Adjusting System Time.....	436
Other OPC Products	437
OPC Data Manager	437
OPC Redundancy Broker	437
Appendix C Sequence of Events Recorder.....	439
Overview of Functionality	440
SOE Setup in TriStation 1131	442
SOE Configuration	442
Defining SOE Block Properties.....	442
Assigning Event Variables to SOE Blocks.....	444
Specifying a Trip Variable	445
SOE Development in TriStation 1131	447
Programming Notes.....	447
SOESTRT (SOE Start)	448
SOESTOP (SOE Stop)	449
SOECLR (SOE Clear).....	450
SOESTAT (SOE Statistics)	451
SOE Recorder Setup	452
Installing SOE Recorder.....	452
Uninstalling SOE Recorder	453
Copying the SOE Definition File	454
Specifying Display Options	454
Specifying Periodic Snapshots.....	455
Specifying Shift Snapshots	456

Specifying Auto Snapshot Options	456
Specifying an Event File Folder	457
Specifying Controllers for Retrieval.....	458
Specifying Automatic Export of Event Data.....	459
SOE Recorder Event Retrieval	461
Setting Up Event Retrieval	461
Retrieving Events	462
Turning Off Event Retrieval.....	463
Appendix D Triconex DDE Server	465
Overview	466
Configuring the DDE Server Application.....	466
Configuring Triconex Host Information	467
Configuring Server Properties for 802.2 Protocol (Tricon Only)	470
Testing a TCP/IP Connection.....	471
Configuration Requirements for DDE Network Redundancy	471
Using TCP/IP Protocol	471
Using 802.2 Protocol (Tricon Only)	472
Requesting Data with a DDE Client Application	473
Requesting Network Status.....	473
Monitoring Responses from the Controller	474
Changing View Options	474
DDE Menu Commands.....	474
Appendix E Peer-to-Peer Communication	477
Overview	478
Peer-to-Peer Data Transfer Time.....	480
Estimating Memory for Peer-to-Peer Data Transfer Time.....	480
Configuring Peer-to-Peer Ports for Tricon.....	482
Configuring Peer-to-Peer Ports for Trident.....	483
Allocating Peer-to-Peer Memory.....	485
Using Send and Receive Function Blocks	486
Send and Receive Function Blocks.....	486
Sample Send and Receive Pair.....	486
Restrictions on Data Transmission Speed.....	488
Monitoring Peer-to-Peer Communication	489
Status of Communication Paths	489
Status of Net1 Ports for Tricon.....	489
Status of Ethernet (Net1 and Net2) Ports for Trident.....	489
Using the SYS_CM_STATUS Function Block.....	490
Using the Trident CM System Attributes.....	490
Sample Peer-to-Peer Programs	492
Fast Send to One Controller	492
Sending Data Every Second to One Controller	492

Sending Data Only When Requested	492
Fast Send of Safety-Critical Data	492
Sample Timing Calculations	493

Appendix F Modbus Protocol 495

Overview	496
Message Response Time	497
Determining Message Response Time	497
Modbus Functions and Scan Time	498
Modbus Messages	499
Communication Modes	499
Function Names and Aliases	500
Modbus Message Formats	501
Sample Query and Response Messages	503
Modbus Message Lengths	504
Modbus Functions	505
Read Coil Status Function (Function 01)	506
Read Input Status (Function 02)	507
Read Holding Registers (Function Code 03)	508
Read Input Registers (Function Code 04)	509
Force Single Coil (Function Code 05)	510
Preset Single Register (Function Code 06)	511
Read Exception Status (Function Code 07)	512
Loop-Back Diagnostic Test (Function 08)	513
Force Multiple Coils (Function Code 15)	514
Preset Multiple Registers (Function Code 16)	515
Modbus Alias Range	516
Transmission Errors and Exception Conditions	518
Transmission Errors	518
Exception Conditions	519
Exception Responses	520
Exception Response Codes	521

Appendix G Triconex Trilogger Event 523

Overview	524
About This Manual	524
Reference Documents	524
Product Overview	525
Typical TriLogger Installations	527
Installation	528
Minimum Hardware Requirements	528
Minimum Software Requirements	528
Theory of Operation	529
Tricon Bins	529

Performance	530
Events	530
Configuration	532
Operation	552
Log	552
Trigger	553
Event List	554
Show Data	554
Scan Time	555
Remote	556
Log Files	557
Appendix H Invensys® Wonderware® InTouch®	559
Dynamic Data Exchange (DDE)	560
Wonderware SuiteLink™	560
InTouch I/O Addressing	561
InTouch Access Names	562
Defining an I/O Item in InTouch	566
Defining an I/O Type Tagname	566
Index	571

Preface

TriStation 1131 Developer's Workbench Training Manual provides information and procedures to perform the basic tasks necessary to use TriStation 1131. Included in this manual are:

- High-level programming concepts
- Detailed information about TriStation 1131 features and tools
- Informative tips and tricks
- Procedures to effectively use the TriStation 1131 Workbench

Course Objectives

After completing this course, you will be able to:

- **Use TriStation 1131 Tools** – Use available tools to create, build, and maintain a TriStation 1131 project.
- **Write Program Logic** – Write program logic in each of the IEC 1131-3 standard compliant languages used in TriStation 1131, as well as CEMPLE.
- **Build Configurations** – Test program logic and connect program variables to Tricon point connections.
- **Perform Tricon Operations** – Download configurations to the Tricon controller, perform routine maintenance, and troubleshoot.
- **Administer Project Options** – Set project and TriStation 1131 options, use audit trail, display project history, and manage security.
- **Generate Reports** – Document a project, use variable annotation, and produce reports for the database.

TriStation 1131 Installation

This section explains how to install and uninstall the **TriStation 1131** software, and to verify that the software is correctly installed. The installation also installs the Triconex Diagnostic Monitor and TS1131 Install Check software.

For information on the Triconex Diagnostic Monitor, see the Help documentation for the software.

Installing the TriStation 1131 Software

This procedure explains how to install the **TriStation 1131** software. The setup program provided by Triconex installs all the components of the **TriStation 1131** Developer's Workbench on your PC. If you purchased the optional CEMPLE software, it is installed at the same time.

Before Starting

If you have previously installed the **TriStation 1131** software on your PC, you must uninstall it before installing a new version of the software.

Procedure

- 1 Log on as an administrator or as a user with administrator privileges.
- 2 Close all open applications.
- 3 Insert the **TriStation 1131** CD in the CD-ROM drive.
If the installation starts automatically, go to [step 7](#). Otherwise, go to the next step.
- 4 From the **Start** menu, click **Settings**, and then click **Control Panel**.
- 5 Double-click **Add/Remove Programs**.
- 6 On the **Install/Uninstall** tab of the **Add/Remove Programs Properties** dialog box, click **Install**.
- 7 Follow the InstallShield Wizard instructions.
Triconex recommends installing the **TriStation 1131** software in the default destination folder, which is: C:\Program Files\Triconex.
- 8 To complete the installation, click **Finish**.
- 9 Restart your PC before running the **TriStation 1131** software.

Uninstalling the TriStation 1131 Software

This procedure explains how to uninstall the TriStation 1131 software. If you have installed a previous version of the software, you must uninstall it before installing the new version.

Procedure

- 1 Log on as an administrator or as a user with administrator privileges.
- 2 From the **Start** menu, click **Settings**, and then click **Control Panel**.
- 3 Double-click **Add/Remove Programs**, and select **TriStation 1131**.
- 4 Click **Change/Remove**.
The **Confirm File Deletion** dialog box asks you to confirm the deletion of the selected application and all its components.
- 5 Click **Yes** to remove the previous version of **TriStation 1131**.

Note If you saved projects in the default directory, (C:\Program Files\Triconex\TriStation 1131 4.0\Programs), the uninstall program *does not* remove them.

- 6 Click **Yes** or **Yes to All** if the **Remove Shared File** dialog box asks about removing unused DLLs.
- 7 When the *Uninstall Successfully Completed* message appears, click **OK**.
You can now install a new version of the **TriStation 1131** software.

Verifying the TriStation 1131 Installation

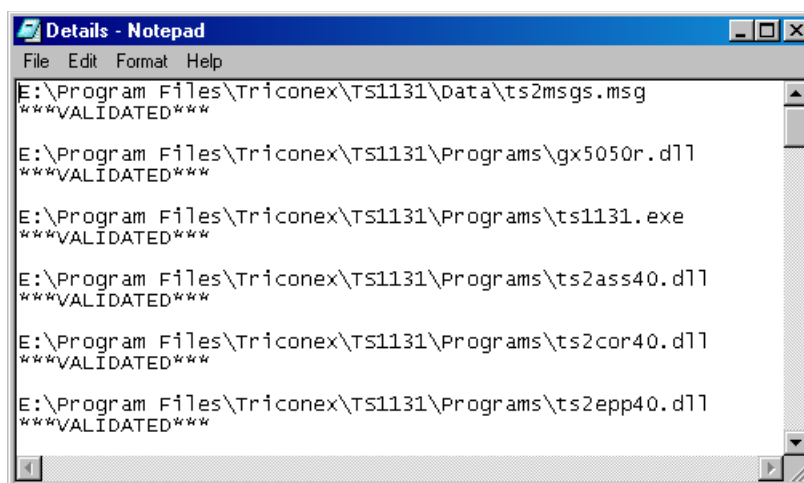
This procedure explains how to verify the **TriStation 1131** software is correctly installed and that associated files are not corrupted. After installing the software and before downloading an application to the controller, you should run the **TriStation 1131** Install Check program. The Install Check software is copied to your hard drive when you install the **TriStation 1131** software.

Note Running TS1131 Install Check is required for safety applications.

For more information, see the *Safety Considerations Guide*.

Procedure

- 1 From the **Start** menu, select **Programs**, **Triconex**, and then **Install Check**.
- 2 Click **Run**.
- 3 Click **Display Details** and verify that the program is validated by viewing each item in the list.



Product and Training Information

To obtain information about Triconex products and in-house and on-site training, see the Triconex Web site or contact your regional customer center.

Web Site

<http://www.triconex.com>

Technical Support

Customers in the U.S. and Canada can obtain technical support from the Customer Support Center (CSC) at the numbers below. International customers should contact their regional support center.

Requests for support are prioritized as follows:

- Emergency requests are given the highest priority
- Requests from participants in the System Watch Agreement (SWA) and customers with purchase order or charge card authorization are given next priority
- All other requests are handled on a time-available basis

If you require emergency or immediate response and are not an SWA participant, you may incur a charge. Please have a purchase order or credit card available for billing.

Telephone

Toll-free number 866-746-6477, or

Toll number 508-549-2424 (outside U.S.)

Fax

Toll number 508-549-4999

E-mail

ips.csc@ips.invensys.com

Introduction

Overview	2
TriStation 1131 Developer's Workbench	3
Programmable Logic Controller (PLC)	4
Triconex Controllers	5
TriStation 1131 Projects	7
TriStation Programming Languages	14
TriStation Tools	19
Help System	26

Overview

The **TriStation 1131™** Developer's Workbench Training Manual provides you with information to create controller program logic and download the applications to Tricon and Trident controllers.

This course includes:

- Introduction
- Project Administration
- Writing Function Block Diagram Logic
- Writing Ladder Diagram Logic
- Writing Structured Text
- Writing CEMPLE Logic
- **TriStation 1131** Communication
- Controller Configuration
- Implementation
- Writing Custom Function Block Logic
- Advanced Exercise

Also included are the following Appendixes:

- Diagnostic Monitor
- OPC Server
- Sequence of Events Recorder
- DDE Server
- Peer-to-Peer Communication
- Modbus Protocol
- Trilogger Event
- Wonderware In Touch®

TriStation 1131 Developer's Workbench

TriStation 1131™ Developer's Workbench, Version 4, is a software application for developing, testing and documenting safety-critical and process-control applications.

You can create programs, functions, and function blocks in IEC 61131-3-compliant languages, using:

- Standard Library (STDLIB) complies with the IEC 61131-3 standard
- Triconex Library (TCXLIB) for all Triconex platforms
- Tricon Library (TRILIB or TX1LIB) specifically for Tricon platforms
- Trident Library for Trident platforms

A comprehensive set of tools enable you to generate a project containing all the configuration data required to execute an application in the Tricon and Trident controllers. A separate Diagnostic Monitor allows you to monitor the status of hardware and applications running on multiple controllers.

CEMPLE™ (Cause and Effect Matrix Programming Language Editor) is an optional **TriStation 1131** language editor used with the Developer's Workbench to create a programs based on a cause and effect matrix. Cause and effect methodology is commonly used in the process control industry to define alarms, emergency shutdown systems, and mitigation actions.

TriStation 1131 runs under Windows NT and Windows 2000 operating systems and follows the Microsoft Windows graphical user interface guidelines.

New Features in TriStation 1131 v4.1

These are the new features in **TriStation 1131** version 4.1:

- Microsoft Windows 2000 and Windows XP compatible
- Setup for new Analog Input and Digital Output Modules
- Setup for new Tricon Communication Module
- Backup project file is automatically saved after a Download All or Download Change
- Restore Project To Last Download command – allows the file which was saved after the download to be restored as the project file
- Write to File feature for intermediate ST code

Programmable Logic Controller (PLC)

A programmable logic controller is a device that stores instructions and then implements operations, such as sequencing, counting, and timing, to automate machines and other processes. It is designed to replace physical mechanisms by simulating the functions they perform.

For example, if you wanted to turn a switch on for 15 seconds, and then turn it off, you can attach an external timer to the switch. But on complicated equipment that has multiple switches, attaching an external timer to each switch would be impractical. Instead, you can write a program for a PLC instructing it to scan the status of the switches, and depending on their states, turn them on or off using a timer function block.

The most common applications for PLCs are:

- Relays
- Counters
- Timers

A PLC is a microprocessor, made up of a CPU, memory, input circuits, and output circuits. It scans or evaluates the inputs, stores the information in its memory, executes the instructions in the program, and then updates the output. It continually repeats this scan cycle.

The Tricon/Trident scan cycle is made up of four steps.

Steps

- 1 The PLC performs an internal check of its hardware.
- 2 The PLC scans inputs and saves the information in its memory.
- 3 The PLC evaluates the information and executes the program, one step at a time. It then updates the outputs.
- 4 The PLC copies the output from the memory to the output circuits. The circuits send the operation command to the device.

Programmable logic controllers are flexible, cost-effective devices for controlling complex operations and systems.

Triconex Controllers

Triconex Controllers are based on triple modular redundant (TMR) architecture, using three isolated, parallel control systems integrated into one set of hardware. This provides a high level of system fault tolerance. Fault tolerance is the ability to detect transient and steady-state error conditions, and to take appropriate corrective action on-line.

The system consists of three identical channels (except for the Power Modules which are dual redundant). An application is downloaded and sent over separate communication paths to three main processors in the controller. The logic is scanned and executed by each of the three channels in parallel with the other two. The channels then perform a two out of three (2003) vote of the output data before sending the controlling operation instructions.

Because each path is separate, no single-point failure in one channel can affect another. If a hardware failure occurs in one channel, the other channels override the faulty one. Repairs can be made to the faulty channel while the PLC continues to process programs without interruption. The system then reconfigures itself to full TMR operation. The triple redundancy architecture increases safety, reliability, and critical control.

IEC Standards

IEC-61131-3 is a set of standards for process control software. A programmer can develop an algorithm, called a "Project," using any combination of the following programming languages:

- Function Block Diagram (FBD)
- Ladder Design (LD)
- Structured Text (ST)

The standard specifies the syntax, semantics, and display of programming languages. By using standard languages, users can read, review, develop, or maintain the programs for equipment from all vendors that are standard-compliant.

A project is made up of reusable entities called "program organization units" (POUs), that include:

- Programs
- Functions
- Function blocks

POUs are elements organized in a sequence that is then executed by the controller. When a controller using IEC 61131-3 compliant languages receives a program that includes these elements, it executes the program instructions in order, sending the control operations to the machine.

Each POU has two parts:

- Declaration – The name and data type
- Instruction – Pre-defined programming code

POUs can be used repeatedly in a program and stored in a custom or user-defined library. These collections of elements can be imported and exported as libraries for use in other projects.

Each language is a standard language, allowing a user familiar with standard languages to read, review, develop, or maintain the programs for equipment from all vendors that implement the standard.

The IEC Standard includes a library of POUs that function as pre-defined programming code. As a developer, using IEC Standard ensures uniform programming.

The benefits of using the IEC 61131-3 standards include:

- Reusable software
 - Reducing the need for developing new software
 - Re-using proven, tested function blocks and functions
 - Reducing programming costs
 - Speeding up development
- Error Detection Early in Development Cycle
 - Enabling logic to be tested and corrected before runtime

TriStation 1131 Projects

The first step in developing an application for process control is to create a new project. A project is a database that contains all of the elements for the application. Each project file has the extension PT2.

Parts of an Application

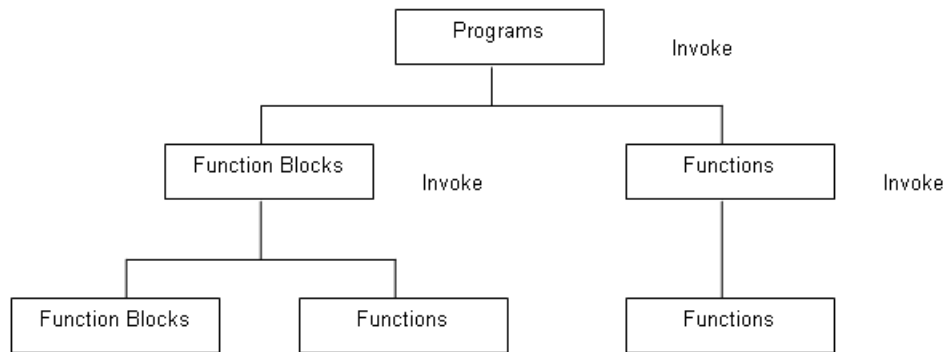
An application is made up of elements (programs, functions, and function blocks) that are connected or sequenced in a specific order. These elements form the logic that runs the PLC operation. An application also includes the libraries used to create the logic, and controller Functions and Function Block

The main elements for an application include:

- Programs
- Function blocks
- Functions
- Variables
- Tagnames
- Data types
- Libraries
- Configuration

Programs

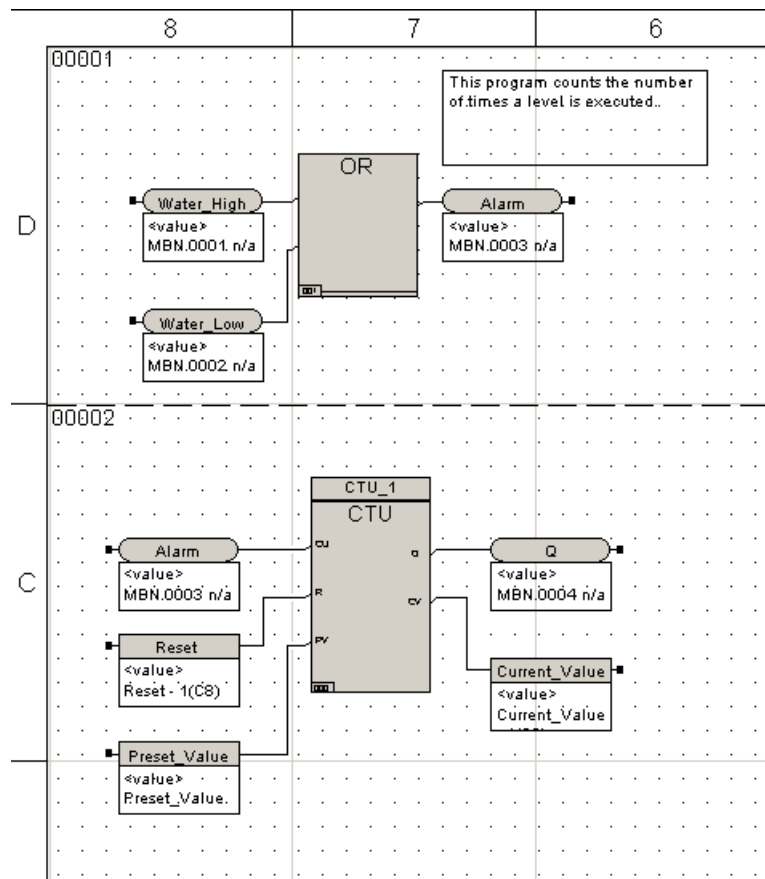
A program is the highest level program organization unit (POU) within a project. It is an assembly of functions and function blocks that provide the logic for the commands executed by the controller. In each program, you can define a maximum of 2,000 variables (inputs, outputs, and locals).



A program can invoke functions and function blocks but cannot invoke another program.

If more than one program is needed for an operation, then multiple programs are created and executed in a specific order. Programs are initiated from the Execution List.

This is an example of a program logic sheet for a water tank alarm. If the water level is too high or too low, an alarm goes off.



The program is divided into two networks. In the first network, an input connects to a sensor (OR function), to detect if the water level is too high. A second input detects if the water level is too low. The output is an alarm if the water level is too high or too low.

In the second network, a count up function block (CTU) counts the number of times the output alarm goes True. When the current value output equals the preset value, then the Q output goes to True.

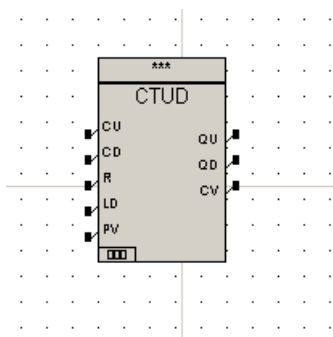
Multiple Programs within a Project

A typical **TriStation** project is made up of multiple programs based on the operation of the different units in the controlled process. Designing a project to break up program logic into smaller routines is called partitioning. Developing a series of small sets of logic results in efficient, reusable routines. For example, the control strategy for a process area within an oil refinery could be partitioned into separate programs that control crude oil distillation towers, furnaces, hydro-desulphurization areas, and redistillation areas.

Function Blocks

A Function Block is a pre-programmed calculation that can have one or more inputs and multiple outputs. A function block retains the values calculated during one evaluation for use in the next evaluation. For certain operations, a set of input values may not yield the same output values. For example a count up function block, may have an input value of one, but because it is counting up, the output value will be increasingly higher than the input.

This is graphic representation of a count up (CTU) function block.



Each input (CU) increases the value of output (Q) by one, until the maximum pre-set value (PV) is reached or exceeded.

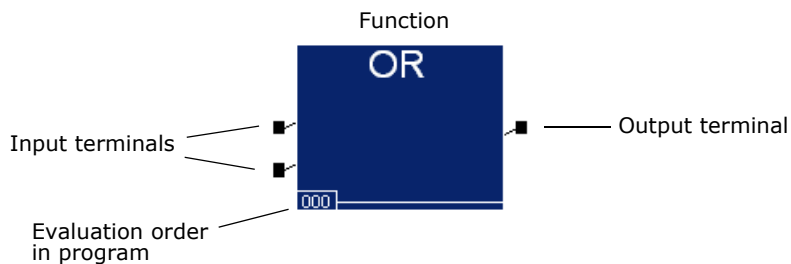
You can also create a user-defined function block. This is helpful when writing logic to control a number of identical process units, especially if the amount of logic required is small. You can use a maximum of 400 variable (input, outputs) and locals) in each function block.

Functions

A function is a pre-programmed calculation that can have one or more inputs but returns only one result. Unlike a function block, values in a function are not retained from one evaluation or scan to the next. The values in a function exist only during each evaluation. You can use a maximum of 400 variables in each function.

You can also create a user-defined function and invoke it each time it is needed. The functions are specific to that project, but can be imported to other projects. This reduces the amount of memory required for the program logic, as well as makes the program easier to read.

In FBD and LD languages, each function or function block is graphically represented on a logic sheet. This is an example of an OR function.



Variables

In a program, function, or function block, a variable represents a value. In a controller, a variable refers to a named area in the memory that stores the value. All variables used in a project must be declared as either local to a program, function, or function block, or global to the project.

A local variable means that the value is temporary and ends when a certain part of the program ends. For example, input to reset a counter when it reaches 32,767 would be a local variable or integer data type (INT). Upon reaching that number, the program ends and starts over again. The value is reset to zero, the value it had at the beginning of the program.

A global variable means that the value is valid until it is replaced with a different value. The output on a counter that counts the number of times an alarm sounds would be a global variable, the value changing only when another alarm sounds. It is important to remember that in **TriStation 1131**, global variables are called *tagnames*.

Tagnames

Tagnames, also known as global variables, are references to physical locations in the controller memory. Tagnames describe the type of point (input, output, or memory) and properties associated with the point.

Data Types

Data types identify the type of data used in tagnames and local variables. A tagname must be one of three data types:

- BOOL – Boolean variable
- DINT – Double integer
- REAL – Real number

These variables point to hardware addresses in the controller and are accessible to all programs in a **TriStation** project.

Local variables can be any of the data types supported by **TriStation 1131**.

TriStation 1131 uses elementary and generic data types.

Elementary Data Types

An elementary data type is defined by IEC standards and identifies the characteristics and size of the data used in a program, function, or function block, and the operations that can be applied to the data.

This table describes the elementary data types and how they can be used.

Data Type	Description	Tagnames	Constants and Variables
BOOL	A Boolean, 1 bit in length, and has two possible values: false (0) or true (1).	✓	✓
DATE	A specific date expressed as the year, month, and day, 8 bits in length. Syntax: D#CCYY-MM-DD		✓
DINT	A double integer, 32 bits in length.	✓	✓
DT	A specific date and time. To specify the time of day, you can use fractions (FFF) of a second. Values are stored internally in microseconds and displayed in the TriStation 1131 Controller Panel in milliseconds. Syntaxes: DT#CCYY-MM-DD-HH:MM:SSDT#CCYY-MM-DD-HH:MM:SS:FFDATE_AND_TIME#CCYY-MM-DD-HH:MM:SS		✓
DWORD	A double word, 32 bits in length.		✓
INT	An integer, 16 bits in length.		✓
LREAL	A real long number, 64 bits in length and has 15 decimal digits of precision.		✓
REAL	A real number, 32 bits in length and has 6 decimal digits of precision.	✓	✓
STRING	An alphanumeric sequence, up to 132 characters in length, delimited by single quotation marks (')		✓
TIME	A period of time (duration) in days, hours, minutes, seconds, and milliseconds for any operation to take place. The range is ± 9999 years and the precision is 0.1 milliseconds. Syntaxes: T#11d T#22.2h T#33.3m T#44.4s T#55.5ms T#11d22h33m44s55.5ms		✓

Data Type	Description	Tagnames	Constants and Variables
TOD	<p>A specific time of day expressed in hours, minutes, seconds, and fractions (FFF) of a second. The precision is 0.001 seconds.</p> <p>Syntaxes:</p> <p>TOD#HH:MM:SS</p> <p>TOD#HH:MM:SS.FFF</p> <p>TIME_OF_DAY#HH:MM:SS</p>		✓

Generic Data Types

Generic data types are used to organize elementary data types that have similar properties. They can be used with IEC 61131-3 standard functions that supports overloaded inputs and outputs. Overloaded is a variable that can be used for different types of data. Generic data type names use the prefix ANY.

TriStation Libraries

TriStation 1131 provides four standard libraries that contain functions and function blocks you can use for application development:

- Standard Library (STDLIB) complies with the IEC 61131-3 standard
- Triconex Library (TCXLIB) for all Triconex platforms
- Tricon Library (TR1LIB or TX1LIB) specifically for Tricon platforms
- Trident Library (TRDLIB) for Trident platforms

You can also create a custom library of functions, function blocks, and data types that can be used in other projects.

Configuration

The controller configuration specifies:

- Operating parameters
- Memory and hardware allocations
- Communication configurations
- Time Synchronization

The controller must be configured before an application can be downloaded and implemented on that controller.

TriStation Programming Languages

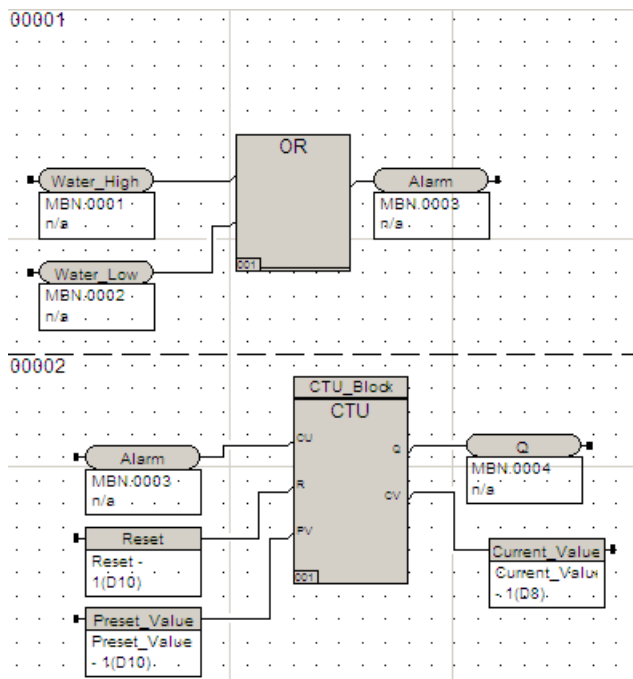
The **TriStation 1131** Developer's Workbench supports three IEC 61131-3 Standard compliant languages and an optional Triconex programming language for developing, testing, and documenting process control applications that execute in the Tricon and Trident controllers:

- Function Block Diagram
- Ladder Diagram
- Structured Text
- Cause and Effect Matrix Programming Language Editor (CEMPLE) for turbomachine applications

Function Block Diagram Language

A Function Block Diagram (FBD) is a graphically-oriented language that corresponds to circuit diagrams. Elements are represented by blocks that are wired together to form circuits. The wires communicate binary and other types of data between FBD elements.

In FBD, a group of elements visibly interconnected by wires is known as a network. An FBD diagram can contain one or more networks. FBD diagrams are always interpreted from the left to the right, and from the top to the bottom.



FBD Language Editor

Function Block Diagrams are created using the FBD Editor. Buttons on the toolbar are used to select and place elements in the workspace.



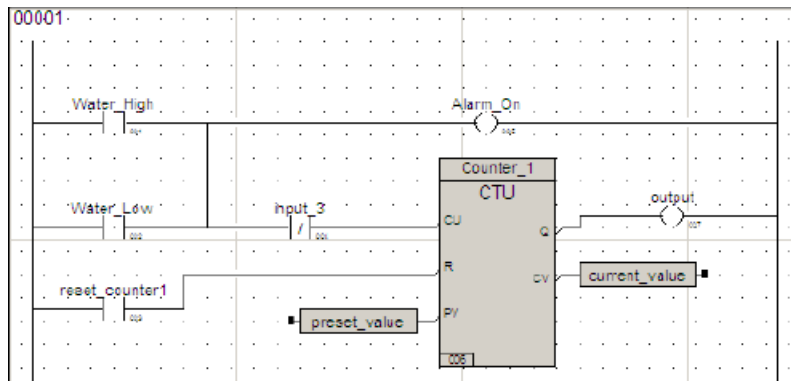
For a description of the toolbar buttons, see *Commands and Properties* in the Developer's Workbench in the Content section of the Help documentation.

Ladder Diagram Language

Ladder Diagram (LD) language is a graphical language that uses a standard set of symbols to represent relay logic. The basic elements are contacts (discrete inputs) and coils (discrete outputs) diagramed like rungs on a ladder, and connected by links. The links are different from the wires used in FBD in that they transfer only binary data between LD symbols, following the power flow characteristics of relay logic.

Function blocks and function elements that have at least one binary input and output can be used in LD diagrams.

This example shows the water tank alarm shown in the previous section, written in LD language.



The two contacts are connected to the left rail, which represents the power rail. The coil is connected to a ground power rail on the right. If either of the contacts is a TRUE state, the "Alarm_On" coil responds with a TRUE state, and sets an alarm.

LD Editor

Ladder Diagrams are created using the LD Editor. Buttons on the toolbar are used to select and place elements in the workspace.



For a description of the toolbar buttons, see *Commands and Properties* in the Developer's Workbench in the Content section of the Help documentation.

Structured Text Language

Structured Text (ST) is a general purpose, high-level programming language, similar to PASCAL or C. It is useful for complex arithmetic calculations, and can be used to implement complicated procedures that are not easily expressed in graphical languages such as FBD or LD.

ST can be used to create Boolean and arithmetic expressions, and conditional statements such as IF...THEN...ELSE.

This example shows the water tank alarm written in ST and performs the operation as the previous FBD and LD examples.

```

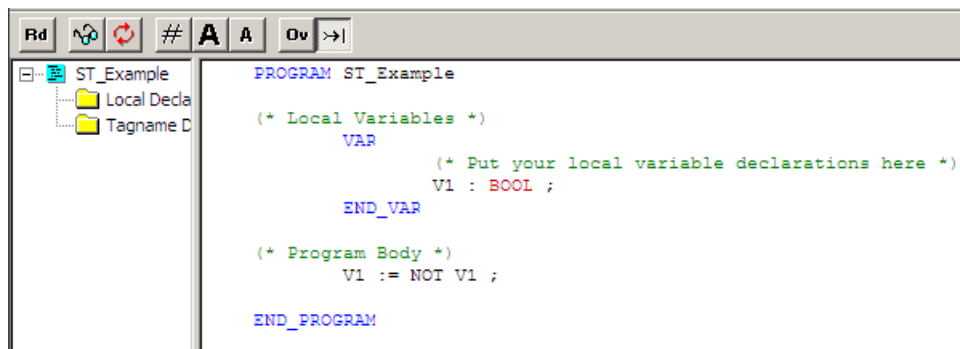
PROGRAM ST1
VAR_EXTERNAL
    Water_High : BOOL ;
    Water_Low : BOOL;
    Alarm : BOOL ;
    Q : BOOL ;
END_VAR
VAR
    Reset : BOOL ;
    Preset_Value : INT ;
    Current_Value : INT ;
    CTU_Block : CTU ;
END_VAR
(* First Network *)
Alarm := Water_High OR Water_Low ;
(* Second Network *)
CTU_Block ( CU := Alarm, R :=Reset, PV :=Preset_Value ) ;
Q := CTU_Block.Q ;
Current_Value := CTU_Block.CV ;
END_PROGRAM

```

The program logic is written between the statements PROGRAM and END_PROGRAM. VAR_INPUT represents the two water level inputs. VAR_OUTPUT sets the alarm if the water level is too high or too low.

ST Editor

The Structured Text (ST) editor allows you to develop programs and functions by writing code. The buttons on the toolbar are used to select editing features. The editor displays a template on the worksheet to help you create the logic in the correct format.



Cause and Effect Matrix Programming Language (CEMPLE)

Cemple™ is an optional **TriStation 1131** language editor that automates the process of creating a program based on a cause and effect matrix. The matrix is then converted to executable program controller code.

Cause and Effect Matrix is a methodology that is commonly used throughout the process industry to define Emergency Shutdown (ESD) strategies. The matrices are frequently used for applications like fire and gas systems in the programming logic is simple, but the volume of inputs and outputs that need to be controlled is high.

CEMPLE allows you to associate a problem in a process with one or more actions that must be taken to correct the problem. The problem is known as a cause and the action is known as an effect.

In a typical matrix, a cause is represented by a row in the matrix and an effect is represented by a column. An X in the intersection of a cause row and an effect column establishes a relationship between the cause and effect.

This example shows a cause and effect matrix controlling five water tank alarms.

			OR	OR	OR	OR	OR
			Effect				
			Description	High level alarm indicator for tank 1	High level alarm indicator for tank 2	High level alarm indicator for tank 3	High level alarm indicator for tank 4
				UNIT_1_ALARM	UNIT_2_ALARM	UNIT_3_ALARM	UNIT_4_ALARM
				UNIT_5_ALARM			
Cause	Description		E01	E02	E03	E04	E05
LEVEL_1_	TRUE=Fluid level in tank 1 is high	C01	X				
LEVEL_2_HI	TRUE=Fluid level in tank 2 is high	C02		X			
LEVEL_3_HI	TRUE=Fluid level in tank 3 is high	C03			X		
LEVEL_4_HI	TRUE=Fluid level in tank 4 is high	C04				X	

Loc	Terminal	Var/Const	VarType	DataType	Description
C01		P1_LEVEL_1_HI	Tagname	BOOL	

To Begin the Course

It is recommended that you open an existing project to view the tools, workspaces, and elements you will be using.

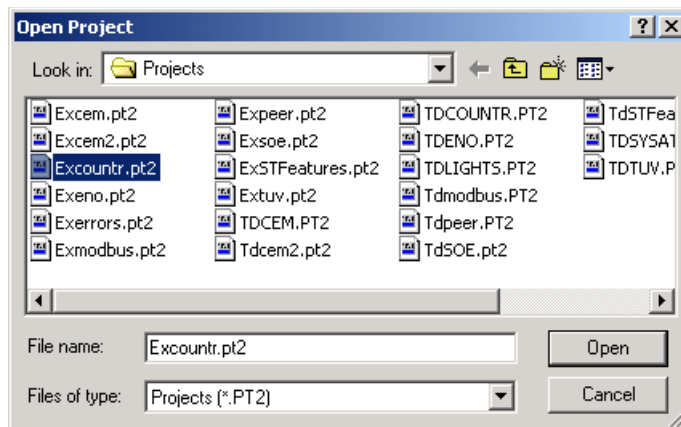
Open Existing Project

A **TriStation** project includes the application and its elements, interface, security, libraries, reports, and configuration settings for a PLC operation.

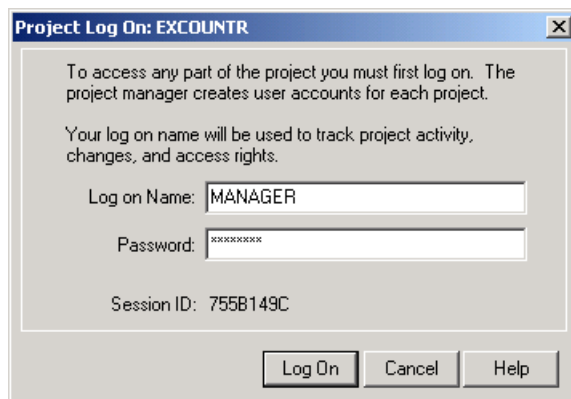
Only one project can be opened at a time.

Procedure

- 1 Open **TriStation 1131**. On the **File** menu, click **Open Project**.



- 2 Click the project name, and then click **Open**.

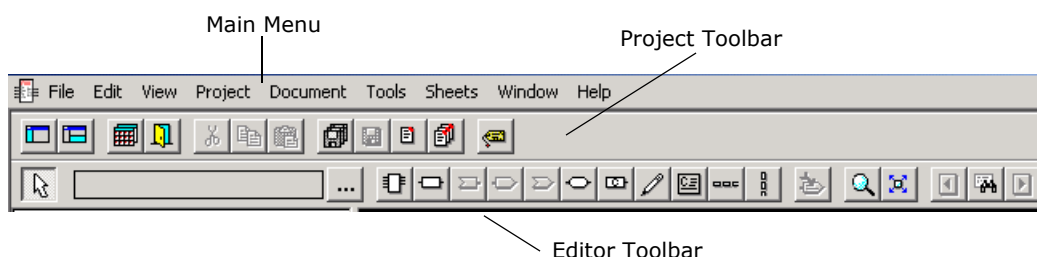


- 3 Enter the default user name **MANAGER**. Enter the default password **PASSWORD**.
- 4 Click **Log On**. The project opens.

TriStation Tools

Main Menu and Project Toolbar

The main menu for an editor or control panel appears above the toolbar. The toolbar has two parts: the project toolbar, which is common to all views of a **TriStation** project, and the toolbar for a specific control panel or editor.



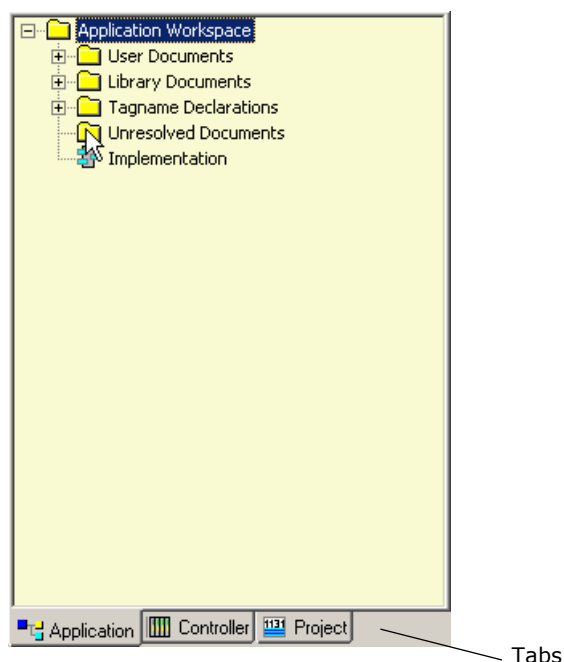
Workspaces

The **TriStation 1131** Developer's Workbench has three workspaces:

- Application
- Controller
- Project

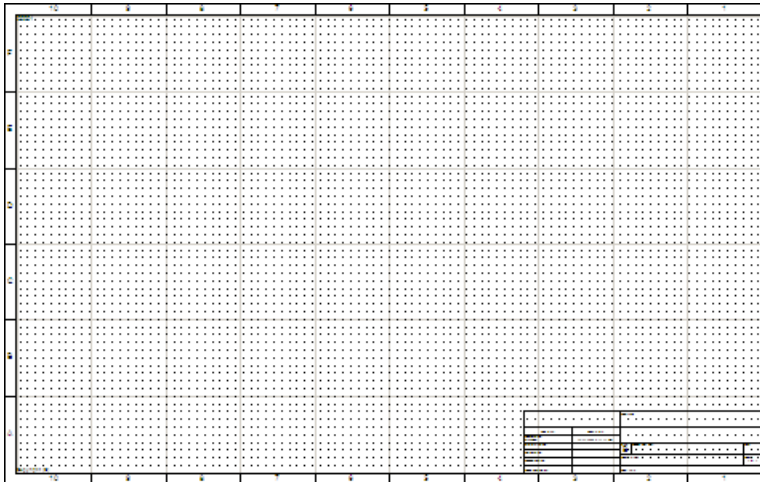
Directory trees list the elements and operations associated with each workspace.

You can change the workspace by clicking the tabs at the bottom of the workspace view.



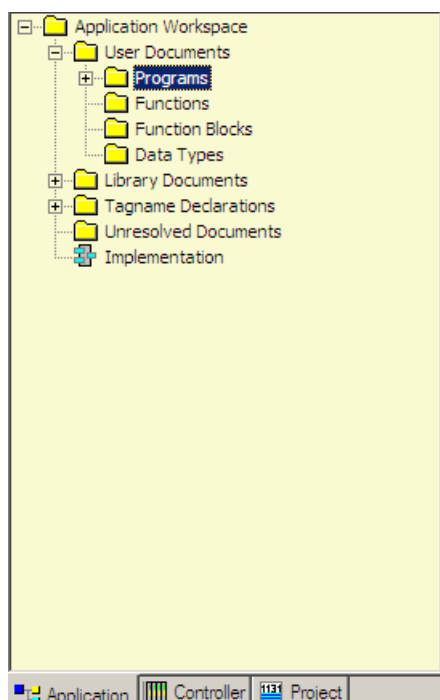
Logic Sheet

The logic sheet is used to visually represent programs in graphics and/or text. It is displayed in each of the workspaces.



Application Workspace

The Application Workspace is used to develop program logic, using the programming languages.

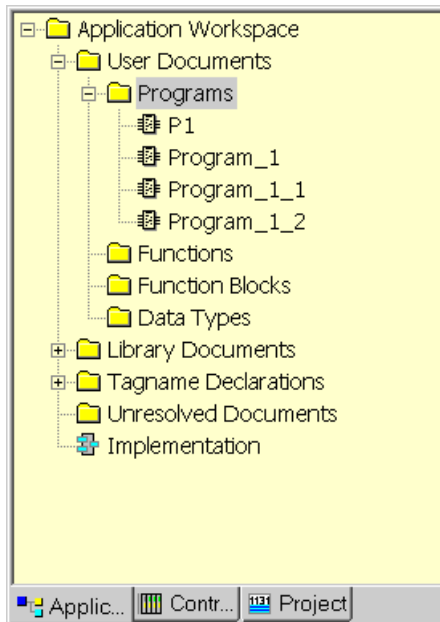


The folders in the directory tree include:

- User Documents – Includes the file folders for programs, function, function blocks, and data types.
- Library Documents – Displays the available functions and function blocks attached to the project.
- Tagname Declarations – Lists the tagnames of each variable used in the program
- Unresolved Documents – Lists elements that are missing and provides a command to find where the missing elements are used.
- Implementation – Lists the Execution List (programs and scan time), SOE (Sequence of Events) and Peer-to-Peer Configuration.

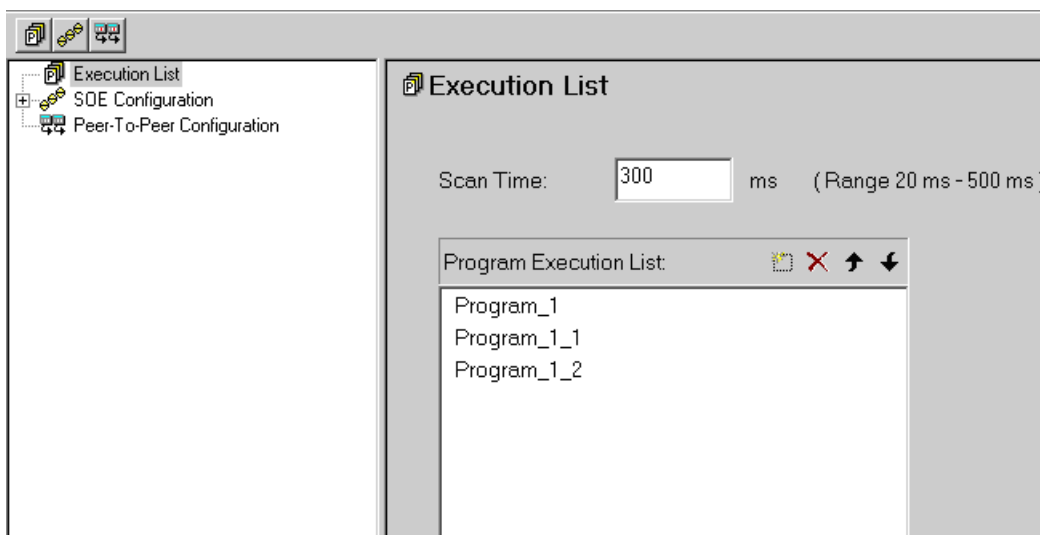
Declaration Tree

The Application workspace can be expanded to list the local and tagname declarations for the currently opened program, function, or function block.



Implementation Tree

The Implementation Tree lists the Execution List (programs and scan time), SOE (Sequence of Events) Configuration, and Peer-to-Peer Configuration.



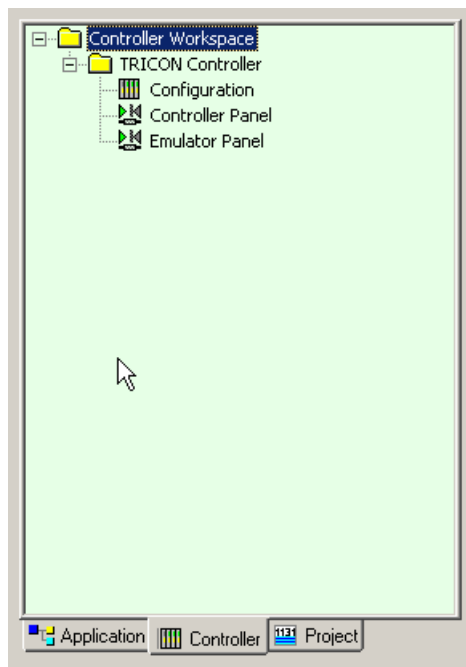
Controller Workspace

The Controller Workspace is used to specify the configuration for the project. It includes:

- The Controller Tree
- The Configuration Tree
- The Controller Panel
- The Emulator Panel

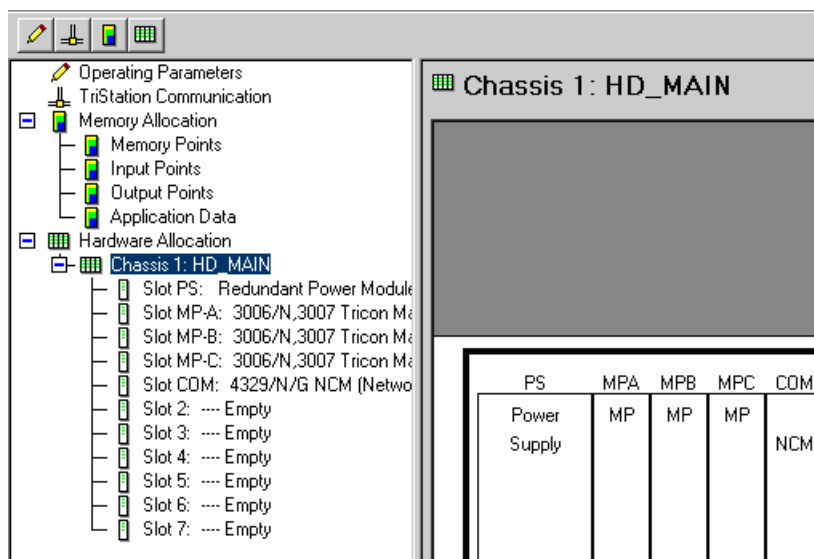
Controller Tree

The Controller tree includes the operating parameters, communication settings, memory and hardware allocation that can be configured. When the application is built, this information is required. The tree also includes the Controller and Emulator Panels used to emulate and run an application.



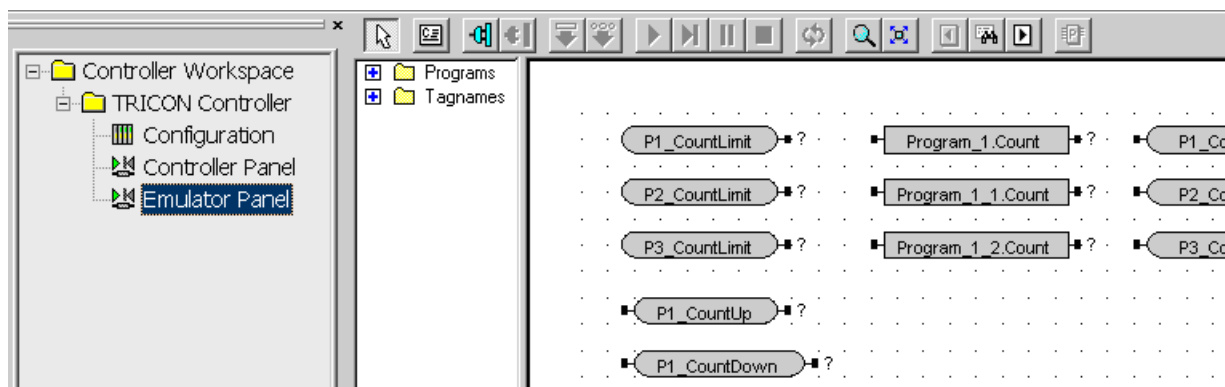
Configuration Tree

The Configuration Tree includes operating parameters (Tricon only), communication, memory allocation, hardware allocation, and Control and Status Attributes (Trident only).



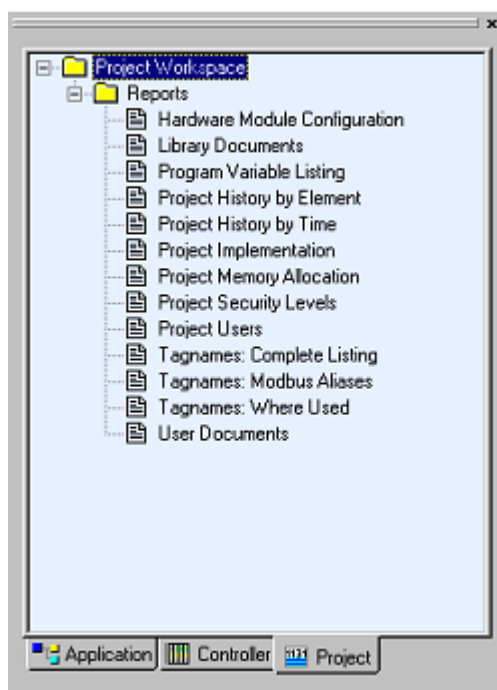
Controller and Emulator Panels

The Controller and Emulator Panels display the programs either running in emulation or in the controller.



The Project Workspace

The Project Workspace is used to view and print reports related to the project. You can also export reports in a variety of formats that can be saved to disk or sent to an email address.

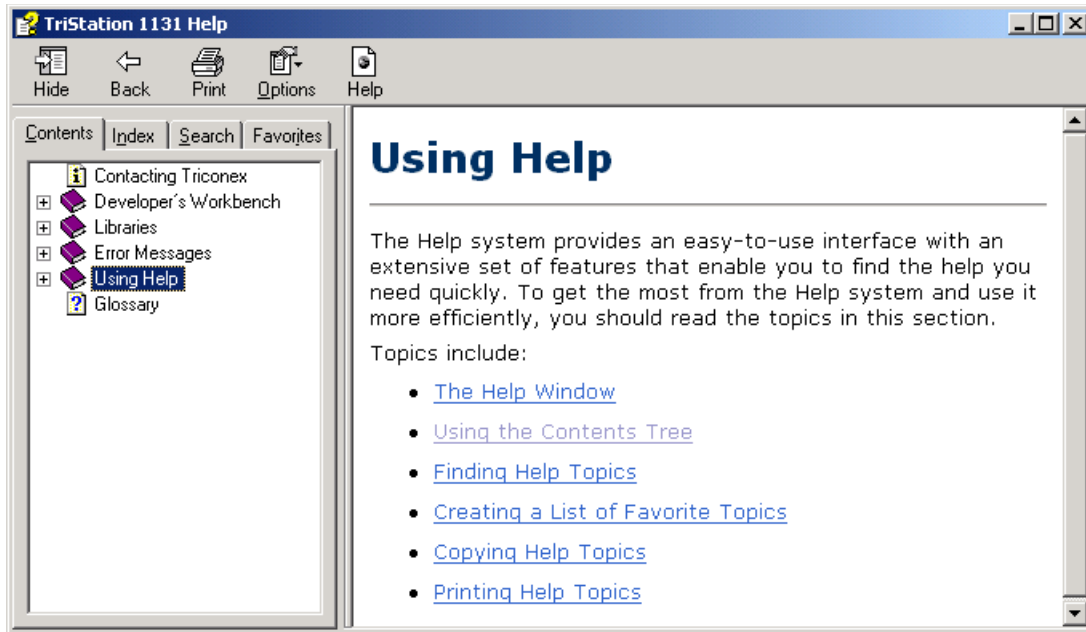


Help System

The **TriStation 1131** Help system provides an extensive set of features that enable you to find information you need quickly.

There are two ways to access the Help system:

- Clicking the **Help** menu on the toolbar
- Pressing the **F1** key



The toolbar at the top provides display, printing, and other options.

The navigation tabs include:

- Contents – An outline of all topics in the Help system
- Index – Displays topics by looking for keywords
- Search – Searches for any word or phrase in the Help system
- Favorites – Creates an easily accessible list of your favorite Help topics

Information is displayed in the Topic Viewer. Links to other topics or web pages are displayed in blue, underlined text and can be accessed by clicking the link.

Topics with additional related information can be accessed by clicking the Related Topics button.

You have completed Chapter 1. Close the sample project by clicking **Close Project** on the **File** menu.

Lab 1: Help System

Use the Help System to define the following:

- 1 Project
- 2 Program
- 3 Function Block
- 4 Function
- 5 Global Variable
- 6 Local variable
- 7 Tagname
- 8 Document
- 9 Application
- 10 Memory Point
- 11 Library
- 12 User-Defined Library
- 13 Alias
- 14 Data Type

15 BOOL

16 DINT

17 REAL

18 Constant

19 Function Block Diagram

20 Ladder Diagram

21 Structured Text

22 CEMPLE

23 Download All

24 Download Change

25 Scan

26 TMR

27 Fault Tolerance

28 De-energize to Trip

29 IEC 61131-3

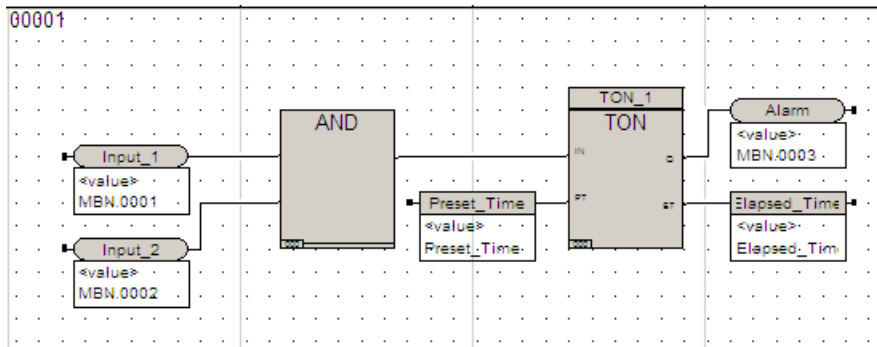
30 TÜV Rheinland

Lab 2: Creating a Timer

Review what you have learned by completing the tutorial.

Tutorial: Creating a Timer

This is a tutorial for creating a Timer. An alarm is set after a delay of a preset length of time.



Inputs are connected to an AND function. The output of the AND function is connected to the IN terminal of an on-delay (TON) function block. The TON function block delays setting the alarm by a Preset Time (PT).

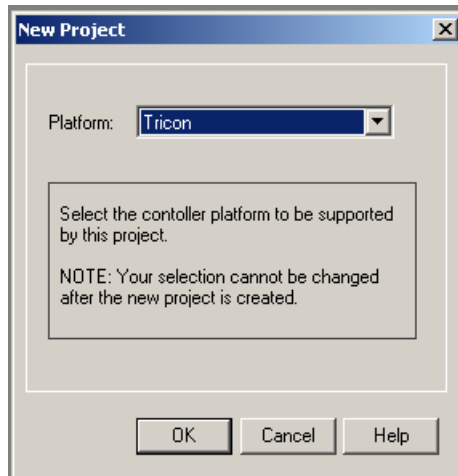
When both inputs to the AND gate go true, the Output is true. The true output from the AND function to the IN input of the timer starts the Elapsed Time (ET). When the elapsed time reaches the time specified by the PT input, the output Q goes true, and the alarm is set. If at any time the input IN changes to false, the timer resets, changing Q to false, and the elapsed time is reset to zero.

You will:

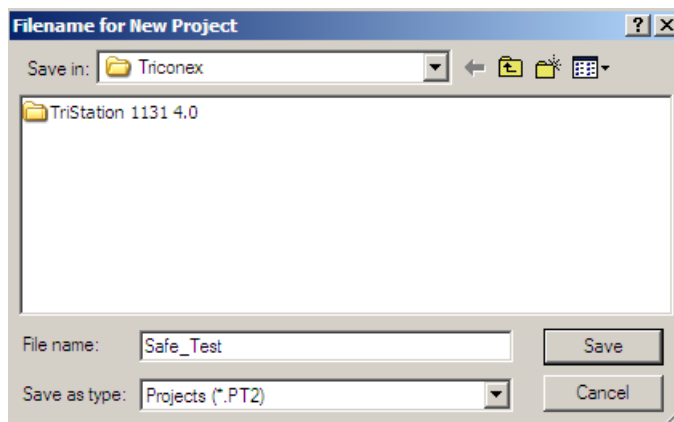
- Create the logic as a Function Block Diagram
- Compile the program and correct any errors
- Test the program in the Emulator

Create Function Block Diagram

- 1 On the **File** menu, click **New Project**.



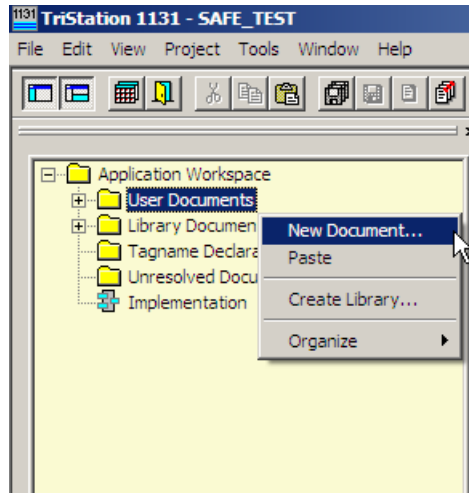
- 2 Select **Tricon** as the platform, and then click **OK**.



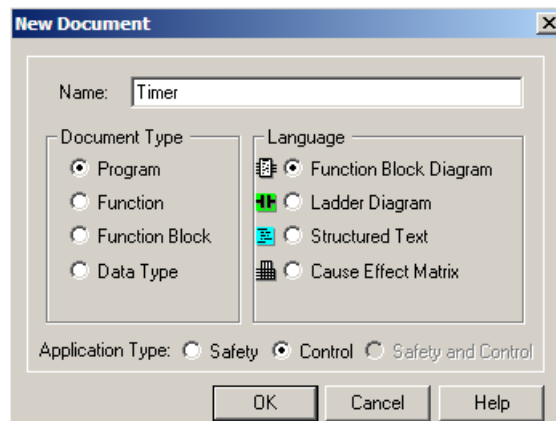
- 3 Enter **Safe_Test** as the filename of the project. Click **Save**.
The libraries are attached to the project and the Application workspace is displayed.

Create a New Program

- 1 In the **Application** tree, right-click **User Documents**, and then click **New Document**.

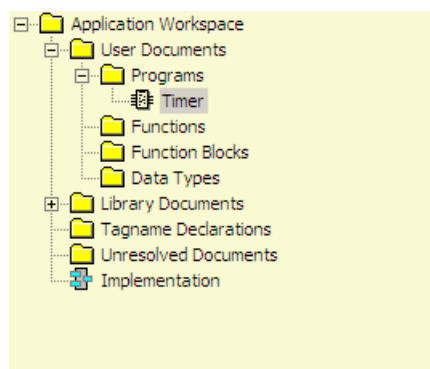


The **New Document** screen is displayed.



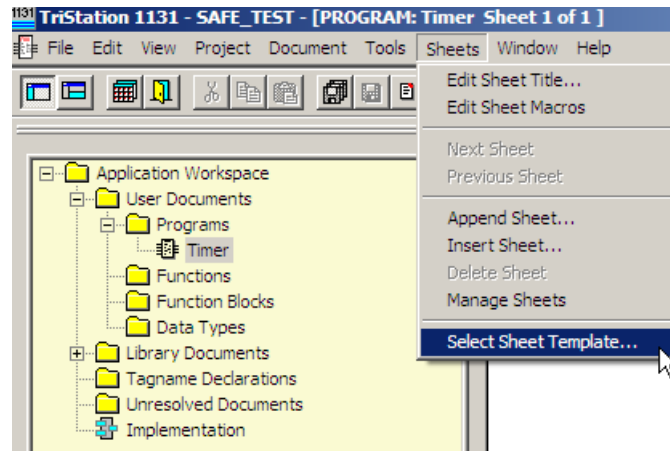
- 2 Enter **Timer** for the filename. Keep the defaults: **Program**, **Function Block Diagram**, and **Control**. Click **OK**.

The **Application** tree displays the timer function under the **Programs** folder.

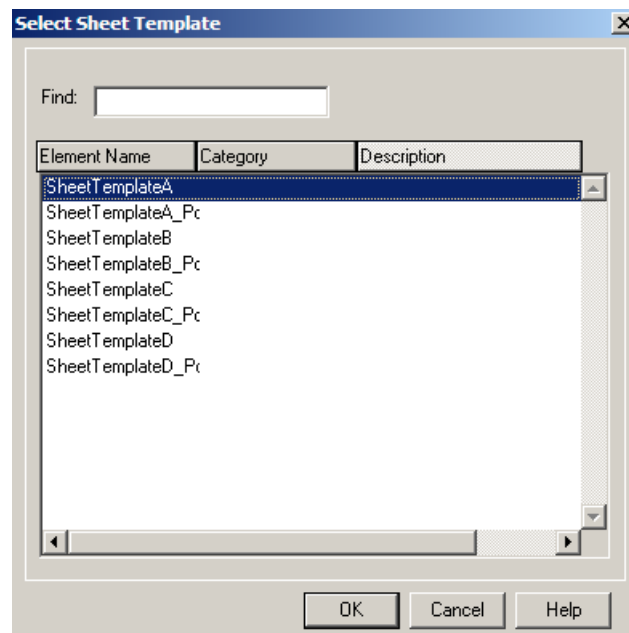


Setup Worksheet

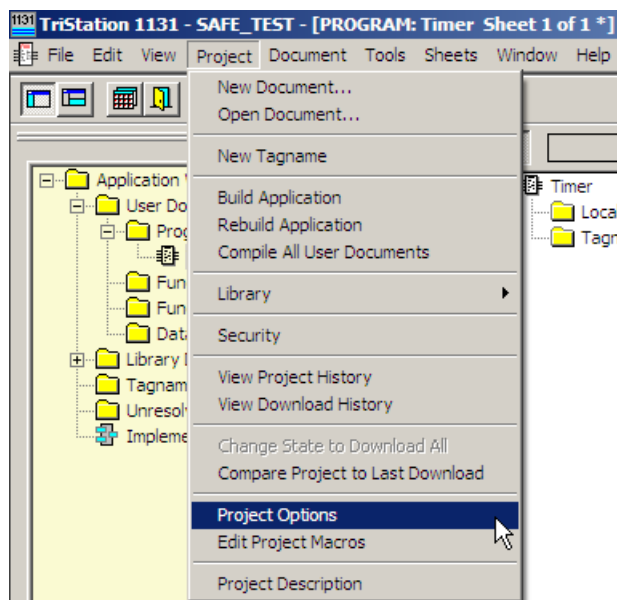
- 1 On the **Sheets** menu, click **Select Sheet Template**.



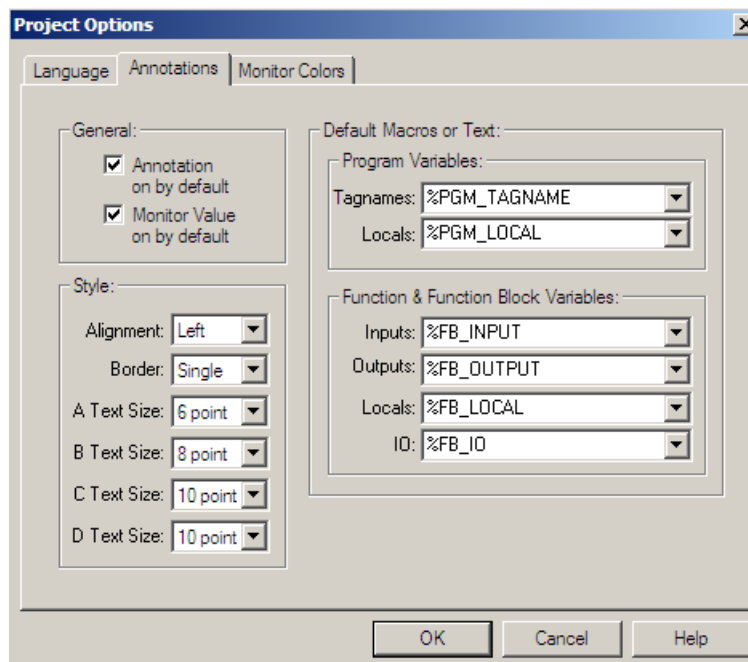
- 2 Click **SheetTemplateA**, and then click **OK**.



- 3 On the **Project** menu, click **Project Options**.

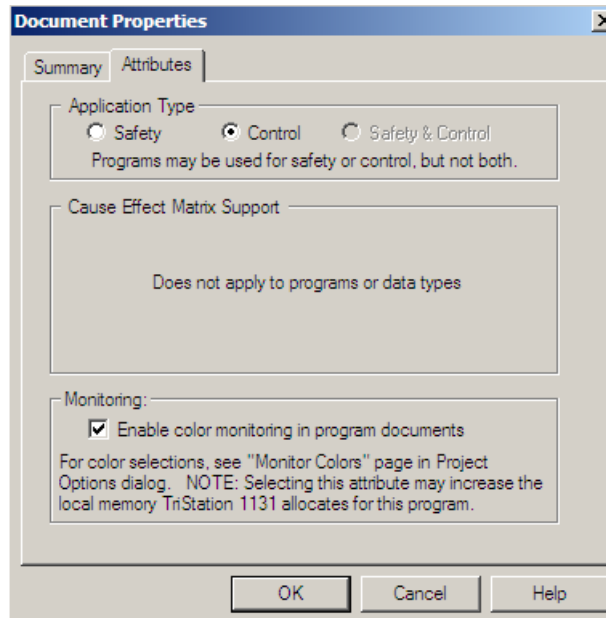


The **Project Options** screen displays.




- 4 Click the **Annotation** tab. Check **Annotation on by Default** and **Monitor Value on by Default**, and then click **OK**.

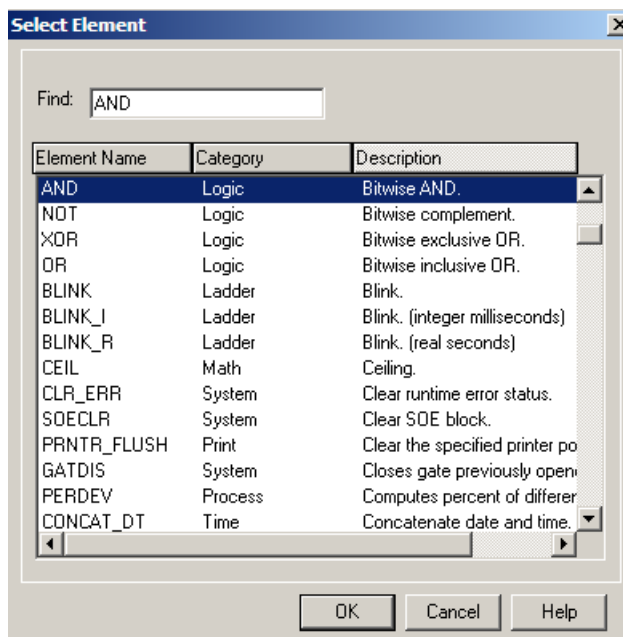
- 5 On the **Document** menu, click **Properties**. The **Document Properties** screen is displayed.



- 6 Click the **Attributes** tab. Check **Enable color monitoring in program documents**, and then click **OK**.

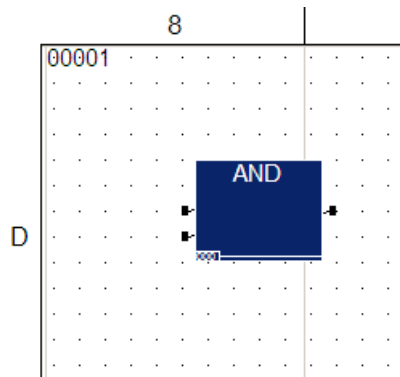
Place the AND Function on Logic Sheet

- 1 Click the **Select Function (Block) Tool** button  on the toolbar. The **Select Element** screen is displayed.

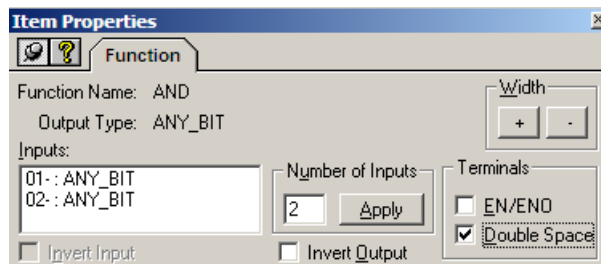


- 2 In the **Find** field, enter **AND**, and then click **OK**.

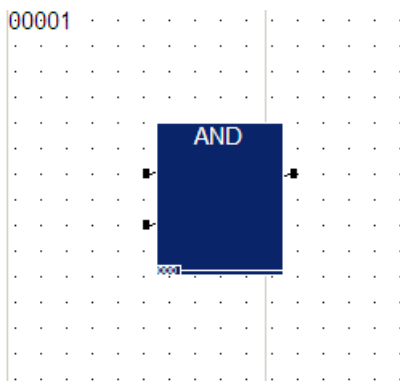
- 3 Move the pointer to the logic sheet. The pointer changes to the function graphic. Click once to place the function on the logic sheet.




- 4 Double-click the AND function. The **Item Properties** screen is displayed.

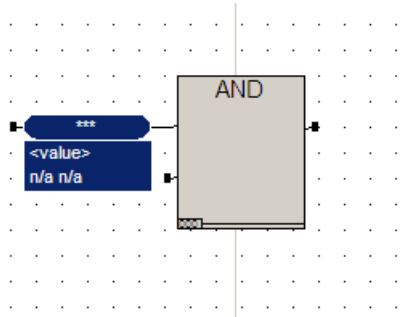


- 5 Check **Double-space**. This increases the size of the element on the sheet, which makes it easier to connect variables. Close the **Item Properties** screen.

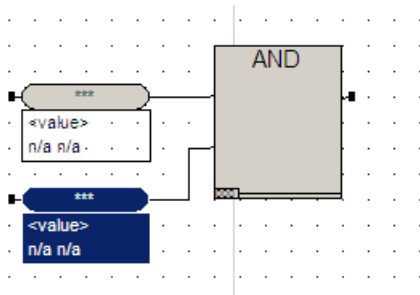


Place Tagnames on Logic Sheet

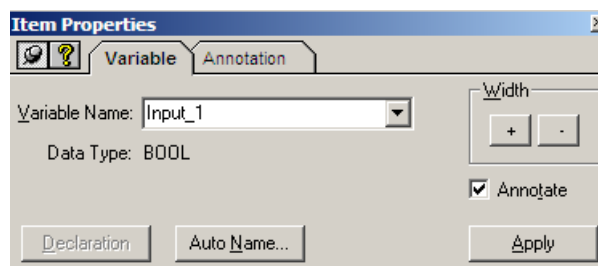
- 1 Click the **Tagname Tool** button  and move the pointer to the logic sheet. Connect it to the top input terminal of the **AND** function by clicking once.



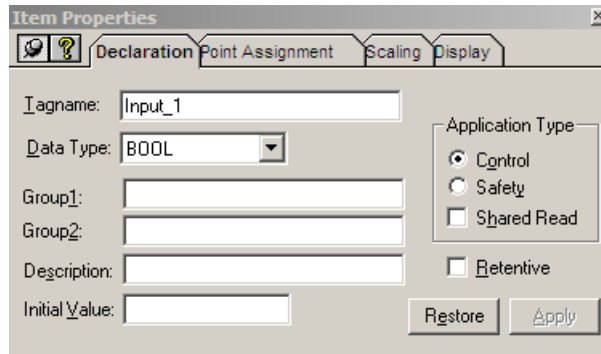
- 2 Click the **Tagname Tool** button and move the pointer to the logic sheet. Connect it to the bottom input terminal of the **AND** function by clicking once.



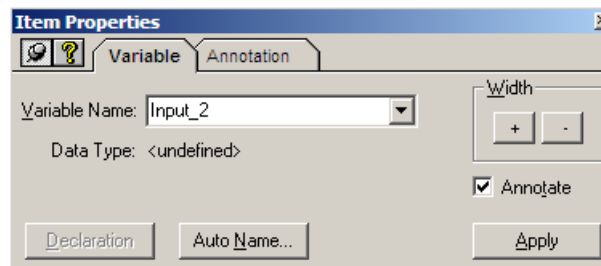
- 3 Double-click the top tagname to display the **Item Properties** screen.



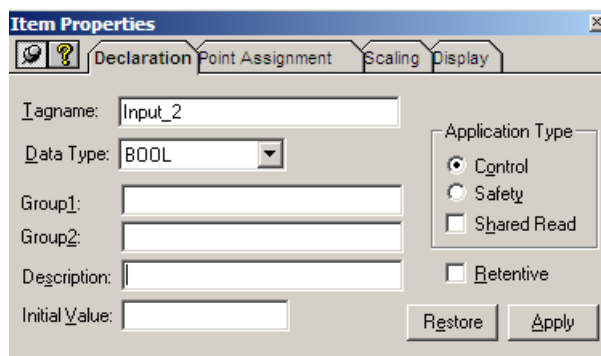
- 4 Enter **Input_1** in the **Variable Name** field. Click **Apply**. The **Declaration** tab is displayed.



- 5 Enter **SampleText** in the **Description** field. Confirm **BOOL** as the Data Type and **Control** as the Application Type. Click **Apply**, and then close the **Declaration** tab.
- 6 Double-click the second tagname to display the **Properties** screen.




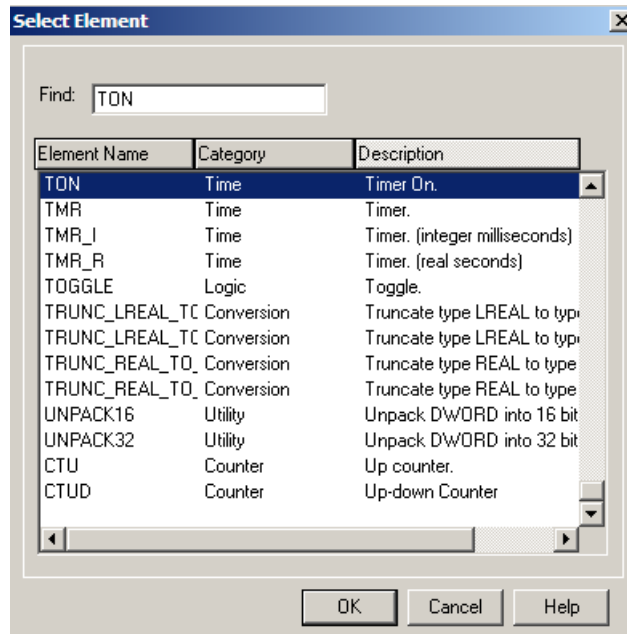
- 7 Enter **Input_2** in the **Variable Name** field. Click **Apply**. The **Declaration** tab is displayed.



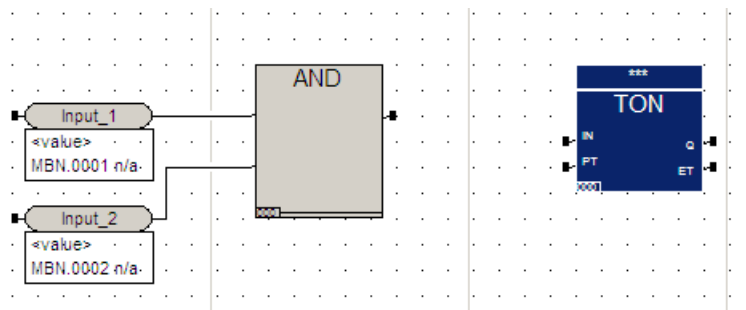
- 8 Enter **SampleText** in the **Description** field. Click **Apply**, and then close the **Declaration** tab.

Place TON Function Block on Logic Sheet:

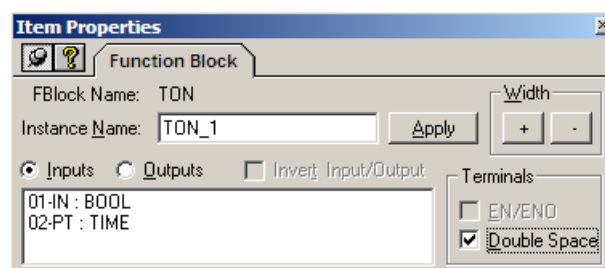
- 1 Click the **Select Function (Block) Tool** button  on the toolbar. The **Select Element** screen is displayed.



- 2 In the **Find** field, enter **TON**, and then click **OK**.
- 3 Place the **TON** function block on the logic sheet to the right of the **AND** function.

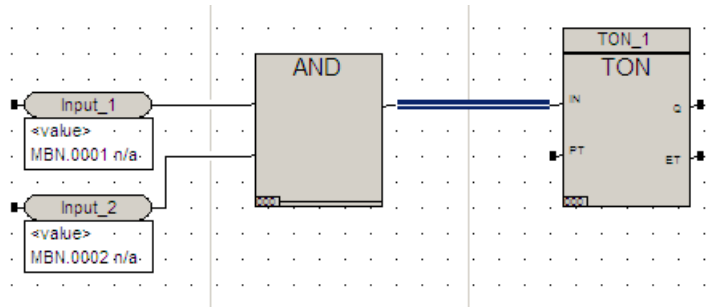


- 4 Double-click **TON** to display the **Item Properties** screen.




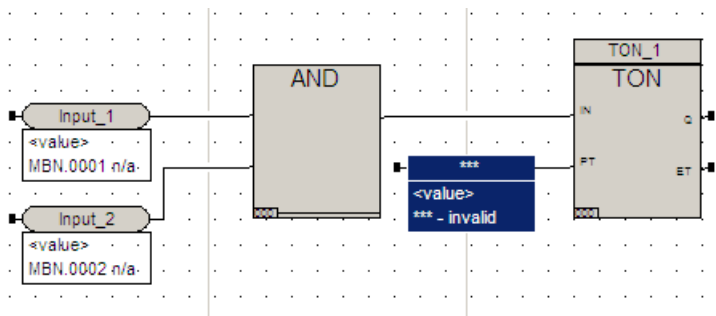
- 5 Enter **TON_1** the **Instance Name** field. Click **Apply**.
- 6 Check **Double Space**, and then close the **Item Properties** screen.

- 7 Connect the output terminal of the **AND** function to the **IN** input terminal of the **TON** function block by drawing a line between the terminals.

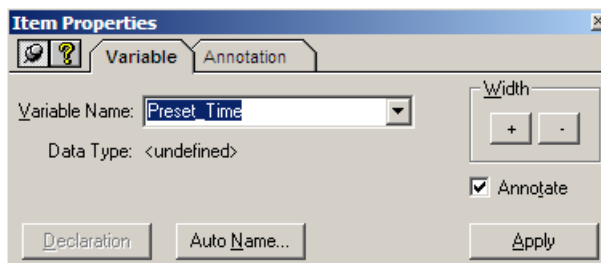


Place Variables on Logic Sheet

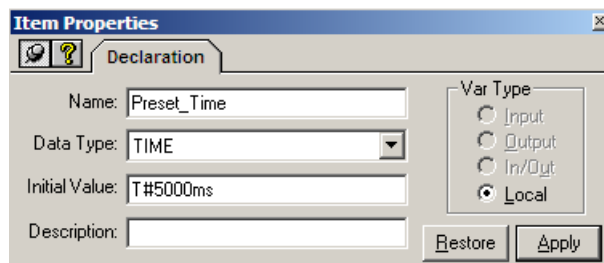
- 1 Click the **Local Variable Tool** button  and connect the variable to the **PT** (Preset Value) input of the **TON** function block.



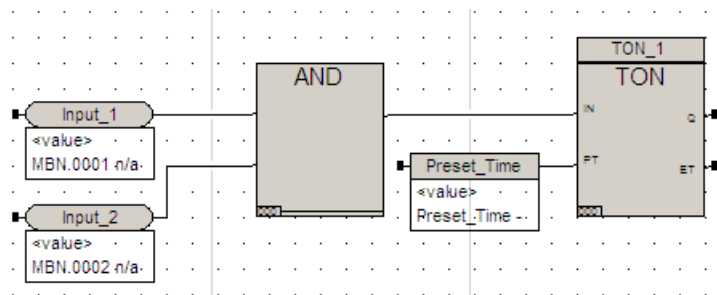
- 2 Double-click the local variable to display the **Item Properties** screen.



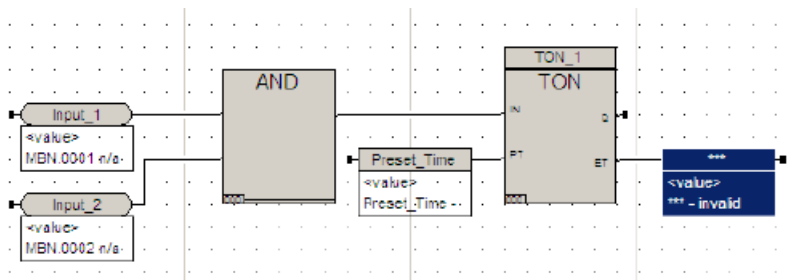
- 3 Enter **Preset_Time** in the **Variable Name** field. Click **Apply**. The **Declaration** tab is displayed.



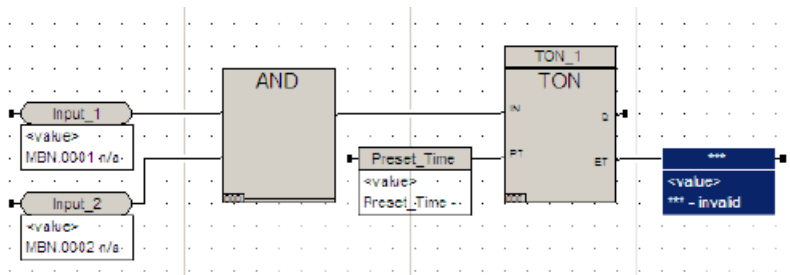
- 4 Select **Time** as the Data Type. Enter **T#5000ms** in the **Initial Value** field. This is the standard syntax for 5000 milliseconds. Click **Apply** and then close the **Item Properties** screen.



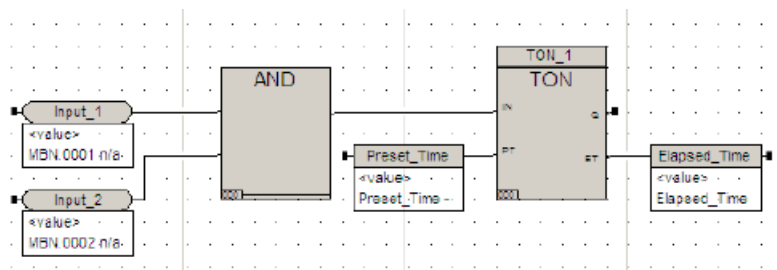
- 5 Click the **Local Variable** tool button and connect the variable to the **ET** (Elapsed Time) terminal of the **TON** function block.



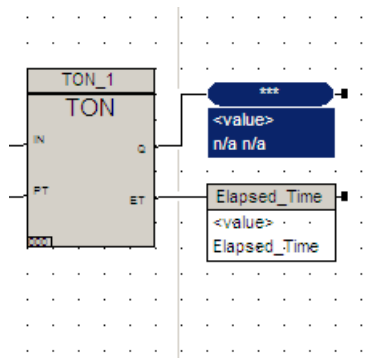
- 6 Double-click the local variable to display the **Item Properties** screen.



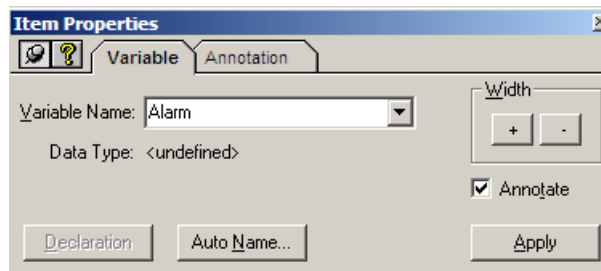
- 7 Enter **Elapsed_Time** in the **Variable Name** field. Click **Apply** and then close the **Item Properties** screen.



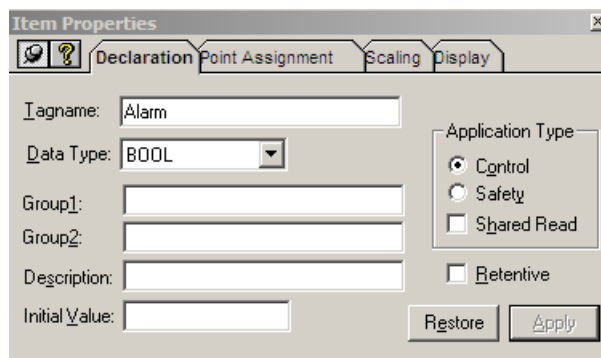
- 8 Click the **Tagname Tool** button and connect the variable to the **Q** output terminal on the **TON** function block.



- 9 Double-click the tagname to display the **Item Properties** screen.

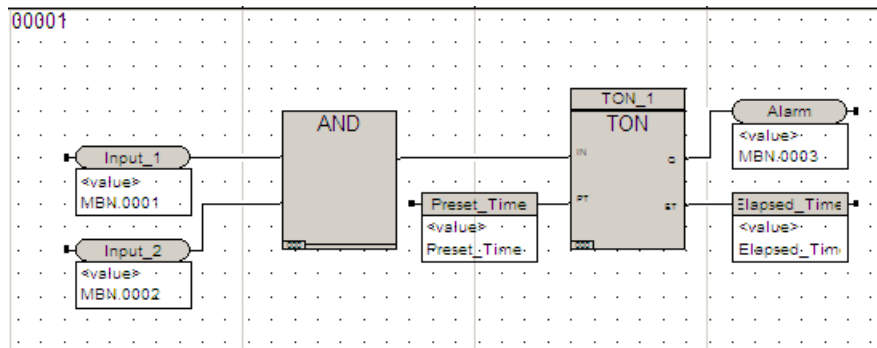


- 10 Enter **Alarm** in the **Variable Name** field. Click **Apply**. The **Declarations** tab is displayed.



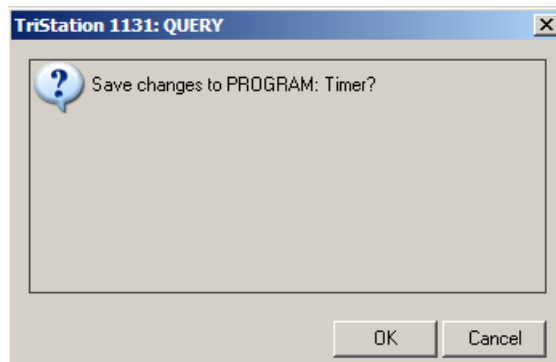
- 11 You may enter text in the **Description** field. Confirm **BOOL** as the Data Type and **Control** as the Application Type. Then close the **Item Properties** screen.

You have completed the logic for the Timer program.

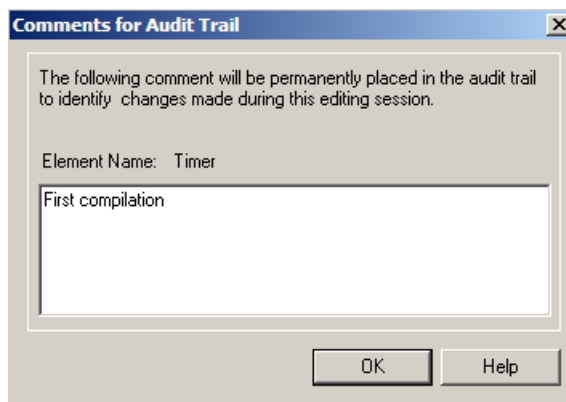


Compile the Program

- 1 Click **Compile** button .

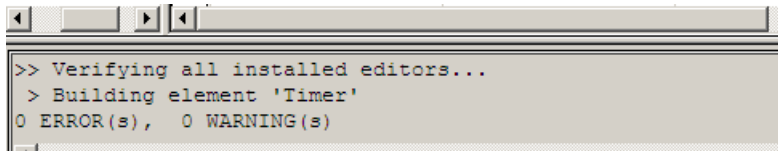


- 2 Click **OK** to save the changes. The **Comments for Audit Trail** screen is displayed.



- 3 You may enter text for the audit trail. Click **OK**.

The program is compiled and the results are displayed in the Message bar at the bottom of the screen.

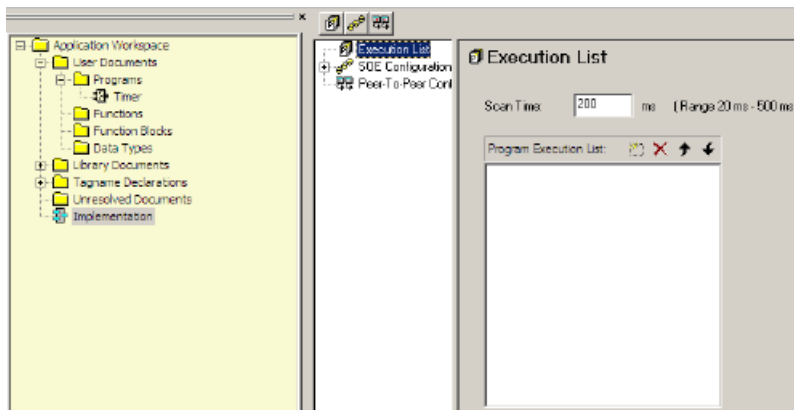


```
>> Verifying all installed editors...
> Building element 'Timer'
0 ERROR(s), 0 WARNING(s)
```

- 4 Verify there are no errors or warnings in the Message bar. If there are errors, click on the text of the message to find the location of the error. Then make the necessary corrections and compile the program again.

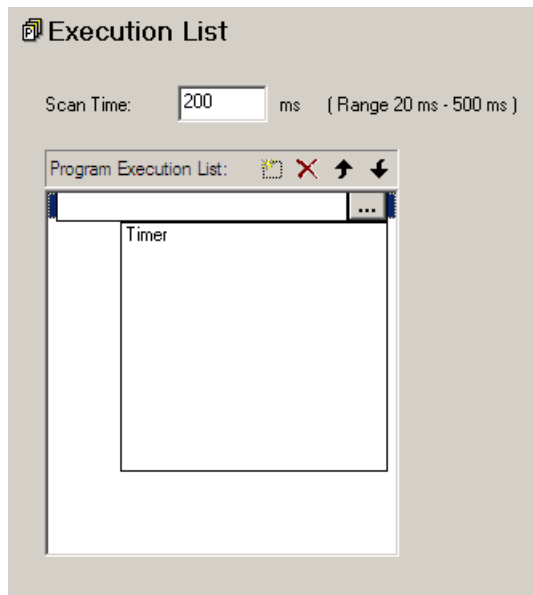
Specify Program Execution

- 1 On the **Application** tree, double-click **Implementation**. The **Execution List** is displayed.

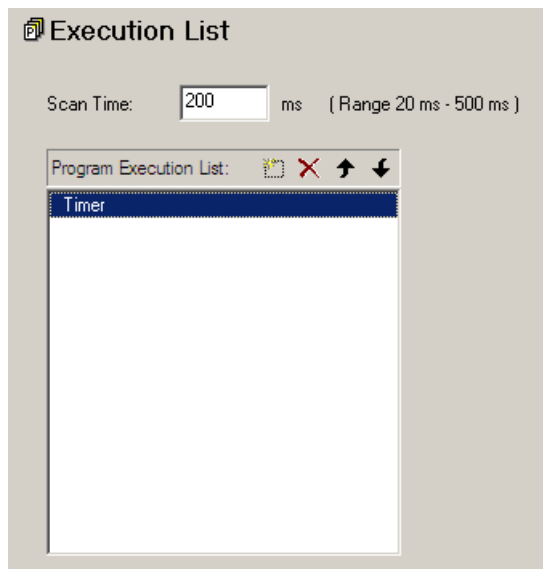


- 2 On the **Program Execution List** toolbar, click the **New Insert** button .

- 3 Click the **Browse Button**  to display the list of programs in this project.





- 4 Click **Timer** to add it to the Execution List.

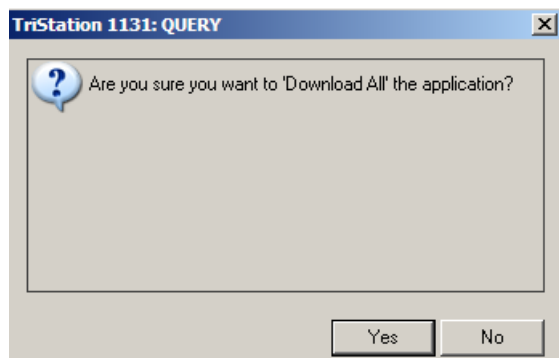


Test Program Logic in Emulator Panel

- 1 Close the program window (not the project) before connecting to the **Emulator Panel**.
- 2 Click the **Controller** tab at the bottom of the **Application** screen.



- 3 Double-click **Emulator Panel**.
- 4 Click the **Connect** button .
- 5 Click the **Download All** button .



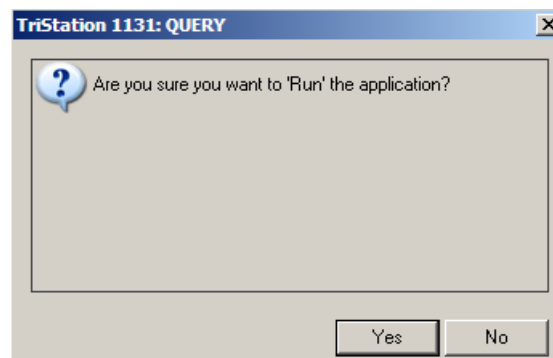
- 6 Click **Yes** to the Download All query. The system builds the application and displays status information in the Message Bar.

```

***** BUILDING FULL APPLICATION *****
>> Verifying the versions of compiler, linker, assembler, and code generator...
>> Validating all tagnames...
>> Verifying all installed editors...
>> Building modified elements...
> Building element 'Timer'
>> Building Configuration...
++ Initializing program 'Timer'...
>> Creating program instances
>> Assembling Libraries for Emulator...
> Linking for emulation...
The estimated stack size is 516 bytes.
0 ERROR(s), 0 WARNING(s)

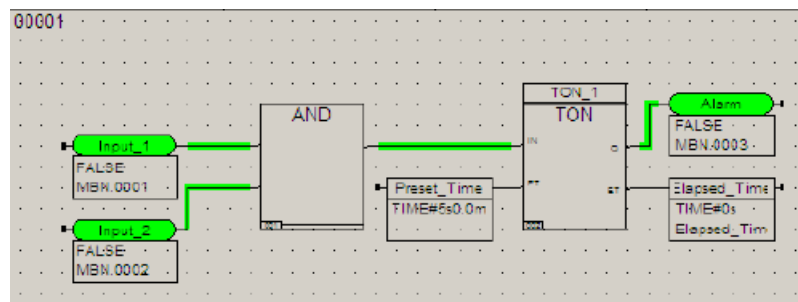
```

- 7 Click the **Run** button .



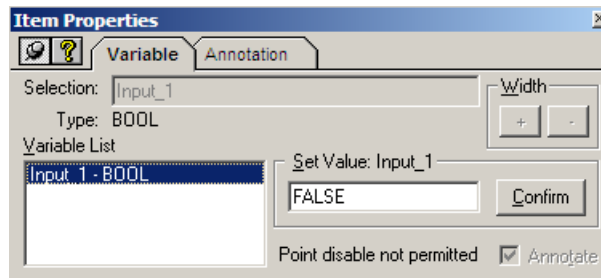
- 8 Click **Yes** to run application.
- 9 On the **Emulator** tree, double-click **Programs** to expand it, and then click **Timer**.

- 10 Click the **Display Program Document** button  to display the program.



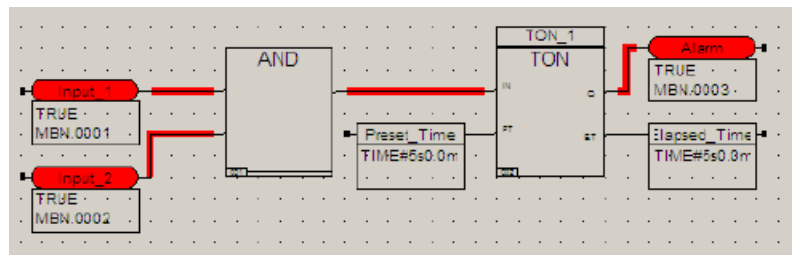
Forcing Points

- 1 On the emulator program display, double-click **Input_1**. The **Item Properties** screen is displayed.



- 2 In the **Item Properties** screen, change the **Set Value Input** to **True**. Click **Confirm** and close the screen.
- 3 On the emulator program display, double-click **Input_2**. In the **Item Properties** screen, change the **Set Value Input** to **True**. Click **Confirm** and close the screen.

The change in input sets the alarm.



- 4 Close the **Emulator Panel**.

Project Administration

Developing a Control Strategy	50
Project Administration	53
Specifying User Access	56
Specifying Project Options	62
Specifying Annotations	64
Setting TriStation 1131 Options	71
Library Documents	82

Developing a Control Strategy

TriStation provides many options for project design and development, which differ slightly depending on the development language you are using. All projects are based on three main elements: programs, functions, and function blocks.

A control strategy involves organizing or partitioning the logic into a series of executable programs that control all of the operations needed within the project. To do this, you need to determine:

- Application type – Safety, Control, or Safety and Control
- Number of programs
- When to create function blocks
- When to create functions

Determining Application Type

An application can include safety programs only, or control programs only, or a combination of safety and control programs.

Safety Application

A safety application is designed to take a process to a safe state when predetermined conditions are violated. It is also referred to as an Emergency Shutdown System (ESD), Safety Instrumented System (SIS), and Safety Interlock System. It is the most restrictive type of application. All elements of the application (programs, functions, function blocks, and tagnames) must be approved or specified for safety. Only programs and tagnames can be designated as safety-only.

Control Application

A control application is designed to control a process and use control, or safety and control functions and function blocks. The tagnames must be specified as control tagnames.

Safety and Control Application

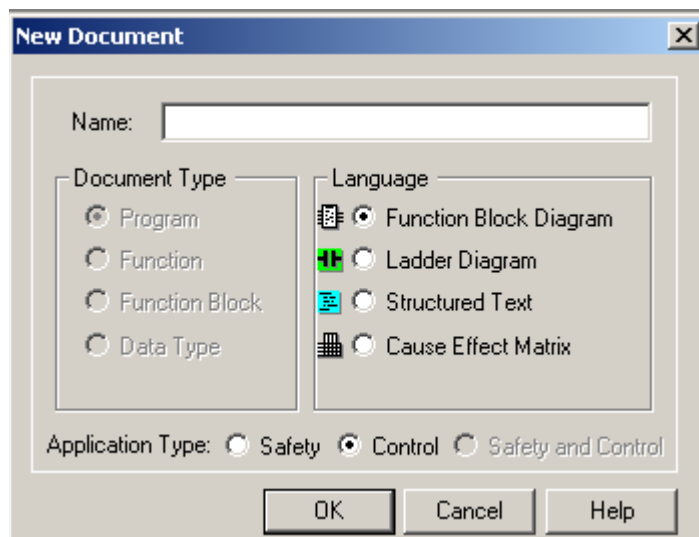
A safety and control application includes both safety and control components.

Most **TriStation** applications are used for safety shutdown purposes. However, some **TriStation** applications may support safety shutdown and/or control operations in a single system. For those operations, it is important to separate the safety shutdown applications from the non-safety logic and control operations. You must also verify the separation when you build the applications.

This table describes how programs, functions, function blocks, and tagnames can be used in safety and control applications

Application Elements	Use
Programs	Safety programs cannot use control functions, function blocks, or tagnames. Control programs cannot use safety tagnames.
Functions and Function Blocks	Safety and control, or control. Library functions and function blocks are designated for use in safety and control, or control applications. These designations cannot be changed.
Tagnames	Safety or control. If the Shared Read property is checked, a safety program can read a control tagname or vice versa.

To separate the safety and non-safety logic, every program, function, or function block must be defined for a safety and/or control application. This is done when you first create the component in the New Document screen. The system automatically determines the available options for the Application type. In this example, a program written in Function Block Diagram language can only be a Safety or Control application.



Determining the Number of Programs

A program is the highest-level executable element in a project. A typical project is partitioned into multiple programs based on the operation of particular units in the controlled process.

When to Create Function Blocks

A function block can invoke several functions or other function blocks. You must create a function block (rather than a function) when your logic returns more than one result and/or needs to retain data from one scan to the next. For example, a counter needs to retain data from one scan to the next. It would be used as a function block.

If you create a number of standard function blocks, you can then create your own shared library, which is a convenient way to make the function available company-wide.

When to Create Functions

A function, like a function block, can be used to implement standard, repetitive operations. However, a function can only return one result and does not retain data from one scan to the next. For example, an average of two numbers will return only one result.

A function can be extensible or non-extensible. An extensible function can have more than one input. For example, an ADD function can add two, three or more inputs. A non-extensible function has a fixed number of inputs.

Project Administration

In creating a new project (or modifying an existing project), there are settings and operations that must be defined before you begin writing the logic for your program. You need to define who can access the project and set up how the program logic will look and function.

In this section, you will specify the settings for a new project for a water tank alarm. One input connects with a sensor to detect if the water level is too high. A second input detects if the water level is too low. The output sets an alarm if the water is too high or too low.

You will:

- Create a new project – select the platform and name the project.
- Specify user access.
- Add a project description.
- Set project options.

The project you create will be used throughout this course.

Lesson 1: Creating a TriStation 1131 Project

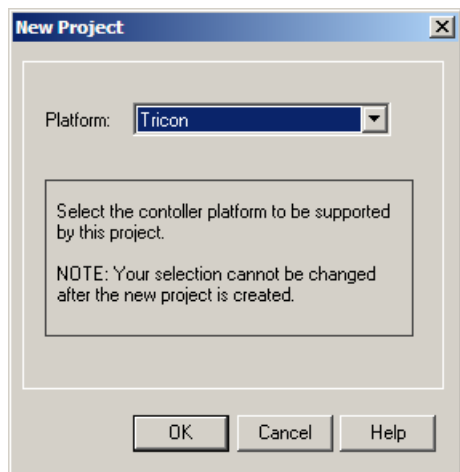
When you create a new project, **TriStation** automatically sets up a default Level 01 (fully privileged) user name **MANAGER**, and **PASSWORD**. It is strongly recommended that you assign personalized level names and passwords during project creation to prevent unwanted access.

In this lesson, you will:

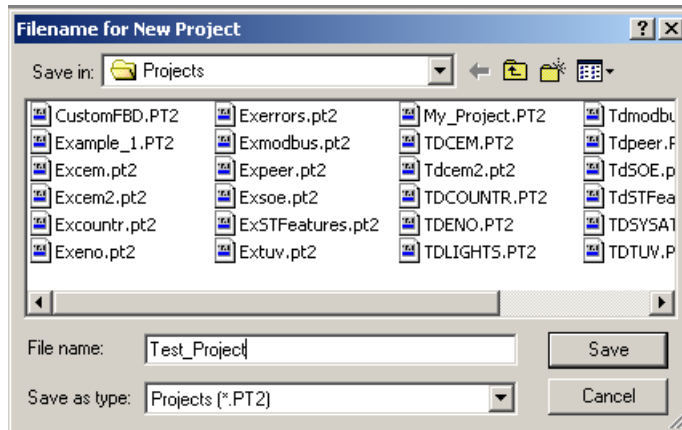
- Create a new project.
- Select Tricon as the platform.
- Name the project.

Procedure

- 1 Open **TriStation 1131**. On the **File** menu, click **New Project**.



- 2 Select **Tricon** as the platform. Click **OK** to continue. The **Filename for New Project** window is displayed.

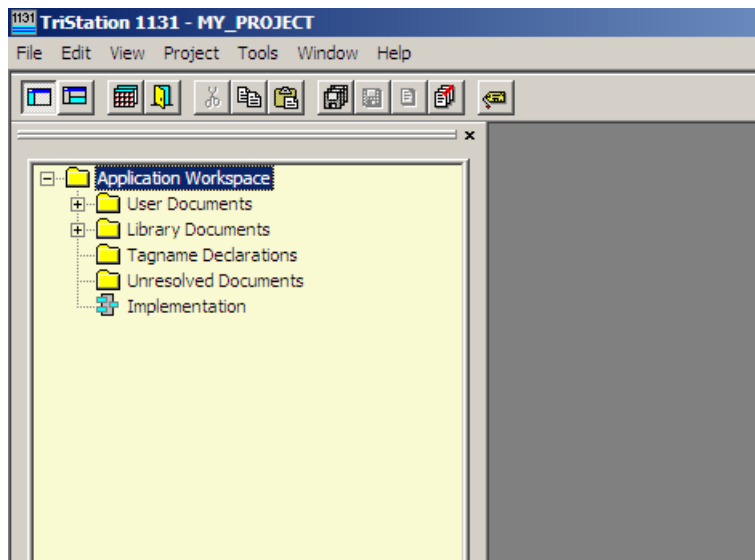


- 3 Enter **Test_Project** in the **File name** field. Click **Save**. The extension **.PT2** is attached automatically.

A name can only contain A to Z, a to z, 0 to 9 and underscores. No spaces or other characters are allowed. To separate words in the file name, use an underscore_ instead of a space. Maximum length is 31 characters.

The system automatically attaches libraries to the project. The type of libraries attached vary with the type of platform selected.

After the file creation process finishes, you are automatically logged on to the project using the default user name (MANAGER) and default password (PASSWORD).



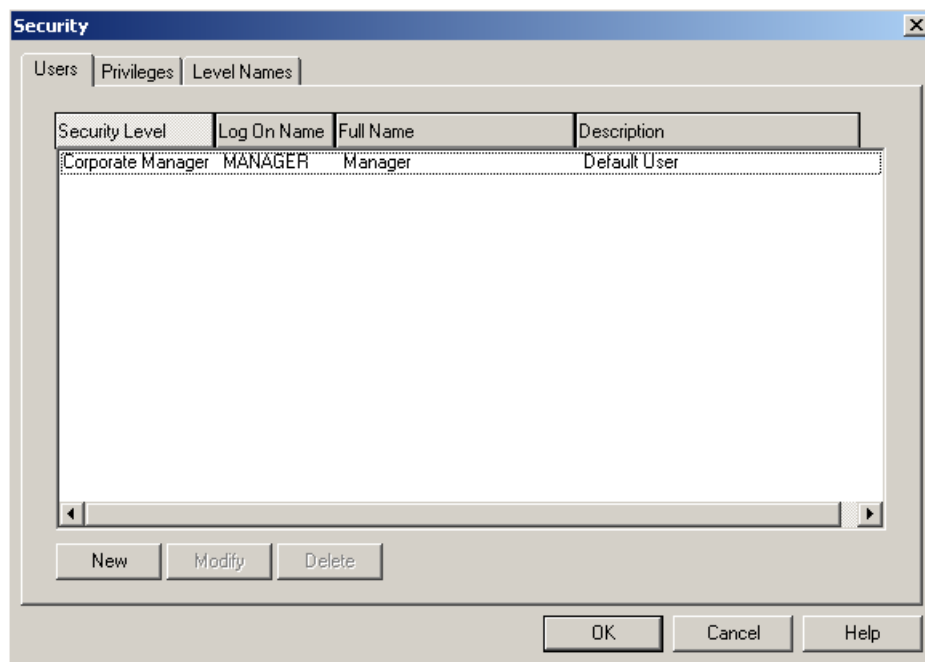
Specifying User Access

In **TriStation**, you can create a security system that defines users and their access to operations.

Access is based on the security level assigned to the user, from 01 to 10. Level 01 is the highest security level with access to all operations. Level 10 is the lowest security level with access to the fewest operations.

Each level allows access to its own operations and the operations associated with all lower levels. For example, a user with security level 03 has access to operations for security levels 04 through 10.

User access is created and defined in the **Security** screen.




Lesson 2: Adding a New User

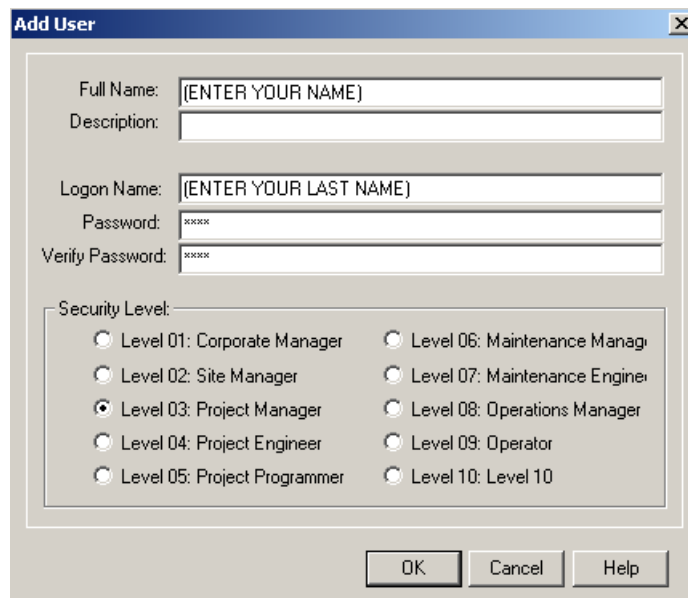
From the **Users** screen, you can add, modify, or delete users from a project.

In this lesson, you will:

- Add yourself as a new user.
- Assign Level 3 Security access to your name.

Procedure

- 1 Display the **Security** screen by doing either of these:
 - Click the **Security** button  on the toolbar, and then click the **Users** tab.
 - On the **Project** menu, click **Security**, and then click the **Users** tab.
- 2 On the **Users** tab, click **New**. The **Add User** screen is displayed.



The **Add User** dialog box is shown. It contains the following fields and options:

- Full Name:** [ENTER YOUR NAME]
- Description:** [Empty text box]
- Logon Name:** [ENTER YOUR LAST NAME]
- Password:** [Masked with x's]
- Verify Password:** [Masked with x's]
- Security Level:** A group box containing ten radio button options:
 - ☐ Level 01: Corporate Manager
 - ☐ Level 02: Site Manager
 - ☒ Level 03: Project Manager
 - ☐ Level 04: Project Engineer
 - ☐ Level 05: Project Programmer
 - ☐ Level 06: Maintenance Manag
 - ☐ Level 07: Maintenance Engine
 - ☐ Level 08: Operations Manager
 - ☐ Level 09: Operator
 - ☐ Level 10: Level 10

At the bottom are three buttons: **OK**, **Cancel**, and **Help**.

3 Enter the following information:

Full Name:	Enter your name
Description:	Manager
Logon Name:	Your Last Name
Password:	Select a password
Verify Password:	Repeat the password
Security Level:	Level 3

Triconex recommends assigning personalized level names and passwords when creating a project.

4 Click **OK** to save settings.

Note: Clicking **OK** saves settings and closes the **Security** screen.

Lesson 3: Changing the Security Level for Privileges

TriStation 1131 has two types of operations that can be assigned to security levels:

- **Controller Operations:** These operations run and manage the controller, such as downloading and running the application, setting operating parameters, and enabling and disabling points.
- **TriStation Operations:** These operations manage user aspects of the project, such as creating and modifying program elements, importing and exporting libraries, printing, and security.

For a description of the operations, see *Operations* in the **Help** system.

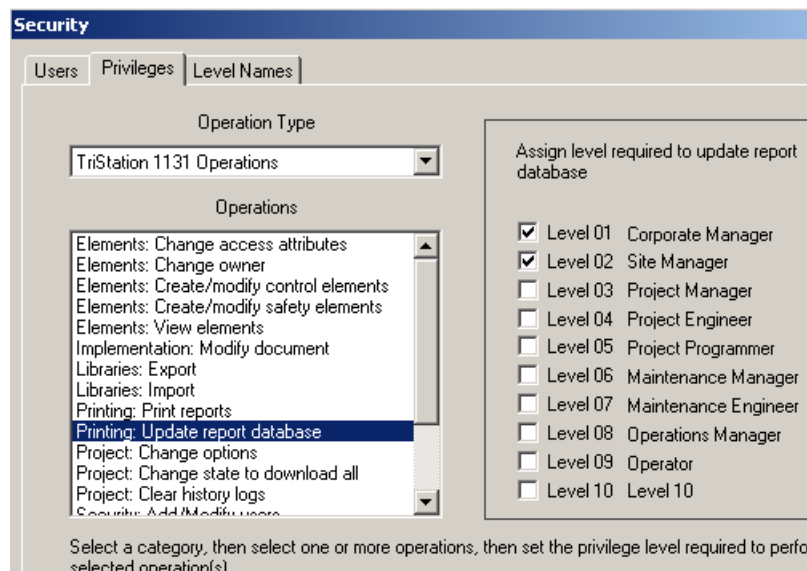
By default, each operation is assigned a security level from the highest level 01 to the lowest 10. You can change the level of security required to perform an operation.

In this lesson, you will change the security level for the **TriStation 1131** operation "Printing: Update report database" to Level 08 Operations Manager.

This allows the Operations Manager and those with higher security levels to update the database.

Procedure

- 1 On the **Security** screen, click the **Privileges** tab.



- 2 Select **TriStation 1131 Operations** from the **Operation Type** list.
- 3 Click **Printing: Update report database**, then click the **Operations Manager** check box.
- 4 Click **OK** to save settings.

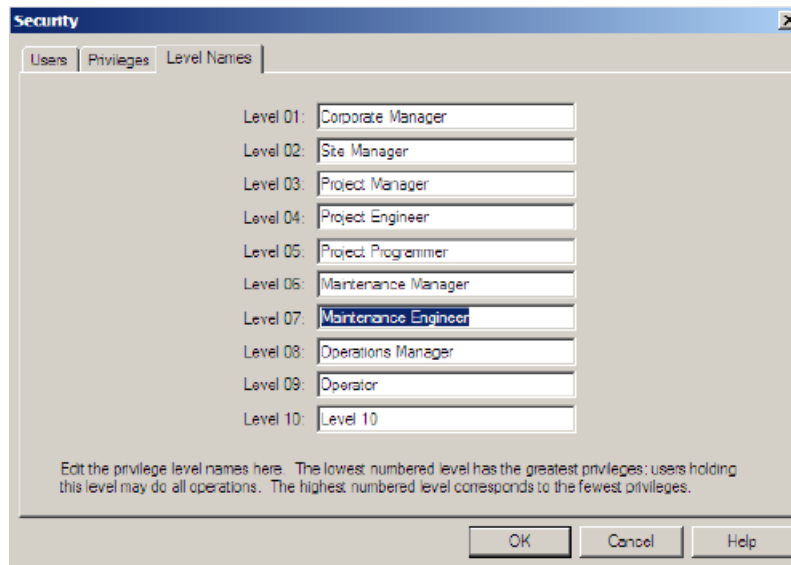
Lesson 4: Changing Security Level Names

You can assign user titles to security levels. The levels are displayed 01 through 10, from highest level to lowest level. If you change a name, it is changed on the other security tabs. Changing a name does not affect its security level assignment.

By default, Corporate Manager has the highest level of privileges and access to all operations. In this lesson you will change the name of Maintenance Engineer to Maintenance Personnel.

Procedure

- 1 From the **Security** screen, click the **Level Names** tab.



- 2 Click **Maintenance Engineer** and change the name to **Maintenance Personnel**.
- 3 Click **OK** to save the new name and close the **Security** window.

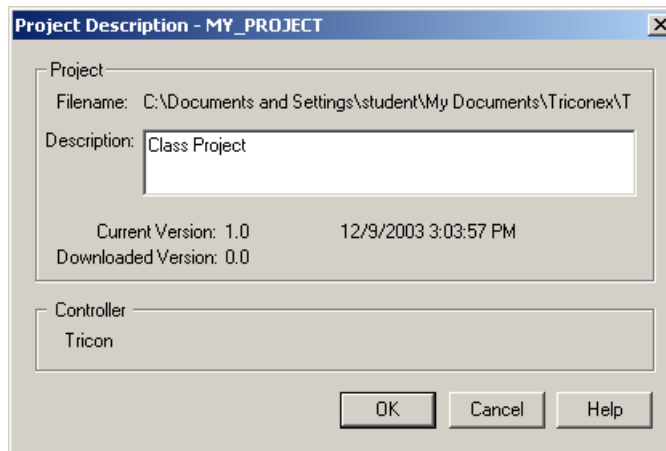
Lesson 5: Adding a Project Description

The Project Description screen displays information about the project, including the version of the project being developed and the version of the project downloaded to the controller, which may be different. You can change or add to the project description.

In this lesson, you will enter **Class Project** in the Description box.

Procedure

- 1 On the **Project** menu, click **Project Description**.



- 2 Enter **Class Project** as the description.
- 3 Click **OK** to save description.

Specifying Project Options

Project Options are settings that specify:

- Language
- Annotation
- Monitor color displays

When you create new elements in a project, default settings are used unless you specify different settings. For example, the Default Language property is set to Function Block Diagram. This means that a new function is automatically created in FBD language unless you specify another language.

You can change the default settings when you begin a new project or any time during project development. The settings only affect new elements.

Lesson 6: Specifying Language Options

TriStation 1131 supports four programming languages:

- Function Block Diagram (FBD)
- Ladder Diagram (LD)
- Structured Text (ST)
- CEMPLE

The Language tab lets you select the language to use when creating a program, function, function block, or data type. You can also select the elements that can be created in the project.

Default Language specifies the type of programming language to use when creating a program, function, function block or data type. The default is Function Block Diagram.

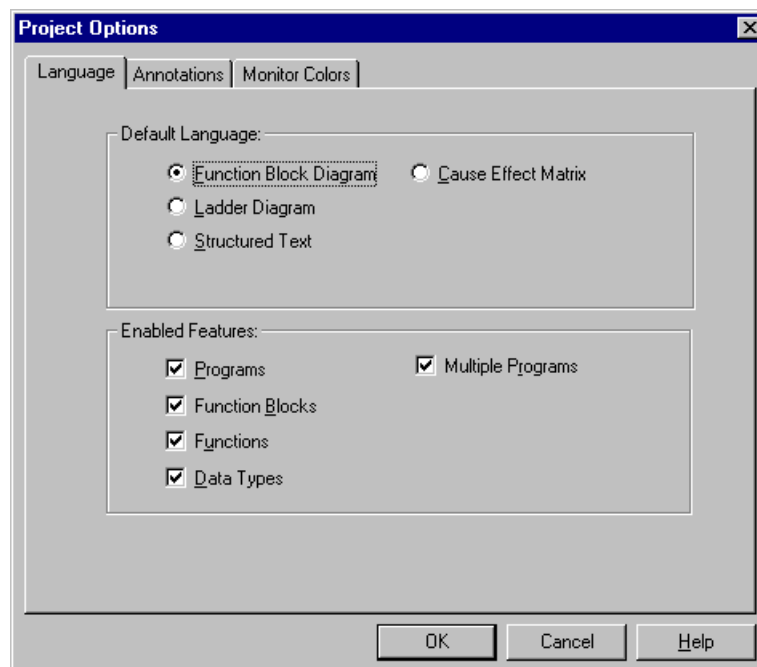
Enabled Features specifies the features that can be created in the project. The default enables all features.

In this lesson, you will:

- Accept the Default Language Property FBD.
- Disable the Multiple Programs feature.

Procedure

- 1 On the **Project Options** screen, click the **Language** tab.

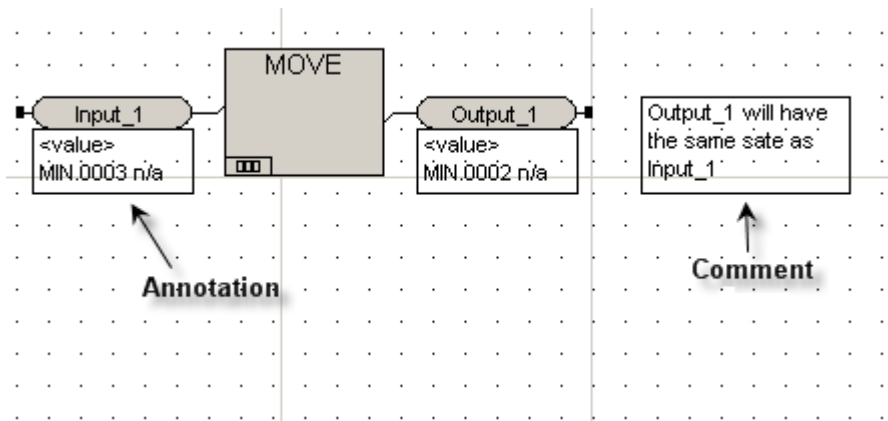


- 2 Keep **Function Block Diagram** as the Default Language. Disable **Multiple Programs** by unchecking the box.
- 3 Click **OK** to save settings.

Specifying Annotations

TriStation offers two methods for documenting program logic: comments and variable annotations. Both methods allow you to enter descriptive text and use macros that describe various characteristics of the project, its elements, and the logic sheet.

A comment is a text box drawn and placed on the logic sheet using the Comment Tool button

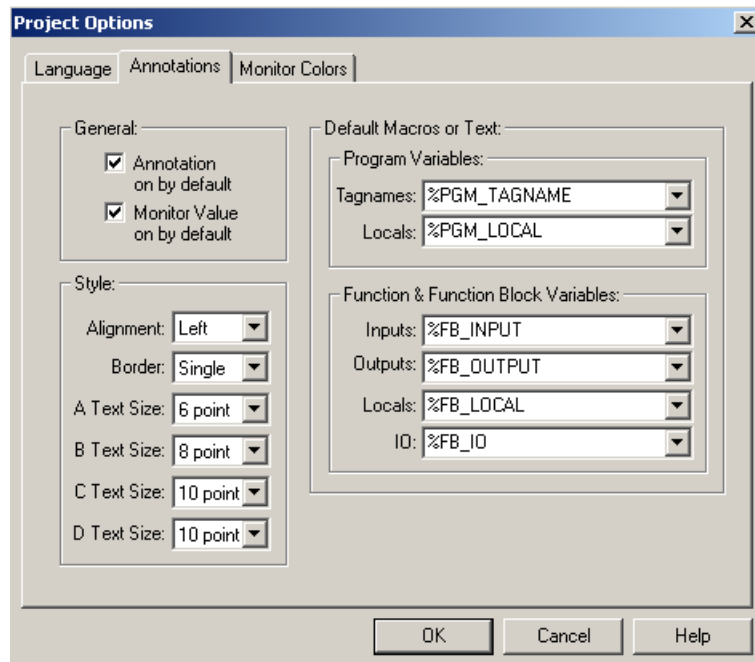


An Annotation is displayed in a box attached to the input, output, input/output or local variable. You can also display the value of a variable during program execution in the emulator and controller.

An Annotation can include:

- The monitor value (value of the variable as the configuration executes)
- The default macro for the particular type of variable
- Other standard macros
- User-modified macros
- Text that you type in

Annotations Tab



General

The General box on the Annotations tab enables you to select:

- Annotation on by default
- Monitor value on by default

Annotation on by Default option automatically adds annotation boxes to variables used with a program, function, or function block.

Monitor Value on by Default displays the value of each variable in the upper left corner of the annotation box when the program or function block is executed.

It is recommended that you enable both features when you are first creating a project.

Style

You can change how the text appears by setting the text alignment in annotation boxes, the borders around them, and the text size used in each sheet template.

The sheet template sizes are:

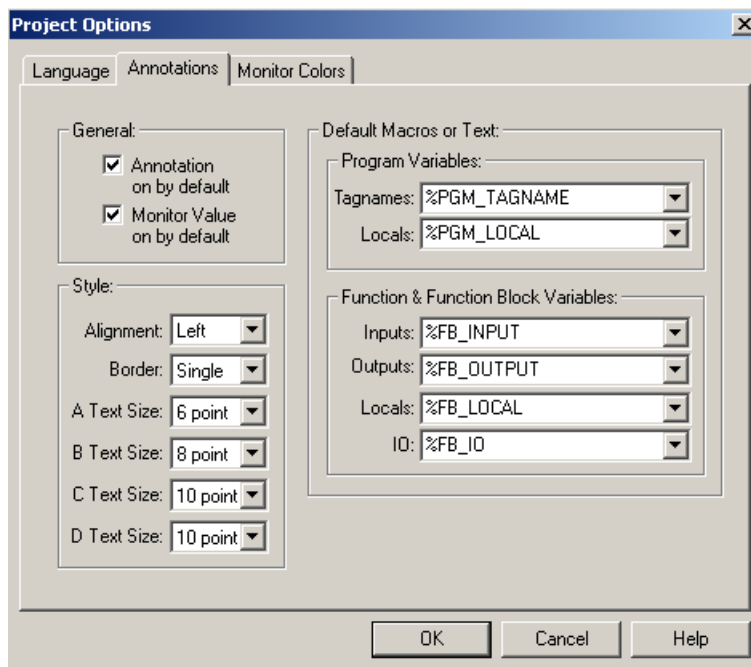
- Sheet A - 8.5" x 11"
- Sheet B - 11" x 17" (default)
- Sheet C - 17" x 22"
- Sheet D - 22" x 34"

These settings also apply to Annotations and Comments.

Default Macros or Text

Macros, used in annotations and comments, are placeholders for text supplied by the system or the user. The value is displayed when the element is run in the emulator or controller.

Macros can only be used in FBD and LD development.



There are two types of macros:

- System macros are supplied values by the system and cannot be changed. For example, the %CREATED_BY macro includes the user ID of the person who created the element. You cannot change values for these macros.
- User-modifiable macros, identified by a pencil icon, are values you can specify. For example, the %APPROVED_NAME macro can include any name you enter.

The Default Macros or Text property specifies a default macro or text to include with a variable in a program, function, or function block.

The default macros used with annotations vary depending on the element and variable type. For example, if %PGM_TAGNAME is selected, the tagname, type of location on controller and alias number will display in the annotation.

For a description of %TAG macros, see *Default Macros for Annotations* in the Help documentation.

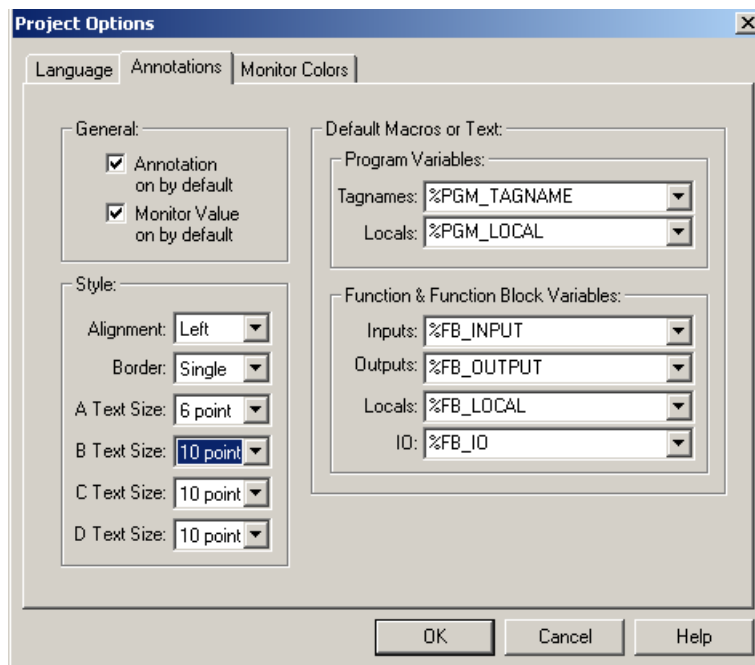
Lesson 7: Specifying Annotation Properties

In this lesson you will:

- Enable the **Annotation on by default** feature that automatically adds annotation boxes to variables.
- Enable the **Monitor Value on by default**.
- Change the text size of the B sheet template to 10 point.

Procedure

- 1 On the **Project Options** screen, click the **Annotations** tab.



- 2 Check **Annotation on by default**. Check **Monitor Value on by default**. Click the **B Text Size** arrow, then select **10 point** from the list. Click **OK** to save settings.

Lesson 8: Specifying Monitor Colors for BOOL Values

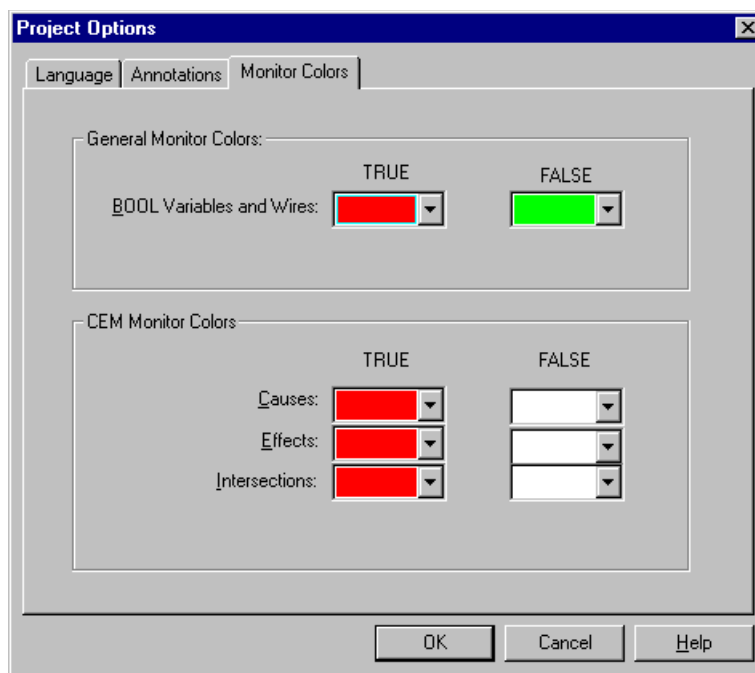
The Monitor Colors tab is used to select the colors displayed for BOOL True and False values and wires. This allows you change to the colors to meet a particular industry or corporate standard for color displays.

In this lesson, you will:

- Change the BOOL Variables and Wires TRUE color to green.
- Change the BOOL Variables and Wires FALSE color to RED.

Procedure

- 1 On the **Project Options** screen, click the **Monitor Colors** tab.



General Monitor Colors: Specifies the colors to display for the value of BOOL variables and wires when the application is run in the emulator and controller.

The default for True is red.

The default for False is green.

CEM Monitor Colors: Specifies the colors to display for cause, effect, and intersection cells in a CEM (Cause and Effect program).

The default for True is red.

The default for False is white.

- 2 Change the **BOOL Variables and Wires** color for **True** to **green** by selecting green from the list.

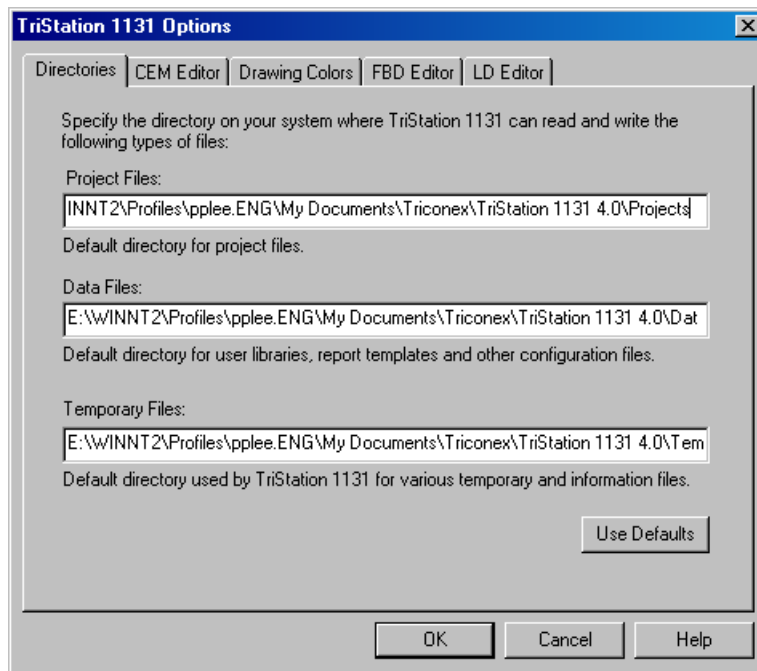
- 3 Change the **BOOL Variables and Wires** color for **False** to **red** by selecting red from the list.
- 4 Click **OK** to save settings.

Setting TriStation 1131 Options

TriStation 1131 Options include properties that affect user aspects of the **TriStation 1131** interface. You can specify directories, drawing color, and set options for the language editors.

The Directories tab enables you to accept the default path or change where the files will be stored on your system. The language editor tabs and The Drawing Colors tabs are used to modify how the elements and documents appear.

All the properties included have default settings that specify how features are initially configured. You can change these settings at any time during project development.



Lesson 9: Specifying Directories

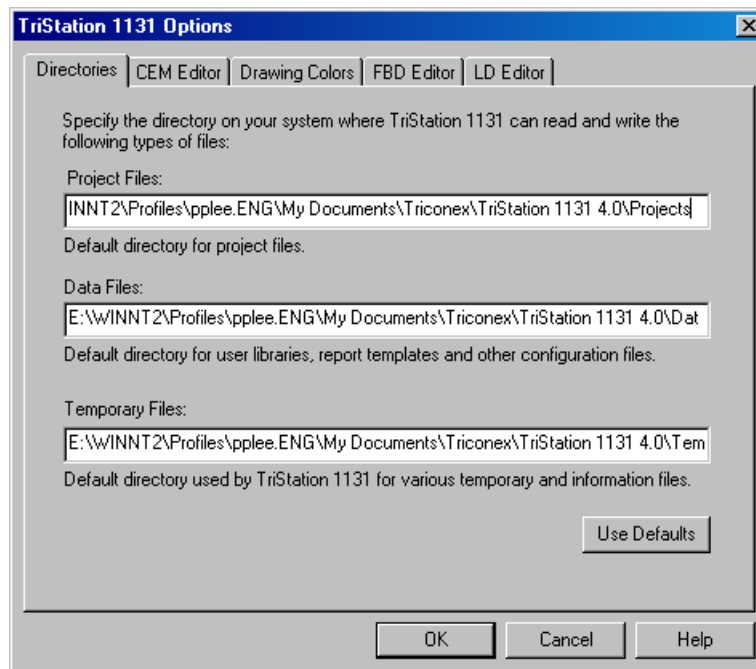
The Directories tab displays the pathnames for the project, library, and report files.

In this lesson, you will:

- View the directory locations for the Project, Data, and Temporary files.
- Keep the default settings.

Procedure

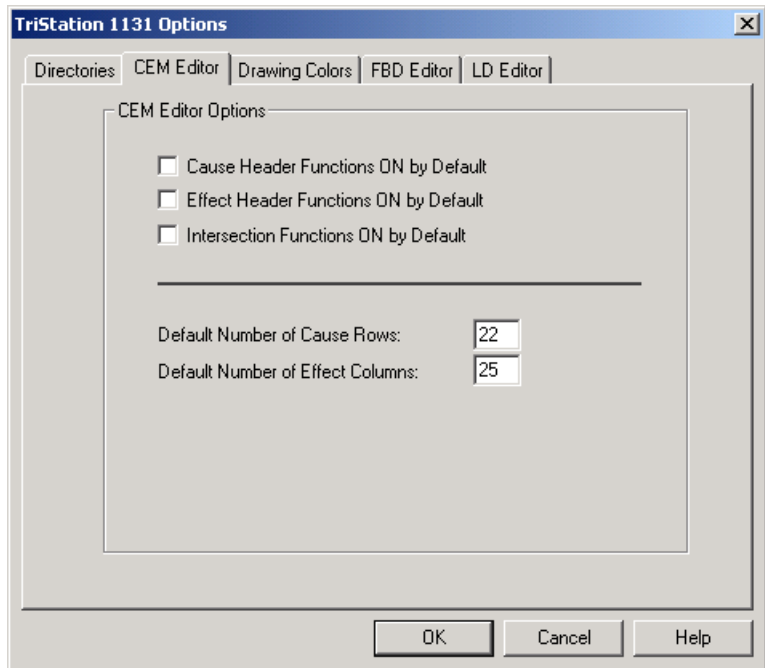
- 1 On the **Tools** menu, click **TriStation 1131 Options**, and then click the **Directories**.



- 2 Keep the default locations or change them by entering a different path.
- 3 Click the **Use Defaults** button to return the settings to the default paths.
- 4 Click **OK** to save settings.

Lesson 10: Specifying CEM Editor Options

The CEM Editor tab displays the initial settings for all the CEM (Cause and Effect Matrix) programs in a project.



Like a table, the CEMPLE matrix is made up of rows and columns, with cells at the intersection of each row and column.

			Effect	Description	OR	OR	OR	OR	OR
					UNIT_1_ALARM	UNIT_2_ALARM	UNIT_3_ALARM	UNIT_4_ALARM	UNIT_5_ALARM
					High level alarm indicator for tank 1	High level alarm indicator for tank 2	High level alarm indicator for tank 3	High level alarm indicator for tank 4	High level alarm indicator for tank 5
	Cause	Description		E01	E02	E03	E04	E05	
	LEVEL_1_	TRUE=Fluid level in tank 1 is high	C01	X					
	LEVEL_2_HI	TRUE=Fluid level in tank 2 is high	C02		X				
	LEVEL_3_HI	TRUE=Fluid level in tank 3 is high	C03			X			
	LEVEL_4_HI	TRUE=Fluid level in tank 4 is high	C04				X		
4									
Loc	Terminal	Var/Const	VarType	DataType	Description				
C01		P1_LEVEL_1_HI	Tagname	BOOL					

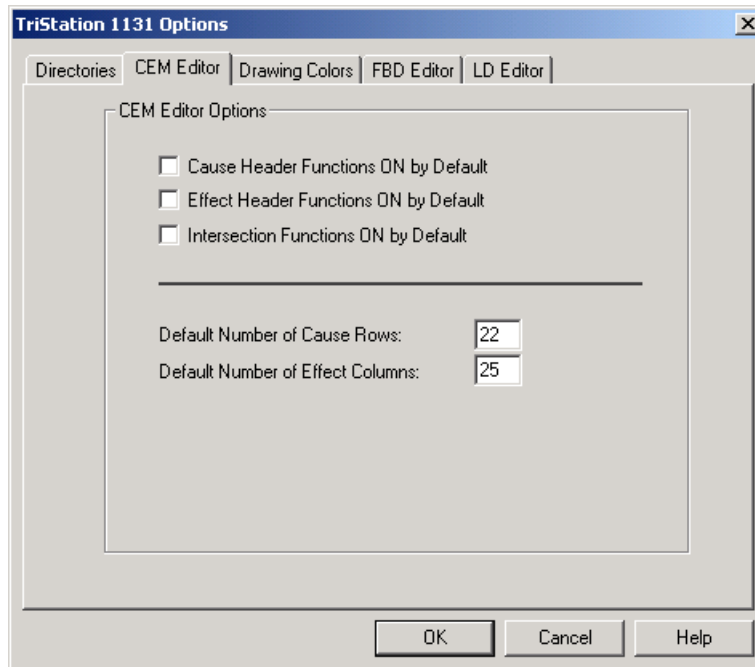
You can select format options that are added automatically as you create the matrix. After a program is created, you can modify these settings on a program-by-program basis.

In this lesson, you will:

- Enable the Cause Header Function by Default.
- Enter 15 as the Default Number of Cause Rows.
- Enter 18 as the Default Number of Effect Columns.

Procedure

- 1 On the **TriStation 1131 Options** screen, click the **CEM Editor** tab.



Cause Header Functions ON by Default:	Adds input and function columns.
Effect Header Functions ON by Default:	Adds output and function columns.
Intersection Functions ON by Default:	Adds function columns.
Default Number of Cause Rows:	Number of rows to include in a new matrix. The default is 25.
Default Number of Effect Columns:	Number of columns to include in a new matrix. The default is 25.

- 2 Check **Cause Header Function ON by Default**.
- 3 Change **Default Number of Cause Rows to 25**.
- 4 Change **Default Number of Effect Columns to 18**.
- 5 Click **OK** to save settings.

Lesson 11: Specifying Drawing Colors

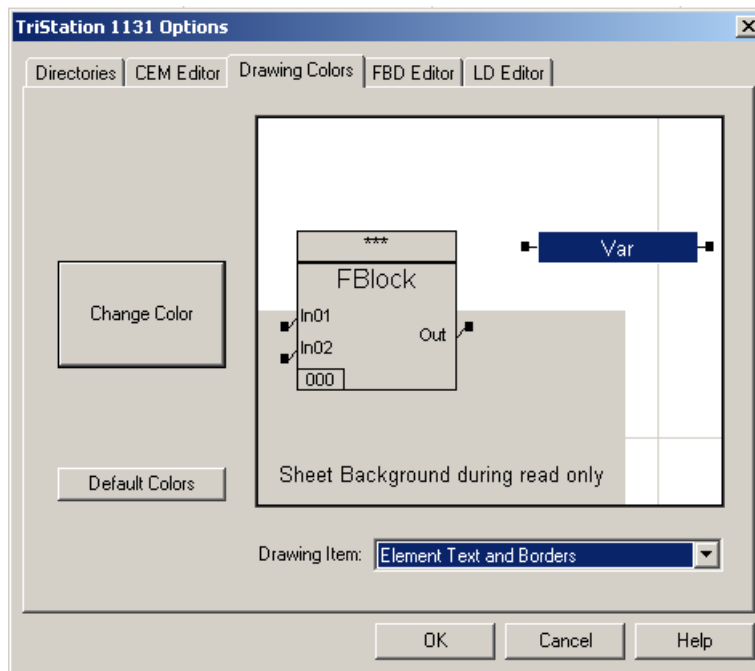
The Drawing Colors tab lets you customize the appearance of your program by selecting standard or custom colors for the drawing items used in the programming editors.

For a description of the default colors, see *Default Colors Command* in the Help documentation.

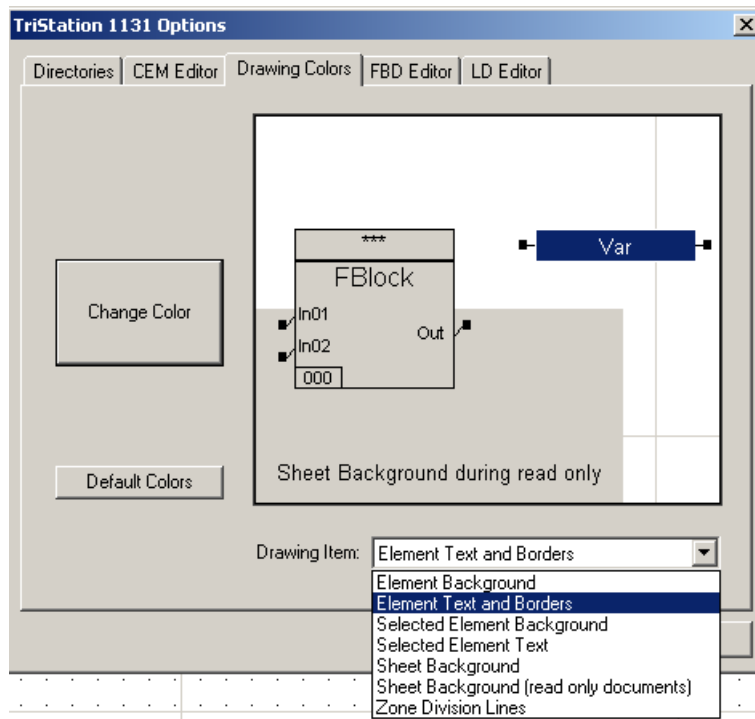
In this lesson, you will change the color of the element text and borders.

Procedure

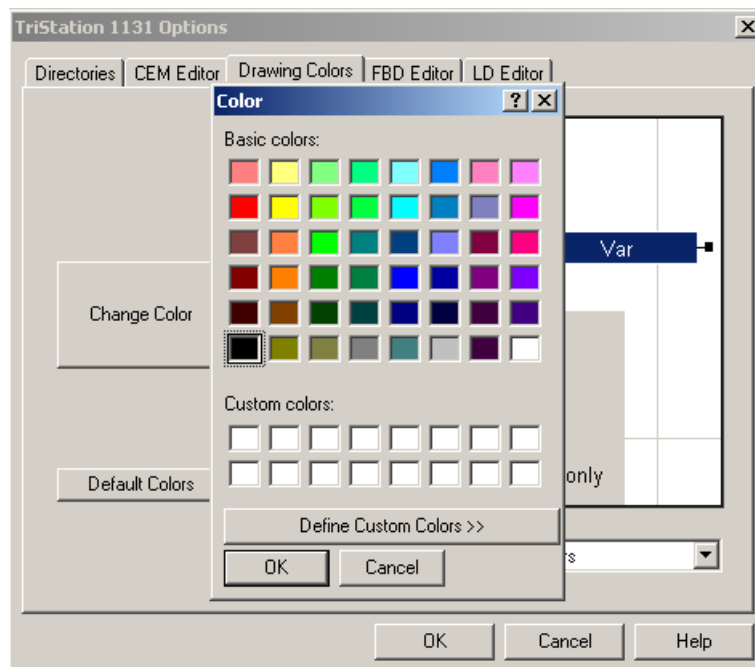
- 1 On the **TriStation 1131 Options** screen, click the **Drawing Colors** tab.



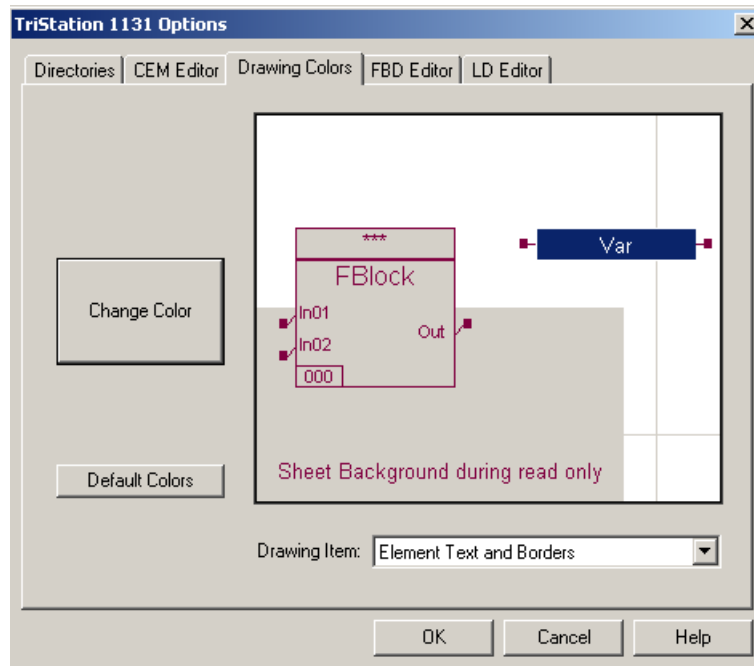
- 2 From the **Drawing Item** list, select **Element Text and Borders**.



- 3 Display the color palette by clicking the **Change Color** button.



- 4 Select a color from the palette by clicking the color, and then clicking **OK**. The item is displayed in the new color.

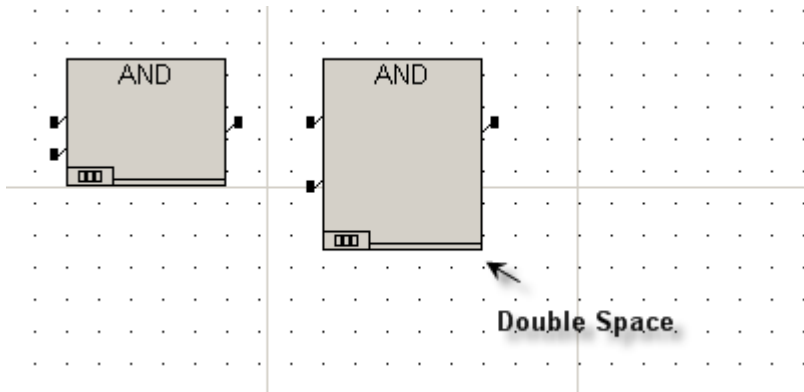


- 5 Click **OK** to save the new color setting.

Lesson 12: FBD Editor Options

The FBD Editor Options settings are used throughout the project. As you write the program logic, you can change these settings for a specific FBD function.

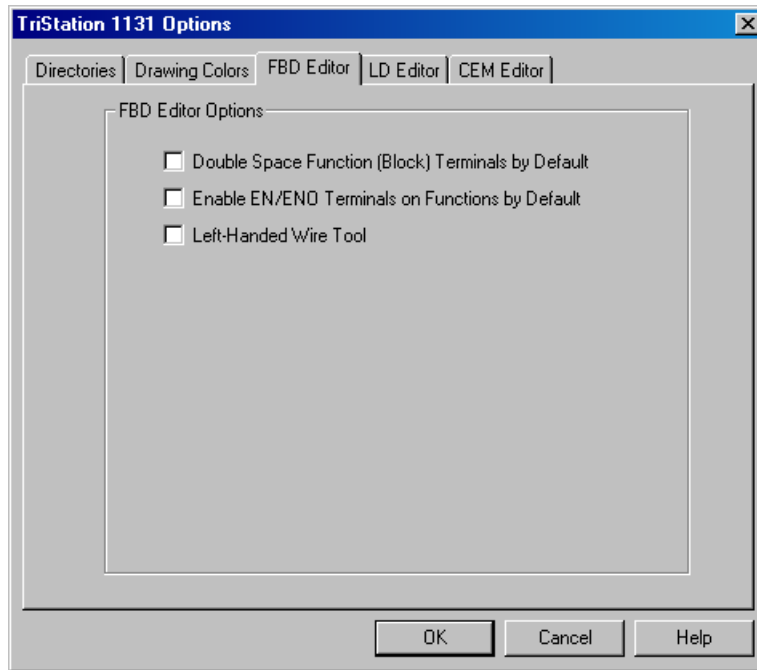
You can also change the Double Space property for functions and function blocks. This doubles the spaces between the input and output terminals, adding more space for annotations and comments.



In this lesson, you will enable Double Space Function (Block) Terminals by Default.

Procedure

- 1 On the **TriStation 1131 Options** screen, click the **FBD Editor** tab.



Double Space Function (Block) Terminals by Default:

Doubles the spacing between terminal (input and output) parameters on the function block, adding space for annotation.

The default is unchecked.

Enable EN/ENO Terminals on Functions by Default:

Automatically enables EN/ENO (input/output) terminals on functions.

The default is unchecked.

Left-Handed Link Tool:

Enables link tool for left-handed use.

The default is unchecked.

- 2 Check **Double Space Function (Block) Terminals by Default**.

CAUTION: If you check Double Space after input and output variables have been attached to the function or function block, the variables will have to be reattached.

- 3 Click **OK** to save settings.

Lesson 13: LD Editor Settings

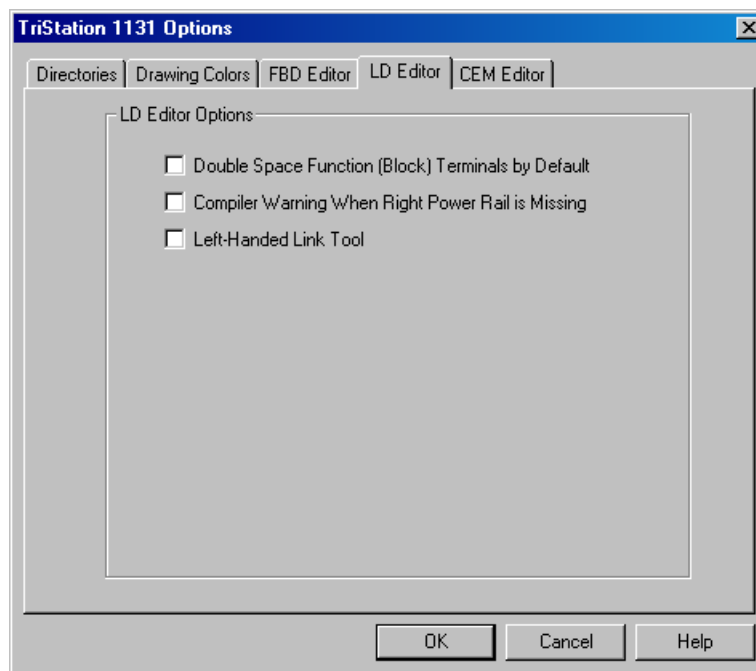
The LD Editor Options settings selected are used throughout the project unless changed on a specific LD function. You change the Double Space property for functions and function blocks.

You can also have a compiler warning display when the right power rail is missing in an LD program. In Ladder Diagrams, power rails act as delimiters, which are characters or graphics that separate program elements. Left power rails are considered On or in a TRUE state at all times; right power rails have an undefined status which can be explicit or implied.

In this lesson, you will enable the Compiler Warning When Right Power Rail is Missing.

Procedure

- 1 On the **TriStation 1131 Options** screen, click the **LD Editor** tab.



Double Space Function (Block) Terminals by Default:

Doubles the spacing between terminal (input and output) parameters on the function block, adding space for annotation.

Compiler Warning When Right Power Rail is Missing:

Displays a compiler warning if the right power rail is missing from a Ladder diagram function.

Left-Handed Link Tool:

Enables link tool for left-handed use.

2 Check Compiler Warning When Right Power Rail is Missing.

CAUTION: If you check Double Space after input and output variables have been attached to the function or function block, the variables will have to be reattached.

Library Documents

You can create a library of project elements (programs, functions, function blocks, and data types) that can be added to other projects. The library function also allows you to manage libraries that can include **TriStation 1131** and user-created libraries.

The library specification (.lsp) and export library (.lt2) files are created in the imported libraries directory. To identify or change the library, go to the Tools menu, click TriStation 1131 Options, and click the Directories tab.

The default location is: c:\...\My Documents\Triconex\TriStation 1131 4.0\Data.

In this section, you will:

- Create a library of project elements
- Manage libraries
- Add libraries
- Update libraries
- Delete libraries
- Verify a library version

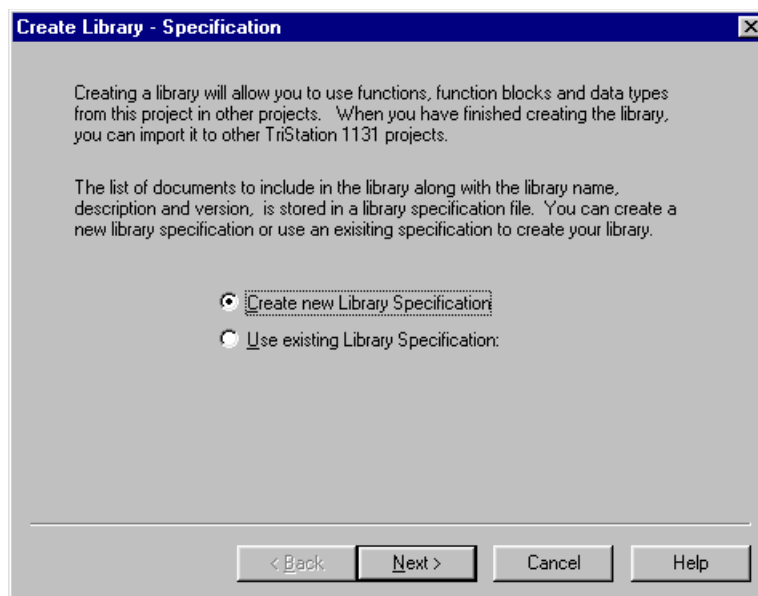
Lesson 14: Creating Libraries

In this lesson, you will create a library of project elements.

You will first create and compile a function block and use that to complete the procedures in this lesson.

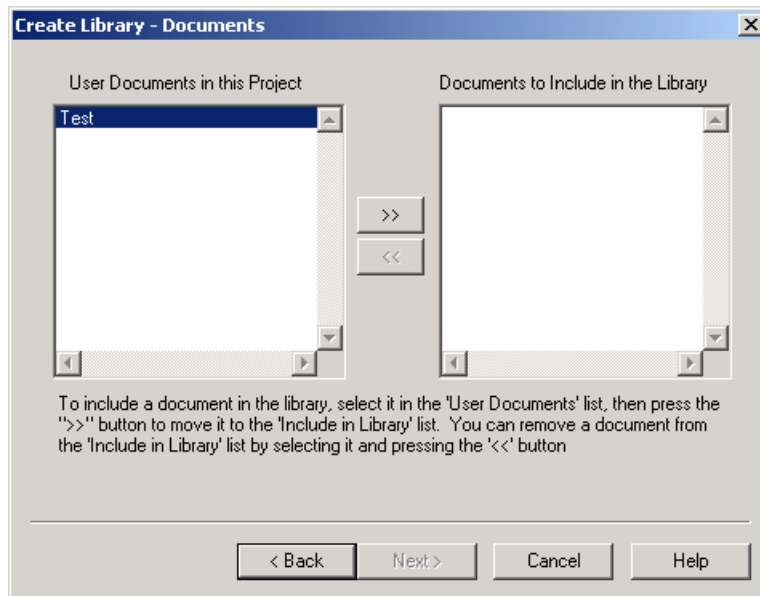
Procedure

- 1 Open **Test_Project**.
- 2 Create a function block named **TEST**. Write an **AND** gate with two inputs and one output.
- 3 Compile the function block, and then close the function block window.
- 4 Expand the **Application** tree, right-click **User Documents**, and then click **Create Library**. The **Create Library - Specification** screen is displayed.

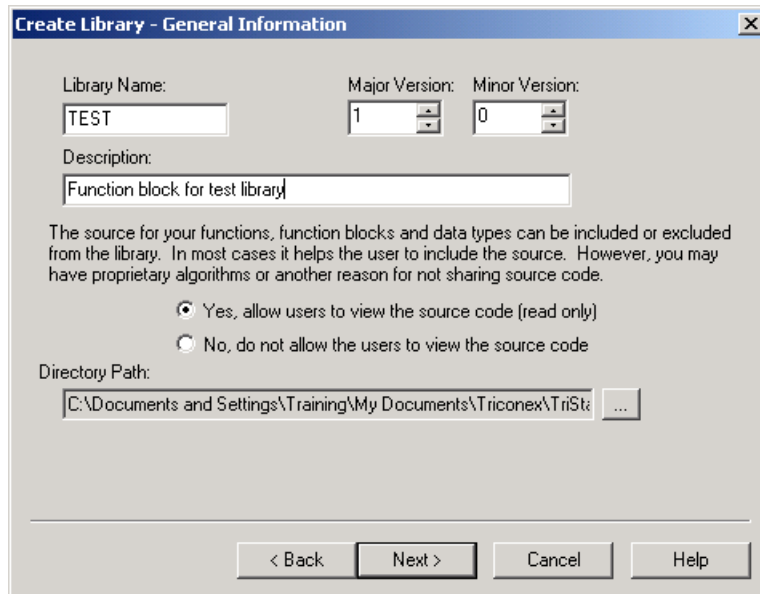


- 5 Click **Create New Library Specification**, and click **Next**.

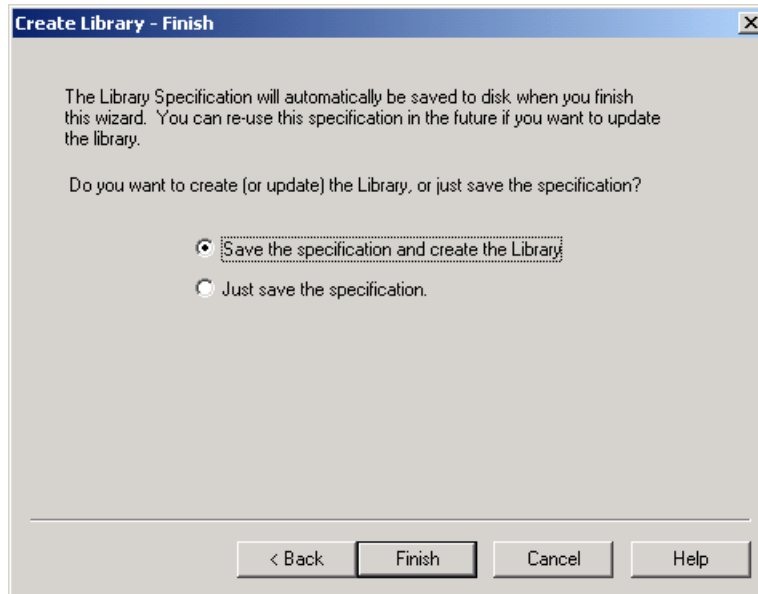
The **Create Library - Documents** screen is displayed.



- 6 Click the **Test** document in the left box, click the **angle brackets (>>)** to move the element to the right side, and then click **Next**.



- 7 Add information about the library, including a name, description, and major and minor version number. You can also specify whether the source code can be viewed or restricted from other users. Click **Next** to continue.



- 8 Click **Save the specifications and create the Library**, and click **Finish**. A test.lsp file and a test.lt2 file are created.

Note You can save the specification, but not create the library, by clicking **Just save the specification**. You might want to do this if you are planning to create a project library, but are not finished with the documents in the project. If you save just the specification, a library.lsp file is created. You can open this file later and create the file based on the specifications.

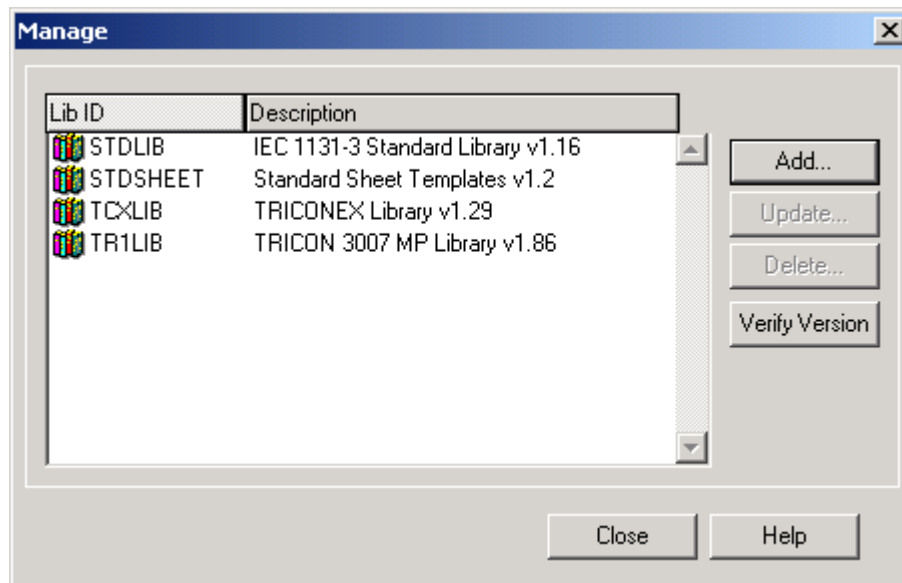
- 9 The function block named **TEST** is now stored in the library and can be deleted from this project. In the **Application** tree, expand **User Documents**, and then expand **Function Blocks**. Right-click **TEST**, click **Delete**, and click **Yes**.

Lesson 15: Managing Libraries

This lesson explains how to manage libraries, which are collections of project elements (programs, functions, and data types) that can be used in a project. **TriStation 1131** automatically includes IEC libraries with functions, function blocks, and data types that can be copied and sometimes modified for a project. You can also add libraries of project elements that were created in other **TriStation 1131** projects.

Procedure

- 1 Expand the **Application** tree, right-click **Library Documents**, and click **Manage**.



- 2 These are the actions that can be taken on the **Manage** screen.

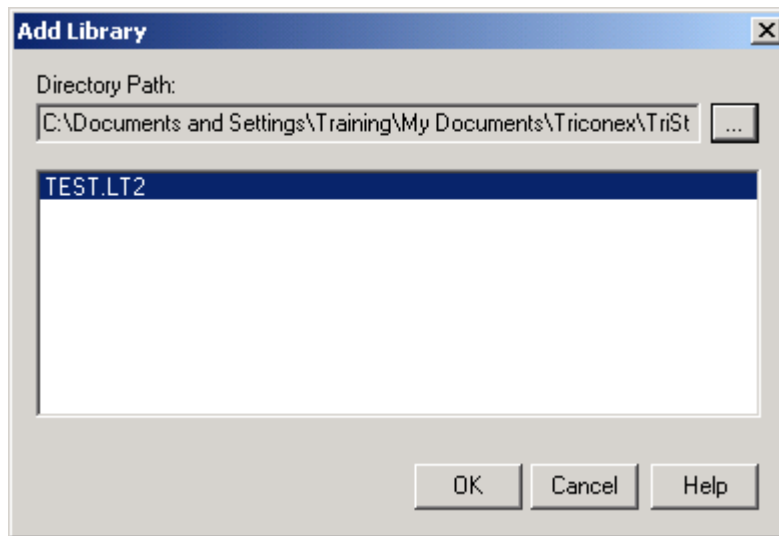
Command	Action
Add	Click to add a new library.
Update	Click to update an existing library.
Delete	Click to delete and existing library.
Verify Version Command	Click to verify the most current version is loaded.

Lesson 16: Adding Libraries

This lesson explains how to add libraries to a **TriStation 1131** project, which allows you to update libraries provided by Triconex and to add libraries of project elements from other projects.

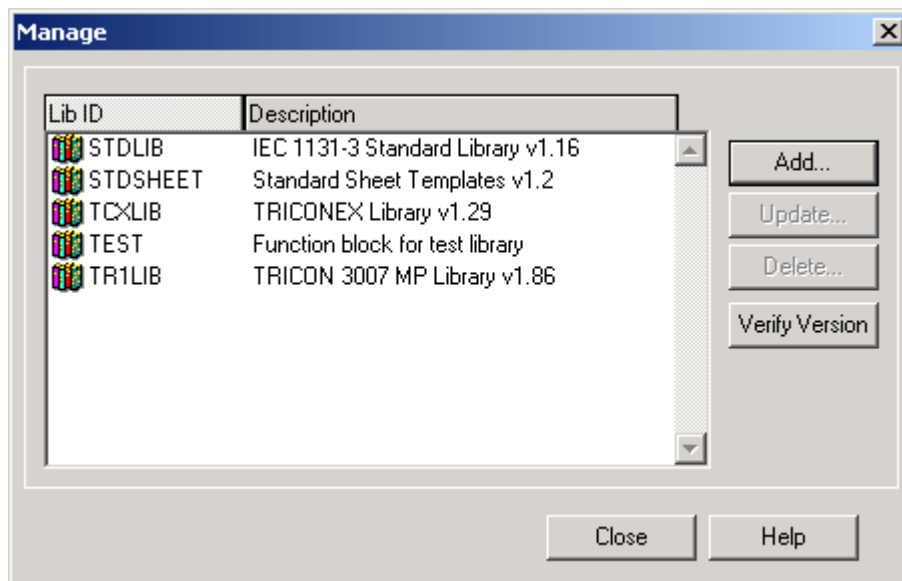
Procedure

- 1 Expand the **Application** tree, right-click **Library Documents**, click **Manage**, and then click **Add**.



- 2 Click **TEST.LT2**, and then click **OK**.
- 3 Click **Yes** on the Query screen.

The library is added to the project, and is displayed on the list.

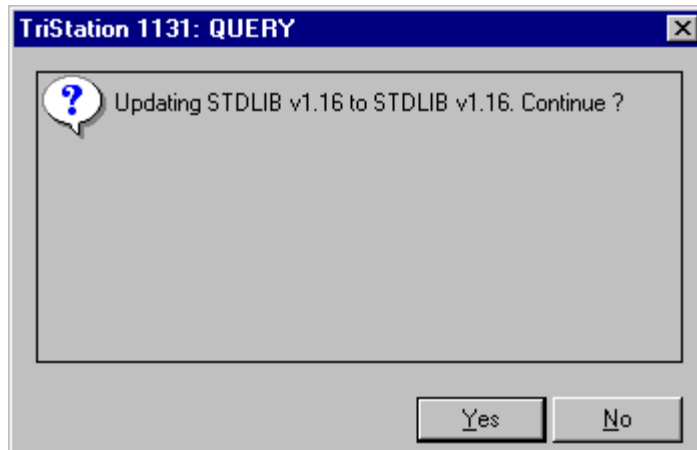


Lesson 17: Updating Libraries

This lesson explains how to update **TriStation 1131** libraries for your project. When you request an update, **TriStation 1131** compares the library in the project with the most current library installed and displays a message indicating the versions of each. You can update the library or cancel the operation.

Procedure

- 1 Expand the **Application** tree, right-click **Library Documents**, and then click **Manage**.
- 2 On the **Manage** screen, select **STDLIB**, and then click **Update**.



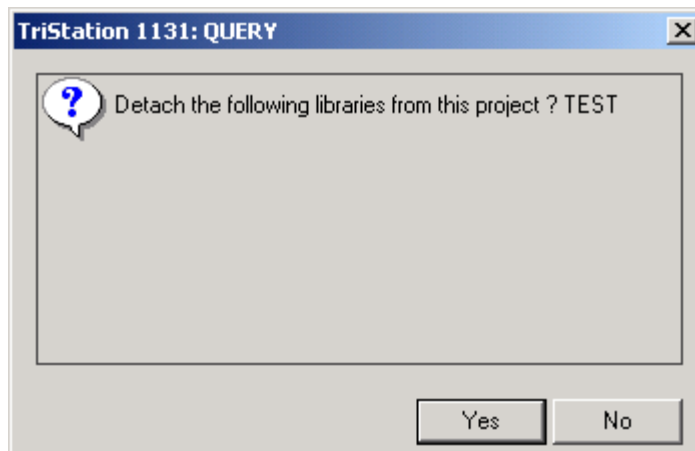
- 3 Click **Yes** to update.

Lesson 18: Deleting Libraries

This lesson explains how to delete a library from a **TriStation 1131** project.

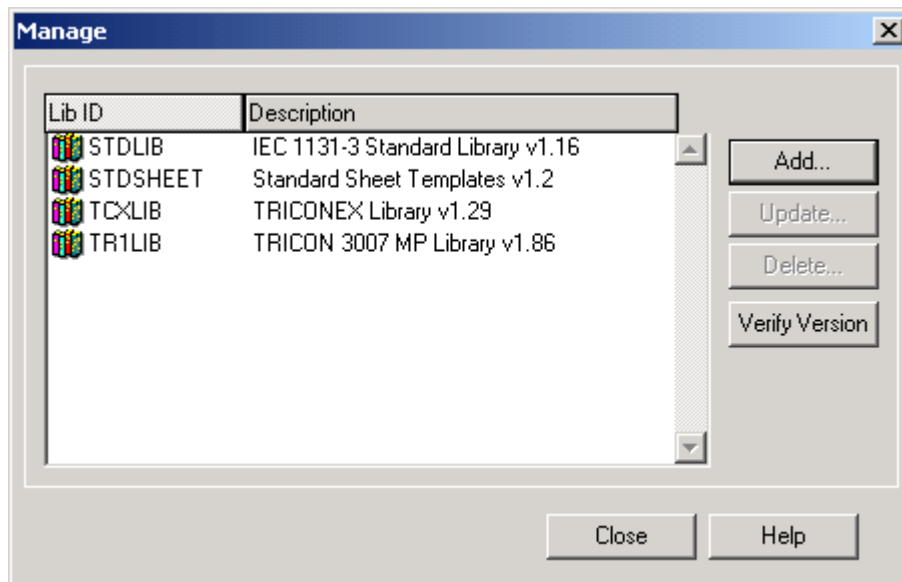
Procedure

- 1 Expand the **Application** tree, right-click **Library Documents**, click **Manage**.
- 2 On the **Manage** screen, select **TEST**, and then click **Delete**.



- 3 Click **Yes** on the Query screen.

The library is deleted from the project, and is removed from the list.

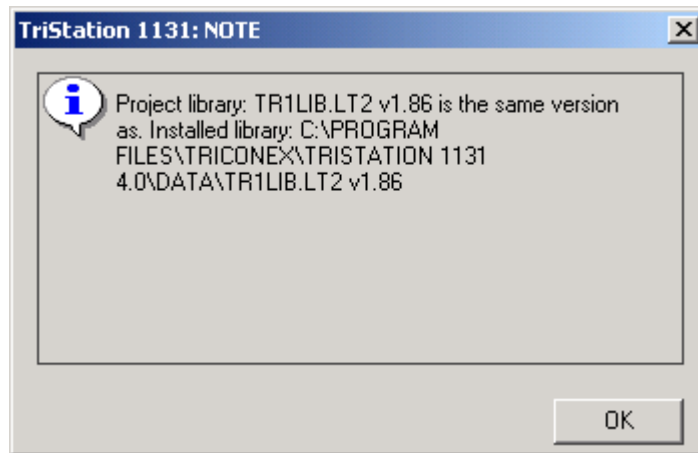


Lesson 19: Verifying a Library Version

This lesson explains how to determine the version of **TriStation 1131** libraries used in your project.

Procedure

- 1 Expand the **Application** tree, right-click **Library Documents**, click **Manage**, and then click **Verify**.



- 2 Click **OK** to close the screens.

Lab 3

Review the setup procedures by doing the following:

- 1 Open the **Test_Project** and add another user named Client with Level 10 security.
- 2 Change the user title of Level 10 to Client.
- 3 For Controller Operations, assign Level 03 to Run Application.
- 4 For TriStation 1131 Operations, assign Level 10 to Print Reports.
- 5 In project description, add "Tank Alarm."
- 6 For project options:
 - Change the border of the annotation box to double.
 - Change the CEM Monitor Color TRUE to green for Causes, Effects, and Intersections.
 - Change the CEM Monitor Color FALSE to red for Causes, Effects, and Intersections.
- 7 For TriStation 1131 options:
 - Change the Element Text and Borders to the default color.
 - Change the sheet background color to light blue.
 - Change the CEM Default Number of Cause Rows to 10.
 - Change the CEM Default Number of Effect Columns to 10.
 - Enable EN/ENO Terminals on Functions by Default in the FBD Editor.
 - Enable Double Space Function (Block) Terminals by Default in the LD Editor.

Writing Function Block Logic

Function Block Diagrams	94
Selecting the Sheet Template	100
Placing a Function on the Logic Sheet	103
Placing Variables on the Logic Sheet	108
Adding a Comment	118
Inserting a Network Divider	122
Compiling the Program	123
Printing Logic Sheets	125

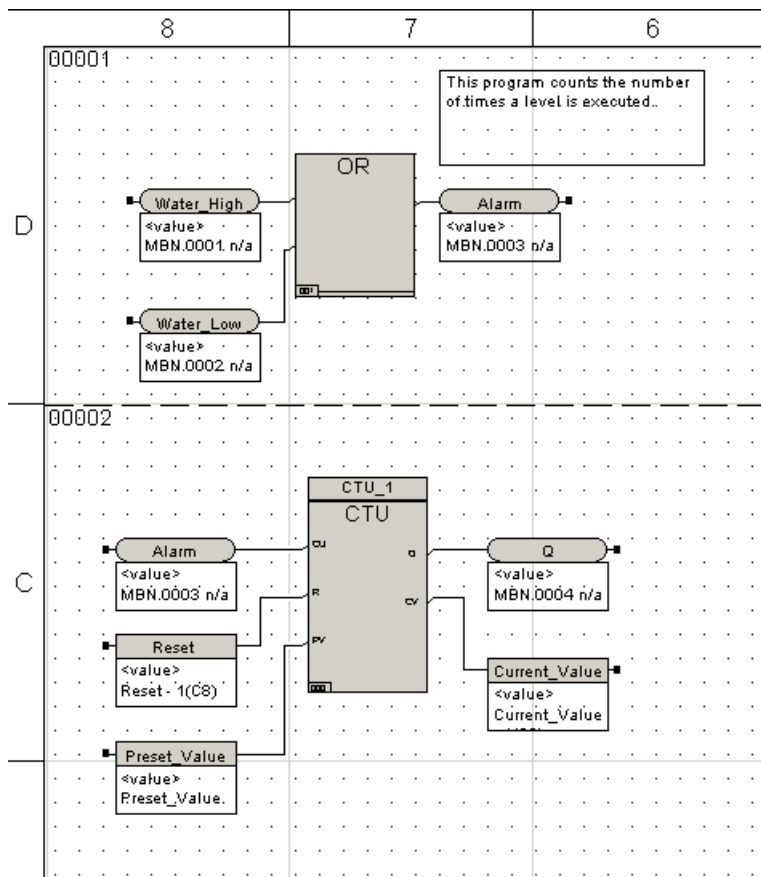
Function Block Diagrams

Function Block Diagrams are created using the FBD editor, which is the default programming language.

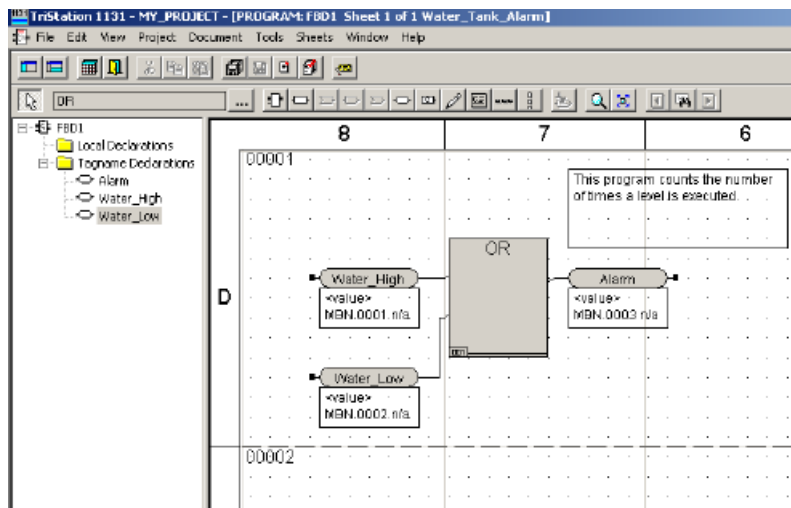
The basic steps for writing FBD logic are:

- Creating a user document.
- Placing and connecting the functions, function blocks, and variables on the logic sheet.
- Declaring the variables.
- Adding annotations and comments.
- Compiling the program and correcting any errors.
- Printing a copy of the logic sheet, as needed.

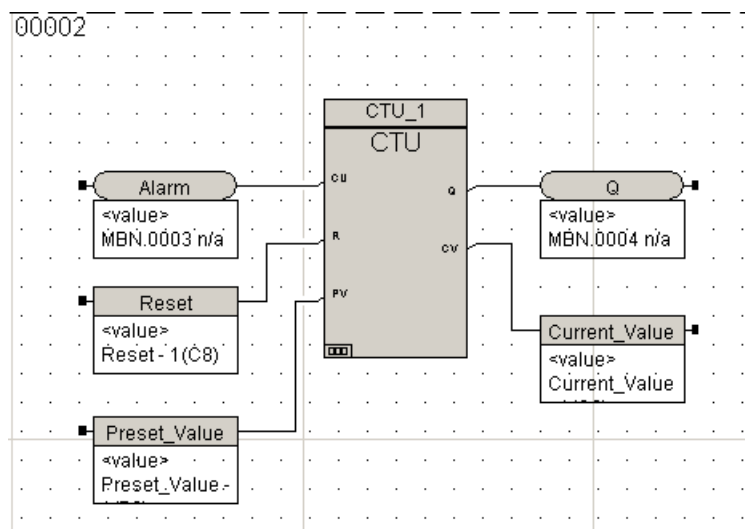
In this chapter, you will write FBD logic for a water tank alarm.



In the upper network, an input connects to a sensor (OR function) to detect if the water level is too high. A second input detects if the water level is too low. The output of the logic sets an alarm if the water level is too high or too low.



In the second network, a Count Up (CTU) function block counts the number of times the output alarm goes TRUE. The counter receives input from the alarm each time it goes off. When the count up (CU) input is true, the value is incremented by one each time until it is greater than or equal to the preset value (PV).



Lesson 20: Creating a User Document

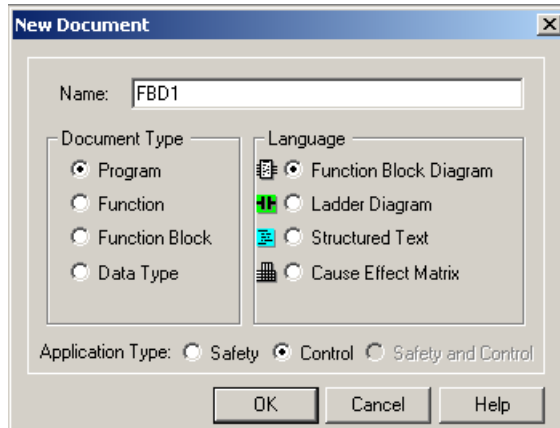
Project logic is created as user documents. User documents include programs, functions, function blocks, and data types.

In this lesson, you will:

- Create a new user document.
- Name the program FBD1.
- Enable Annotation on by default.
- Enable Monitor Value on by default.

Procedure

- 1 Open the project you created in Chapter 2.
- 2 Expand the **Application** tree, right-click the **User Documents** folder, and then click **New Document**.

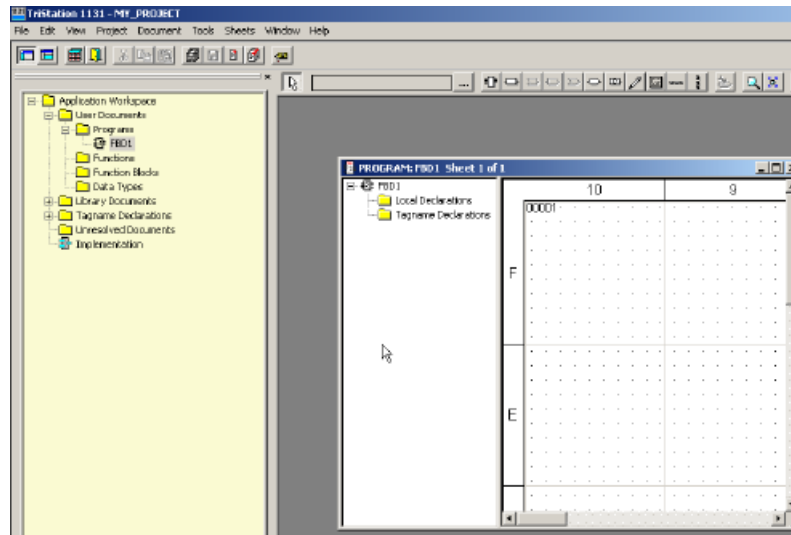


- 3 Specify the following:

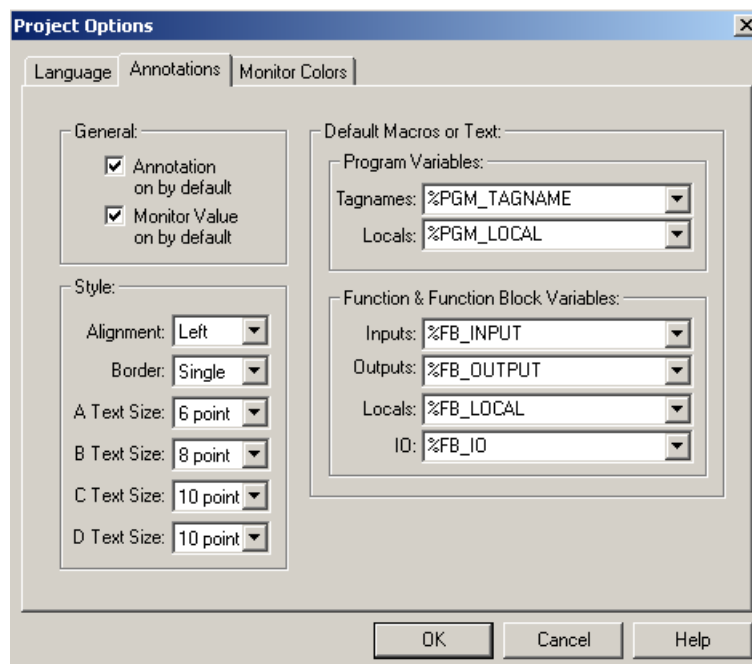
Name:	Enter FBD1
Document Type:	Click Program
Language:	Click Function Block Diagram
Application Type:	Click Control

- 4 Click **OK** to save.

The document is opened in the FBD editor, displaying the logic sheet.

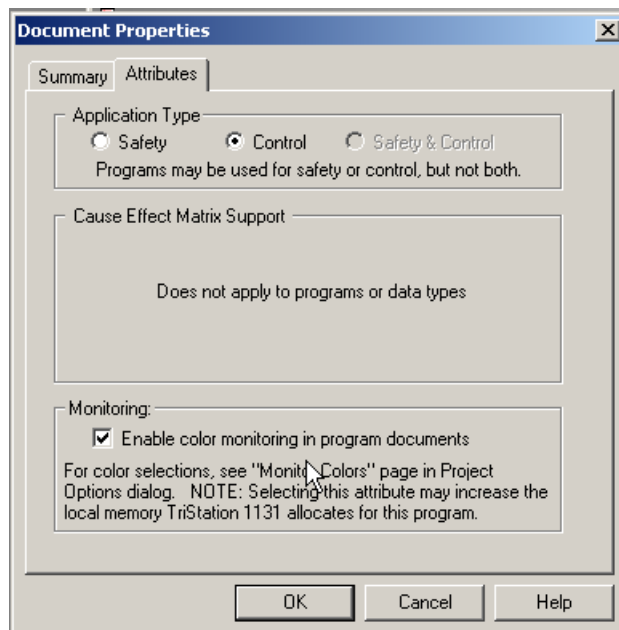


- 5 On the **Project** menu, click **Project Options**, and then click the **Annotations** tab.




- 6 Check **Annotation on by default**. Check **Monitor Value on by default**. Click **OK**.

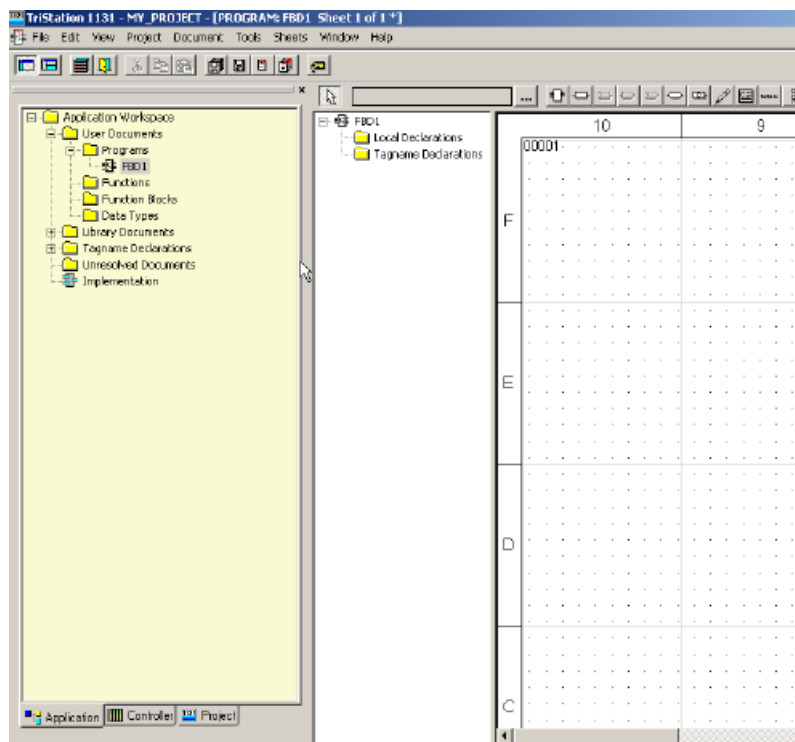
- 7 On the **Documents** menu, click **Properties**, and then click the **Attributes** tab.



- 8 Check **Enable color monitoring in program documents**, and then click **OK**.

This enables the system to display the logic in color when run in the emulator or controller.

- 9 Click the **Workspace View** button  to maximize the display.



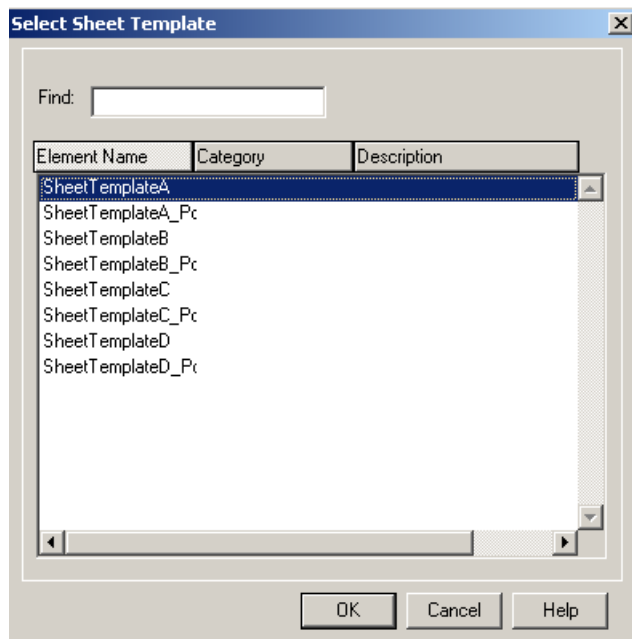
Lesson 21: Selecting the Sheet Template

When you create a new document, **TriStation 1131** automatically displays a default template for the logic sheet. The Sheets command allows you to change the default template, give the logic sheet a title, and enter information in the sheet descriptions fields.

In this lesson, you will change the default to Sheet A.

Procedure

- 1 On the **Sheets** menu, click **Select Sheet Template**.



The following sheet sizes are available:

- Sheet A- 8.5" x 11"
- Sheet B- 11" x 17" (default)
- Sheet C- 17" x 21"
- Sheet D- 21" x 33"

- 2 Click **Sheet Template A**.
- 3 Click **OK** to save.

Lesson 22: Specifying Sheet Title Properties

The Sheet Title box includes the title of logic sheet and other information, such as approval, drawing number and title. The sheet title is displayed in the title block, the window caption bar, and the Window Menu list.

When you open a new program, function or function block, some of the fields in the sheet description are automatically filled by **TriStation**. You can modify these fields and fill in other fields.

There are two ways to enter a title and information on the logic sheet:

- Using the Edit Sheet Title option in the Sheets menu
- Using the Item Properties screen

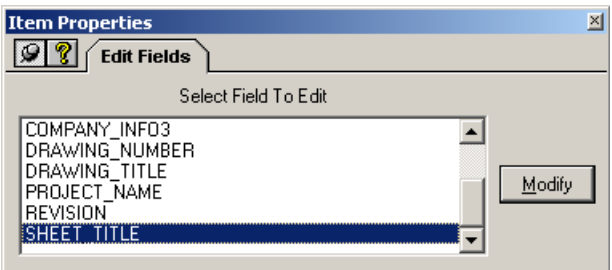
In this lesson, you will enter a title for the logic sheet using the Item Properties screen.

Procedure

- 1 Scroll to the bottom right corner of the logic sheet to the **Sheet Title** box.

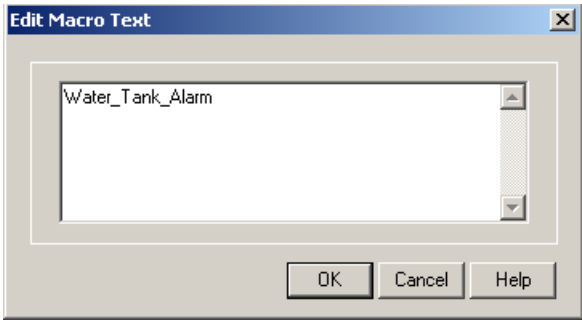
PROJECT		Class Project	
ACTIVITY	DATE/TIME	DRAWING TITLE	
CREATED BY MANAGER	12/18/2003 4:28:05 PM		
MODIFIED BY		SIZE A	DRAWING NO.
PRINTED BY		SHEET TITLE	
APPROVED BY		SHEET 1 of 1	
SHEET TEMPLATE SheetTemplateA		CLIENT FBD1 v0.0	

- 2 Double-click anywhere in the **Sheet Title** box. The **Item Properties** screen is displayed.



- 3 Click **Sheet_Title**.

The **Edit Macro Text** screen is displayed.



- 4 Enter **Water_Tank_Alarm**.
- 5 Click **OK** to save **Water_Tank_Alarm** as the Sheet Title.

		PROJECT		Class Project	
ACTIVITY		DRAWING TITLE			
CREATED BY Manager	DATE/TIME 12/18/2003 4:25:05 PM				
MODIFIED BY		SIZE A	DRAWING NO.	REV.	
PRINTED BY					
APPROVED BY		SHEET TITLE Water Tank Alarm		SHEET 1 of 1	
SHEET TEMPLATE SheetTemplateA		SUBJECT FBD110.0			

- 6 To edit or enter information in other fields, click a field name in the **Edit Fields** window.
- 7 Enter text, click **OK** to save, then exit the **Item Properties** screen.

Lesson 23: Placing a Function on the Logic Sheet


There are two ways to place a function on the logic sheet:

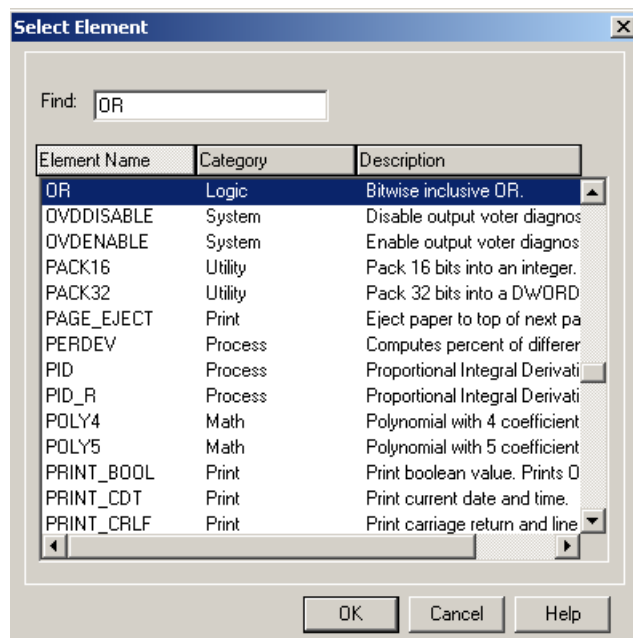
- Clicking the Select Function (Block) Tool button
- Using the Tools Menu

After placing a function on the logic sheet, you can specify properties, such as number of inputs and terminal features.

In this lesson, you will place an OR function on the logic sheet.

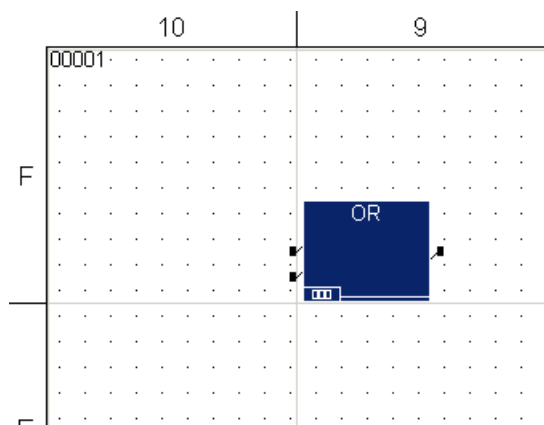
Procedure

- 1 Click the Function Block Selection button  on the toolbar. The Select Element screen is displayed.



- 2 To find the OR function, do either of the following:
 - Enter **OR** in the field to highlight the function, then click OK.
 - Scroll down the Element Name list, highlight the function, and then click OK.

- 3 Move the pointer to the logic sheet. The pointer changes to the function graphic. Click once to place the function on the sheet.



Lesson 24: Specifying Function Properties

The Item Properties screen for a function has only one tab with several modifiable fields.

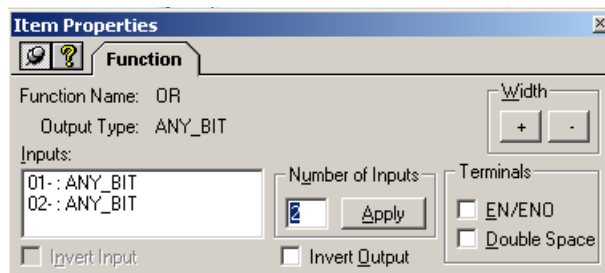
All functions follow these IEC 61131-3 standards:

- The inputs of most functions have a generic data type that begins with the prefix ANY.
- All of the inputs to a function must have the same data type. Otherwise, a type mismatch error displays when you build the application.
- A function has only one output, which must have the same data type as the inputs to the function.
- Some functions are extensible, which means you can add up to 50 inputs to the function.

In this lesson, you will specify the properties of the OR function.

Procedure

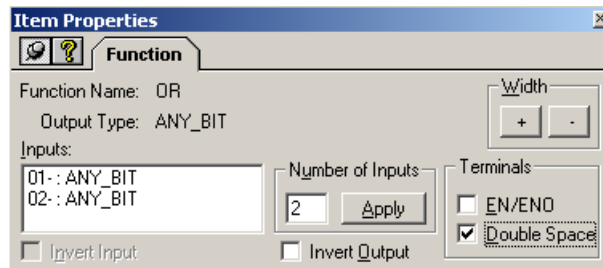
- 1 Double-click the OR function to display Item Properties screen.



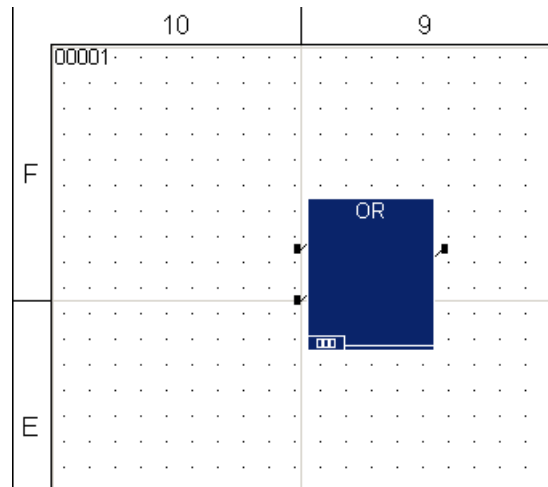
The Function tab has the following fields:

Property	Action
Inputs	Lists the names and data type of the inputs to the function.
Number of Inputs	If the number of inputs is enabled, the function is extensible and the number of inputs can be changed.
Invert Output	If an input is inverted, the value is changed to the opposite (True to False, or False to True) when the function is executed. Only available for BOOL inputs.
Width	The plus (+) or (-) minus button expands or shrink the width of the variable symbol so you can use a longer name or fit the symbol into a smaller space.
EN/ENO	Includes an input and output parameter that detects errors in FBD and LD logic.
Double Space	Doubles the space between the inputs and outputs to provide more space for annotations and comments.

- 2 Check Double Space to double the spaces between the terminals on the OR graphic.



- 3 Close the **Item Properties** screen. The OR function graphic is displayed with double spaces between the terminals.



Lesson 25: Placing Variables on the Logic Sheet


There are two ways to place variables on the logic sheet:

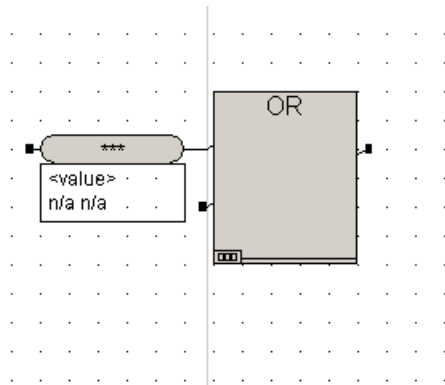
- Clicking the corresponding icon on the toolbar
- Using the Tools Menu

In this lesson, you will:

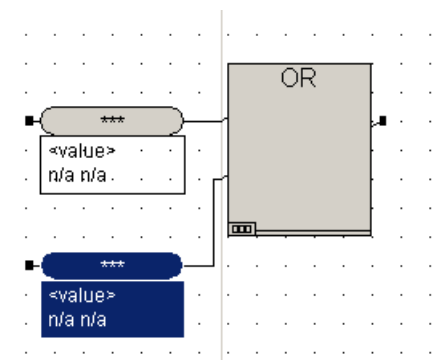
- Place two input variables and an output variable on the logic sheet.
- Connect the variables to the function.

Procedure

- 1 Click the **Tagname Tool** button  and move the pointer to the logic sheet.
- 2 Connect the tagname to the top input terminal of the function by clicking once. The input variable is displayed.

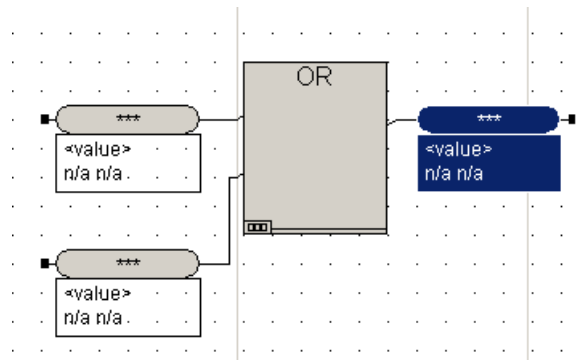


- 3 Add a second tagname to the bottom input terminal by clicking the **Tagname Tool** button, moving the pointer to the bottom terminal, and clicking once. The bottom input variable is displayed.



Note: You may need to reposition the variables with the pointer so that all elements and connections can be seen.

- 4 Add an output variable to the output terminal by clicking the **Tagname Tool** button, moving the pointer to output terminal, and clicking once. The output variable is displayed.



Note that the default annotations are displayed for each variable.

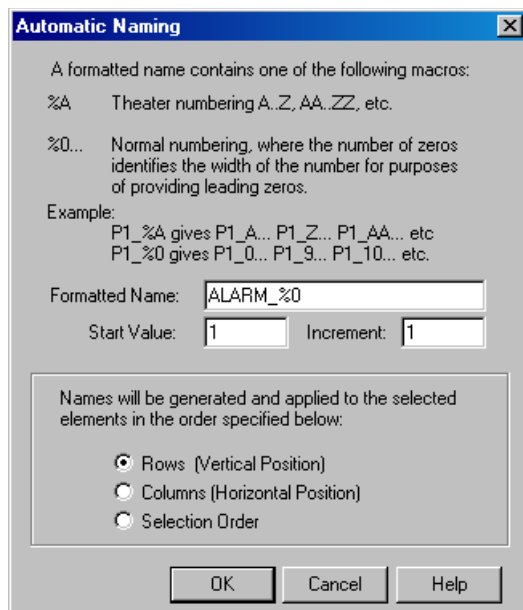
Declaring Variables

Variables store values and must be "declared" or assigned a name and data type to run in a Triconex controller. Declaring assigns each variable to a memory point located in the controller memory.

A variable must be named before other properties can be specified. When naming a variable, you can enter up to 31 alphanumeric and underscore characters. Names are not case sensitive. Remember to use underscore _ to separate characters; no blank spaces.

The Auto Name feature is useful when you have several similar variables and want to name them according to a convention. You can specify a name, starting value, and incremental value. You can also specify the order in which to apply the names to the selected elements and the increment to use.

For example, if you have 10 alarm detector inputs, and you want to name them ALARM_1, ALARM_2, ALARM_3, etc., you can use Auto Name to name them automatically.



In this example, you would enter ALARM_DETECT_%0 to start the automatic names at 1.

You can also enter a text description and group names, and specify an initial value to be used when the system starts up. The initial value specifies the beginning value assigned to variable on the first scan of the application. The default is zero.

Lesson 26: Declaring Program Variables for OR Function

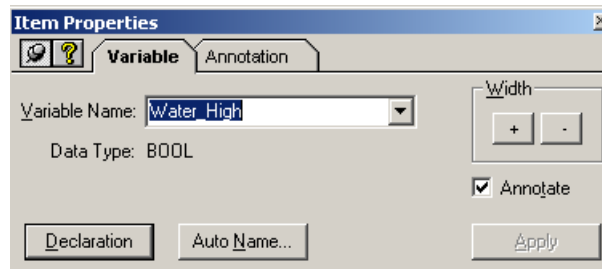
All variables must be declared to be used in a program or function. You have created three tagnames (global variables) for the FBD program: two inputs and one output.

In this lesson, you will:

- Declare the input and output variables.
- Specify properties for the variables.

Procedure

- 1 Double-click the top input variable. The **Item Properties** screen is displayed.

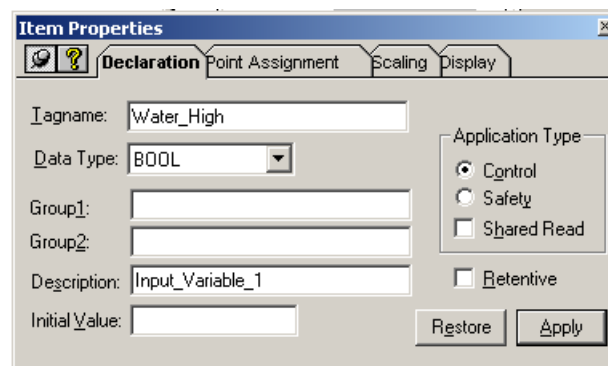


- 2 Enter **Water_High** in the Variable Name field.

The Annotate box is checked because it was specified when you first set up the project. It will automatically add an annotation box to this variable.

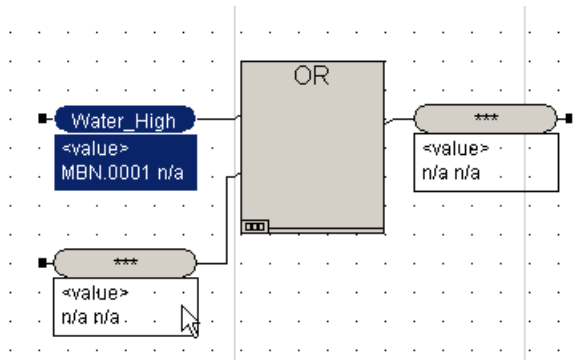
Note that the data type is displayed. If a tagname is placed or dragged to the logic sheet using the pointer, the data type is automatically set to the correct type for the variable.

- 3 Click **Apply**. The **Declaration** tab is displayed.

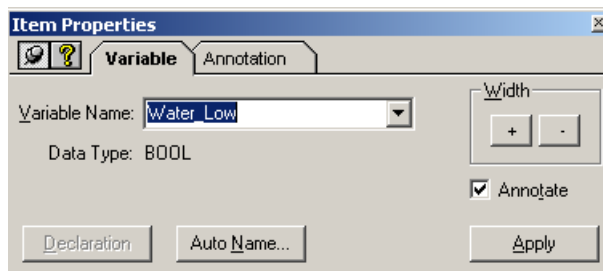


- 4 Enter **Input_Variable_1** as the description. Confirm **BOOL** as the **Data Type**. Click **Apply**, and then close the **Declaration** tab.

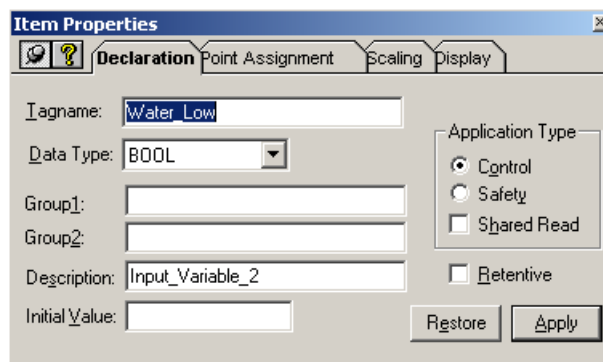
The named variable is displayed on the logic sheet.



- 5 Double-click the second input variable. The **Item Properties** screen is displayed.

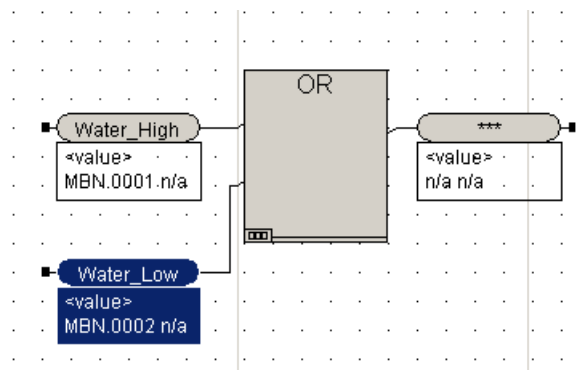


- 6 Enter **Water_Low** in the **Variable Name** field. Click **Apply**. The **Declaration** tab is displayed.

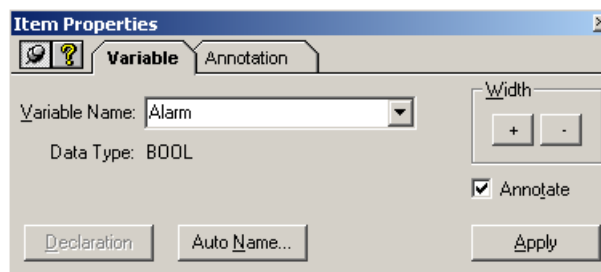


- 7 Enter **Input_variable_2** in the **Description** field. Confirm **BOOL** as the **Data Type**. Click **Apply**, and then close the **Declaration** tab.

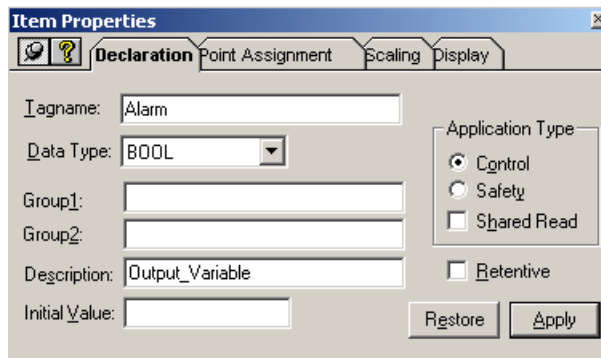
The named variable is displayed on the logic sheet.



- 8 Double-click the output variable. The **Item Properties** screen is displayed.

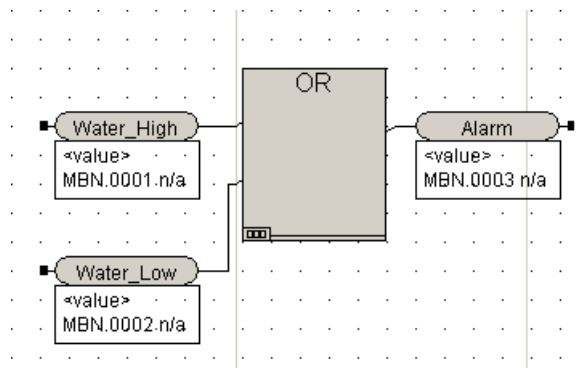


- 9 Enter **Alarm** in the **Variable Name** field. Click **Apply**. The **Declaration** tab is displayed.



- 10 Enter **Output_Variable** in the **Description** field. Confirm **BOOL** as the **Data Type**. Click **Apply**, and then close the **Declaration** tab.

The named variable is displayed on the logic sheet.

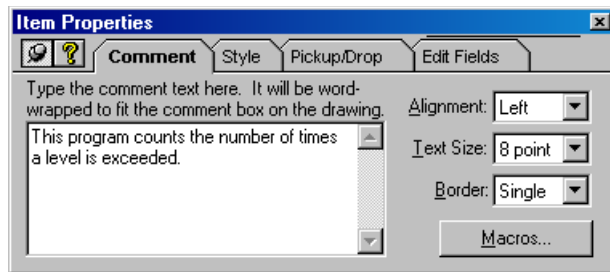


Adding Comments

Comments are text boxes used to describe operations or tasks performed by a program, function, or function block. There is no limit on the number of comment boxes you can place on a logic sheet.

The comment box can be placed anywhere on the logic sheet. You can draw comment boxes around or on top of any project element and specify properties for the text, alignment and borders. You can also include macros in the comment.

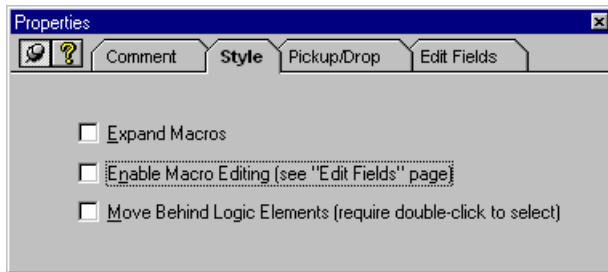
Comment Tab



You can enter text and macros in the comment field. You can adjust comment text in three ways:

- Alignment- Left, right, or center
- Text Size- 3 points to 24 points
- Border- Single, double, or no border around the comment box

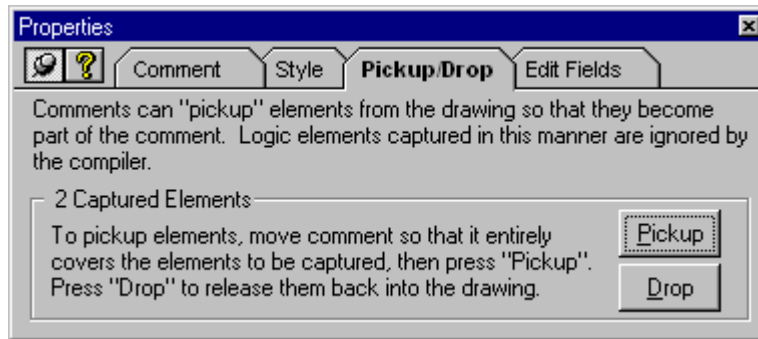
Style Tab



There are three fields under the Style tab:

Property	Action
Expand Macros	Specifies whether the value or the name of the macro is displayed. If checked the value (not the name) of the macro is displayed when the application is run in the emulator and controller. For example, if expanded, the macro %DATE_CREATED displays the month, day, and year when the project was created.
Enable Macro Editing	This property allows you to edit the values of macros that can be modified, using the Edit Fields tab.
Move Behind Logic Elements	This property allows you to create a comment box behind a logic element, enclosing the element with descriptive text.

Pickup/Drop Tab

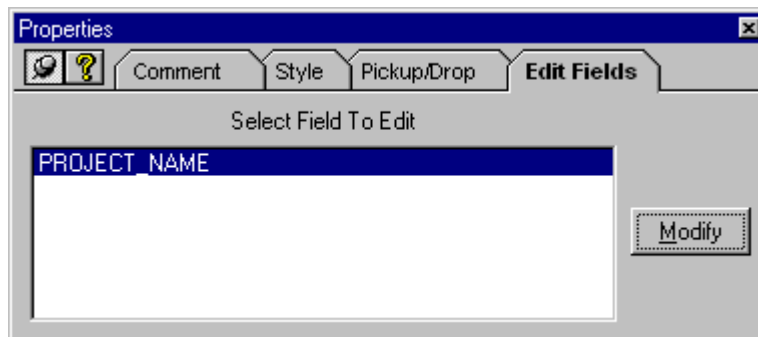


The Pickup/Drop tab allows a comment box to include elements so that they become part of the comment.

Pickup is used to include the element in the comment box. Drop releases the element from the comment box. Logic elements captured in a comment box are not executable. You must release them from the comment box to make them executable.

This feature is useful for isolating and testing a specific logic element.


Edit Fields Tab

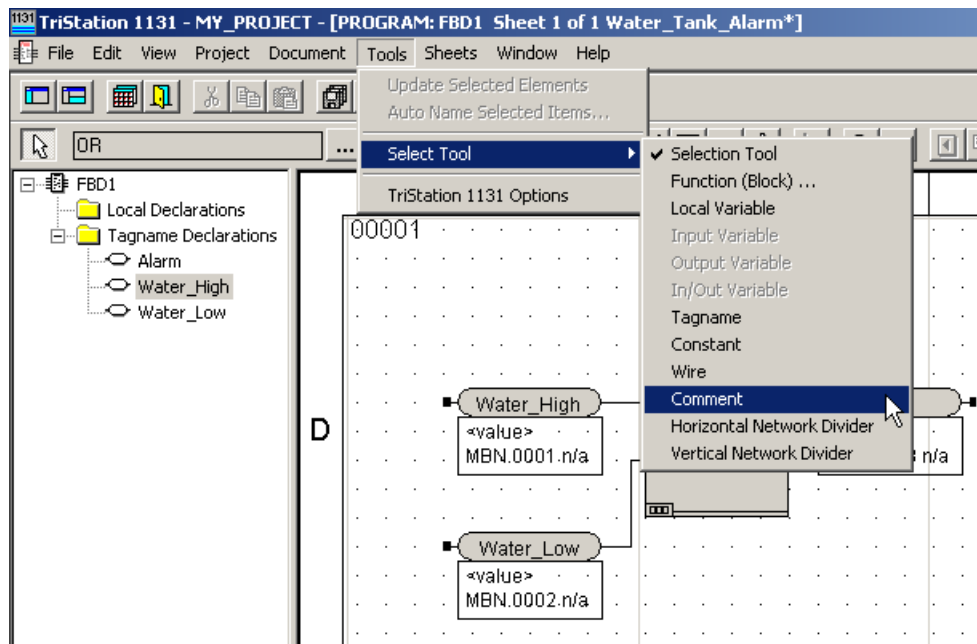


If you have included modifiable macros in your comment text, you can use the Edit Fields tab to modify them.

Lesson 27: Adding a Comment


There are two ways to place a comment box on the logic sheet:

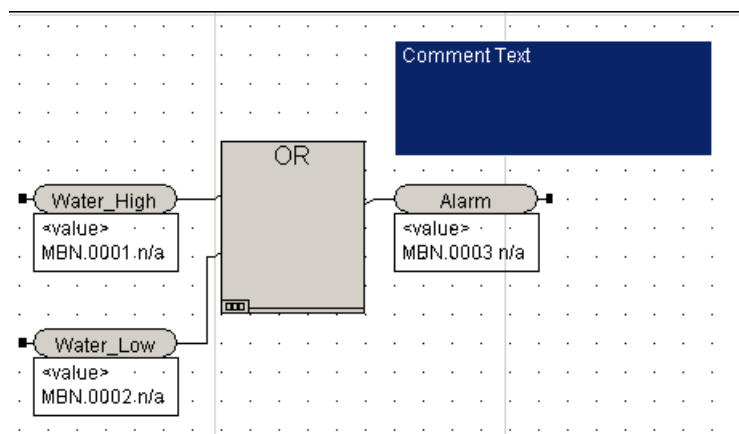
- Using the Comment Tool button  on the FBD toolbar
- Using the Comment option on the Select Tool feature of the Tools menu



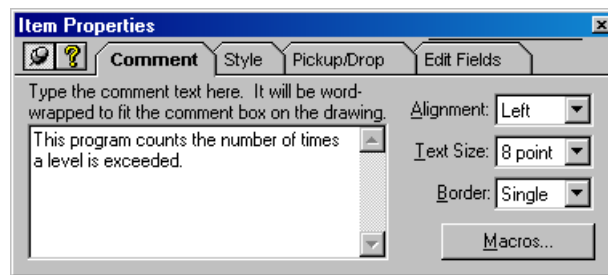
In this lesson, you will add a comment box to the FBD logic sheet using the Comment button.

Procedure

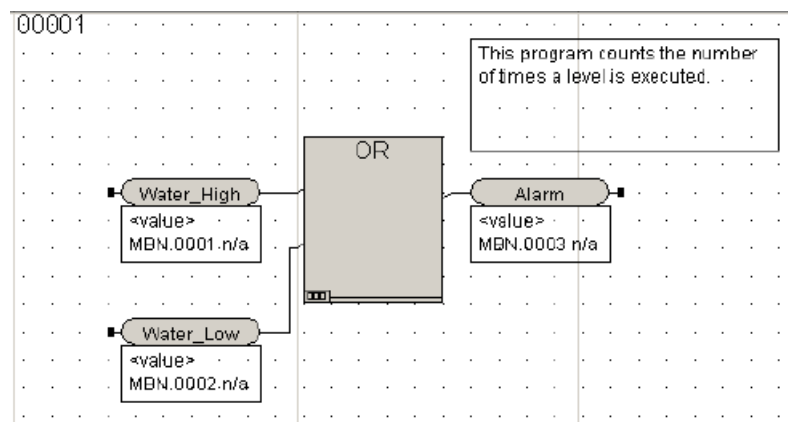
- 1 On the toolbar, click the **Comment** button .
- 2 On the logic sheet, click and drag to create a comment box.



- 3 Double-click the box to display the Item Properties screen.



- 4 Enter text to be included as a comment. You can specify other properties, such as alignment, text size, and border style from this tab.
- 5 Close the **Comment** tab. The comment text is displayed on the logic sheet.




Lesson 28: Using Macros with Comments

Macros are placeholders for text or information supplied by the system or the user. The value is displayed when the element is run in the emulator or controller.

Macros can only be used in FBD and LD development.

There are two types of macros:

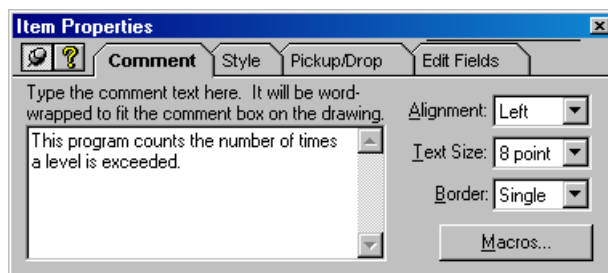
- System macros are supplied values by the system and cannot be changed. For example, the %CREATED_BY macro includes the user ID of the person who created the element. You cannot change values for these macros.
- User-modifiable macros, identified by a pencil icon  are values you can specify.

CAUTION When you change the value of a macro, it is changed for all comments and annotations that use the macro in the project.

In this lesson, you will add a macro to the comment.

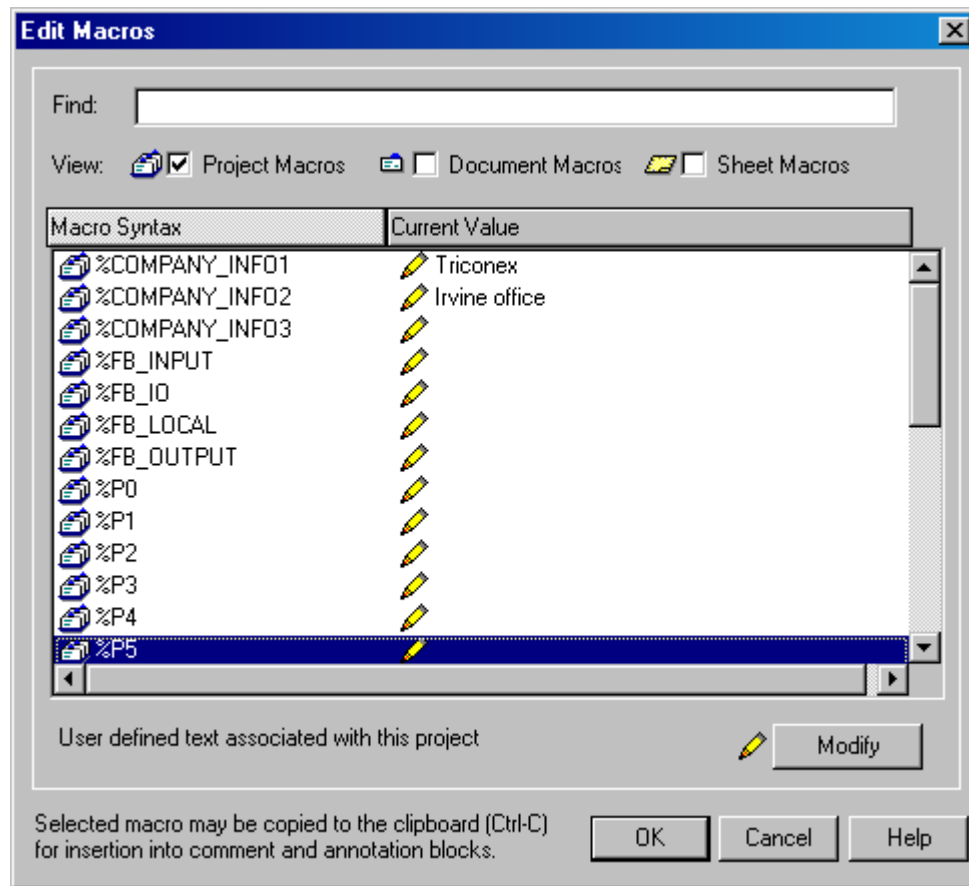
Procedure

- 1 Double-click the **Comment** box to display the **Item Properties** screen.



- 2 Click the Macros button.

The Edit Macros screen displays a list of macros that can be included with the comment text.





- 3 Scroll down to %DESCRIPTION, click to highlight, and copy it by pressing **Ctrl+c**.
- 4 Click **OK** to close the **Edit Macro** screen, click inside the text area of the **Comment** field, and paste the macro by pressing **Ctrl+v**. The macro is added to the comment.
- 5 Close the **Item Properties** screen.

Lesson 29: Inserting a Network Divider

In FBD, you can divide the logic sheet into different networks using horizontal and vertical dividers. The program logic executes in the order the network divider is placed: from left to right, and from top to bottom.


There are two ways to place network dividers on the logic sheet:

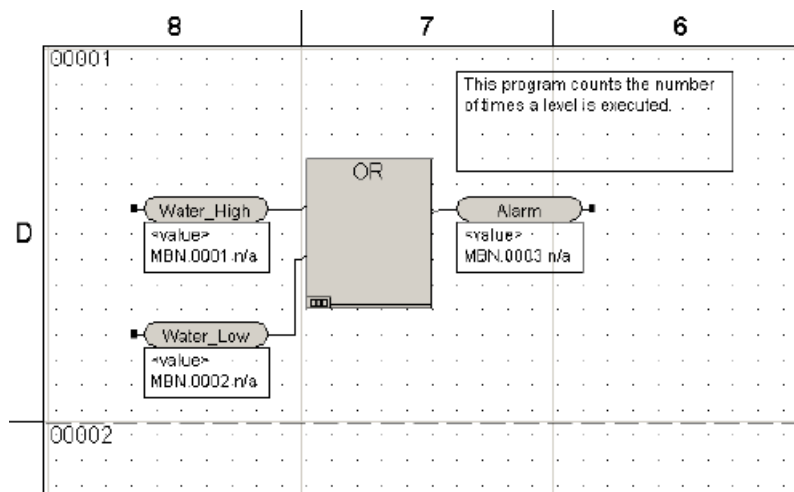
- Using the Horizontal Network Divider button  and the Vertical Network Divider button  on the FBD toolbar
- Using the Select Tool options on the Tools Menu

In this lesson, you will place a horizontal network divider on the logic sheet using the Horizontal Network Divider button.

In the practice exercise at the end of this chapter, you will write the program logic for this second network.

Procedure

- 1 Click the **Horizontal Network Divider** button  on the FBD toolbar. Move the pointer beneath the OR function to divide the logic sheet horizontally.
- 2 Click once to place the network divider on the logic sheet.

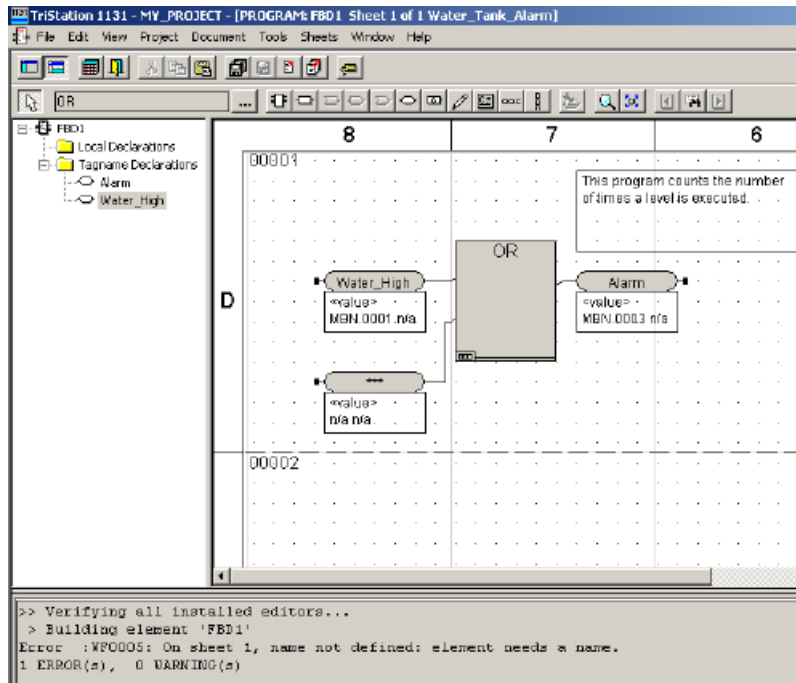


Note: A new reference number 00002 is placed on the logic sheet.

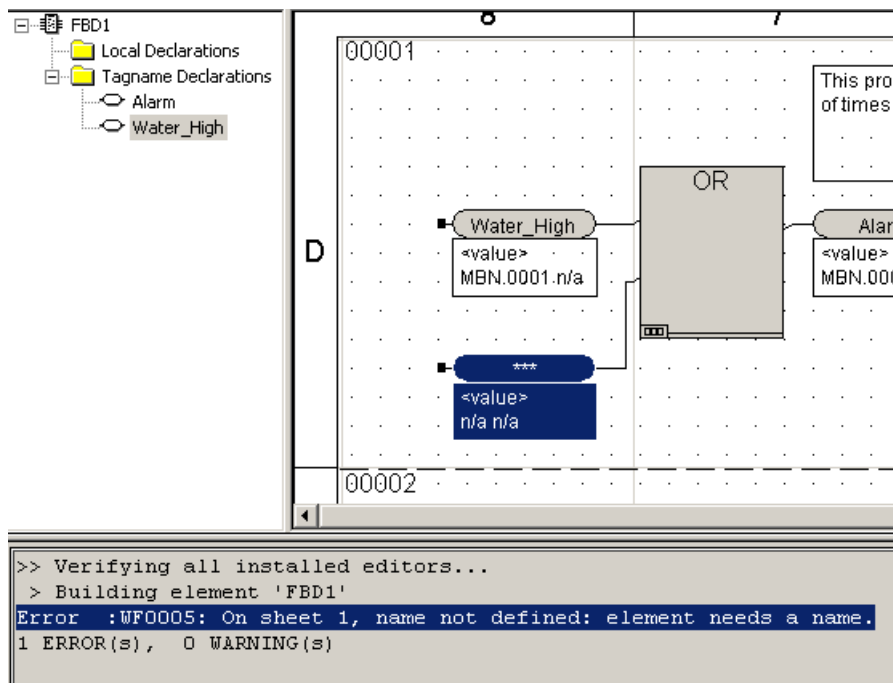
Lesson 30: Compiling the Program

Compiling a program is a way to verify your logic before building the application. Any errors will be displayed in the Message View.

For example, in this program an input element does not have a name and is flagged as an error.



You can click on each error statement and the location of the error is displayed.

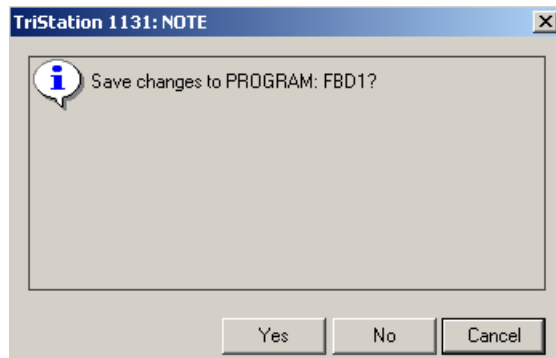


All errors must be corrected before building the program.

In this lesson, you will compile the program for the water tank alarm.

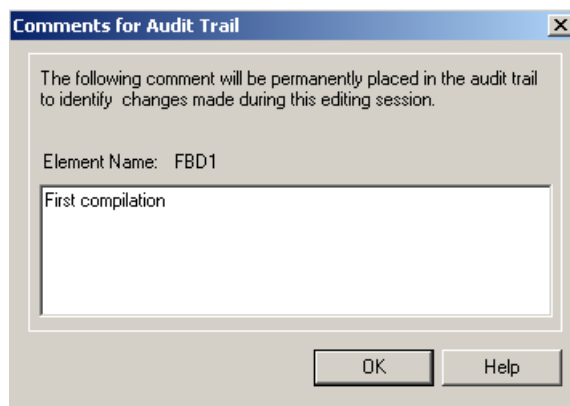
Procedure

- 1 On the **Document** menu, click **Compile**.

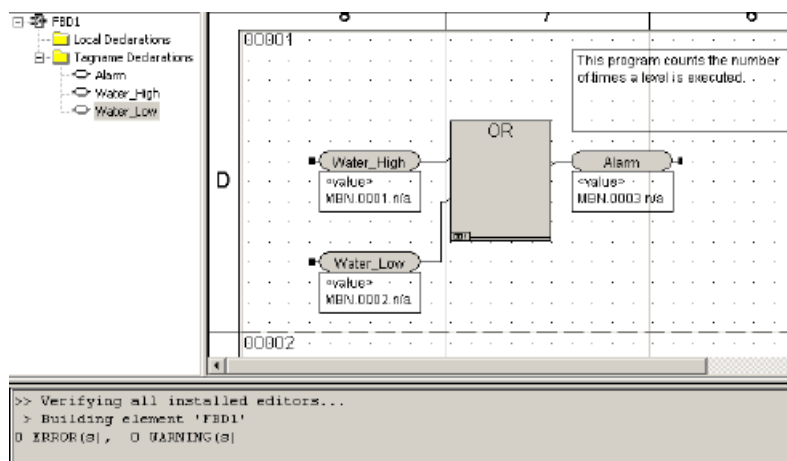


- 2 Click **OK** to save changes.

The **Comments for Audit Trail** screen is displayed. You can add comments when you change a program, function, or function block. The comments provide an audit trail that can be viewed on reports.



- 3 Enter a comment and click **OK**. The **Message View** automatically opens and displays the status of the compile process. Fix any errors and compile the program again.



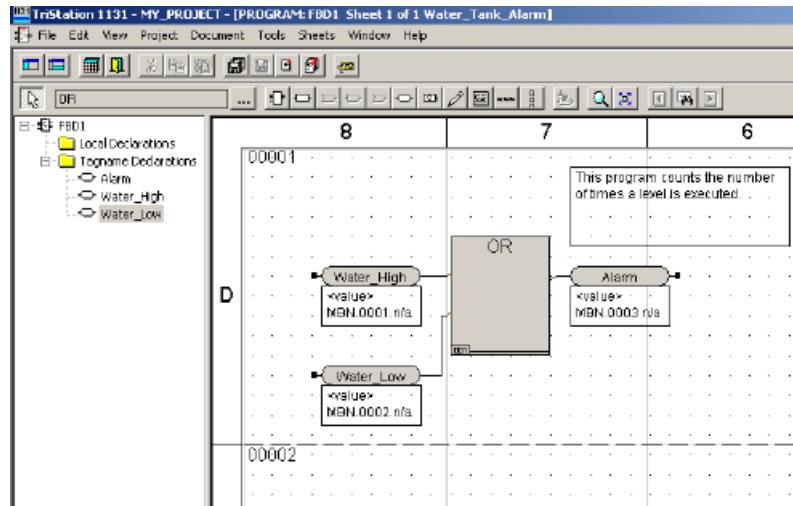
Lesson 31: Printing Logic Sheets

From the FBD, you can print the logic sheets displayed in the current window.

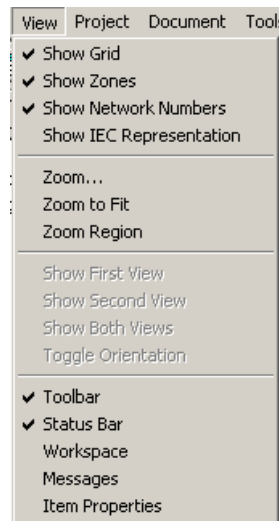
In this lesson, you will print the logic sheet for the OR function you created.

Procedure

- 1 Display the logic sheet.



- 2 Use the **View** menu to adjust the view of the logic sheet.



3 On the File menu, use the following commands as needed:

- **Print Preview** - Displays the sheets to be printed.
- **Print Setup** - Displays printer settings.
- **Print** - Displays copy and property settings.

4 On the **File** menu, click **Print**, and then click **OK**.

You have completed the logic for the first part of the water tank alarm.

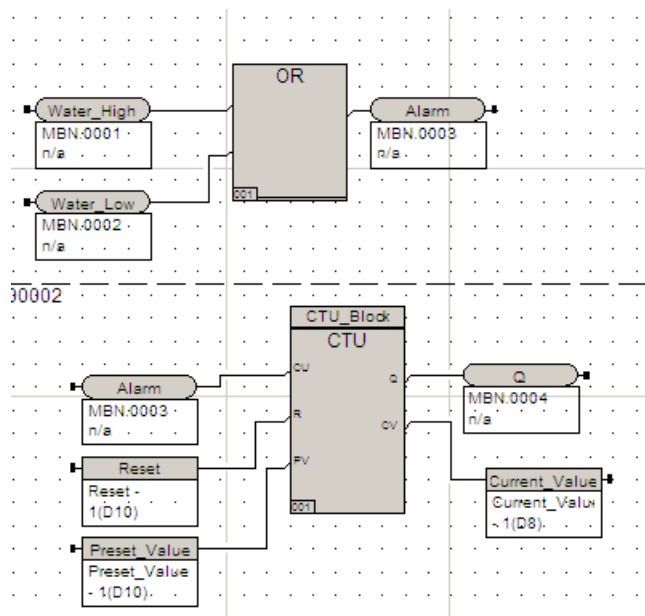
Lab 4

Review the procedures for writing FBD logic by completing the water tank alarm. You will add a counter that will count the number of times the water level alarm is set off.

In the second network, you will:


- Place a CTU function block on the logic sheet.
- Place variables on the logic sheet.
- Declare the variables for the CTU function block.
- Compile the program and print the logic sheet.

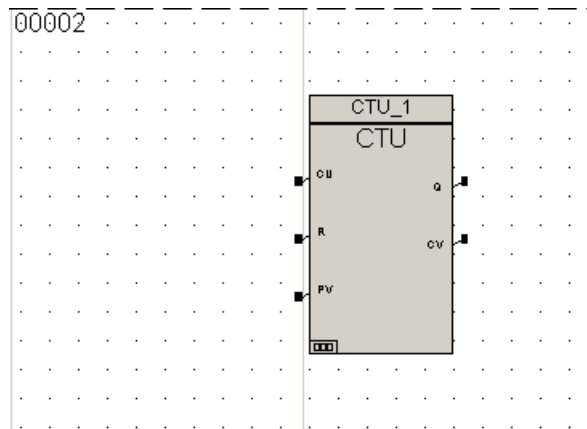
This is the completed logic sheet for the water tank alarm, showing both networks.



Procedure

Place a CTU function block on the logic sheet



- 1 Open the **Test_Project** project.
- 2 Use the **Select Function (Block)** button  to place a **CTU** on the logic sheet in the second network.
- 3 Double-click the **CTU** function block.
- 4 On the **Item Properties** screen, do the following:
 - Enter **CTU_1** in the Instance Name field.
 - Click **Apply**.
 - Check **double space**.
 - Close the **Item Properties**.



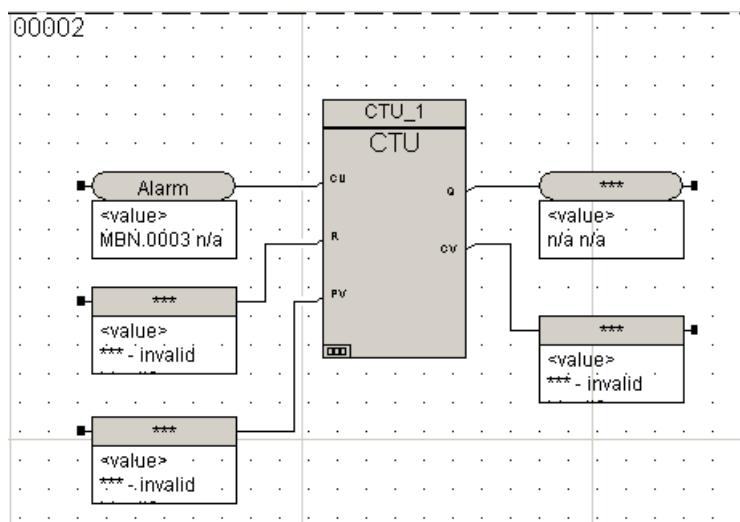
Place variables on the logic sheet

- 1 Pass the output value of the alarm to the input of the counter. To do this, copy and paste the output variable, **Alarm**, from the first network to top input (CU) of the CTU.

Note: When you use the output variable from a previous network as the input to another block of logic, the value passes from the output resultant to the input of the second network.

- 2 Use the **Local Variable Tool** button  and connect a variable to the middle input terminal (R for Reset) of the **CTU Function Block**.
- 3 Use the **Local Variable Tool** button and connect a variable to the bottom input terminal (PV for PreSet Value) of the **CTU Function Block**.
- 4 Use the **Tagname Tool** button  and connect a variable to the top output terminal (Q for True if Current Value = Preset Value) of the **CTU Function Block**.

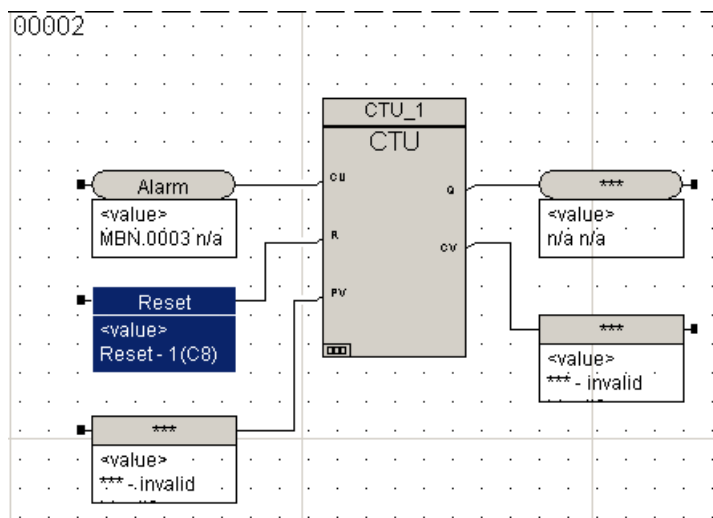
- 5 Use the **Local Variable Tool** button and connect a variable to the bottom input terminal (**CV for Current Value**) of the **CTU Function Block**.



Declare the variables for the CTU function block

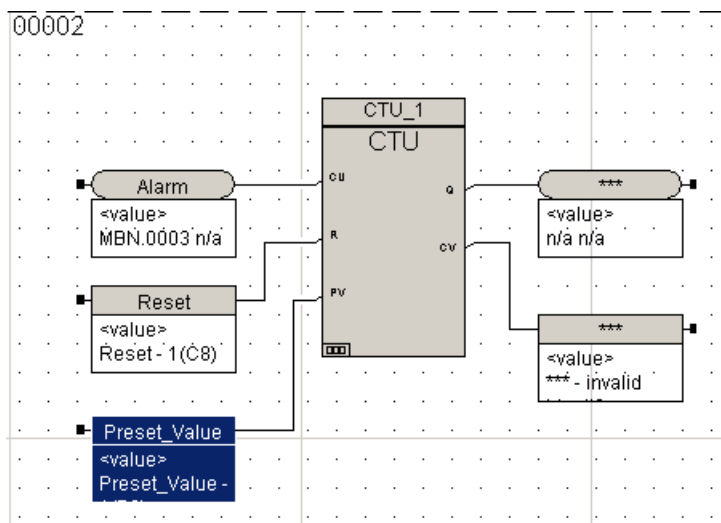
Reset Variable

- 1 Double-click the **Reset** input variable to display the **Item Properties** screen and do the following:
 - Enter **Reset** in the Variable Name field.
 - Click **Apply**, and then click the **Declaration** button.
- 2 On the **Declaration** tab, enter **BOOL** as the **Data Type** and click **Apply**.
- 3 Close the **Item Properties** screen.



Preset Variable

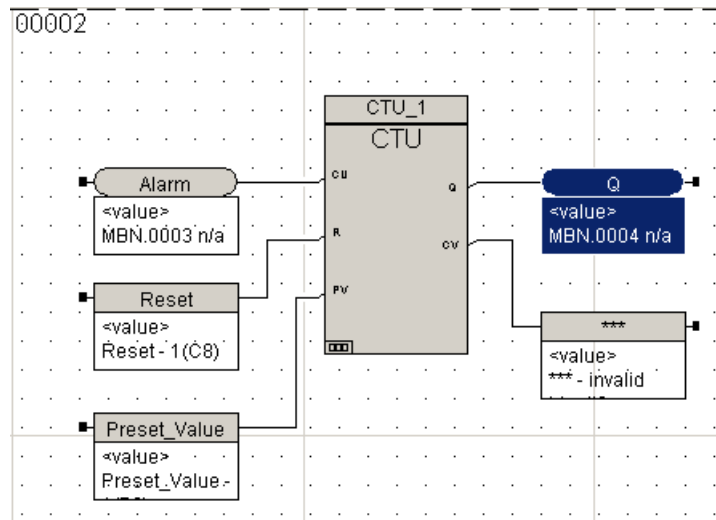
- 1 Double-click the Preset input variable to display the Item Properties screen and do the following:
 - Enter **Preset_Value** in the **Variable Name** field.
 - Click **Apply**, and then click the **Declaration** button.
- 2 On the **Declaration** tab, confirm **INT** as the **Data Type** and click **Apply**. Then, close the **Item Properties** screen.



Q Variable

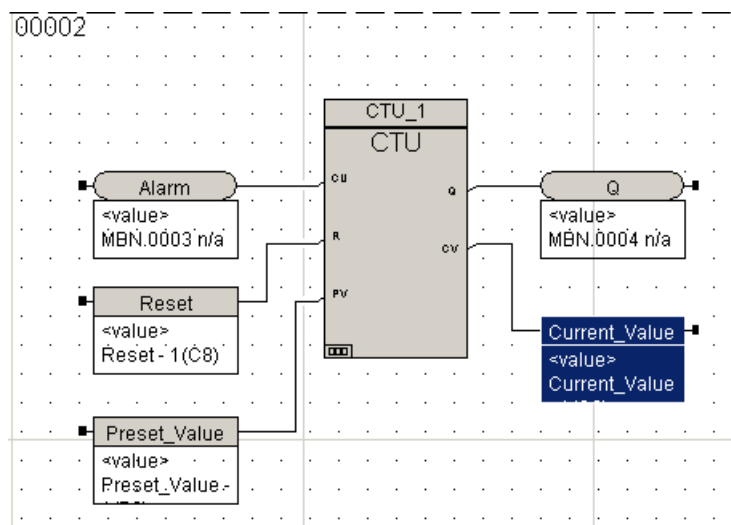
- 1 Double-click the Q output variable to display the Item Properties screen and do the following:
 - Enter **Q** in the **Variable Name** field.
 - Click **Apply**, and then click the **Declaration** button.

- 2 On the **Declaration** tab, confirm **BOOL** as the **Data Type**. Click **Apply** and then close the **Item Properties** screen.



CV Variable

- 1 Double-click the **CV** output variable to display the **Item Properties** screen and do the following:
 - Enter **Current_Value** in the **Variable Name** field.
 - Click **Apply**, then click the **Declaration** button.
- 2 On the **Declaration** tab, enter **INT** as the **Data Type**. Click **Apply** and then close the **Item Properties** screen.



Compile the Program and Print Logic Sheet

- 1** Compile the program and correct any errors.
- 2** Print the logic sheet.

Writing Ladder Diagram Logic

Ladder Diagram Logic 134

Contacts 135

Coils 135

Links 136

Power Rails 136

Ladder Diagram Logic

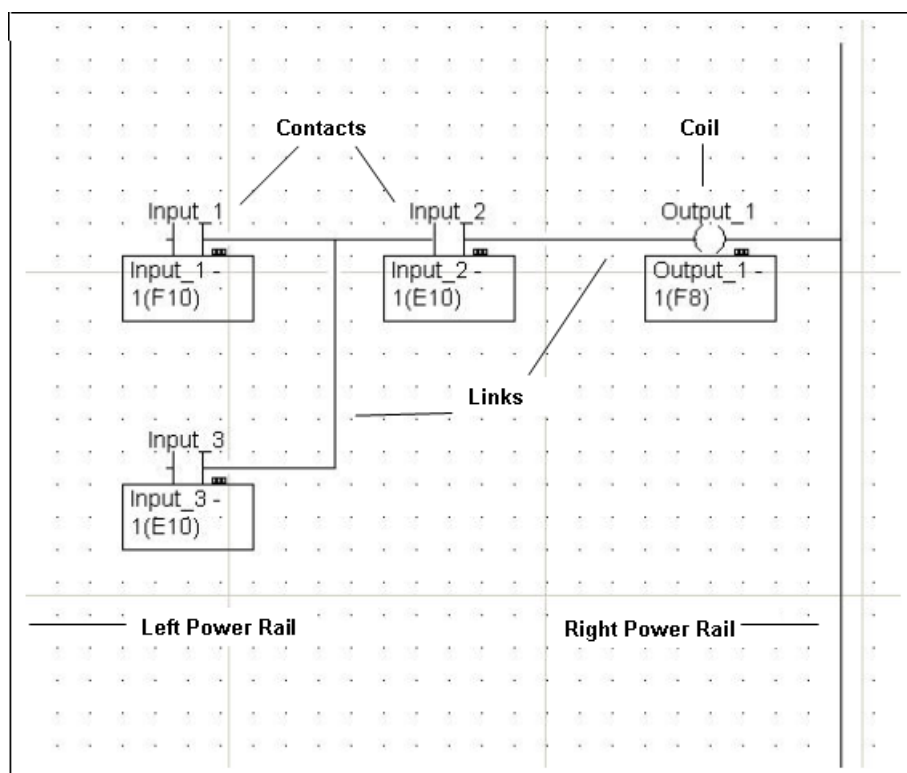
Ladder Diagram (LD) is a graphical language that uses a standard set of symbols to represent relay logic. A ladder diagram resembles an electrical wiring diagram and may be divided into sections, called rungs.

The basic elements are coils and contacts connected by links and defined by left and right power rails. A ladder diagram follows the power flow characteristics of relay logic.

Functions and function blocks that have at least one binary input and output can be used in LD diagrams.

Components of a ladder diagram are:


- Contacts
- Coils
- Links
- Power Rails



Contacts

A contact is an element in LD language that represents an input or state of a variable. The state is referred to as ON or OFF, and is equivalent to Boolean states True (1) and False (2). A contact is not used to change the value of a variable.


Contact Type	Symbol	Description
Normally Open	— —	The state of the left link is copied to the right link if the state of the Boolean variable is On.
Normally Closed	— / —	The state of the left link is copied to the right link if the state of the Boolean variable is Off.
Positive Transition	— P —	The state of the right link is On from one evaluation to the next when the associated variable changes from Off to On while the state of the left link is On.
Negative Transition	— N —	The state of the right link is On from one evaluation to the next when the associated variable changes from On to Off while the state of the left link is On.

The Contact Tool button  is used to place contacts on the logic sheet.

Coils

A coil represents an output state. It can be used to change the state of the associated Boolean variable.


Coil Type	Symbol	Description
Normal (Momentary)	()	The state of the left link is copied to the associated Boolean variable and to the right link.
Negated (Momentary)	(/)	The inverse of the state of the left link is copied to the associated Boolean variable and to the right link.
Positive Transition	(P)	The state of the associated Boolean variable is On from one evaluation to the next if the left link changes from Off to On.
Negative Transition	(N)	The state of the associated Boolean variable is On from one evaluation to the next if the left link changes from On to Off.
Set (Latch)	(S)	The associated Boolean variable is set to On if the left link is in the On state and remains On until reset by the RESET coil.
Reset (Unlatch)	(R)	The associated Boolean variable is reset to Off if the left link is in the On state and remains Off until set by the SET coil.

The Coil Tool button  is used to place coils on the logic sheet.

Links


In an LD program, logic elements are connected by links. Links may be horizontal or vertical. The direction of the link determines how it is evaluated in the logic sequence. Links are different from the wires used in FBD because they transfer only binary data between LD symbols.

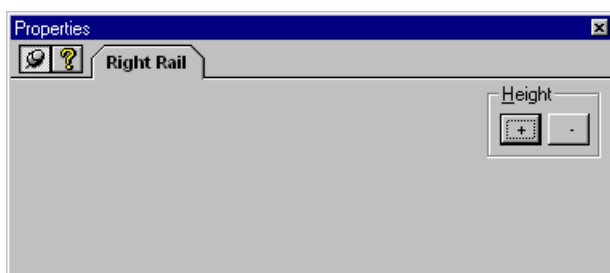
Link Direction	Description
Horizontal	<p>Transmits the state of the element on its immediate left to the element on its immediate right.</p> <p>The state represents the Inclusive OR of the On states of the horizontal links on its left side.</p>
Vertical	<p>Intersects one or more horizontal link elements on each side.</p> <p>The state is copied to all attached horizontal links on its right, but is not copied to attached horizontal links on its left.</p> <p>Is Off if the states of all attached horizontal links to its left are Off.</p> <p>Is On if the state of one or more attached horizontal links to its left is On.</p>

The Link Tool button  is used to connect the element on the logic sheet.

Power Rails

In Ladder Diagrams, power rails act as delimiters. The **TriStation 1131** automatically places a left power rail on the logic sheet when you create a new document. The first element in every network must be linked to the left power rail, which is considered ON at all times.

The right power rail has an undefined status and may be explicit or implied. The Right Power Rail button  is used to place right power rails on the logic sheet. You can change the height of the rail by double-clicking the rail and using the + or - buttons on the Right Rail tab.

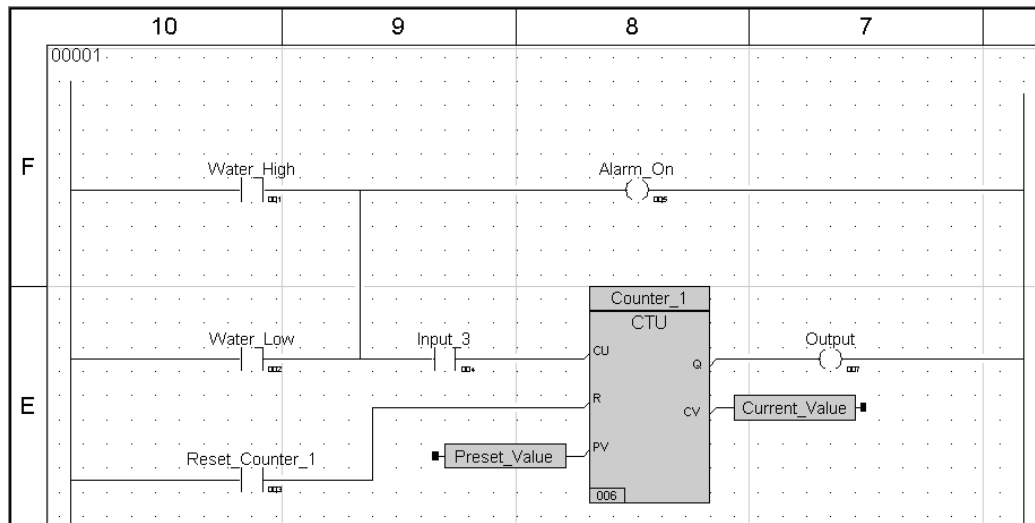


In this chapter, you will write LD logic for a water tank alarm.

One of the program inputs, represented by a contact, connects to a sensor to detect if the water level is too high. A second input (contact) detects if the water level is too low. The output sets an alarm, represented by a coil, if the water level is too high or too low.

A counter counts the number of times the alarm is set.

This is the completed LD logic.



Lesson 32: Creating a User Document

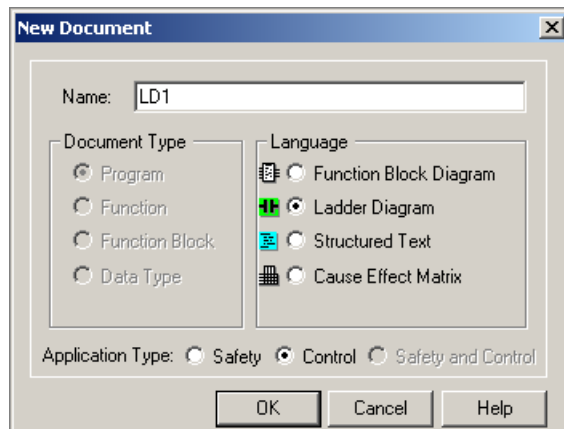
Project logic is created as user documents. User documents include programs, functions, function blocks, and data types.

In this lesson, you will:

- Create a new user document.
- Name the program LD1.


Procedure


- 1 Open the project you created in Chapter 2.
- 2 Expand the **Application** tree, right-click the **User Documents** folder, and then click **New Document**.

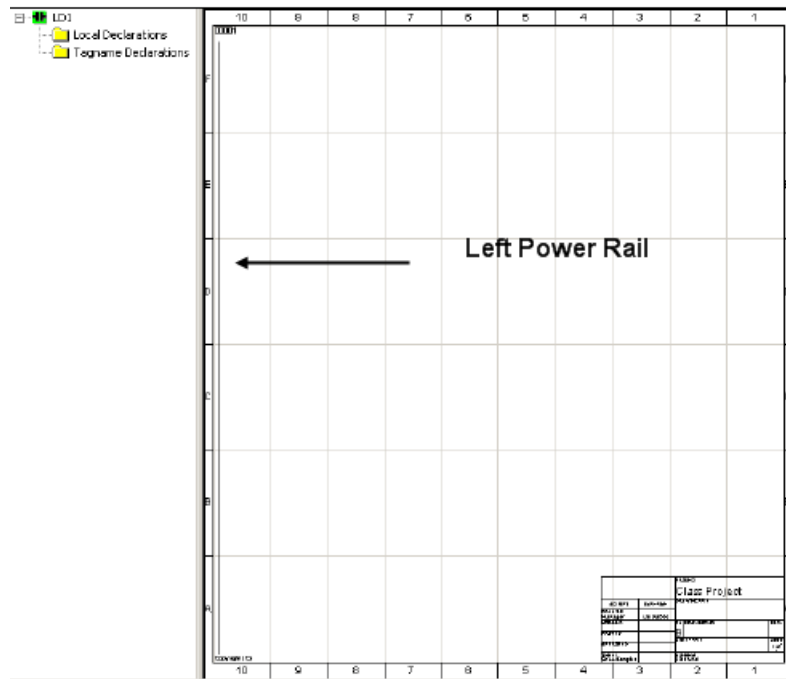


- 3 Specify the following:

Name:	Enter LD1
Document Type:	Click Program
Language:	Click Ladder Diagram
Application Type:	Click Control

- 4 Click **OK** to save. The document is opened in the **LD** editor, displaying the logic sheet.
- 5 Click the **Workspace View** button  to maximize the display.

- 6 Click the **Zoom to Fit** button  to display the full logic page.




Note: The left power rail is displayed on the logic sheet.

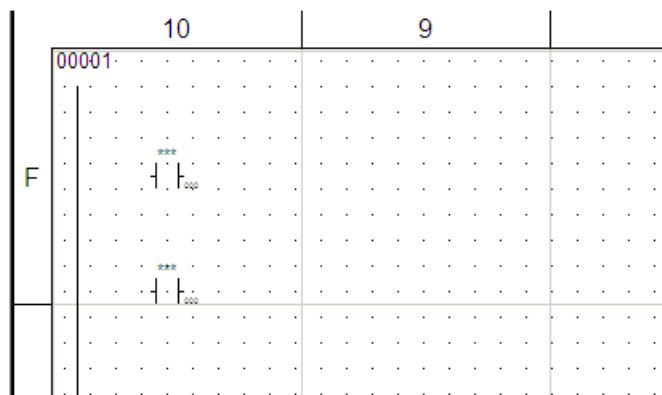
- 7 On the **Projects** menu, click **Project Options**, then click the **Annotation** tab.
- 8 Enable **Annotation on by default**. Enable **Monitor Value on by default**. Then click **OK**.


Lesson 33: Placing Contacts and Coils on the Logic Sheet

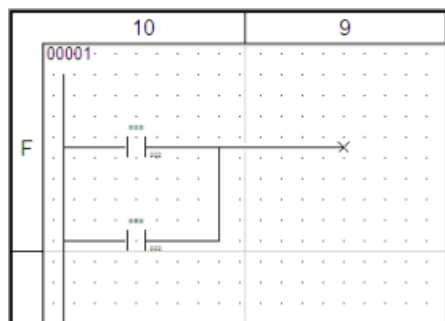
In this lesson, you will place two contacts and one coil on the logic sheet.


Procedure

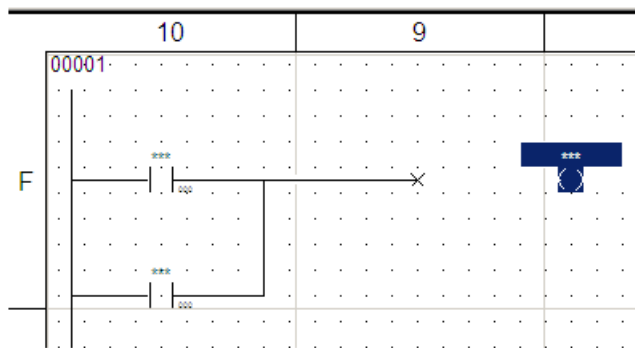
- 1 Click the **Contact Tool** button  on the toolbar and place it on the logic sheet. This represents the water level input.
- 2 Click the **Contact Tool** button on the toolbar and place the second contact below the first contact. This represents the second water level input.



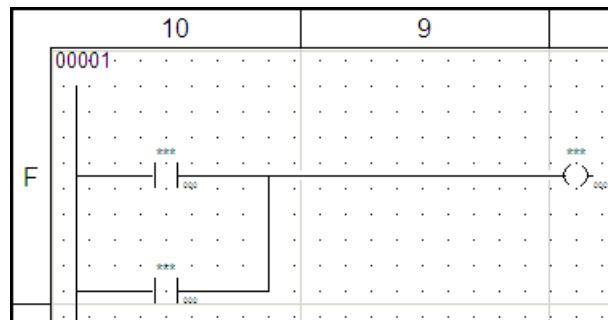
- 3 Click the **Link Tool** button  and draw wires connecting the contacts as follows:
 - Connect each contact to the left power rail.
 - Connect the second contact to the first contact.



- 4 Click the **Coil Tool** button  on the toolbar and place it on the logic sheet. This represents the alarm output.



- 5 Click the **Link Tool** button and connect the horizontal link to the coil.

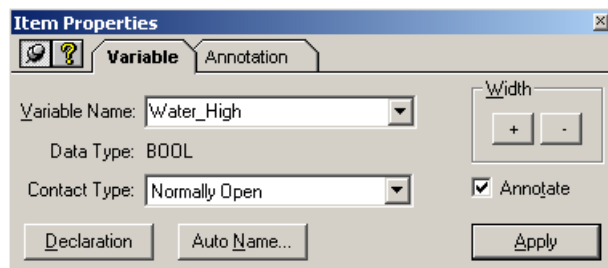


Lesson 34: Declaring Contacts and Coils

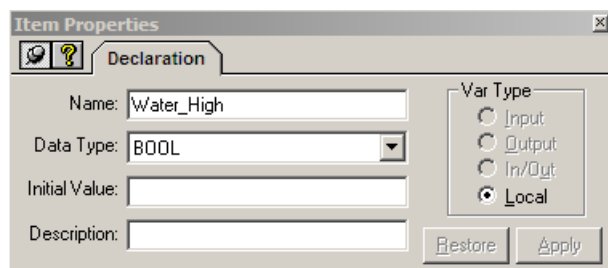
In this lesson, you will declare the contacts and coil for the water tank alarm.

Procedure

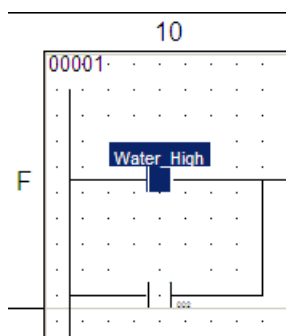
- 1 Double-click the first contact to display the **Item Properties** box.



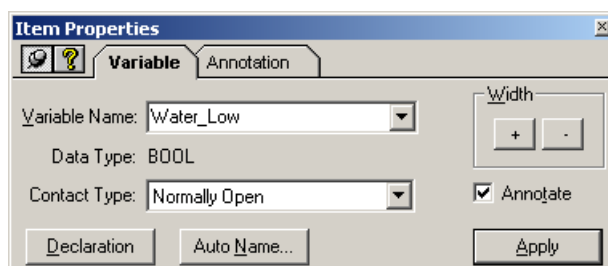
- 2 Enter **Water_High** in the **Variable Name** field. Click **Apply**. The **Declaration** screen is displayed.



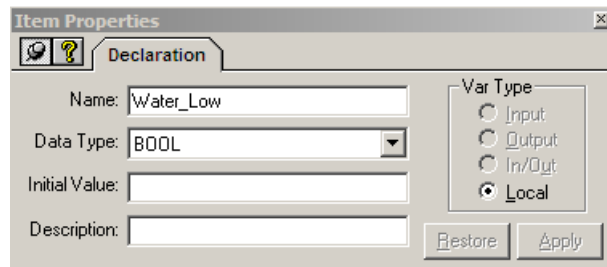
- 3 Confirm the Name, Data Type as **BOOL**, and Variable Type as **Local**. Close the **Item Properties** screen. The contact is labeled on the logic sheet.



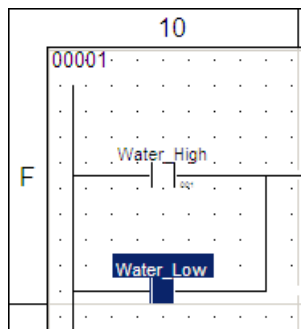
- 4 Double-click the second contact to display the **Item Properties** box.



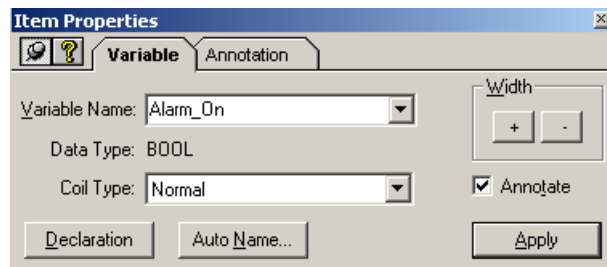
- 5 Enter **Water_Low** in the **Variable Name** field. Click **Apply**. The **Declaration** screen is displayed.



- 6 Confirm the Name, Data Type as **BOOL**, and Variable Type as **Local**. Close the Item Properties screen. The contact is labeled on the logic sheet.

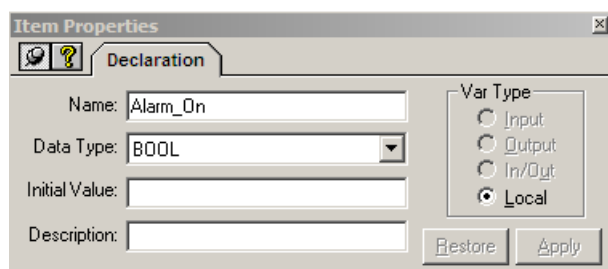


- 7 Double-click the coil to display the **Items Properties** box.



- 8 Enter **Alarm_On** in the **Variable Name** field. Click **Apply**.

The **Declaration** screen is displayed.



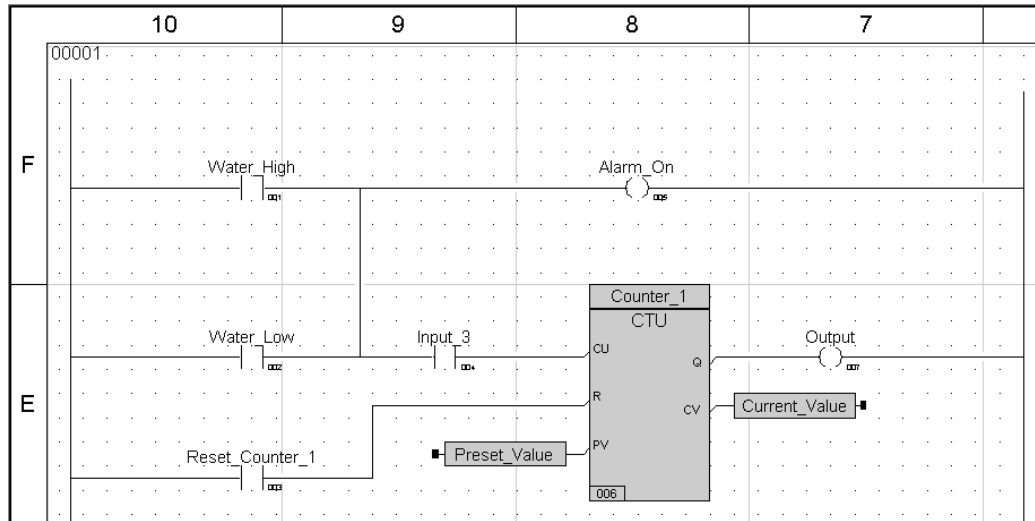
- 9 Confirm the Name, Data Type as **BOOL**, and Variable Type as **Local**. Close the **Item Properties** screen. The coil is labeled on the logic sheet.

You have completed the Ladder Diagram logic for the first network.

Lab 5



Review what you have learned by creating adding a counter to the water tank alarm.

This is the completed Ladder Diagram logic.





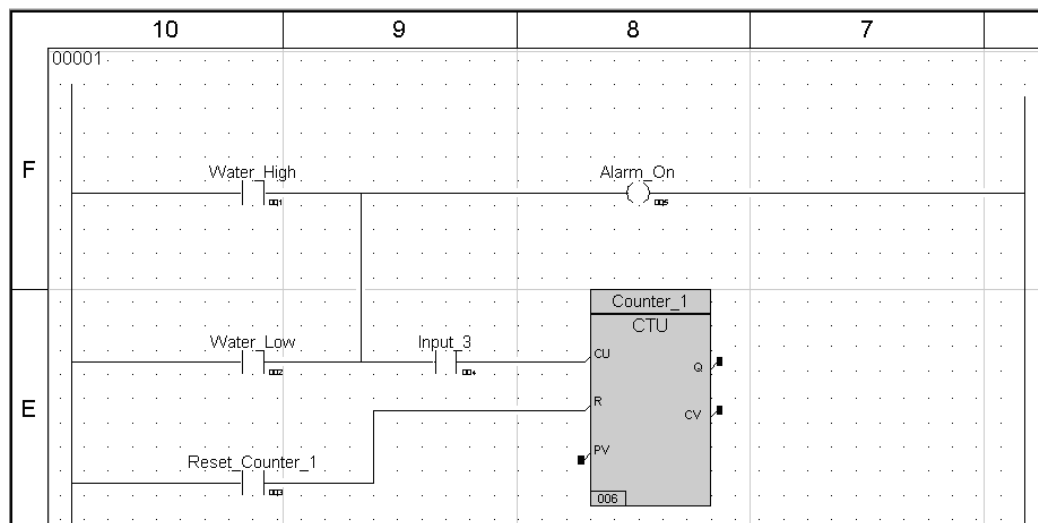
Procedure

Place a CTU function block on the logic sheet


- 1 Open the **Test_Project** project.
- 2 Open **LD1**.
- 3 Click the **Power Rail Tool** button  on the LD toolbar and place an explicit right power rail on the logic sheet.
- 4 Use the **Select Function (Block)** button  to place a **CTU** on the logic sheet in the second network.
- 5 Display the **Item Properties** screen for the CTU function block and do the following:
 - Enter **Counter_1** in the **Instance Name** field.
 - Check **Double Space**.
 - Click **Apply**.
 - Close **Item Properties**.

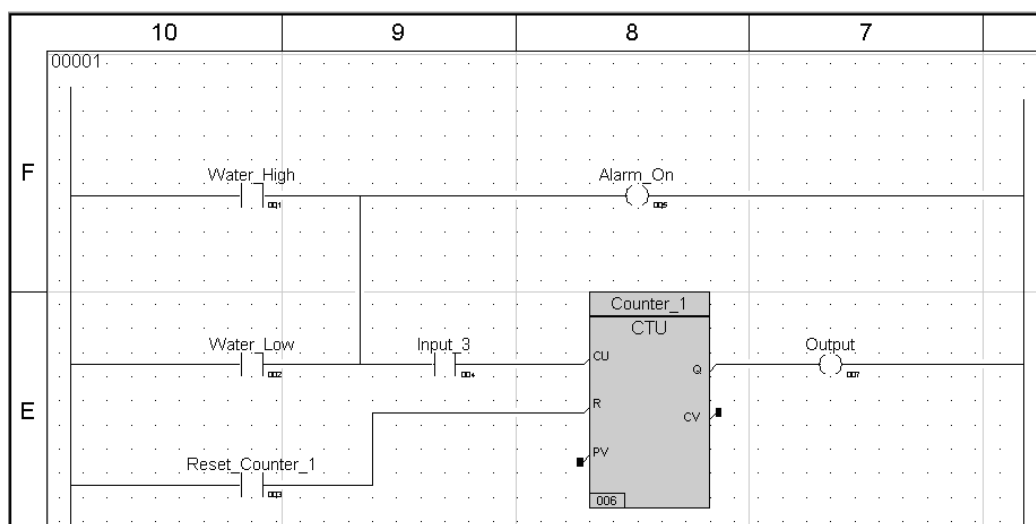
Place contacts on the logic sheet

- 1 Pass the output value of the alarm to the input of the counter. To do this, click the **Link Tool** button  and connect the **Alarm** to the right power rail.
- 2 Use the **Contact Tool** button  to place a contact on the logic sheet and connect it to the **CU** terminal on the counter.
- 3 Display the **Item Properties** screen for the contact and do the following:
 - Enter **Input_3** in the **Variable Name** field.
 - Confirm the contact as **Normally Closed**.
 - Click **Apply**.
 - Confirm Name, Data Type as **BOOL**, and Variable Type as **Local**.
 - Close **Item Properties**.
- 4 Place a second contact on the logic sheet and connect it to the **Reset** terminal on the counter.
- 5 Display the **Item Properties** screen for the contact and do the following:
 - Enter **Reset_Counter_1** in the **Variable Name** field.
 - Confirm the contact as **Normally Open**.
 - Click **Apply**.
 - Confirm Name, Data Type as **BOOL**, and Variable Type as **Local**.
 - Close **Item Properties**.




Place a coil on the logic sheet

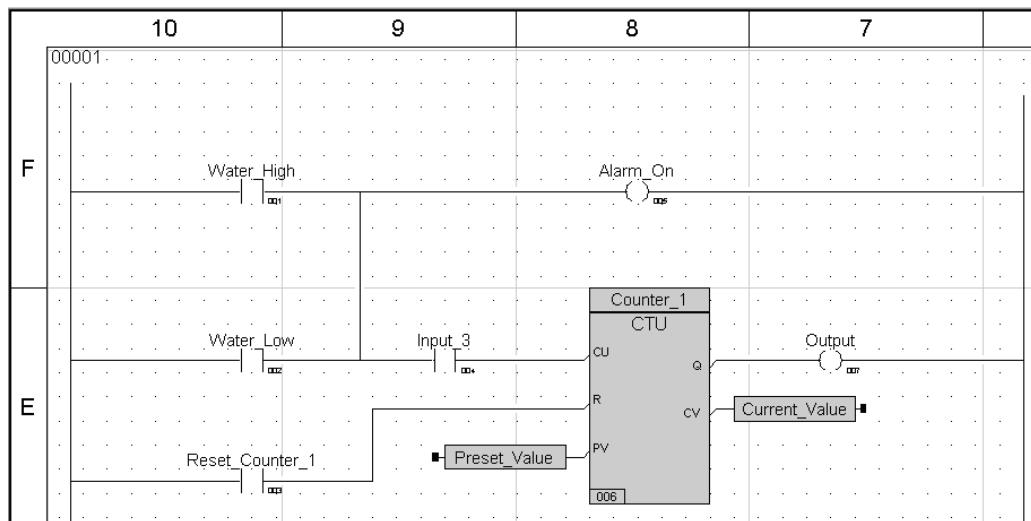
- 1 Use the **Coil Tool** button  to place a coil on the logic sheet and connect it to the **Q** terminal on the counter.
- 2 Display the **Item Properties** screen for the coil and do the following:
 - Enter **Output** in the **Variable Name** field.
 - Confirm the coil as **Normal**.
 - Click **Apply**.
 - Confirm Name, Data Type as **BOOL**, and Variable Type as **Local**.
 - Close **Item Properties**.



Place local variables on the logic sheet

- 1 Use the **Local Variable Tool** button  to place a local variable on the logic sheet and connect it to the **Preset Value** (PV) terminal of the **CTU** function block.
- 2 Display the **Item Properties** screen for the variable and do the following:
 - Enter **Preset_Value** in the **Variable Name** field.
 - Click **Apply**.
 - Confirm Name, Data Type as **INT**, and Variable Type as **Local**.
 - Close **Item Properties**.
- 3 Place a second local variable on the logic sheet and connect it to the **Current Value** (CV) terminal of the **CTU** function block.
- 4 Display the **Item Properties** screen for the variable and do the following:
 - Enter **Current_Value** in the from the **Drawing Item** list field.
 - Click **Apply**.
 - Confirm Name, Data Type as **INT**, and Variable Type as **Local**.
 - Close **Item Properties**.

You have completed the Ladder Diagram logic for the water tank alarm.



Compile the program

- 1 Compile the program and correct any errors.
- 2 Print the logic sheet.

Writing Structured Text Logic

Structured Text Logic	150
Structured Text Conventions	150
Structured Text Constructs	151
ST Editor	156

Structured Text Logic

Structure Text (ST) is a high-level programming language, similar to Pascal or C. Structured Text is particularly useful for complex arithmetic calculations, and can be used to implement complicated procedures that are not easily expressed in graphical languages such as FBD or LD.

ST allows you to create Boolean and arithmetic expressions, as well as structure programming constructs such as conditional statements (IF...THEN...ELSE). Functions and function blocks can be invoked in ST.

Structured Text Conventions

Programs, functions, and function blocks are declared using keywords that frame the beginning and end of the elements being declared. For example:

```
PROGRAM_Water Tank Alarm
```

A program begins with the declaration PROGRAM, followed by the program name. The end of the program is designated by END_PROGRAM.

Variables are declared using the keyword VAR, followed by the type of variable. The end of the variable list is designated by END_VAR.

Convention	Variable Type
VAR	Local variable
VAR_EXTERNAL	Tagname (global variable)
VAR_INPUT	Input variable. All input variables must be declared before any output variables can be declared.
VAR_OUTPUT	Output variable
VAR_IN_OUT	Variables that function as both input and output
VAR_CONSTANT	A defined constant. The value does not change.
VAR_TEMP	Variables placed in temporary memory and cleared when program ends.

Structured Text Example

```
VAR_EXTERNAL
(* Tagnames for water tank alarm *)
  Water_High : BOOL ;
  Water_LOW : BOOL;
  Alarm : BOOL ;
  Q : BOOL ;
END_VAR
```

In this example, VAR_EXTERNAL declares four tagnames and their data types. Each variable is followed by a colon (:) separator and its data type.

When writing ST logic, each statement ends with a semi-colon (;) separator. Indentations are used in outline form to increase the readability of the logic and to show that one statement is subordinate to another statement or keyword.

Comments must be enclosed in parentheses and framed between asterisks: (* this is comment text*).

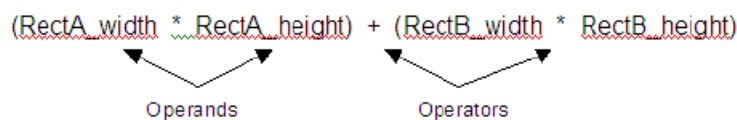
Structured Text Constructs

Structured text uses two types of constructs:

- Expressions
- Statements

Expressions

An expression calculates or evaluates values. It consists of operands and operators that when evaluated computes a value. For example, an expression to calculate the total area of two rectangles can be written as:



Operands and Operators

An operand is an element with which an operation is performed. An operand can be a variable such as X, RectA_width, ALARM_1; or a literal constant such as 1234, 1.35, 16#FFFF.

An operator performs a specific action on one or more operands to compute a value. For example, the multiplication operator (*) is used to compute the product of two or more operands:

RectA_width * RectA_height

Common operators are:

Symbol	Operator
+	Addition
-	Subtraction
*	Multiplication
/	Division
Mod(AB)	Modulo - remainder of an integer divided A/B
A**B	A to the power of B
SQ(A)	Square root of A
NEG(A)	Reverse sign +/-

Order of Evaluation

Expressions are evaluated in a specific order, depending on the precedence of the operators and/or sub-expressions. Expressions within parentheses have the highest precedence and are always evaluated first. Other operators are subsequently evaluated based on their precedence. When operators have equal precedence, they are evaluated from left to right.

Symbol	Operation	Precedence
(...)	Parenthesization	1 - Highest
Function (...)	Parameter list of a function, function evaluation	2
**	Exponentiation; i.e., raising to a power	3
-	Inversion	4
NOT	Complement	4
*	Multiply	5
/	Divide	5
MOD	Modulus operation	5
+	Addition	6
-	Subtraction	6
<, >, <=, >=	Comparison	7
=	Equality	8
< >	Inequality	8
AND	Boolean AND	9
XOR	Boolean Exclusive OR	10
OR	Boolean OR	11 - Lowest

Statements

A statement is used to invoke functions and function blocks, perform conditional evaluation of statements, or repeat specific sections of logic. ST supports the following kinds of statements:

- Assignment statements
- Conditional statements
- Function and function block control statements
- Selection statements
- Reiteration statements

Assignment Statements

An assignment statement changes the value of a variable. The format of an assignment statement is:

```
X := Y
```

Y is the new value of X when the assignment statement is evaluated.

This is an example of an assignment statement for the OR function output of the water tank alarm:

```
Alarm_On := Water_High or Water_Low;
```

Conditional Statements

A conditional statement specifies that a statement or a group of statements will execute when certain conditions exist. A conditional statement uses IF...THEN or IF...THEN...ELSE to define the conditions. This example describes the condition in which the control state of a pump is on or active:

```
IF (Gate = CLOSED) AND  
(PUMP = ON) AND (TEMP > 200.0) THEN  
    Control_State := Active;  
ELSE  
    Control_State := Hold;  
    PumpSpeed := 10.0;  
END_IF;
```

Function and Function Block Control Statements

A control statement invokes function or function blocks within an ST executable element. The format for a control statement is the function or function block name followed by a list of input parameter values and their value assignments.

```
FUNCTION_BLOCK_1 (Input 1 := Value1, Input 2 := Value 2, ...);
```

Selection Statements

A selection statement selects one value from a number of given values depending on specific criteria.

```
(* Select the maximum flowrate *)  
FlowMAX := MAX ( RateA, Rate B, RateC, RateD)
```

CASE Statements

A CASE statement is a type of selection statement that allows a selected statement to be executed depending on the value of an expression that returns an integer result.

```
(*Select a SPEED based on the value of SETTING *)
CASE SETTING of
1: SPEED := 5.0;
2: SPEED := 7.5;
3,4,5:SPEED := 12.0;
6: SPEED := 15.0;
7,8:SPEED := 18.0;
9: SPEED := 21.0;
10:SPEED := 25.0;
else
SPEED := 0.0;
end_CASE;
```

Iteration Statements

An iteration statement can be used when it is necessary to repeat one or more statements a number of times, depending on the value of an iteration variable. **TriStation 1131** supports two iteration statements: For...Do and Exit.

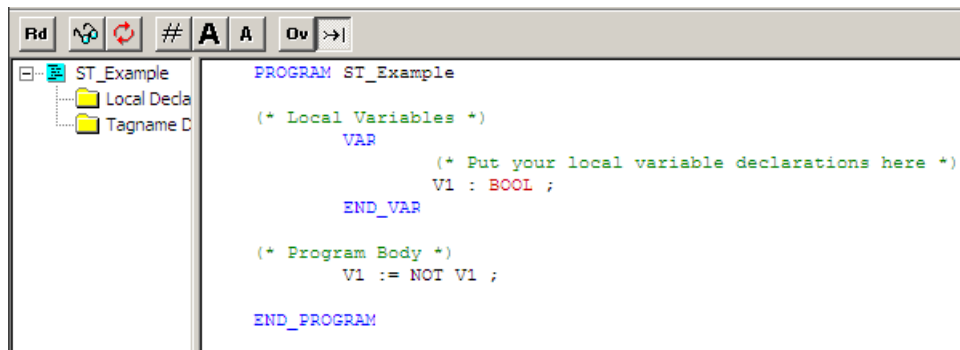
For...Do is used to repeat a set of statements depending on the value of an iteration variable. Exit can only be used within an iteration statement and ends the iteration loop.

In this example, a list of Level readings is scanned. If any level reading is True, the scan is terminated. The Exit statement is used in each FOR loop to terminate the loop when a level reading is found.

```
Level := FALSE;
FOR A := 100 TO 200 DO
  FOR B := 0 TO 9 DO
    IF Level (A,B) THEN
      LevelNo := A*10 + B;
      Level := TRUE; EXIT;
    END_IF;
  END_FOR;
IF Level THEN EXIT; END_IF;
END_FOR;
```

ST Editor

The Structured Text (ST) editor is used to develop programs and functions.



The ST editor displays a basic template that allows you to insert tabs, spaces, and comments between keywords and identifiers wherever a space is required.

In this chapter, you will create structured text for the water tank alarm.

One input connects to a sensor and detects if the water level is too high. The second input detects if the water level is too low. The output sets an alarm if the water level is too high or too low.

In the second network, a counter function block counts the number of times the alarm is set.

This is the completed ST logic for the water tank alarm:

```
PROGRAM ST1
  VAR_EXTERNAL
    Water_High : BOOL ;
    Water_LOW : BOOL;
    Alarm : BOOL ;
    Q : BOOL ;
  END_VAR

  VAR_
    Reset : BOOL ;
    Preset_Value : INT ;
    Current_Value : INT ;
    CTU_Block : CTU ;
  END_VAR

  (* First Network *)
  Alarm := Water_High OR Water_Low ;

  (* Second Network *)
  CTU_Block ( CU := Alarm, R :=Reset, PV :=Preset_Value ) ;
  Q := CTU_Block.Q ;
  Current_Value := CTU_Block.CV ;
END_PROGRAM
```


Lesson 35: Creating a User Document

Project logic is created as user documents. User documents include programs, functions, function blocks, and data types.

In this lesson, you will:

- Create a new user document.
- Name the program ST1.

Procedure

- 1 Open the project created in Chapter 2.
- 2 Expand the **Application** tree, right-click the **User Documents** folder, and then click **New Document**.
- 3 Specify the following:

Name:	Enter ST_1
Document Type:	Click Program
Language:	Click Structured Text
Application Type:	Click Control

- 4 Click **OK**. The document is opened in the ST editor, displaying the logic sheet with a text template displayed.

```
PROGRAM ST_1

(* Local Variables *)
VAR
    (* Put your local variable declarations here *)
    V1 : BOOL ;
END_VAR

(* Program Body *)
V1 := NOT V1 ;

END_PROGRAM
```

- 5 Delete the template text by highlighting it and pressing **Delete**.

Lesson 36: Entering Structured Text Logic

In this lesson, you will enter ST logic for the water tank alarm.

Procedure

The keyword, PROGRAM, declares the program. In this program, PROGRAM ST_1 defines the program name. The program logic is written between the statements PROGRAM and END_PROGRAM.

```
PROGRAM ST_1

    VAR_EXTERNAL
        Water_High : BOOL ;
        Water_LOW : BOOL;
        Alarm : BOOL ;
        Q : BOOL ;
    END_VAR
END_VAR
```

1 Enter logic declaring the tagnames:

```
VAR_EXTERNAL
    Water_High : BOOL ;
    Water_LOW : BOOL;
    Alarm : BOOL ;
    Q : BOOL ;
END_VAR
```

The keyword, VAR_EXTERNAL, declares the variables listed between VAR_EXTERNAL and END_VAR. The statements name the variables and specify the data types.

```

PROGRAM ST_1

    VAR_EXTERNAL
        Water_High : BOOL ;
        Water_LOW : BOOL;
        Alarm : BOOL ;
        Q : BOOL ;
    END_VAR

END_VAR

VAR
    Reset : BOOL;
    Preset_Value : BOOL;
    Current_Value : INT;
    CTU Block : CTU;

END_VAR

END_PROGRAM

```

2 Enter logic declaring the local variables:

```

VAR_
    Reset : BOOL;
    Preset_Value : BOOL;
    Current_Value : INT;
    CTU Block : CTU;

END_VAR

```

The keyword, VAR_, declares the variables listed between VAR_ and END_VAR. The statements name the variables and specify the data types.

```

PROGRAM ST_1

    VAR_EXTERNAL
        Water_High : BOOL ;
        Water_LOW : BOOL;
        Alarm : BOOL ;
        Q : BOOL ;
    END_VAR

END_VAR

VAR_
    Reset : BOOL;
    Preset_Value : BOOL;
    Current_Value : INT;
    CTU Block : CTU;

END_VAR
(* First Network *)
    Alarm := Water_High OR Water_Low ;
END_PROGRAM

```

3 Enter the comment text and operation:

```

(* First Network *)
Alarm := Water_High OR Water_Low ;

```

You have completed the ST logic for the first network of the water tank alarm.

```
PROGRAM ST_1

    VAR_EXTERNAL
        Water_High : BOOL ;
        Water_LOW : BOOL ;
        Alarm : BOOL ;
        Q : BOOL ;
    END_VAR

    VAR
        Reset : BOOL ;
        Preset_Value : INT ;
        Current_Value : INT ;
        CTU_Block : CTU ;
    END_VAR

    (*First Network *)
    Alarm := Water_high OR Water_Low ;

    (*Second Network *)
    CTU_Block (CU := Alarm, R := Reset, PV :=Preset_Value) ;
    Q := CTU_Block.Q ;
    Current_Value := CTU_Block.CV ;

END_PROGRAM
```

4 Enter comment text:

```
(* Second Network *)
```

5 Enter logic for CTU function block:

```
CTU_Block( CU:=Alarm, R:=Reset, PV:=Preset_Value ) ;
Q := CTU_Block.Q ;
Current_Value := CTU_Block.CV ;
```

6 The keyword **END_PROGRAM** designates the end of PROGRAM ST_1.

You have completed the ST logic for the water tank alarm.

Lesson 37: Compiling the Program

Compiling a program is way to verify your logic before building the application. Any errors will be displayed in the Message View. All errors must be corrected before building the program.

In this lesson, you will compile the program for the water tank alarm.

Procedure

- 1 On the **Project** menu, click **Compile All User Documents**.

The Message View automatically opens and displays the status of the compile process.

- 2 If there are errors, they must be fixed before building and testing the application.

6

Writing Cemple Logic

Writing CEMPLE Logic	164
Matrix Planning	165
Using the CEM Editor	167
User-Defined Functions and Application-Defined States	175

Writing CEMPLE Logic

CEMPLE™ is an optional **TriStation 1131** language editor that automates the process of creating programs, based on a cause and effect matrix. Cause and effect matrix is a methodology that is commonly used in the process control industry to define alarms, emergency shutdown strategies, and mitigation actions.

A matrix created in CEM language can be as basic or complex as your situation requires. In a basic matrix, causes are identified as True or False inputs related to one or more effects through the intersections between them. The state of a cause (True or False) determines the state of the related effect.

If more than one cause is related to an effect, the state of the effect is based on how the matrix is evaluated. You can specify the matrix evaluation as a de-energize-to-trip (fail-safe) or energize-to-trip system. In a typical de-energize-to-trip system, if one of the inputs changes to False, the related outputs also change to False. In an energize-to-trip system, the reverse is True; if one of the inputs changes to True, the related outputs also change to True.

For more complex processes, CEM language allows you to add functions or function blocks to causes, intersections, and effects. This feature can be used for many purposes; for example, to accept non-Boolean input and convert to Boolean output, to set timers before evaluating the input, and to pass additional input variables to output variables.

CEM language includes these features:

- Ability to specify up to 99 causes, 99 effects, and 1,000 intersections
- Ability to invoke functions and function blocks to evaluate cause, intersection, and effect states
- Choice of de-energize-to-trip or energize-to-trip matrix evaluation
- Automatic conversion of matrix to Function Block Diagram language
- Customized view monitoring of active causes, intersections, and effects
- Multiple levels of undo and redo editing

Matrix Planning

Planning includes determining the causes (problems) to be monitored, and determining how the matrix is to be evaluated.

Restrictions and Limitations

These are the restrictions on matrix development:

- No more than 99 causes, 99 effects, and 1,000 intersections.
- Variables with a variable type of In/Out (VAR_IN_OUT) are not allowed in CEM programs, function blocks that are invoked by matrix programs, or any safety program or function block.

Matrix Evaluation Options

When planning a matrix, you must determine how the matrix is evaluated when it includes multiple causes and effects.

If the matrix is based on an energize-to-trip system, such as a fire suppression system, an OR evaluation is typically used because the normal state of inputs is False. If one of the inputs changes to True, the related outputs also change to True. The default setting is OR.

If the matrix is based on a de-energize-to-trip (fail-safe) system, an AND evaluation is typically used because the normal state of inputs is True. If one of the inputs changes to False, the related outputs also change to False. This is why it is typically used with systems that are designed to be fail-safe.

How a Matrix is Evaluated

When a matrix is executed, the states of causes, effects, and intersections are evaluated in a specific order. The states of causes, intersections, and effects are saved in internal variables. An internal Move function moves the cause state to the intersection state, and then to the effect state.

The order of evaluation is shown and described in the following figures.

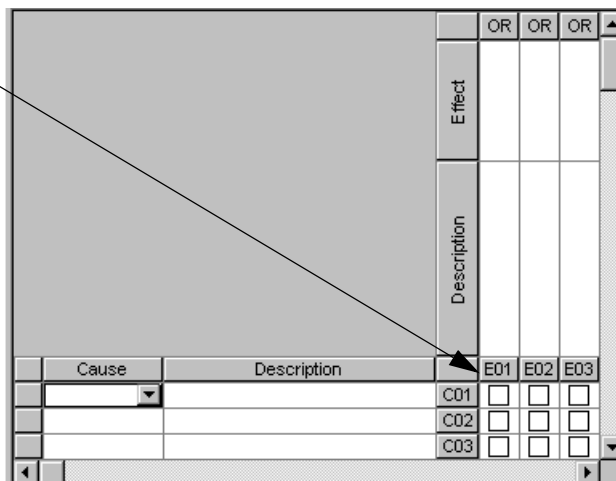
1. For each cause from the top row to the bottom, a state is determined based on the inputs and function associated with the cause.

			OR	OR	OR	▲
		Effect				
		Description				
	Cause	Description		E01	E02	E03
	▼		C01	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
			C02	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
			C03	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
◀						▶

2. For each intersection from the bottom to the top, a state is determined based on the cause state and intersection function.

For typical AND evaluations, all cause states must be True for the intersection state to be True and one False state makes the output False.

For typical OR evaluations, all cause states must be False for the intersection state to be False and one True state makes the output False.



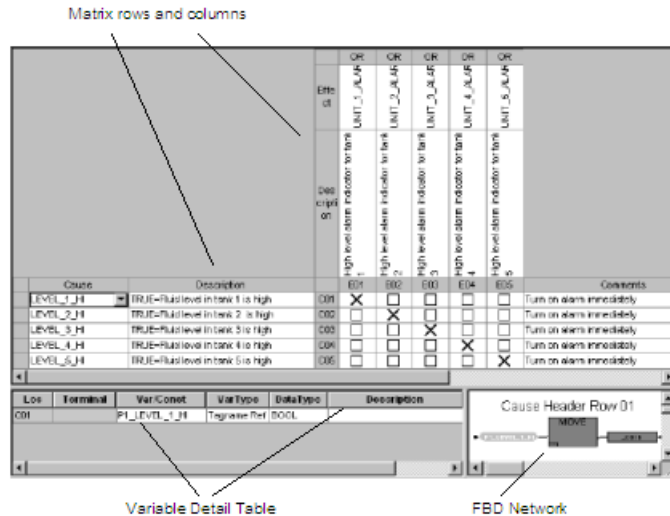
			OR	OR	OR
		Effect			
		Description			
Cause	Description		E01	E02	E03
		C01	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
		C02	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
		C03	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

3. For each column from the left to the right, an effect state is determined based on the intersection state and function associated with the effect.

		OR			OR	OR	▲
		Effect					
Description							
Cause	Description		E01	E02	E03		
	▼	C01	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		
		C02	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		
		C03	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	▼	

Using the CEM Editor

The CEM editor allows you to create a **TriStation 1131** program based on a cause and effect matrix.



The editor includes the following areas:

Matrix: identifies causes, effects, and the intersections between them. Can also include inputs, outputs, functions, and function blocks related to causes, effects, and intersections.

FBD Network: displays the Function Block Diagram (FBD) related to the cause, effect, or intersection that you select in a matrix. It also allows you to specify properties and to invert the values of variables.

Variable Detail Table: displays the inputs and outputs of an FBD Network that are generated when a cause, effect, or intersection is selected. It also allows you to specify variable type and data type.

Matrix

The matrix area of the CEM editor includes the rows, columns, and intersections of a matrix. In a basic matrix that does not use functions, causes can be directly related to effects through intersections.

In a complex matrix, functions can be included for causes, effects, and intersections. When functions are included, the inputs and outputs of those functions can be specified in the matrix. A text label appears at the cause and effect intersection.

<div> <div>Effect Items</div> <div>Cause Items</div> </div>						OR	OR	OR	Intersections	
					Output					
					Function					
					Effect					
					Description					
	Input	Function	Cause	Description		E01	E02	E03	Comments	
					C01					
					C02					
					C03					

FBD Network

The FBD Network area of the CEM editor displays the Function Block Diagram (FBD) related to the cause, effect, or intersection that you select in the matrix. The FBD network uses internal Boolean variables to save and move results to associated cells so that causes and effects can be evaluated. When you create a cause, intersection, or effect, an internal variable is automatically created for each.

The screenshot displays the FBD Network area. At the top, there is a matrix with columns for Cause, Description, and five Effect units (UNIT_1_ALARM to UNIT_5_ALARM). Below this is a table of internal variables:

Loc	Terminal	Var/Const	VarType	DataType	Description
C01		P1_LEVEL_1_HI	Tagname	BOOL	

To the right of the table is a diagram of a MOVE function block. The input is labeled 'Cause Header Row 01' and the output is labeled 'FBD Network area'.

The CEM editor uses internal variables to store and move results between cells. Although you cannot directly access the internal variables, you can create variables and copy the values to those variables. You can also specify properties and invert values of variables.

If you select a cause, effect, or intersection that does not contain a function or if you make multiple selections, the FBD Network cannot display appropriate information.

Variable Detail Table

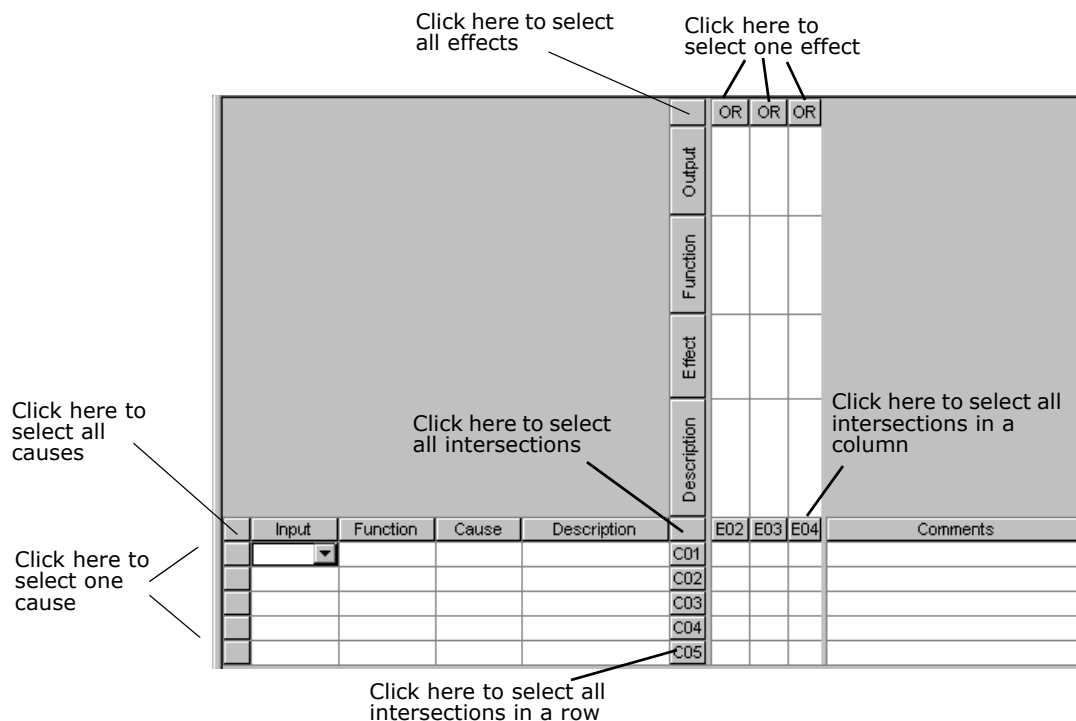
The Variable Detail Table area of the CEM editor displays the inputs and outputs of an FBD Network that are generated when a cause, effect, or intersection is selected. This figure shows the names, variable types, and data types related to the function block.

Loc	Terminal	Var/Const	VarType	DataType	Description
C01	HI_LEV	LEVEL_1_HI	Tagname Refe	BOOL	
C01	LOW_LEV	LEVEL_1_LOW	Tagname Refe	BOOL	
C01	BYP	BYP_LEV_1	Tagname Refe	BOOL	
C01	LVLALRM				

Selecting and Editing Cells in a CEM Program

This table and figure explain how to select one or more cells in a CEM program.

To ...	Do This ...
Select a single cell	Right-click anywhere in the cell.
Select contiguous (adjacent) cells	Click the first cell, hold down the shift key, and click the last cell in the area.
Select discontinuous (non-adjacent) cells	Click a cell, hold down the Ctrl key, and click the rows or column.



This table explains how to edit cells in a matrix.

To ...	Do This ...
Enter edit mode	Click directly over the text in an editable cell.
Complete a cell entry	Press the tab key or Enter to complete a cell entry and move the cursor to the next cell to the right.
Move to the next cell	Press the tab key or Enter to move the cursor to the next cell to the right.
Delete the contents of a cell or group of cells	Select a cell or group of cells (but do not place in edit mode) and press the delete key.

Displaying and Sizing Cells from the Matrix

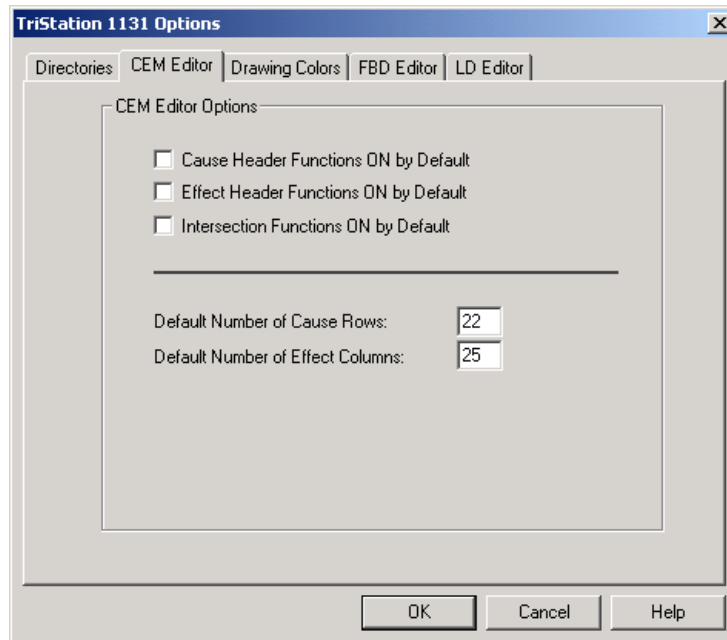
This table describes how to change the display and size of cells in a matrix. You can also make changes by using commands on the View menu. These actions do not affect the matrix evaluation.

To drag or double-click a cell boundary, you must use the double-arrow cursor, which is only active in the gray cells of a matrix.

To ...	Do This ...
Change width of column	Drag the column boundary left or right.
Restore size of column	Drag the column boundary to the left so that the column is almost hidden, then release the button.
Change height of row	Drag the row boundary up or down.
Restore default size of row	Drag the row boundary upward until the row is almost hidden, then release the button.
Hide a column	Drag the column boundary to the left until it meets the nearest boundary. For an effect column, double-click the thickened column boundary.
Unhide a column	For a cause column, double-click the boundary between the currently displayed columns and the hidden column.
Hide a row	Drag the row boundary upward until it meets the nearest boundary.
Unhide a row	Double-click the thickened row boundary.

Specifying CEM Editor Options

The CEM Editor options are used as initial settings for all the CEM (cause and effect matrix) programs in a project. After a program is created, you can modify these settings on a program-by-program basis. The CEM Editor tab is accessed from the TriStation 1131 menu on the toolbar.



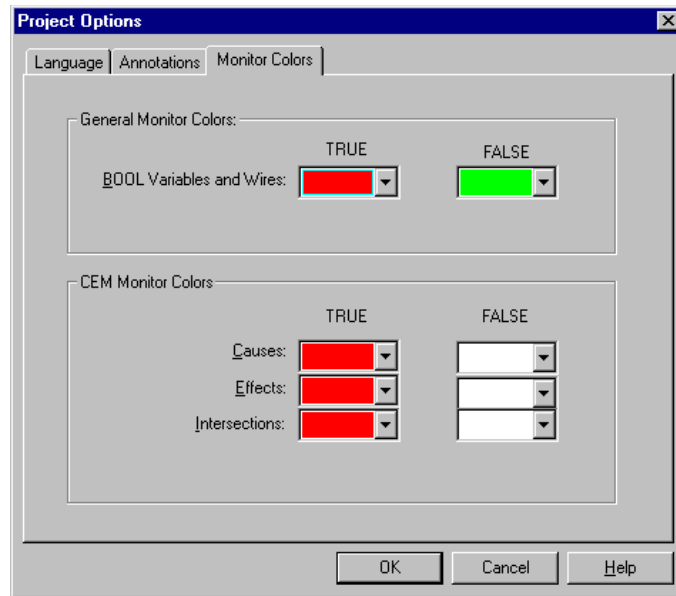
This table describes the properties on the CEM Editor tab.

Property	Action
Cause Header Functions On by Default	Check to have an input and function columns added. The default is unchecked.
Effect Header Functions On by Default	Check to have output and function columns added. The default is unchecked.
Intersection Functions On by Default	Check to have function columns added. The default is unchecked.
Default Number of Cause Rows	Enter the number of rows to include in a new matrix. The default is 22.
Default Number of Effect Columns	Enter the number of columns to include in a new matrix. The default is 25.

Specifying Monitor Colors and Names

You can specify the colors that are displayed for True and False BOOL values when an application is monitored in the emulator and controller using the Monitor Colors tab.

The Monitor Colors tab is accessed from the Project Options on the Project Menu.

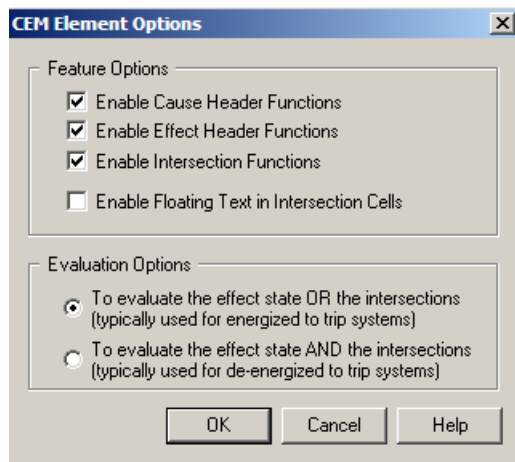


The default for True is red; False is white.

Specifying CEM Element Options

CEM Element Options specify whether functions are used in a specific matrix program, and how the matrix is evaluated.

The CEM Element Options screen is accessed from Options on the Document menu.



This table describes the properties on the CEM Element Options screen.

Property	Action
Enable Cause Header Functions	Check to add an input and function column to the cause header. The default is checked.
Enable Effect Header Functions	Check to add an output and function column to the effect header. The default is checked.
Enable Intersection Functions	Check to add a function column to the intersection. The default is checked.
Enable Floating Text in Intersection Cells	Check to allows the name of the function or function block to be displayed in a neighboring cell if that cell is empty. The default is unchecked.
Evaluation Options	Specifies how the matrix is to be evaluated when it includes multiple intersections between a cause and effect. The default is OR.

User-Defined Functions and Application-Defined States

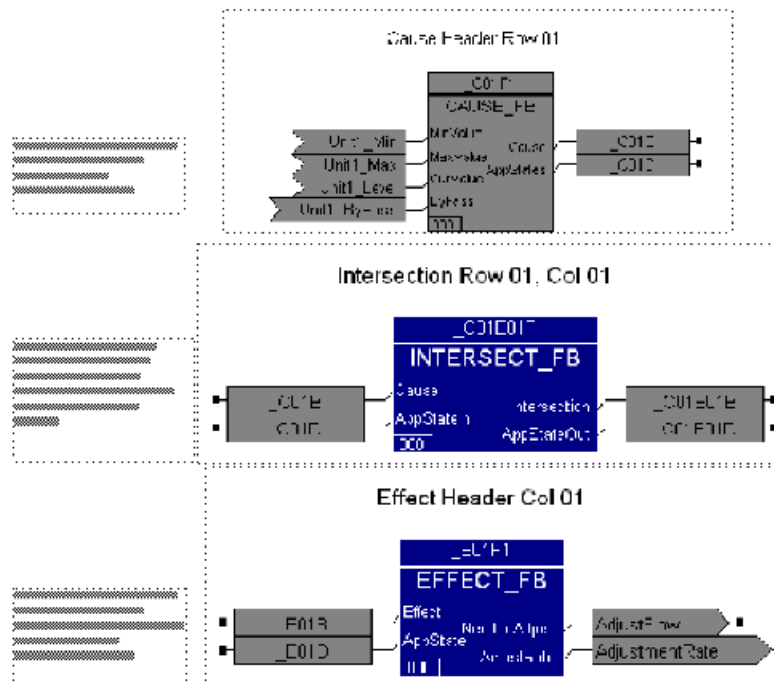
User-Defined Functions

User-defined functions must be enabled before they can be used in a CEM program. (Triconex-supplied functions and function blocks do not have to be enabled.) When you enable a function, it is validated to ensure it can be used in that part of the matrix. For example, a function used in an intersection must have a Boolean primary input and Boolean primary output. If not enabled, the function is not available for selection.

Application States

User-defined functions can include a variable that stores application states and that is evaluated in the same way as the cause, intersection, and effect internal variables. This means you can include application information that is evaluated with an AND or OR operation when the matrix is run. Application state inputs and outputs must be a DWORD datatype, which is a 32-bit string.

This figure shows an example of using a variable to store the application state.



Enabling User-Defining Functions and Application-Defined States

User-defined functions and application-defined states must be enabled for a specific matrix. If a user-defined function is not enabled, it cannot be used in a matrix.

User-defined functions and application-defined states are enabled by right-clicking the user-defined function, clicking Properties, and then clicking the Attributes tab.

This table describes how to enable user-defined functions and application-defined states.

Property	Action
Application Type	Click either Control or Safety and Control.
Supports Use in Cause Rows with... Inputs	Check to enable for use in cause rows. The default is unchecked.
Supports Use in Effect Columns With... Inputs	Check to enable for use in effect columns. The default is unchecked.
Supports Use in Intersections	Check to enable for use in intersections. The default is unchecked.
Supports Application Defined States	Check to enable the function to add a variable to store the application defined state. The default is unchecked.

The compile process determines whether the function can be used.

Specifying Local Variables, Tagnames, and Constants in a CEM Program

When functions are used with causes or effects, the inputs and outputs to the functions must be specified as variables or constants. These are specified in the Input or Output columns, or in the Var/Const column in the Variable Detail Table.

In/Out variables (VAR_IN_OUT) are not allowed in CEM programs, function blocks that are invoked by CEM programs, or any safety program or function block.

Specifying Properties in the Variable Detail Table

Properties in the Variable Detail Table, located in the lower left part of the CEM program, can be modified.

Loc	Terminal	Var/Const	Var Type	Data Type	Description
C01		Input_2	Tagname Reference	BOOL	
C01		Alarm_Flag	Local	DINT	Flag to detect alarm
C01	AND				

This table describes how to specify properties in the Variable Detail Table.

Property	Action
Var/Const	Enter a variable or constant.
Var Type	Click either Local or Tagname.
Data Type	Specify a data type.
Description	Enter a description.

Lesson 38: Creating a CEMPLE Project

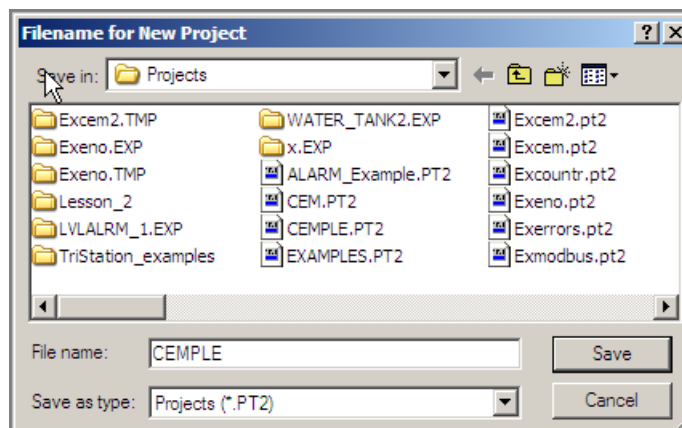
In this chapter, you will create matrixes in CEMPLE. First, you will create a function named ALARM and a function block named SHUTDOWN. The function and function block will be invoked by the matrixes created later in this chapter. You will then create four cause and effect matrixes.

In this lesson, you will:

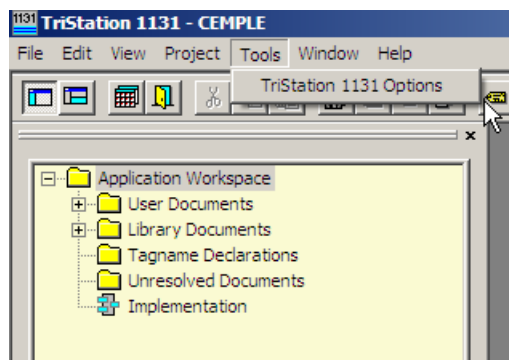
- Create a new project named Cemple.
- Create FBD logic for an alarm.
- Set CEM Default Options.

Procedure

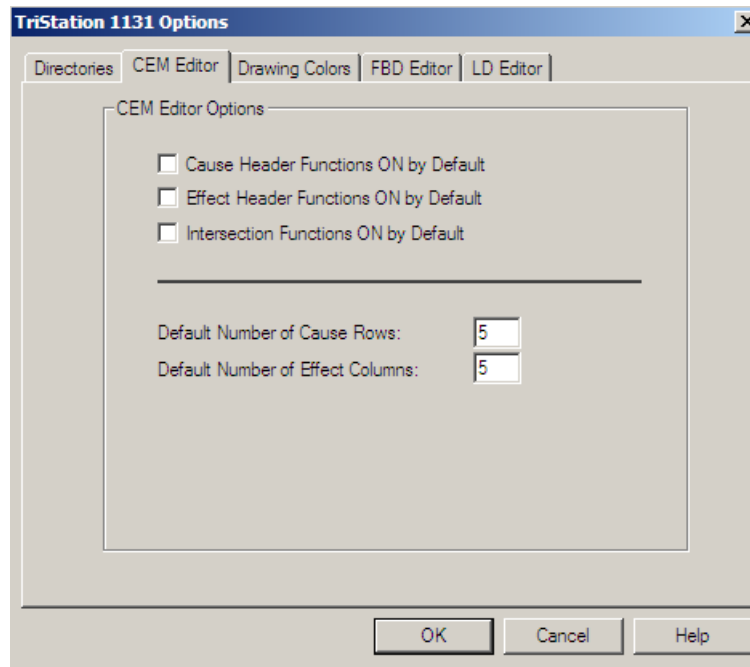
- 1 Open **TriStation 1131**. On the **File** menu, click **New Project**.
- 2 Select **Tricon** as the platform.
- 3 Click **OK** to continue. The **Filename for New Project** window is displayed.



- 4 Enter **CEMPLE** as the project name. Click **Save**.
- 5 Set the **CEM** editor options. Click **Tools** on the toolbar, and then click **TriStation 1131 Options**.



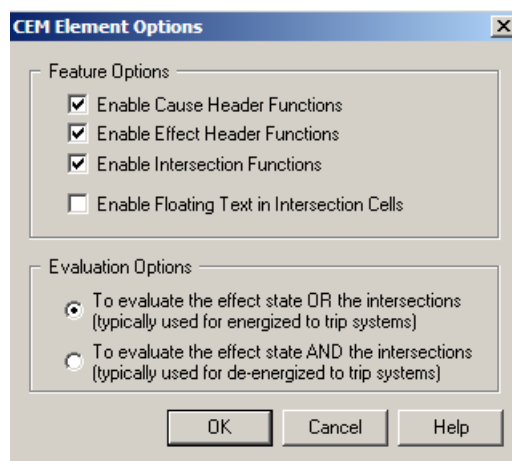
- 6 Click the **CEM Editor** tab.



- 7 Specify the following:

Default Number of Cause Rows:	5
Default Number of Effect Columns:	5

- 8 Click **OK**.
- 9 Click the **Document** menu, and then click **Options**. The **CEM Element Options** screen is displayed.

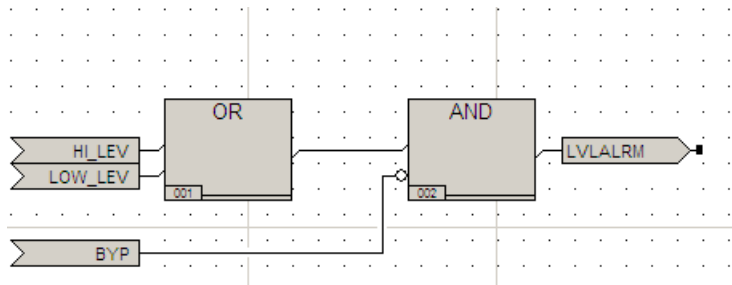


- 10 Specify the following:
 - Enable Cause Header Functions
 - Enable Effect Header Functions
 - Enable Intersection Functions
- 11 Click OK. Save and close the project.

Lesson 39: Creating a Function

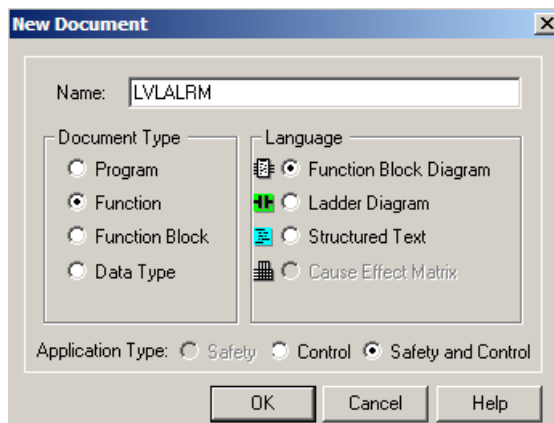
In this lesson, you will create a function named LVLALRM that will be invoked from a matrix in another lesson. You can review the steps for creating FBD logic in Chapter Three: Writing FBD Logic.

This is the completed FBD logic.



Procedure

- 1 Open the CEMPLE project you created in a previous lesson.
- 2 Expand the **Application** tree, right-click the **User Documents** folder, and then click **New Document**.

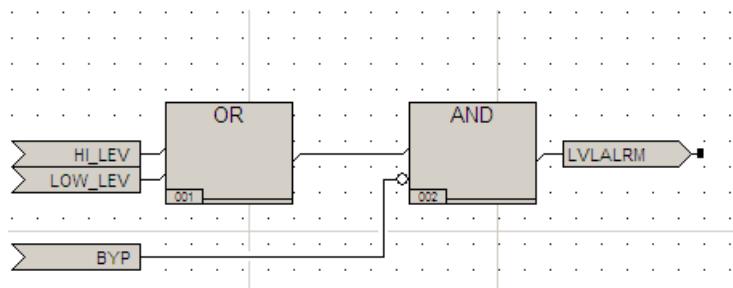


- 3 Specify the following:

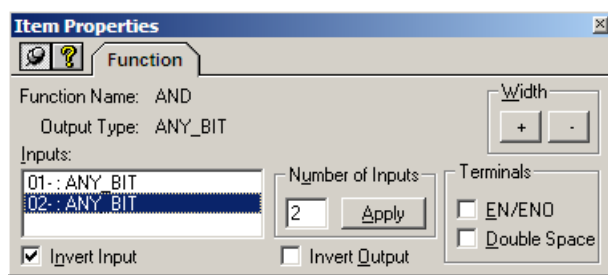
Name:	Enter LVLALRM
Document Type:	Click Function
Language:	Click Function Block Diagram
Application Type:	Click Safety and Control

- 4 Click **OK**. The document is opened in FBD and the logic sheet is displayed.

- 5 Create the FBD logic for the Alarm. Declare all variable types as **BOOL**.

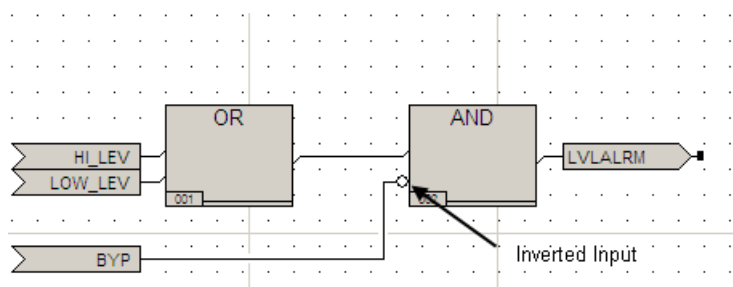


- 6 After declaring the inputs, double-click the **AND** function. The **Item Properties** screen is displayed.



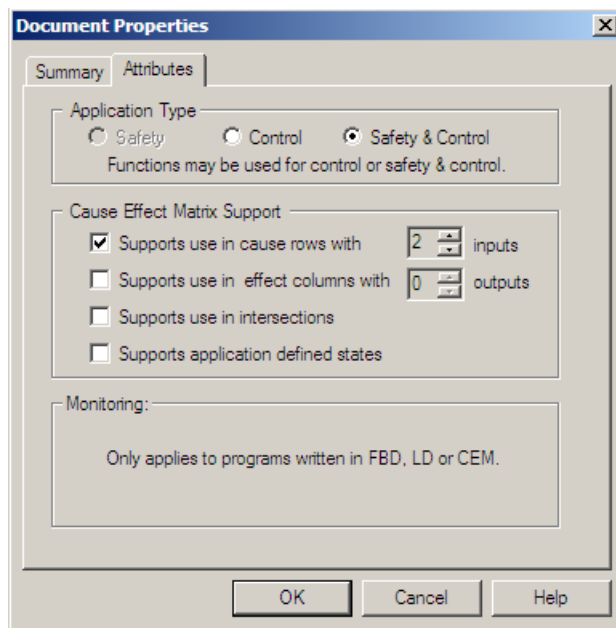
- 7 Highlight **02:ANY_BIT** and check **Invert Input**. Close the **Item Properties** screen.

If an input or output is inverted, the value is changed to the opposite value (True to False or False to True) when the function is executed.



This function will be used in creating the Cause and Effect Matrix in a subsequent lesson. You must enable the function so it can be invoked by the matrix.

- 8 On the **Document** menu, click **Properties**, and then click the **Attributes** tab.

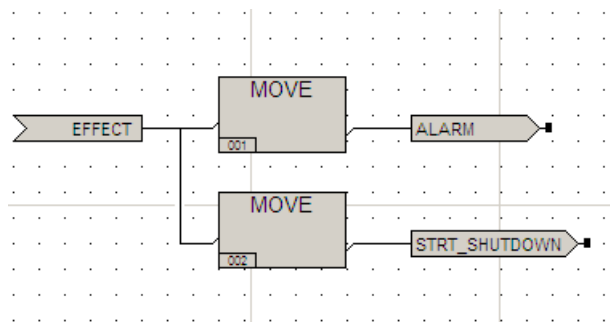


- 9 Check **Support use in cause rows**. Use the up/down arrow to display **2 inputs**. Click **OK**.
- 10 Compile the function by clicking the **Compile** button. Correct any errors.
- 11 Save and close the project.

Lesson 40: Creating a Function Block

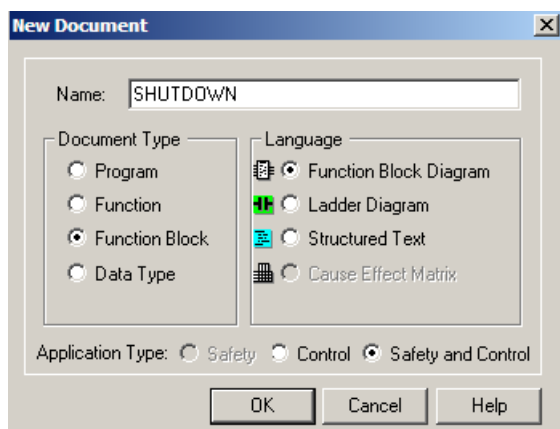
In this section, you will create a function block named SHUTDOWN that will be invoked from a matrix in another lesson. You can review the steps for creating FBD logic in Chapter Three: Writing FBD Logic.

This is the completed FBD logic.



Procedure

- 1 Open the CEMPLE project you created in a previous lesson.
- 2 Expand the **Application** tree, right-click the **User Documents** folder, and then click **New Document**.

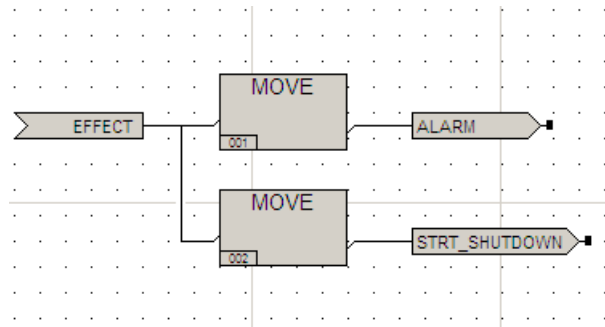


- 3 Specify the following:

Name:	Enter SHUTDOWN
Document Type:	Click Function Block
Language:	Click Function Block Diagram
Application Type:	Click Safety and Control

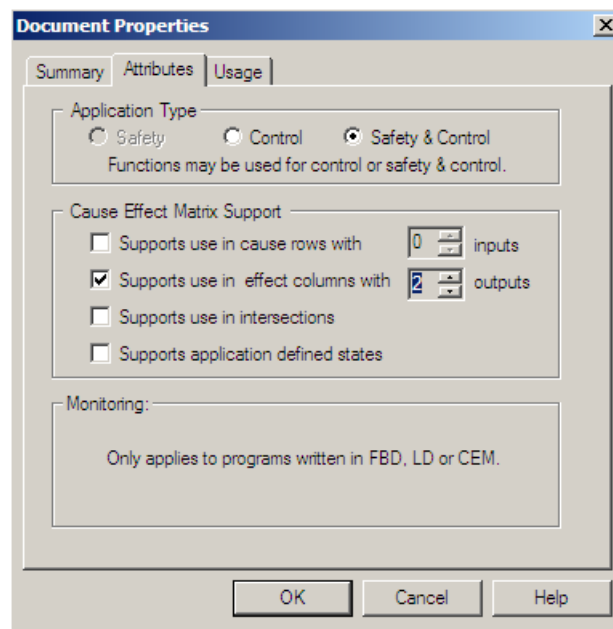
- 4 Click **OK** to save. The document is opened in FBD and displays the logic sheet.

- 5 Create the following FBD logic for the Alarm. Declare all variable types as **BOOL**.



This function block will be used in creating the Cause and Effect Matrix in a subsequent lesson. You must enable the function block so that it can be invoked by the matrix.

- 6 On the **Document** menu, click **Properties**, and then click the **Attributes** tab.



- 7 Check **Support use in effects columns**. Use the up/down arrow to display **2 outputs**. Click **OK**.
- 8 Compile the FBD logic by clicking the **Compile** button. Correct any errors.
- 9 Save and close the project.

Lesson 41: Creating a Simple Matrix

In this lesson, you will create a matrix controlling five unit alarms. It will have five causes and five effects.

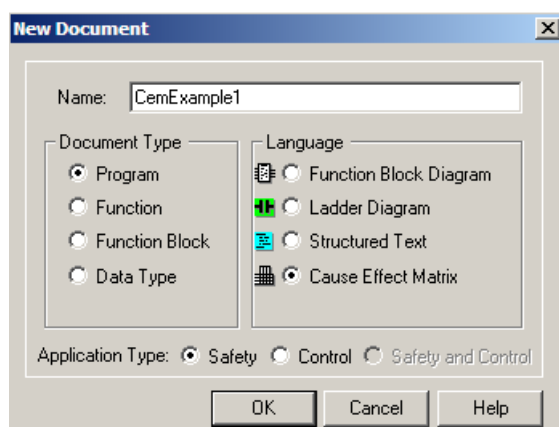
In this type of matrix, the causes are directly mapped to effects. An "X" in an intersection associates a cause with an effect, so when the cause is True, the effect is True. Conversely, when the cause is False, the effect is also False.

This is the completed matrix.

		Effect						
		UNIT_1_ALARM	UNIT_2_ALARM	UNIT_3_ALARM	UNIT_4_ALARM	UNIT_5_ALARM		
		Description						
		High level alarm indicator for Tank 1	High level alarm indicator for Tank 2	High level alarm indicator for Tank 3	High level alarm indicator for Tank 4	High level alarm indicator for Tank 5		
Cause	Description	E01	E02	E03	E04	E05	Comments	
LEVEL_1_HI	True=Fluid level in Tank 1 is high	C01	X				Turn on alarm immediately	
LEVEL_2_HI	True=Fluid level in Tank 2 is high	C02		X			Turn on alarm immediately	
LEVEL_3_HI	True=Fluid level in Tank 3 is high	C03			X		Turn on alarm immediately	
LEVEL_4_HI	True=Fluid level in Tank 4 is high	C04				X	Turn on alarm immediately	
LEVEL_5_HI	True=Fluid level in Tank 5 is high	C05					X	Turn on alarm immediately

Procedure

- 1 Open the CEMPLE project that you created in a previous lesson.
- 2 Expand the **Application** tree, right-click the **User Documents** folder, then click **New Document**.



- 3 Specify the following:

Name:	Enter CemExample1
Document Type:	Click Program
Language:	Click Cause Effect Matrix
Application Type:	Click Safety

- 4 Click **OK** to save. The document is opened in CEMPLE and displays the logic sheet.

		OR OR OR OR OR							
		Effect							
		Description							
	Cause	Description	E01	E02	E03	E04	E05	Comments	
			C01	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
			C02	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
			C03	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
			C04	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
			C05	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

- 5 Click the cause cell in the first cause row, key in **LEVEL_1_HI** and press **Enter**.

Cause	Description	E01	E02	E03	E04	E05	Comments
LEVEL_1_HI		C01					
		C02					
		C03					
		C04					
		C05					

- 6 Click the description cell in the first cause row, key in **True = Fluid level in Tank 1 is high** and press **Enter**.

Cause	Description	E01	E02	E03	E04	E05	Comments
LEVEL_1_HI	True=Fluid level in Tank 1 is high	C01					
		C02					
		C03					
		C04					
		C05					

- 7 Click the effect cell in effect column 1, key in **UNIT_1_ALARM** and press **Enter**.

				OR	OR
			Effect	UNIT_1_ALARM	
			Description		
	Cause	Description		E01	E02
	LEVEL_1_HI	True=Fluid level in Tank 1 is high	C01	<input type="checkbox"/>	<input type="checkbox"/>
			C02	<input type="checkbox"/>	<input type="checkbox"/>

- 8 Click the description cell in effect column 1, key in **High level alarm indicator for Tank 1** and press **Enter**.

				OR	OR
			Effect	UNIT_1_ALARM	
			Description	High level alarm indicator for Tank 1	
	Cause	Description		E01	E02
	LEVEL_1_HI	True=Fluid level in Tank 1 is high	C01	<input type="checkbox"/>	<input type="checkbox"/>
			C02	<input type="checkbox"/>	<input type="checkbox"/>

- 9 Associate **Cause C01** with **Effect E01** by clicking the intersection cell. An **X** appears in the cell. Then click in the first **Cause** row of the **Variable Detail Table**.


		OR	OR
		Effect	UNIT_1_ALARM
		Description	High level alarm indicator for Tank 1
Cause	Description	E01	E02
LEVEL_1_HI	True=Fluid level in Tank 1 is high	C01	X
		C02	

The Variable Detail Table displays the local variables. The FBD Network pane displays a graphic of the Move function created in a previous lesson.

Cause	Description	E01	E02	E03	E04	E05	Comments
LEVEL_1_HI	True=Fluid level in Tank 1 is high	C01	X				
		C02					
		C03					
		C04					
		C05					

Loc	Terminal	VarConst	VarType	Data Type	Description
C01		LEVEL_1_HI	Local	BOOL	

Cause Header Row 01



FBD Network Pane


The input and output to this MOVE function are local variables automatically defined by CEMPLE and cannot be changed. In more complex matrixes, the Variable Detail Table contains elements that can be modified.

- 10 Click the comment cell in row 1, key in **Turn on alarm immediately**, and then click **Enter**.

Cause	Description	E01	E02	E03	E04	E05	Comments
LEVEL_1_HI	True=Fluid level in Tank 1 is high	C01	X				Turn on alarm immediately

- 11 Complete the matrix by repeating steps 6 through 10 for **Cause rows**, **Effects columns**, and **Comment rows 2 through 5**. This is the completed matrix:

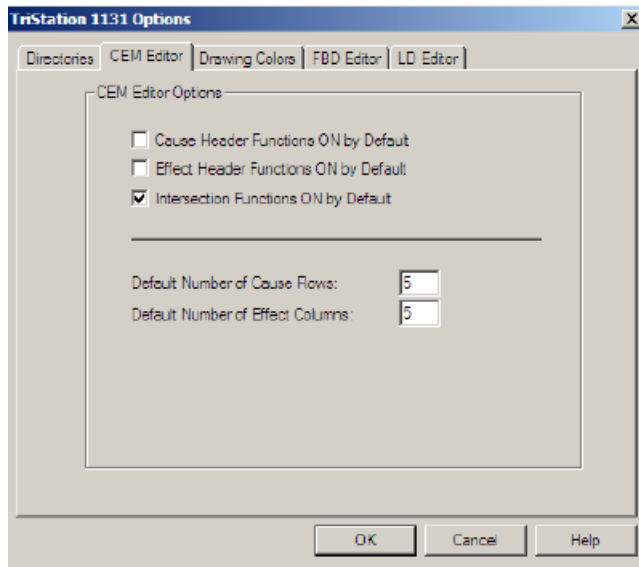
			OR						
			Effect						
			Description						
			High level alarm indicator for Tank 1						
			High level alarm indicator for Tank 2						
			High level alarm indicator for Tank 3						
			High level alarm indicator for Tank 4						
			High level alarm indicator for Tank 5						
Cause	Description		E01	E02	E03	E04	E05	Comments	
LEVEL_1_HI	True=Fluid level in Tank 1 is high	C01	X					Turn on alarm immediately	
LEVEL_2_HI	True=Fluid level in Tank 2 is high	C02		X				Turn on alarm immediately	
LEVEL_3_HI	True=Fluid level in Tank 3 is high	C03			X			Turn on alarm immediately	
LEVEL_4_HI	True=Fluid level in Tank 4 is high	C04				X		Turn on alarm immediately	
LEVEL_5_HI	True=Fluid level in Tank 5 is high	C05					X	Turn on alarm immediately	

- 12 Compile the program by clicking the **Compile** button  on the toolbar.
- 13 Save the program and close the project.

Lesson 42: Creating a Matrix with Intersection Functions

In the matrix created in the previous section, causes were directly mapped to effects. However, in some situations, when a cause becomes TRUE, you may want the program to execute some additional logic before turning on the effect. For example, you can delay turning on the effect for a few milliseconds.

The Intersection Functions ON by Default option on the CEM Editor tab allows you to specify a function or function block in the intersection area rather than an X. An intersection function can monitor the cause state as well as other criteria to determine what the effect state should be.



In this lesson, you will create a matrix that uses the TON function block to delay the effect. The matrix uses the structure of the previous matrix, changing it to a TON function block, and adding Intersection Functions. You will:

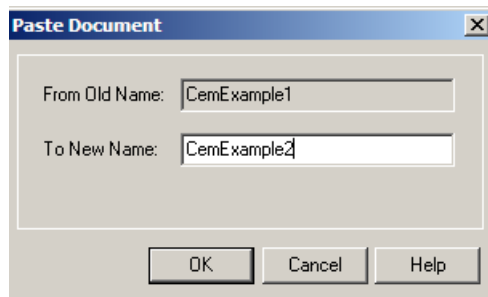
- Copy CemExample1 and rename it CemExample2.
- Specify a TON function block.
- Enable Intersection Functions.
- Define the input variable.

This is the completed matrix.

			Effect	OR	OR	OR	OR	OR
			Description	High level alarm indicator for Tank 1	High level alarm indicator for Tank 2	High level alarm indicator for Tank 3	High level alarm indicator for Tank 4	High level alarm indicator for Tank 5
Cause	Description		E01	E02	E03	E04	E05	Comments
LEVEL_1_HI	TRUE=FLUID level in Tank 1 is high	C01	TON					Turn on alarm immediately
LEVEL_2_HI	TRUE=FLUID level in Tank 2 is high	C02		TON				Turn on alarm immediately
LEVEL_3_HI	TRUE=FLUID level in Tank 3 is high	C03			TON			Turn on alarm immediately
LEVEL_4_HI	TRUE=FLUID level in Tank 4 is high	C04				TON		Turn on alarm immediately
LEVEL_5_HI	TRUE=FLUID level in Tank 5 is high	C05					TON	Turn on alarm immediately
Loc	Terminal	Var/Const	VarType	DataType	Description			
C01		LEVEL_1_HI	Local	BOOL				

Procedure

- 1 Open the CEMPLE project created in the previous lesson.
- 2 Expand the **User Documents** folder.
- 3 Expand the **Programs** folder, right-click **CemExample1**, and then click **Copy** from the menu.
- 4 On the **Edit** menu, click **Paste**.



- 5 Name the new program **CemExample2**. Click **OK**. The new program appears in the Application tree.

- 6 Double-click **CemExample 2** to open it.

The matrix now includes a drop-down list of pre-defined functions and function blocks.

				OR	OR	OR	OR	OR	
				UNIT_1_ALARM	UNIT_2_ALARM	UNIT_3_ALARM	UNIT_4_ALARM	UNIT_5_ALARM	
				High level alarm indicator for Tank 1	High level alarm indicator for Tank 2	High level alarm indicator for Tank 3	High level alarm indicator for Tank 4	High level alarm indicator for Tank 5	
		Cause	Description	E01	E02	E03	E04	E05	Comments
		LEVEL_1_HI	TRUE=FLUID level in Tank 1 is high	C01	MOVE				Turn on alarm immediately
		LEVEL_2_HI	TRUE=FLUID level in Tank 2 is high	C02	MOVE				Turn on alarm immediately
		LEVEL_3_HI	TRUE=FLUID level in Tank 3 is high	C03		TON			Turn on alarm immediately
		LEVEL_4_HI	TRUE=FLUID level in Tank 4 is high	C04			MOVE		Turn on alarm immediately
Loc	Terminals	Var/Const	VarType	DataType	D TOGGLE TON TP TP_I TP_R				
C03E0	PT	ALRM_DELAY	Local	TIME					

- 7 Specify the **TON** function block for **Cause Row 1 (01)** and **Effect Column 1 (E01)** by selecting **TON** from the drop-down list in the intersection cell.

		Cause	Description	E01	E02	E03	E04	E05	Comments
		LEVEL_1_HI	TRUE=FLUID level in Tank 1 is high	C01	TON				Turn on alarm immediately
		LEVEL_2_HI	TRUE=FLUID level in Tank 2 is high	C02	MOVE				Turn on alarm immediately
		LEVEL_3_HI	TRUE=FLUID level in Tank 3 is high	C03		MOVE			Turn on alarm immediately
		LEVEL_4_HI	TRUE=FLUID level in Tank 4 is high	C04			MOVE		Turn on alarm immediately
		LEVEL_5_HI	TRUE=FLUID level in Tank 5 is high	C05				MOVE	Turn on alarm immediately

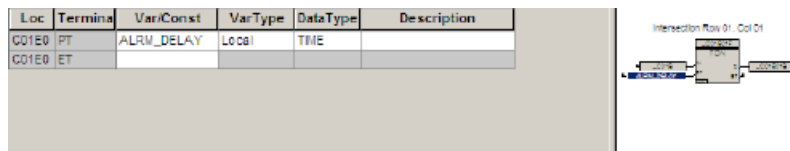
The matrix is now associated with a TON function block. The text TON appears in the cell. The Variable Detail Table and FBD Network are displayed.

- 8 To complete the TON function, you must add a PT (time) input. Click in the **CO1** and **E01** intersection cell to display the **Variable Detail Table**. Enter **ALRM_DELAY** in the white cell of the **Var Name** column and press **Enter**.

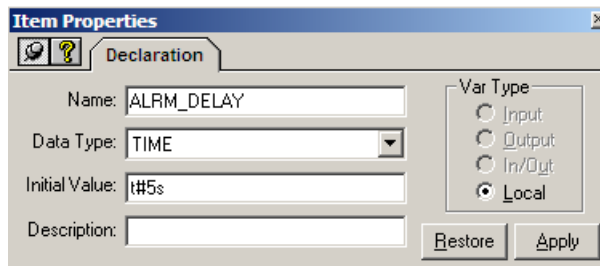
The input name, variable type and data type are displayed.

Loc	Terminals	Var/Const	VarType	DataType	Description
C01E0	PT	ALRM_DELAY	Local	TIME	
C01E0	ET				

The TON function graphic in the FBD Network pane displays the PT input.



- 9 Declare the **PT** variable by clicking the **ALRM_DELAY** graphic in the FBD Network to display the **Item Properties** screen. Click the **Declaration** tab.



- 10 Enter the initial value **t#5s**, click **Apply**, and then close the screen.
- 11 Complete the matrix by repeating **steps 7 and 8** for intersections for **Cause Rows 2** through 5. Specify the intersections as **TON** and add an **alarm delay** for each intersection in the **Var Name** column of the **Variable Detail**.

This is the completed matrix.

				Effect		UNIT_1_ALARM		UNIT_2_ALARM		UNIT_3_ALARM		UNIT_4_ALARM		UNIT_5_ALARM			
Description		High level alarm indicator for Tank 1		High level alarm indicator for Tank 2		High level alarm indicator for Tank 3		High level alarm indicator for Tank 4		High level alarm indicator for Tank 5							
Cause		Description		E01		E02		E03		E04		E05		Comments			
LEVEL_1_HI		TRUE=FLUID level in Tank 1 is high		CO1		TON								Turn on alarm immediately			
LEVEL_2_HI		TRUE=FLUID level in Tank 2 is high		CO2				TON						Turn on alarm immediately			
LEVEL_3_HI		TRUE=FLUID level in Tank 3 is high		CO3						TON				Turn on alarm immediately			
LEVEL_4_HI		TRUE=FLUID level in Tank 4 is high		CO4								TON		Turn on alarm immediately			
LEVEL_5_HI		TRUE=FLUID level in Tank 5 is high		CO5										TON		Turn on alarm immediately	

Loc	Terminal	Var/Const	VarType	DataType	Description
CO1		LEVEL_1_HI	Local	BOOL	

- 12 Compile the program. Correct any errors.
- 13 Save the program and close the project.

Lesson 43: Creating a Matrix with Cause Header Functions

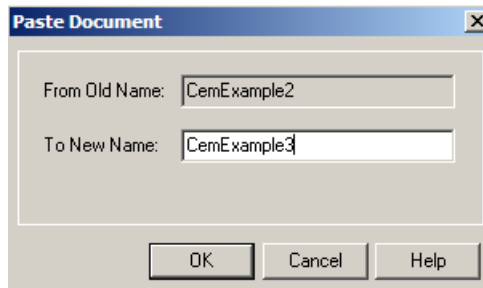
In a typical CEM, the cause is the input to the Cause Row. However, some applications may require:

- Multiple process values as the inputs to a Cause Row
- Function or function block that evaluates the process values to determine the actual cause state

In this lesson you will create a matrix controlling five alarms, using a TON (Timer On) function block, and a user-defined LVLALRM function that allows two process inputs. The matrix uses the structure of the previous matrix, which included the Intersection Functions. The new matrix adds Cause Header Functions and specifies the LVLALARM function created in a previous lesson.

Procedure

- 1 Open the CEMPLE project created in a previous lesson.
- 2 Double-click the **User Documents** folder.
- 3 Double-click the **Programs** folder, right-click **CemExample2**, and then click **Copy** from the menu.
- 4 Click **Edit** on the toolbar, and then click **Paste**.



- 5 Name the new program **CemExample3**. Click **OK**. The new program appears in the **Application** tree.
- 6 Double-click **CemExample3** to open it. The matrix now displays the Function and Cause columns.

	Input	Function	Cause	Description		E04	E05	Comments
	LEVEL_1_HI			TRUE=FLUID level in Tank 1 is high	C01			Turn on alarm immediately
	LEVEL_2_HI			TRUE=FLUID level in Tank 2 is high	C02			Turn on alarm immediately

- 7 Select the **LVLALRM** function from the drop-down list in the Function column for each of the five Cause Rows.

Input	Function	Cause	Description	E04	E05	Comments
LEVEL_2_HI	LVLALRM		TRUE-FLUID level in Tank 2 is high	C02		Turn on alarm immediately
LEVEL_3_HI	LVLALRM		TRUE-FLUID level in Tank 3 is high	C03		Turn on alarm immediately
LEVEL_4_HI	NOT		TRUE-FLUID level in Tank 4 is high	C04	TON	Turn on alarm immediately
LEVEL_5_HI	R_TRIG		TRUE-FLUID level in Tank 5 is high	C05	TON	Turn on alarm immediately

- 8 The matrix now displays cells for two inputs. Click the second input cell for **LEVEL_1**, key in **LEVEL_1_LOW**, and then click **Enter**. The variable name appears in the field.

Loc	Terminal	Var/Const	VarType	DataType	Description
C01	HI_LEV	LEVEL_1_HI	Local	BOOL	
C01	LOW_LE	LEVEL_1_LOW	Local	BOOL	

- 9 Key in **LEVEL_2_LOW** through **LEVEL_5_LOW** for **Low Level** inputs.
- 10 **BYP** (Bypass) is represented by a blank cell in the Variable Detail Table. Select the **Level1_1** cause row. Key in **BYP_LEV_1**, and then click **Enter**.

Loc	Terminal	Var/Const	VarType	DataType	Description
C01	HI_LEV	LEVEL_1_HI	Local	BOOL	
C01	LOW_LE	LEVEL_1_LOW	Local	BOOL	
C01	BYP	BYP_LEV_1	Local	BOOL	
C01	LVLALR				

- 11 Key in the variable names **BYP_LEV_2** through **BYP_LEV_5** for the bypass variables.
- 12 Compile the program, correcting any errors.
- 13 Save the program and close the project.

Lesson 44: Creating a Matrix with Effect Header Functions

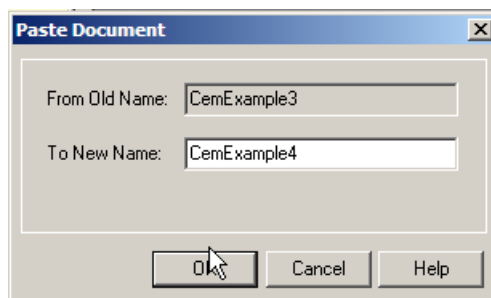
In the previous matrixes, the effect is the output of the Effect Column. However, you can use Effect Header function and function blocks in either of these two ways:

- To process the effect state before setting the outputs of the Effect Column
- To set multiple outputs for the Effect Column

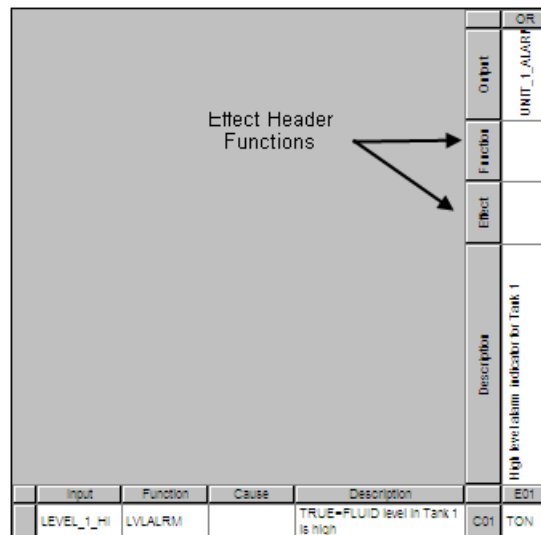
In this lesson, you will create a matrix that specifies the SHUTDOWN function block created in a previous lesson. The matrix uses the structure of the previous matrix, which included the Cause Header Functions. The new matrix adds Effect Header Functions and specifies the SHUTDOWN function with outputs.

Procedure

- 1 Open the CEMPLE project created in a previous lesson.
- 2 Double-click the **User Documents** folder.
- 3 Double-click the **Programs** folder, right-click **CemExample3**, and then click **Copy** from the menu.
- 4 Click **Edit** on the toolbar, and then click **Paste**.



- 5 Name the new program **CemExample4**. Click **OK**. The new program appears in the **Application** tree.



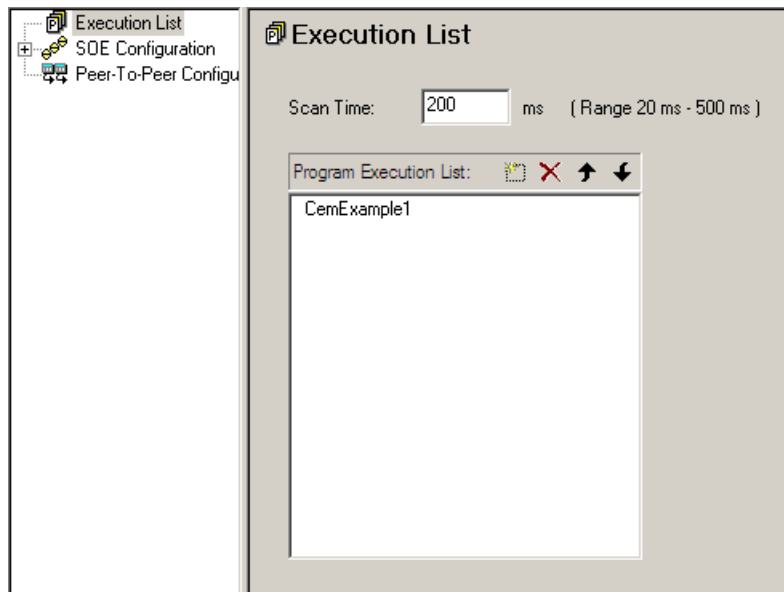
- In the FBD Network pane, the second output is displayed connected to the STRT_SHUTDOWN terminal of the SHUTDOWN function block. CEMPLE automatically assigns the default data type BOOL and the default Variable type of Output to each of the output variables.
- 10 Compile the program, correcting any errors.
 - 11 Save program and close the project.



Lesson 45: Monitoring an Instance View

In this lesson, you will monitor an instance view of a matrix using CemExample1.

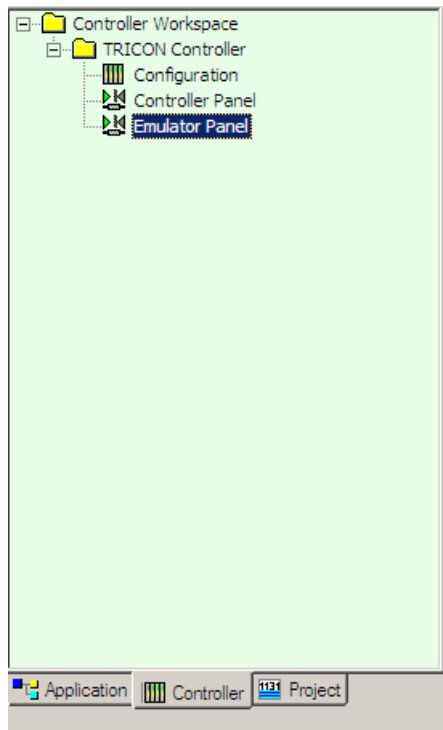
Procedure



- 1 Open the Cemple project.
- 2 Double-click the **User Documents** folder, the **Programs** folder, and then click **CemExample1**.
- 3 On the **Application** tree, double-click **Implementation**. The **Execution List** is displayed.

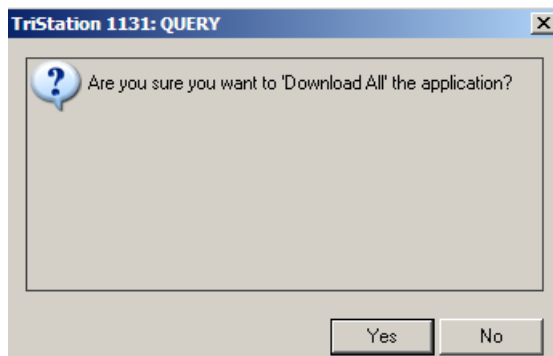


- 4 On the **Program Execution List** toolbar, click the **New (Insert)** button  and then click the **Browse** button  to display the list of programs in the project.
- 5 Click **CemExample1** to add it to the Execution list.
- 6 Close the program before connecting to the Emulator.

- 7 Double-click the **Controller** tab, and then double-click **Emulator Panel**.




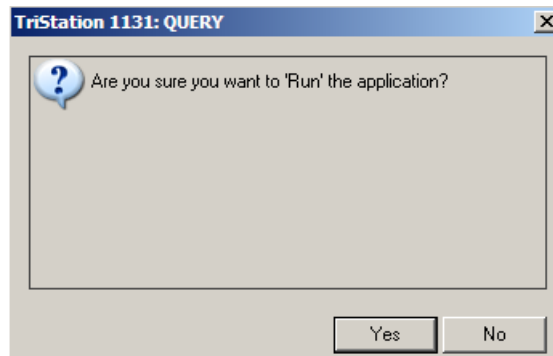
- 8 Click the **Connect** button . Click the **Download All** Button .




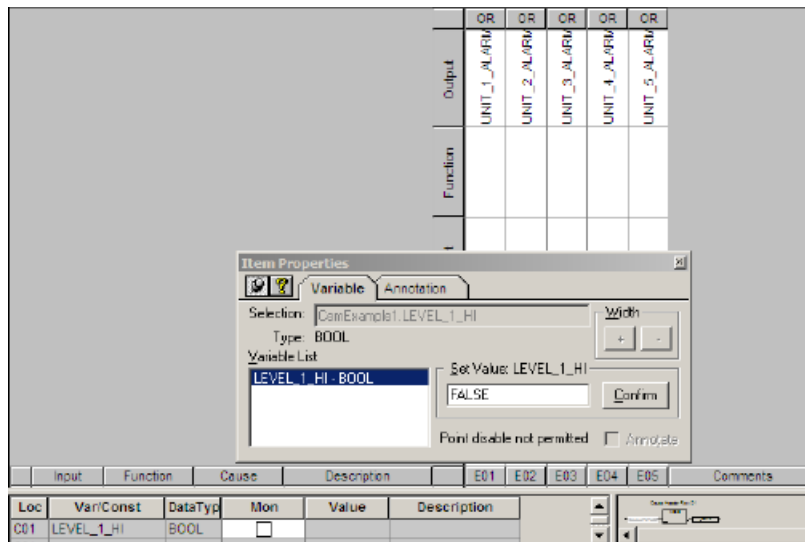
- 9 Click **Yes** to the Download All query. The system builds the application and displays status information in the Message Bar.

```
>> Building Configuration...
    ++ Initializing program 'CemExample1'...
>> Creating program instances
>> Assembling Libraries for Emulator...
    > Linking for emulation...
The estimated stack size is 516 bytes.
0 ERROR(s), 0 WARNING(s)
```

- 10 Correct any errors and then click the **Run** button .



- 11 Click **Yes** to the Run application query.
- 12 On the **Emulator** tree, expand **Programs**, and click the **Display Program Document** button . The matrix is displayed.
- 13 Double-click a **Cause Variable** in the **Variable Detail Table**. The **Item Properties** for that variable is displayed.



- 14 In the **Set Value** field, enter **True** or **1** and click the **Confirm** button. The Cause Row and Effect Column goes active and turns red.

Tricon and Trident Controller Configuration

Tricon Controller Configuration	204
Allocating Memory	211
Allocating Hardware	214
Configuring Tricon Communication with External Devices	234
Tricon Time Synchronization	244
Trident Controller Configuration	250
Configuring Trident Communication with External Devices	262
Trident Time Synchronization	273

Tricon Controller Configuration

Overview

Physically, a basic Tricon system consists of:

- Field-replaceable modules
- Chassis that house the modules
- Field wiring connections
- Programmer's workstation

Modules

Modules are field-replaceable units consisting of an electronic assembly in a protective metal housing. Each module is fully enclosed to ensure that no components or circuits are exposed—even when a module is removed from the chassis. Offset connectors backplane connectors make it impossible to plug a module in upside down, and keys on that prevent the insertion of modules into incorrect slots.

Chassis

Modules are collectively housed in a chassis. A Tricon controller can include up to fifteen chassis housing any combination of input, output, and spare modules, as well as communication modules.

The Main Chassis of the Tricon controller houses the Main Processor modules and up to six I/O sets. Expansion Chassis support up to eight I/O sets each.

Field Wiring Connections

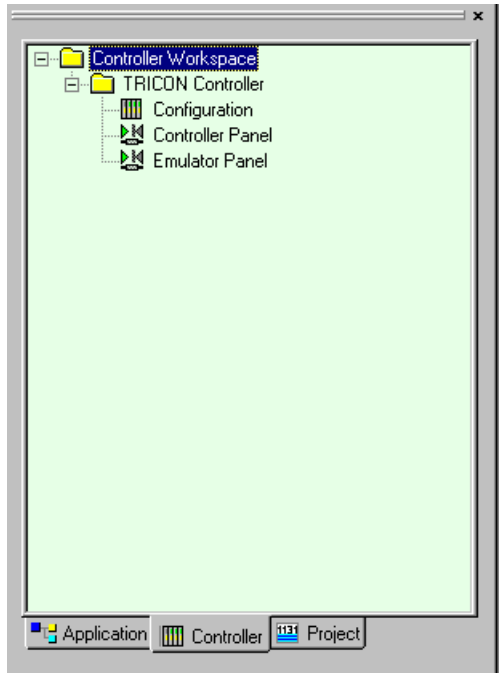
External termination assemblies are available for connection to field devices. You can also connect your own cables directly to the connectors at the top of the Tricon backplane.

Programmer's Workstation

TriStation 1131 supports Windows NT and Windows 2000.

System Configuration

After creating an application, you must configure the controller by defining the system configuration in the Controller Workspace.



A system configuration is created by:

- Setting operating parameters
- Allocating memory and hardware
- Configuring communication with external devices
- Set up time synchronization

The system configuration is used to graphically and textually define I/O modules and memory points for each slot in the Tricon or Trident controller, and assign the global variables associated with each point.

Operating Parameters

Operating parameters are settings that restrict or allow connection access and write access to a Tricon controller.

These types of read and write access are possible:

- Input, output, and memory points can be read by any external device that can communicate with a Tricon controller.
- Write access to input points is not allowed from any external device.
- Write access to an output or memory point is allowed or restricted based on the system, communication, application, and point settings.

External devices must use supported communication protocols (Modbus, TSAA, OPC, and DDE) to communicate with the controller.

This table describes write access to Tricon points from external devices.

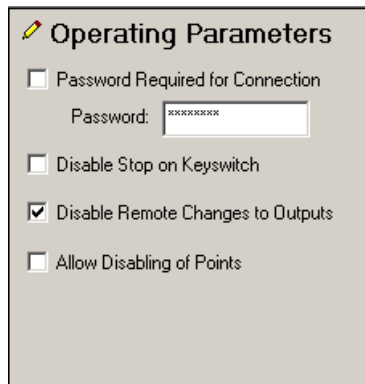
Tricon Write Access

Property or Feature	Description
Tricon Keyswitch	<p>A system setting that determines write access to output and memory points unless overruled by the GATENB function block in the application.</p> <ul style="list-style-type: none"> • Restricts write access when set to the Run position. • Allows write access when set to the Remote or Program position.
GATENB	A Tricon function block that programmatically allows write access to a specified range of aliased memory points when the keyswitch is in the Run position.
GATDIS	A Tricon function block that programmatically restricts remote write access for ranges of aliased memory points that were previously enabled by GATENB.
Disable Remote Changes to Outputs	A system setting on the Operating Parameters screen that determines write access to output points.

Property or Feature	Description
Privilege	<p>A TriCon ACM and NCM module setting that determines whether network devices using DDE, OPC, or TSAA communication have write access to output points and read/write aliased memory points.</p> <p>For Tricon ACM, the default is Read.</p> <p>For Tricon NCM, the default is Read/Write.</p> <p>For Tricon EICM, HIM, and SMM modules do have this property.</p>
Prohibit Writes	<p>A Tricon SMM module setting that determines whether Honeywell devices have write access to output points and read/write aliased memory points.</p> <p>The default is unchecked, which means write access is allowed.</p>
Point Assignment	<p>A tagname setting that determines whether the output and memory point is assigned a Read or Read/Write alias number.</p>

Setting Operating Parameters

Operating Parameters are set using the Operating Parameters screen in the Controller Configuration function.

The screenshot shows a configuration window titled "Operating Parameters" with a pencil icon. It contains four settings: "Password Required for Connection" (unchecked), "Disable Stop on Keyswitch" (unchecked), "Disable Remote Changes to Outputs" (checked), and "Allow Disabling of Points" (unchecked). A password field is visible next to the first option, containing several asterisks.

These include:

- Setting a password
- Disabling Stop on the Keyswitch
- Disabling Remote Changes to Outputs
- Allowing Disabling of Points

Setting the Password

The Password Required for Connection property specifies whether a password is required to connect to the controller. If access is restricted, only users with access privileges can disable points or download changes to the controller.

If the box is checked, you must also enter a password. This setting takes effect after the application is downloaded to the controller. A screen displays requiring the password to be entered before the connection can be attempted.

The default is unchecked, which means a password is not required.

Disabling Stop on the Keyswitch

The keyswitch is a four-position switch (Run, Program, Stop, Remote) on the Tricon Main Chassis. The position of the keyswitch enables or disables control functions for the entire Tricon controller.

The Disable Stop on Keyswitch property specifies whether to logically disable the STOP position of the keyswitch on the Tricon Main Chassis. When stopped, the controller stops reading inputs, forces non-retentive digital and analog outputs to 0, and halts the control program. Retentive outputs are returned to the value they had before the keyswitch was turned to Stop. This may be used for security reasons, or during installation or service of process-related equipment.

If checked, setting the keyswitch to STOP does not halt the application. The default is unchecked, which means that the application is stopped the keyswitch is turned to STOP.

Disabling Remote Changes to Outputs

The Remote position on the keyswitch allows writes to program variables by **TriStation**, Modbus masters and external hosts.

The Disable Remote Changes to Outputs property specifies whether external devices, such as other TriStations, Modbus, and external hosts, can write to output points in the **TriStation 1131** application.

If checked, external devices cannot write to output points regardless of the settings for other properties. You should check this property if the application includes safety-critical outputs.

The default is checked, which means output points cannot be changed by external devices.

Allow Disabling of Points

Disabling or "forcing" points prevents inputs from field instruments from changing the value of the points. It is recommended that you disable points only during initial online test of an application or when field instruments need to be replaced or repaired, avoiding a system shutdown. For example, you can disable a water valve so the water tank can be repaired. Disabling points can increase the scan time.

The Allow Disabling of Points property specifies whether points can be disabled from the **TriStation 1131** PC. A **TriStation 1131** PC cannot write to disabled points. However, external devices such as Modbus masters can write to disabled points, unless you disable external device writes.

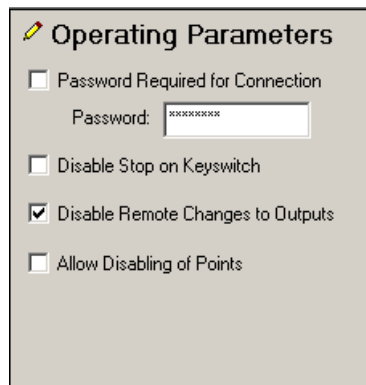
The default is unchecked, which means points cannot be disabled from **TriStation 1131** PC.

Lesson 46: Setting Tricon Operation Parameters

In this lesson, you will set operation parameters for the water tank alarm.

Procedure

- 1 Open the **Test_Project** project.
- 2 Click the **Controller** tab to display the **Controller** tree, and then double-click **Configuration**.
- 3 Double-click **Operating Parameters**.



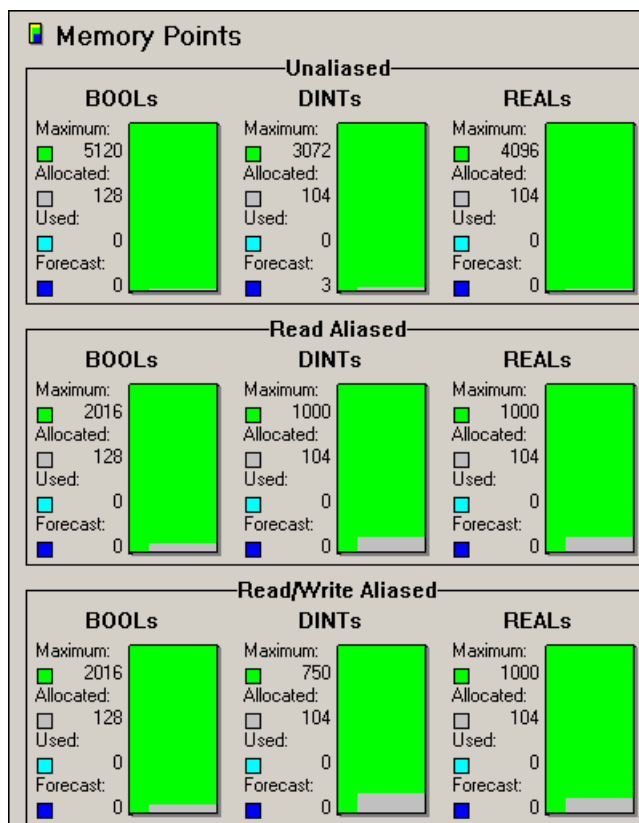
- 4 Leave the **Password Required for Connection** unchecked.
- 5 Check the following boxes:
 - Disable Stop on Keyswitch
 - Disable Remote Changes to Outputs
 - Allow Disabling of Points
- 6 Click **Save Project** to save the operating parameters.

Allocating Memory

Initially, the amount of memory for input and output points is determined by the number and type of I/O modules configured in the **TriStation 1131** project. A default amount of memory for memory points is set when a **TriStation 1131** project is created. With each successful build, **TriStation 1131** allocates memory for each of the memory points, inputs points, and output points.

You can increase the allocation of memory variables when necessary. This is especially important because allocation sizes are frozen for all memory points after you download the application, and additional memory cannot be allocated.

This is an example of how memory allocation is displayed when clicking Memory Allocation on the Configuration tree.



The memory allocation for each point type is displayed as a vertical bar graph. The bar displays:

- Maximum- Maximum number of points available for the specific point
- Allocated- Number of points allocated
- Used- Number of points used as of last Download
- Forecast- Number of points to use the next time the application is downloaded to the controller

Lesson 47: Allocating Memory For Points

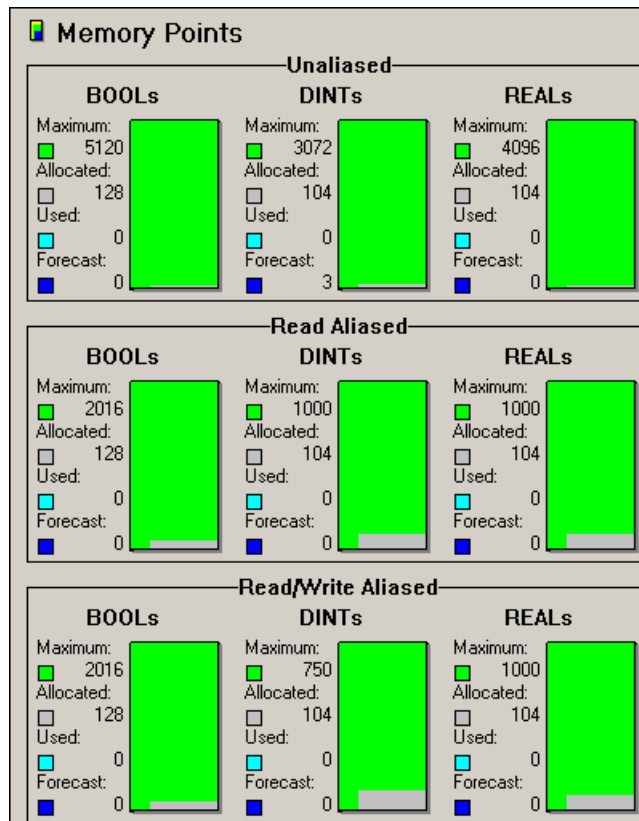
In this lesson you will:

- Access the Memory Allocation display
- Learn how to change memory allocation
- Access the Application Data graph

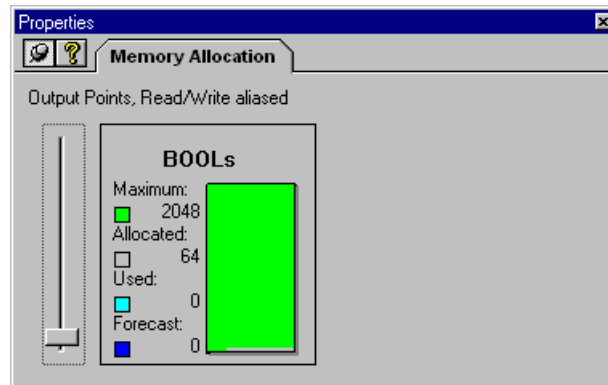
Procedure

Change memory allocation

- 1 Open the project Water_Tank_Alarm.
- 2 Click the Controller tab.
- 3 Expand the Controller tree, double-click Configuration, and then expand Memory Configuration. Memory Allocation for the project is displayed.



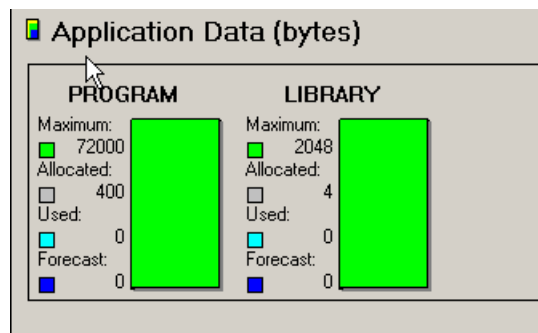
- 4 Double-click the type of point you want to change. The Item Properties screen for that point type is displayed.



- 5 Move the slider up or down to change the memory allocated for the selected point.
- 6 Application Data displays two graphs: Program and Library

Access Application Data

- 1 The Memory Allocation display includes the Application Data graphs.



Program displays the amount of bytes allocated for local variables in program instances.

Library shows the amount of bytes allocated for libraries.

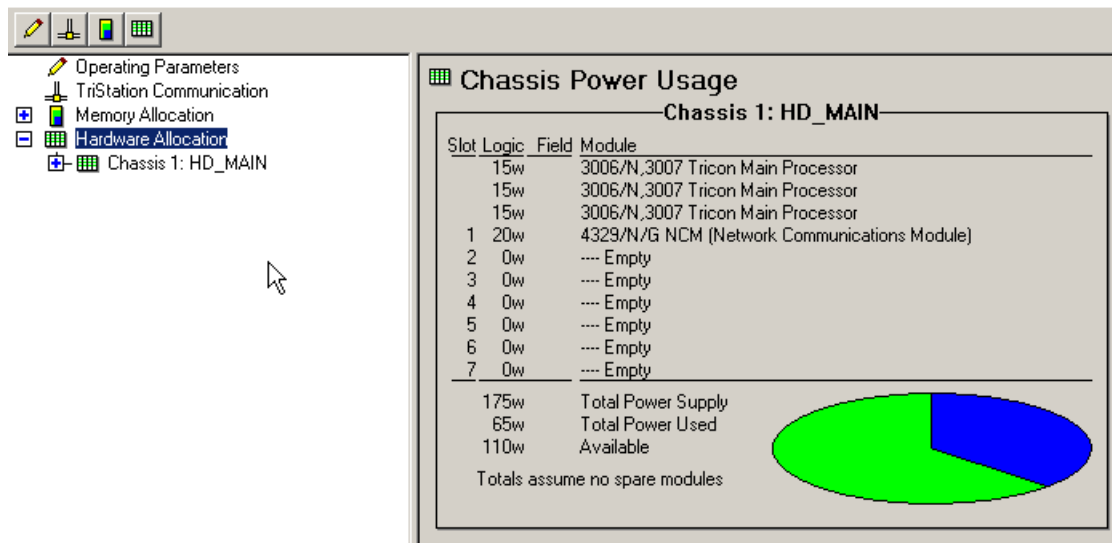
As a user, you cannot directly change the memory allocation for Application Data. However, if you rebuild a project after updating its shared libraries, **TriStation 1131** may change the memory allocation for the application.

Allocating Hardware

Chassis Power Usage

Prior to downloading a configuration, the Tricon checks its physical hardware against the hardware allocated when you configured your project. In allocating hardware, you must first ensure that you have sufficient logic power for your project. Power usage or logic power refers to the amount of logic power consumed by a specific type of module in the chassis.

The Chassis Power Usage screen shows information about the logic power used by each chassis. Each chassis in your Tricon system has a separate display.



The display lists the slots, modules in the chassis, and the logic power used by each modules. The pie chart graphically represents these proportions.

In this example, there are three Tricon Main Processor modules. The 4119 EICM (Intelligent Communications Module) is assigned by default to communications slot. The allocation for the COM slot can be changed at any time based on the specific needs of the configuration being emulated or downloaded.

The Chassis Power Usage display also shows:

- Total Power Supply- maximum logic power that the chassis can support.
- Total Power Used- amount of logic power being used by the current configuration of modules in a chassis
- Available- amount of available (unused logic power)

If the Available power is negative, the chassis is using too much logic power. You must delete one or more modules from the chassis and add them to another chassis in the configuration. You may need to add chassis to the configuration to accommodate the modules.

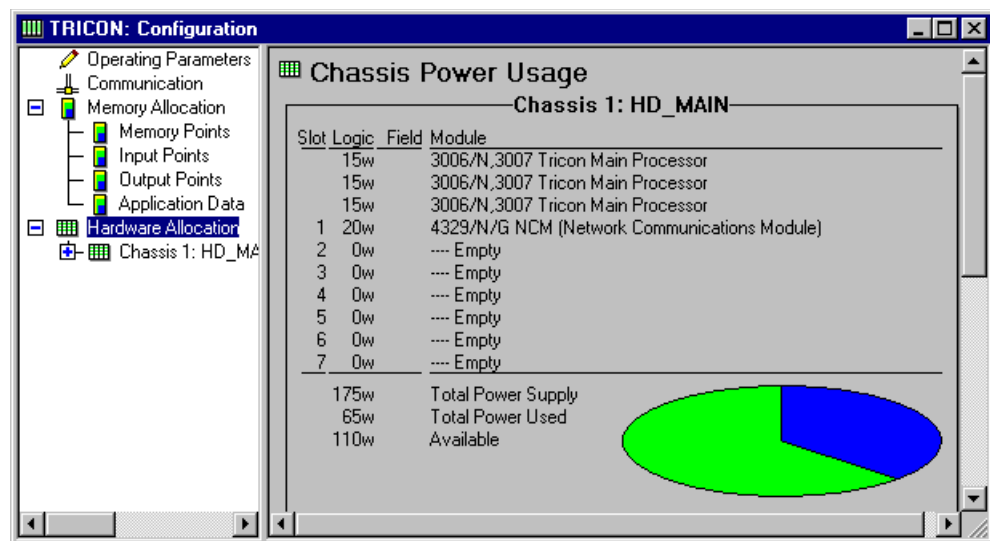
Lesson 48: Determining Chassis Power Usage

In this lesson you will:

- Determine Tricon Power Usage.
- Learn how to add or delete a Tricon chassis.

Procedure

- 1 Open the **Water_Tank_Alarm** project.
- 2 Click the **Controller** Tab.
- 3 Expand the **Controller** tree, double-click **Configuration**, and then click **Hardware Allocation**. The **Chassis Power Usage** screen is displayed.

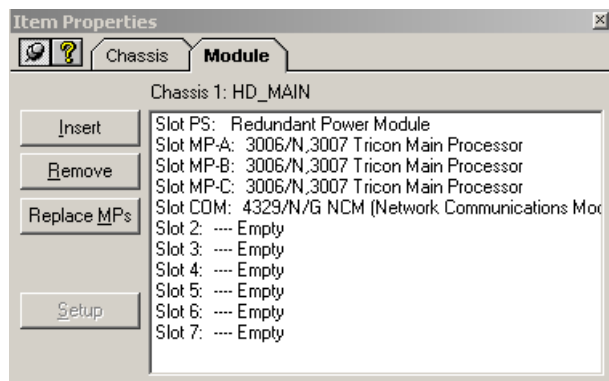


- 4 Check the Available power. If the number is negative, you would delete one or more modules from the chassis and add them to another chassis in the configuration.

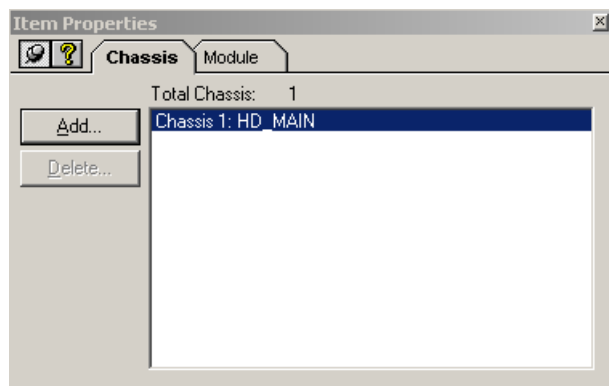
5 To add or delete a Tricon Chassis, do either of the following:

- Double-click the **Chassis Power Usage** screen.
- Double-click **Hardware Allocation**.

The Item **Properties** screen is displayed.

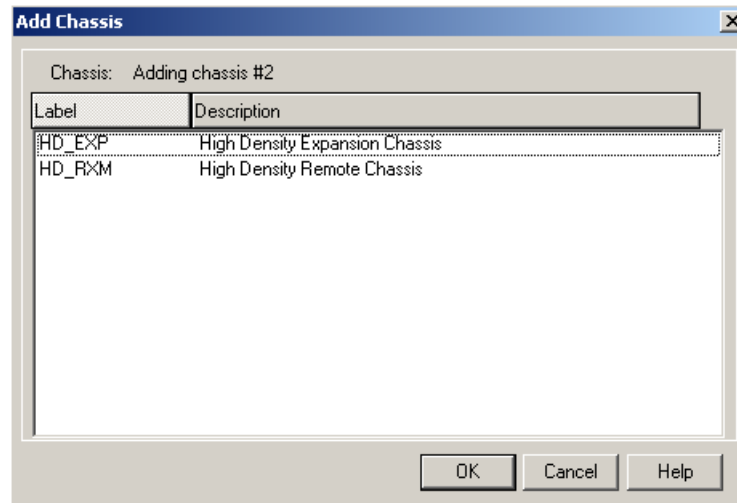


6 Click the **Chassis** tab.



7 Do either of the following:

- To add a chassis, click **Add**. Select the type of chassis, and then click **OK**.



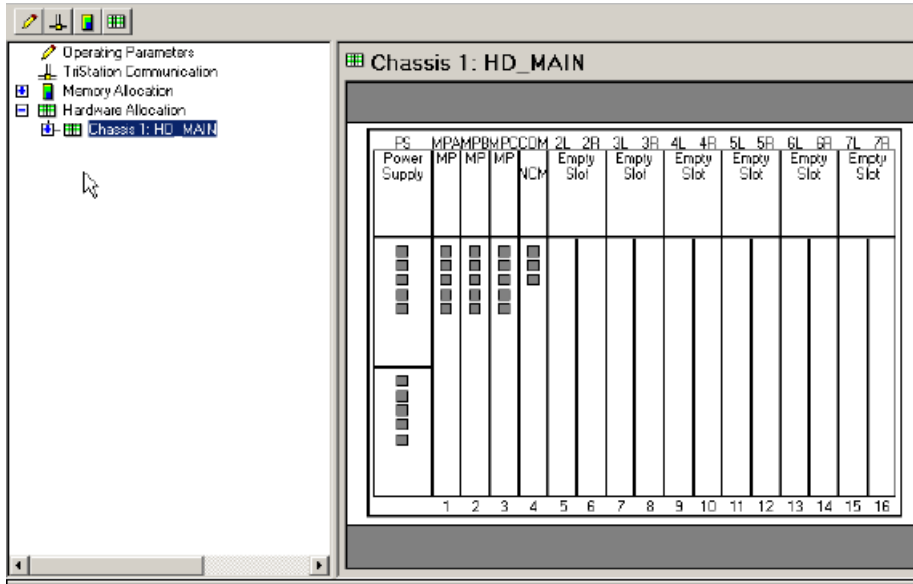
The **Item Properties** screen for the additional chassis is displayed. You can make modifications to the chassis or close the screen.

- To delete a chassis, display the **Item Properties** screen, and then click the **Chassis** tab.

Select a chassis and then click **Delete**. Close the screen. The chassis is removed from the Hardware allocation list.

Chassis Window

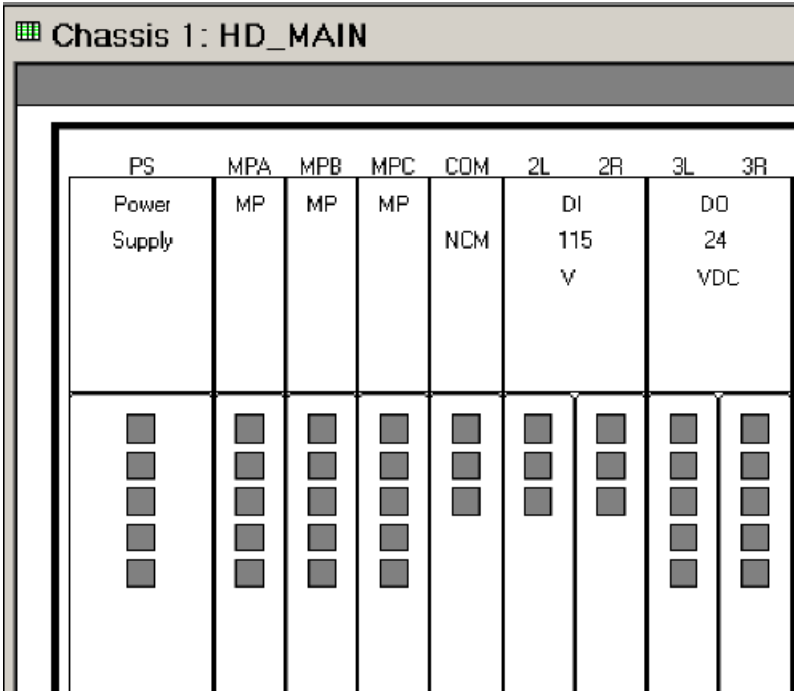
The Chassis window shows the physical setup of modules for each chassis on the controller. Each slot is labeled and corresponds to the slots on the controller. In the software, the slots in the Tricon chassis are marked as "Empty" until they are assigned.



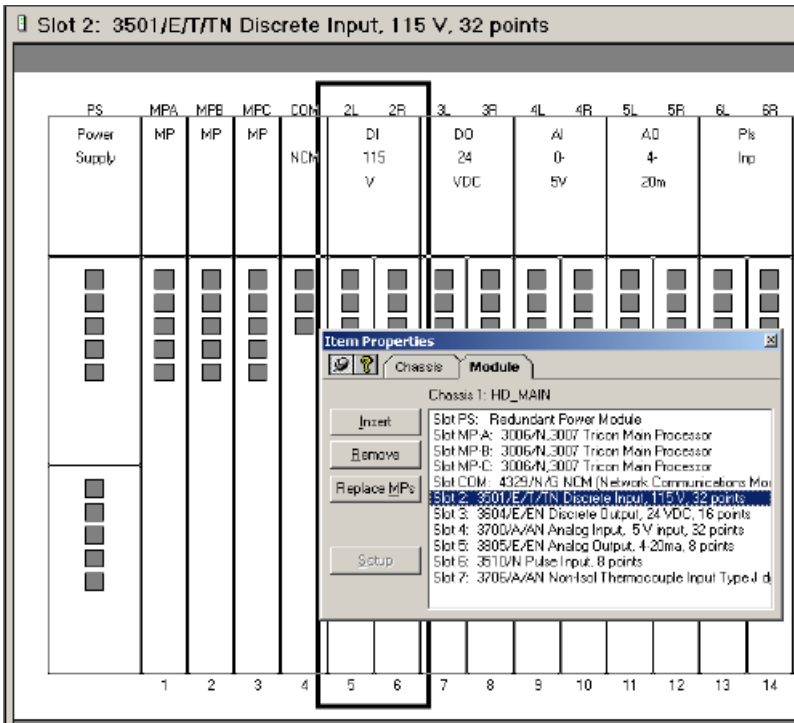
The small blocks at the top of each module represent the status indicators for each function. For example, the blocks in the Power supply column represent the Pass, Fail, Alarm, and Battery Low indicators. The color of each square corresponds to the actual color of the indicator on the module installed in the Tricon controller and indicate the following:

- Green - Pass
- Yellow - Active
- Red - Fault
- Gray - Off

Each I/O module in the Tricon occupies one of two slots that constitute an "I/O set." The left module is label "L" and the right module is labeled "R." At any time, the status of either the left or right module can be "active" or "hot-spare" for online replacement or backup.



Double-clicking a on a slot displays the **Item Properties** screen for that I/O module.



From this screen, you can:

- Insert modules into the chassis.
- Remove modules from the configuration.

Replace Tricon main processors.

Types of Tricon Modules

Tricon Main Processor Modules (MP)

The Tricon controller has three Main Processor modules. Each controls a separate channel of the system and operates in parallel with the other two Main Processors. A dedicated I/O Processor on each Main Processor manages the data exchanged between the Main Processor and the I/O Modules.

Tricon Selectable Modules

The Tricon controller has the following types of selectable modules:

- Digital Input
- Digital Output
- Analog Input
- Analog Output
- Pulse Input
- Pulse Totalizer Input
- Thermocouple Input
- Enhanced Intelligent Communication Module (EICM)
- Network Communication Module (NCM)
- Safety Manager Module (SMM)
- Hiway Interface Module (HIM)
- Advanced Communication Module (ACM)

Digital Input Modules

Each digital input module has three independent channels that process all data input to the modules. On each channel, a microprocessor scans each input point, compiles data, and transmits it to the Main Processor before processing to ensure the highest integrity.

All digital input modules sustain complete, ongoing diagnostics for each channel. Failure of any diagnostic on any channel activates the module's FAULT indicator, which in turn, activates the chassis alarm signal. The FAULT indicator points to a channel fault, not a module failure. The module will continue to operate in the presence of a single fault and may continue to operate with certain kinds of multiple faults.

Digital Output Modules

Each digital output module houses the circuitry for three identical isolated channels. Each channel includes an I/O microprocessor that receives its output table from the I/O communication processor on its corresponding Main processor. The output module executes a voter diagnostic, reading the output value for each point to determine whether a latent fault exists within the output circuit.

Analog Input Modules

All analog input modules have three independent input channels. Each input channel receives variable voltage signals from each point, converts them to digital values and transmits the values to the three Main Processors. One value is then selected to ensure correct data for every scan. Sensing of each input prevents a single failure on one channel from affecting another channel.

Analog Output Modules

Analog Modules receive output signals from the Main Processor on each of three channels. Each set of data is voted, and a functioning channel is selected to drive the outputs.

Pulse Input Modules

Pulse Input Modules are commonly used with tachometers or speed sensors on equipment such as turbines or compressors to monitor speeds of rotation for motors, fans, and pumps.

Pulse Totalizer Input Modules

Pulse Totalizer Input Modules function as counter with equipment such as flowmeters to determine total processes or counts as inputs to batch operations.

Thermocouple Input Modules

A thermocouple is a kind of thermometer consisting of two wires of dissimilar metals. The wires are connected at the "hot junction," which is the temperature to be measured. The other ends of the wires terminate at the controller or the "cold junction," and are held at a fixed lower temperature. Keeping the metals at different temperatures generates voltage that can be measured and interpreted as a temperature reading on equipment connected to the Tricon controller.

Enhanced Intelligent Communication Module (EICM)

The Enhanced Intelligent Communication Module (EICM) enables the Tricon to communicate with Modbus devices (masters or slaves).

Modbus is an industry-standard master/slave protocol that is traditionally used for energy management, transfer line control, pipeline monitoring, and other rugged industrial processes. The Tricon controller uses variables as identifiers. Modbus devices use numeric addresses called aliases as identifiers. Any standard Modbus device can communicate with the Tricon controller by means of the EICM provided that aliases are assigned to the Tricon variables. A Modbus communication link can use either the Remote Terminal (RTU) or ASCII mode of transmission.

Network Communication Module (NCM)

The Network Communication Module (NCM) is used for TCP/IP networking. The NCM supports all Triconex protocols and applications, user-written applications, and TCP/IP networking with external systems. These include:

- TriStation – TriStation Programming System
- Peer-to-Peer – Proprietary network for a maximum of 10 Tricons
- Time Synchronization – Master/Slave protocol for maintaining a consistent time base
- TSAA – Master/Slave protocol used for SOE, DDE, SER, and other Triconex applications
- TSAA/TCP/IP – Master Slave protocol used for user-written application for external hosts

Safety Manager Module (SMM)

The Safety Manager Module acts as an interface between a Tricon controller and Honeywell's Universal Control Network.

Hiway Interface Module (HCM)

The Hiway Interface Module acts as an interface between a Tricon controller and Honeywell's TDC-300 control system, enabling higher-order devices to communicate with Tricon controllers.

Advanced Communication Module (ACM)

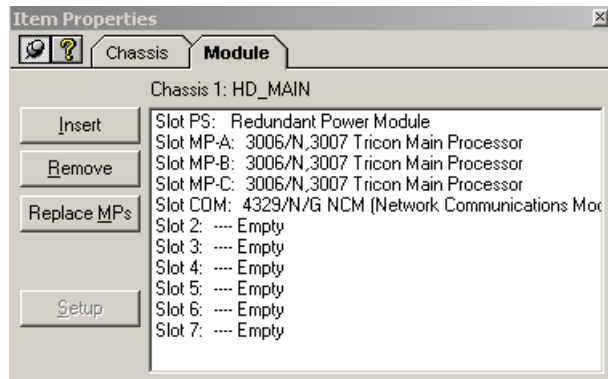
The Advanced Communication Module acts as an interface between a Tricon controller and Foxboro's Intelligent Automation (I/A) Series DCS.

Lesson 49: Inserting Tricon Modules

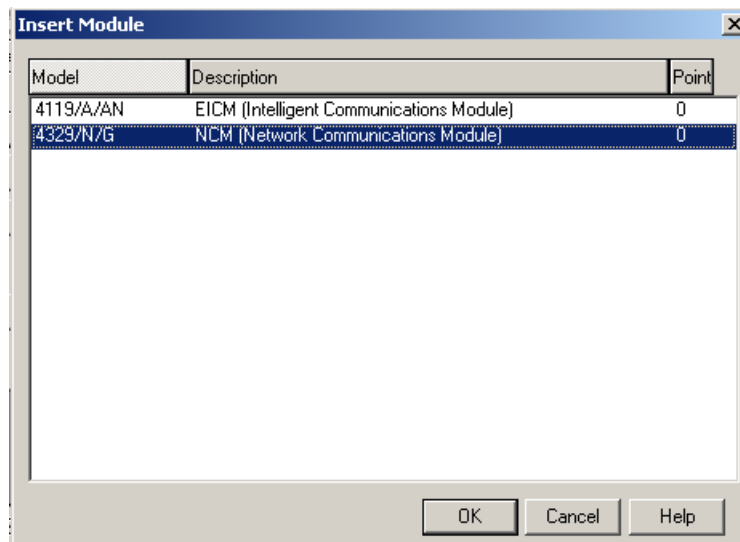
In this lesson, you will insert and install modules into the main chassis.

Procedure

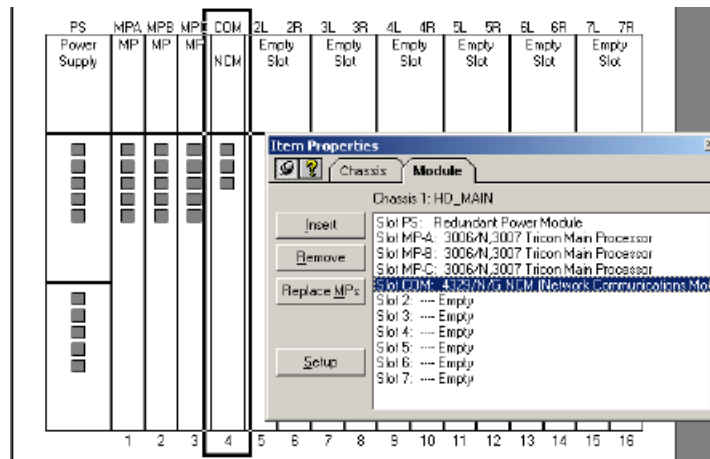
- 1 Expand the **Controller** tree, double-click **Configuration**, and expand **Hardware Allocation**. Double-click **Chassis 1**. The **Item Properties** screen is displayed.



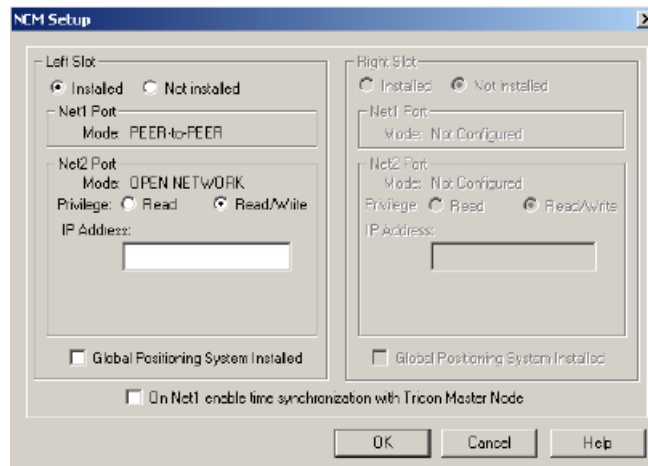
- 2 Click **NCM** (Network Communications Module) and then click **Insert**. The **Insert Module** screen is displayed.



- 3 In this example, two kinds of communication modules are available. Select the **Network Communications Module** and then click **OK**. The module appears in the chassis.



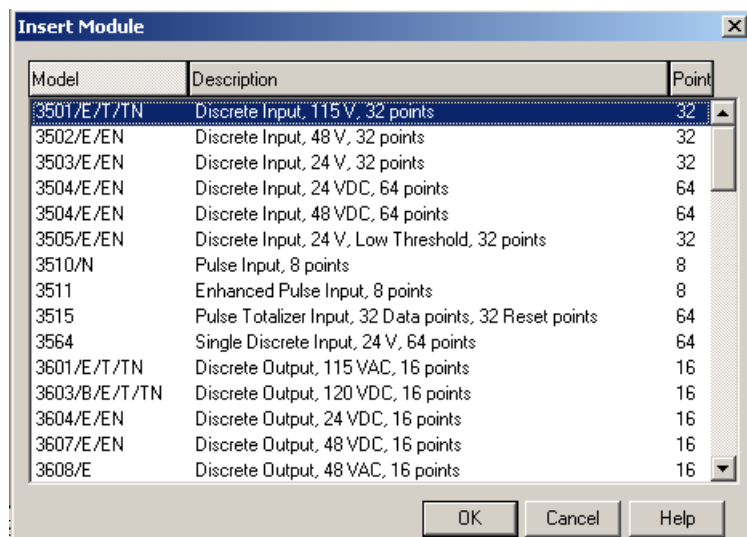
- 4 Click **Setup** on the **Item Properties** screen. The **NCM Configuration** screen is displayed.



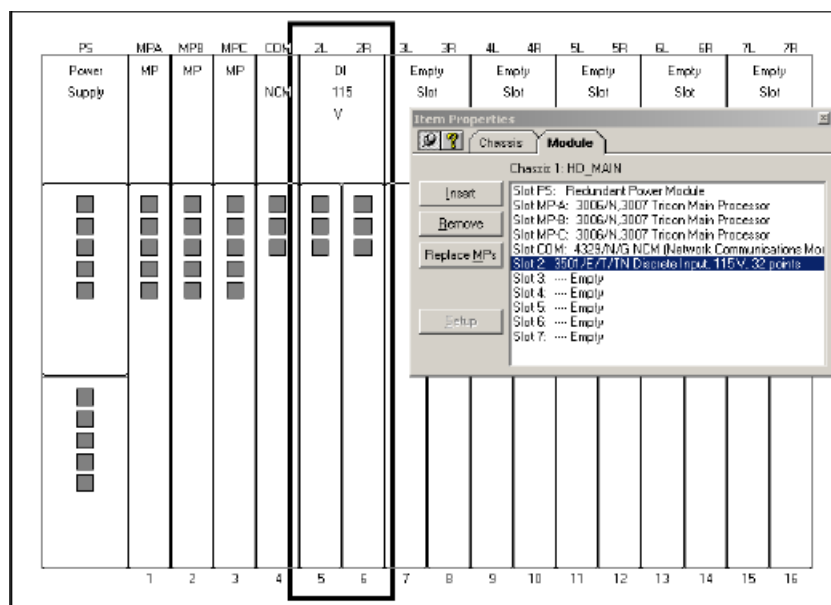
- 5 Under Left Slot, click **Installed**, and then click **OK**. The communication module is now installed and configured for your project.

Next you will install a digital input module.

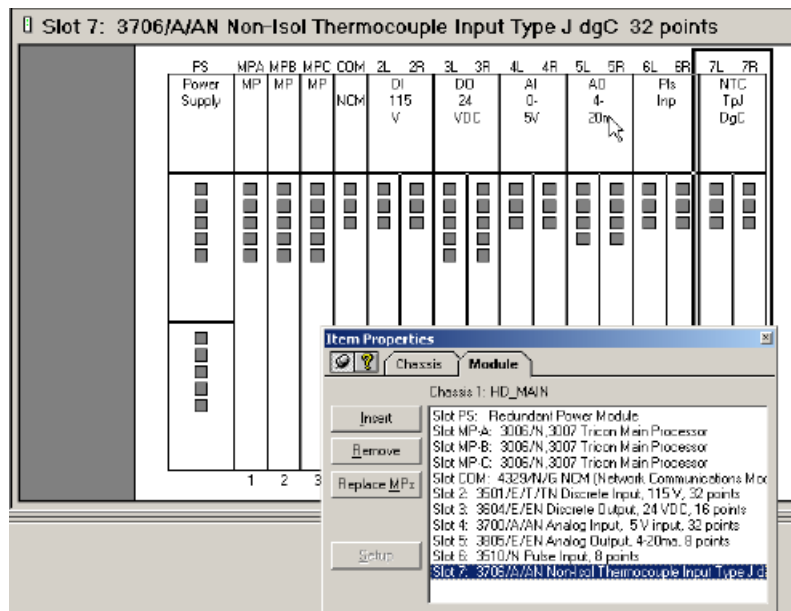
- 6 On the **Item Properties** screen, click **Slot 2** and then click **Insert**. The **Insert Module** screen is displayed.



- 7 Click **3501/E/T/TN Discrete Input 115V 32 points**, and then click **OK**. The module is inserted in the slot.



- 9 To remove a module from the chassis, double-click the module. The **Item Properties** screen is displayed.



- 10 Select the module and then click **Remove**.
- 11 Click **Save Project**.

Lesson 50: Replacing the Tricon MP Model

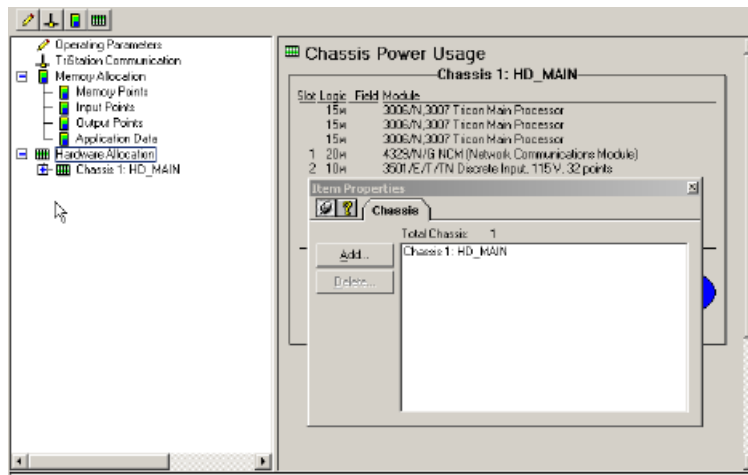
If you install different model Main Processors in the controller, you must logically configure the change in the **TriStation 1131** project. This change requires a Download All.

When you replace MPs, **TriStation 1131** saves the configuration and attaches the Tricon library that supports the installed MPs. Before saving the configuration, you are allowed to back up your project.

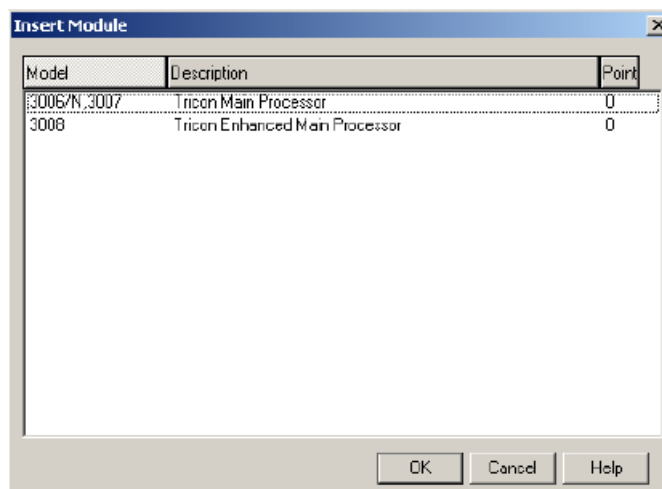
In this lesson, you will learn how to replace the model for the Tricon MPs in the **TriStation 1131** configuration.

Procedure

- 1 Expand the **Controller** tree, double-click **Configuration**, and expand **Hardware Allocation**. Double-click the chassis.



- 2 On the **Item Properties** screen, click **Replace MPs**.



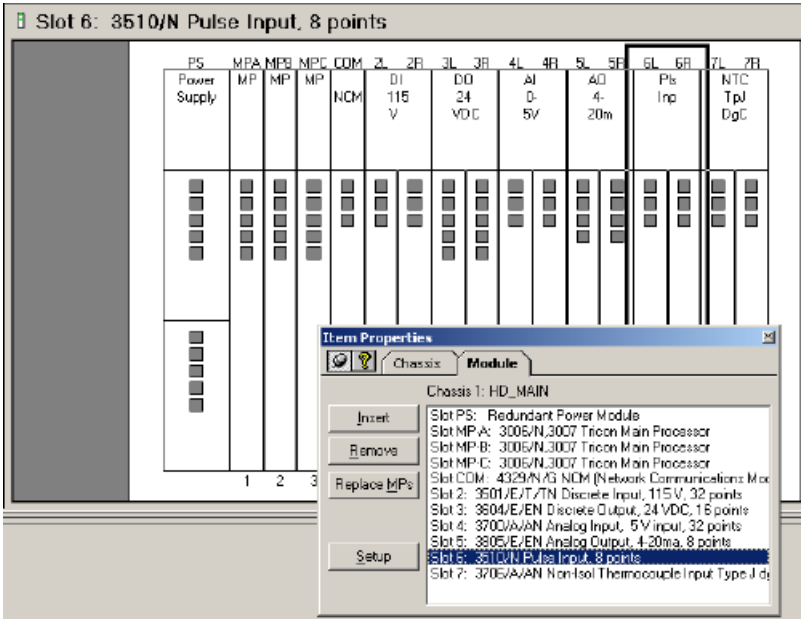
- 3 Click the model that is installed, and then click **OK**.
- 4 Click **OK** to save the project. To finish the process, you must build the application and download it to the controller.

Lesson 51: Configuring a Tricon Pulse Input Module

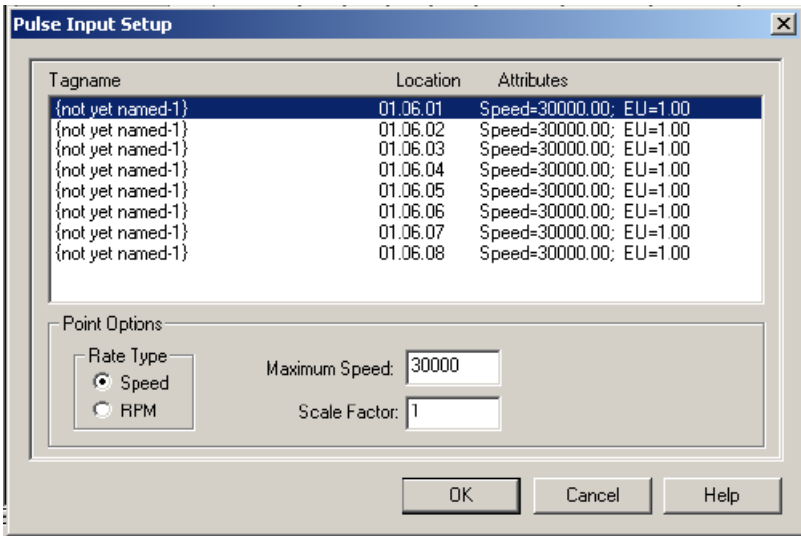
In this lesson you will learn how to configure a Tricon Pulse Input and Enhanced Pulse Input Module, which are used for speed and RPM inputs. Each point can be independently configured.

Procedure

- 1 Expand the **Controller** tree, double-click **Configuration**, and expand **Hardware Allocation**. Double-click the slot assigned to the **PI** module.



- 2 Click **Setup**. The **Pulse Input Setup** screen is displayed.



- 3 Click a tagname to be configured and specify these properties for each point on the **Pulse Input Setup** screen.

Property	Action
Rate Type	Click Speed or RPM to specify the type of rate applied to pulse input signals.
Maximum RPM	Enter the revolutions per minute for the pulse input device; used with RPM.
Maximum Speed	Enter the speed for the pulse input device; used with speed Point Type.
Pulses Per Revolution	Enter the speed for the pulse input device; used with speed Point Type.
Scale Factor	Enter the scaling value to convert pulse input signals into engineering units. For: <ul style="list-style-type: none"> • Pulses per second, set to 0.01 6667 • Pulses per minute, set to 1.0 (default) • Pulses per hour, set to 60.0

- 4 After all the tagnames are specified, click **OK** to save.

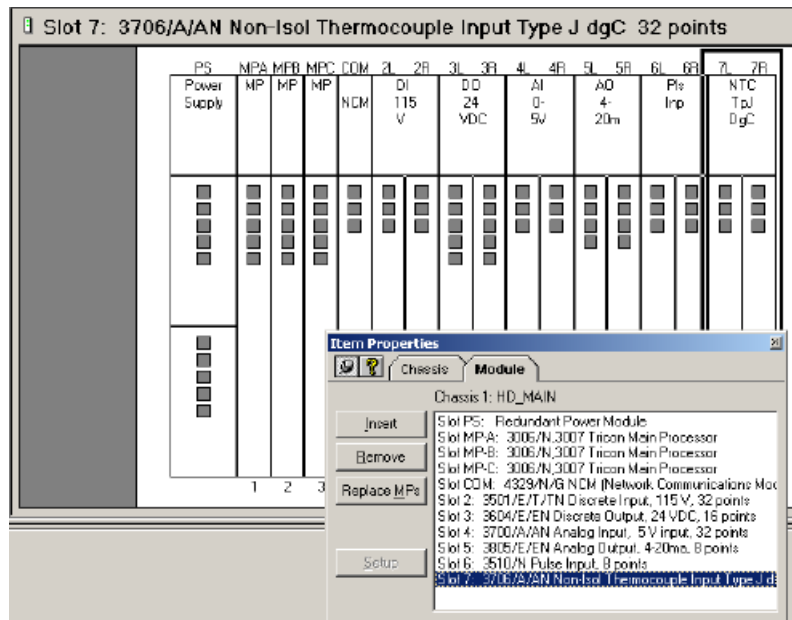
Lesson 52: Specifying a Tricon Thermocouple Module

In this lesson, you will specify the type of Tricon Thermocouple Module, which must match the module that is installed in the system. Because these modules come in a variety of configurations, select carefully.

For more information, see the *Tricon Planning and Installation Guide*.

Procedure

- 1 Expand the **Controller** tree, double-click **Configuration**, and expand **Hardware Allocation**. Double-click the slot assigned to the thermocouple module, and then click **Insert**.



- 2 Select the module type in the configuration, as indicated by these properties.

Property	Action
Model	3706/A/AN or 3708/E/EN
Type E, J, K, T	Specifies the thermocouple type installed. <ul style="list-style-type: none"> • For 3706, J, K, or T • For 3708, E, J, or T
Degree Conversion	dgC is converted to Celsius. dgF is converted to Fahrenheit.
UpS (upscale) DnS (downscale)	Specifies value returned if voltage is out-of-range, or burnout occurs. <ul style="list-style-type: none"> • Upscale returns +32,767 • Downscale returns -32,767 For model 3708E only
Points	For model 3706, 32 points. For model 3708, 16 points.

- 3 Click the type of module installed in the controller, and click **OK**. There are no other properties to specify.

Configuring Tricon Communication with External Devices

To communicate with external devices, Tricon modules must be configured to connect with a specific type of controller (Network or Serial) and to support specific types of port connections.

If these types of modules are assigned to a chassis, they must be configured to communicate with external devices:

- ACM
- EICM
- HIM
- NCM
- SMM

Lesson 53: Specifying the Tricon Default Connection

You can specify the default connection setting either Network or Serial Connection, when you establish communication between a **TriStation 1131** PC and a controller. This initial setting can be changed when you use the Connect command to connect to the controller.

In this lesson, you will learn how to specify the Tricon default connection.

Procedure

- 1 Expand the **Configuration** tree, double-click **Configuration**, and then click **TriStation Communication**.

The screenshot shows the 'Communication' configuration window. It is divided into three main sections. The top section, 'Network TCP/IP Addresses', contains three text input fields: 'Node Name' with the value 'TRINODE06', 'Node Number' with the value '6', and 'TCP/IP Address' with the value '206.32.216.26'. The bottom-left section, 'Default Connection', has two radio buttons: 'TRICON Serial Port' (which is selected) and 'TRICON Network'. Below the selected radio button is a dropdown menu showing 'COM1'. The bottom-right section, 'Debug Message Options', contains five unchecked checkboxes: 'Messages Sent & Received', 'Message Detail', 'State Changes', 'Errors', and 'Information'.

- 2 Specify these properties on the **TriStation Communication** screen.

Property	Action
Select Connections	If using ACM or NCM, click Network Connection. If using EICM, click Serial Connection.
Node Number	For a network connection, enter the number for the controller.
Node Name	For a network connection, enter the name for the controller.
IP Address	For a network connection, enter the IP address.
Serial Port	For a serial connection, select the TriStation 1131 PC port that is connected to the controller.
Default Connection	If only one Selected Connection is checked, the default connection is set based on it. If both network and serial connections are set, you must specify the default connection setting.

- 3 To complete the connection, you must configure a Tricon ACM, EICM, or NCM module.

Lesson 54: Configuring Tricon ACM Ports

The Tricon ACM supports these connections:

- On Net1, a network connection to a Foxboro Intelligent Automation (I/A) Series DCS
- On Net1, time synchronization with a Foxboro Intelligent Automation (I/A) Series DCS
- On Net2, a network connection to a **TriStation 1131** PC or other network devices
- On Net2, time synchronization with the Tricon master node

You can install primary and redundant ACM modules in either chassis 1 or chassis 2.

In this lesson, you will learn how to configure ports on a Tricon ACM.

Procedure

- 1 Expand the **Controller** tree, double-click **Configuration**, and expand **Hardware Allocation**. Double-click the **ACM** slot, and then click **Setup**.

The image shows the 'ACM Setup' dialog box. It has a title bar 'ACM Setup' with a close button. The dialog is divided into several sections:

- Net1 - ACM Connection:**
 - SQE Block:
 - Privilege: ☒ Read ☐ Read/Write ☒ Redundant mode
- Net2 - Left Slot (Network Connection):**
 - ☒ Used ☐ Not Used
 - Privilege: ☒ Read Only ☐ Read/Write
 - IP Address:
 - IP Subnet Mask (hex):
 - Default Gateway IP Address:
- Net2 - Right Slot (Network Connection):**
 - ☒ Used ☐ Not Used
 - Privilege: ☒ Read Only ☐ Read/Write
 - IP Address:
 - IP Subnet Mask (hex):
 - Default Gateway IP Address:
- Time Synchronization:**
 - ☐ On Net1 enable time synchronization with external source
 - ☐ On Net2 enable time synchronization with Tricon Master Node
 - ☒ None

At the bottom are three buttons: OK, Cancel, and Help.

- 2 For a Net1 connection, specify these properties on the **ACM Setup** screen.

Property	Action
SOE Block Name	If using SOE, specify the block number.
Privilege	Specify Read/Write to allow external devices to read and write points.
Redundant Mode	Check to specify that a redundant module is installed.
Time Synchronization	Click Net1 to enable time synchronization with I/A DCS.

- 3 For a Net2 connection, specify these properties on the ACM setup screen.

Property	Action
Redundant Mode	Check to specify that a redundant module is installed.
Used/Not Used	Click Used to specify the slots that have an installed ACM module. To enable the left slot, check Redundant Mode.
Privilege	Specify Read/Write to allow external devices to read and write points. The default is Read.
IP Address	If using a default IP address, leave blank. If not, enter the IP address that identifies the controller on the network. This must be the same IP address as entered on the Communication screen.
IP Subnet Mask	If the controller is on a subnet, enter the subnet address.
Gateway IP Address	If needed, enter an IP address to be used as the default for a gateway.
Time Synchronization	Click Net2 to enable time synchronization with the Tricon Master controller.

- 4 Click **OK** to save.

Lesson 55: Configuring Tricon EICM Ports

The Tricon EICM supports these serial connections:

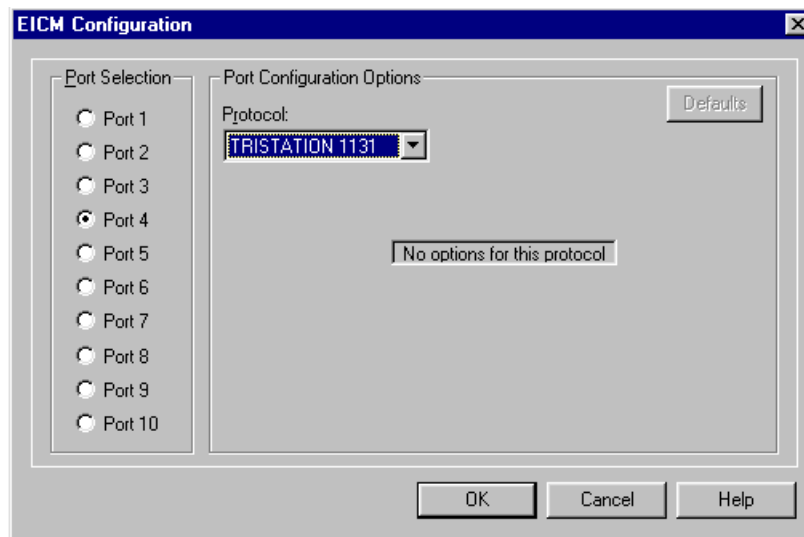
- To a **TriStation 1131** PC using **TriStation 1131** protocol
- To an external device using Modbus protocol (master, slave, and master/slave)
- To a Centronics printer

You can install an EICM module in either chassis 1 or the first expansion chassis.

In this lesson, you will learn how to configure ports on a Tricon ECM.

Procedure

- 1 Expand the **Controller** tree, double-click **Configuration**, and expand **Hardware Allocation**. Double-click the **EICM** slot, and then click **Setup**.



2 Specify these properties on the **EICM Setup** screen.

Property	Action
Port Selection	For TriStation 1131 or Modbus communication, click ports 1-4 or 6-9. For a printer connection, click ports 5 or 10.
Protocol	For TriStation 1131 , select TriStation 1131 . For Modbus, select the specific Modbus protocol.
Modbus Slave Address	Enter the slave address, which can be 1-247. Only available with Modbus slave and Modbus master/slave protocols.
Baud Rate	Click the rate which must be the same as other slaves on the network. The default is 9600. The total rate for all four ports must be less than or equal to 57,600.
Data Bits	Set as needed; must be the same as other Modbus slaves. <ul style="list-style-type: none"> • Modbus slave can use 7 or 8 bits • Modbus master and master/slave must use 8 bits.
Stop Bits	Click 1 Bit or 2 Bits to specify whether to send 1 or 2 bits to indicate that the transmission of a byte of data is complete.
Parity	Must use the same setting as other Modbus slaves.
Handshake	If set to Hardware, see Signal Delays in the Help documentation.
Modbus Alias Range	Set a minimum between 0 and 32767; maximum between 1 and 32767.
Rows and Columns	Type the number of characters for the printer; only available with port 5 and 10. Rows can be 0-255, columns can be 0-255.
SOE Block Name	The SOE Block Name

3 Click **OK** to save.

Lesson 56: Configuring Tricon HIM Ports

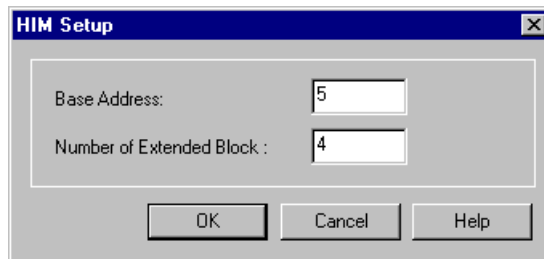
Configuring Tricon HIM Ports

The Tricon HIM supports a connection between a Tricon controller and a Honeywell control system.

In this lesson, you will learn how to configure a Tricon HIM port.

Procedure

- 1 Expand the **Controller** tree, double-click **Configuration**, and expand **Hardware Allocation**. Double-click the **HIM** slot, and then click **Setup**.



- 2 Specify these properties on the HIM Setup screen.

Property	Action
Base Address	Enter a number between 5 and 31 to specify the block address for the HIM based on the Data Hiway memory map. The default is 5.
Number of Extended Block	Enter a number which identifies a pair of HIM blocks consisting of a base block and a secondary block (which is offset from the six-bit block address by 32). The default is 4.

- 3 Click **OK** to save.

Lesson 57: Configuring Tricon NCM Ports

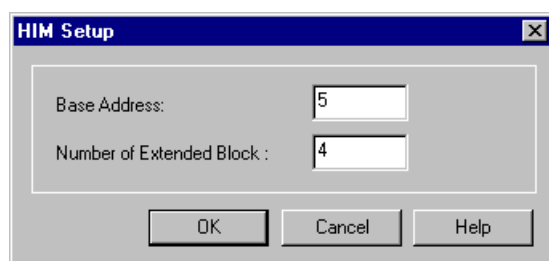
The Tricon NCM supports these connections:

- On Net1, a network connection to other Triconex controllers using Peer-to-Peer protocol
- On Net1, a network connection to an SOE Recorder PC
- On Net1, time synchronization with the Tricon master node
- On Net2, a network connection to a **TriStation 1131** PC or other external devices
- Serial connection between a Tricon NCMG module and a Global Positioning System (GPS) interface

You can install a primary and redundant module in one logical slot.

Procedure

- 1 Expand the **Controller** tree, double-click **Configuration**, and expand **Hardware Allocation**. Double-click the **NCM** slot, and then click **Setup**.



- 2 Specify these properties on the **NCM Setup** screen.

Property	Action
Installed/Not Installed	Click to indicate which slots have modules installed.
Privilege	Select Read or Read/Write.
IP Address	Enter the IP address for the NCM.
IP Subnet Mask	Use as needed.
Global Positioning System Installed	Check to synchronize time with a GPS.
Time Synchronization	Check to synchronize time with the Tricon master node.

- 3 Click **OK** to save.

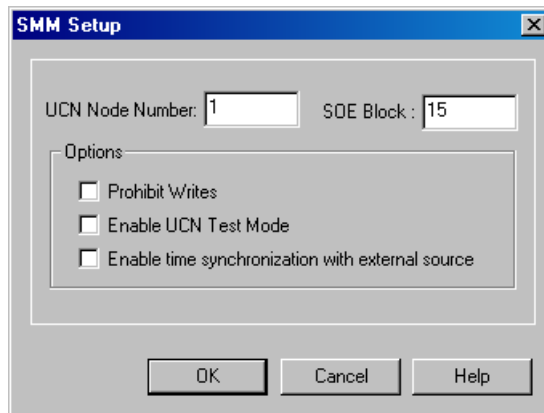
Lesson 58: Configuring Tricon SMM Ports

The Tricon SMM supports a connection between a Tricon controller and a Honeywell Universal Control Network (UCN).

In this lesson, you will learn how to configure a Tricon SMM port.

Procedure

- 1 Expand the **Controller** tree, double-click **Configuration**, and expand **Hardware Allocation**. Double-click the **SMM** slot, and then click **Setup**.



- 2 Specify these properties on the SMM Setup screen.

Property	Action
UCN Node Number	Specify the UCN node number.
SOE Block	Enter 15 or 16, which are Modified External blocks reserved for the SMM.
Prohibit Writes	Check to prevent Honeywell workstations from writing to memory and output points. The default is unchecked.
Enable UCN Test Mode	Should only be checked by Honeywell factory test personnel. The default is unchecked.
Time Synchronization	Check to enable time synchronization with the UCN. The default is unchecked.

- 3 Click **OK** to save.

Tricon Time Synchronization

The Time Synchronization communications protocol enables a network of Triconex controllers to synchronize time with each other or with external devices, such as a DCS or the Global Positioning System (Tricon only).

This table summarizes the ways Tricon controller time can be synchronized to an external device or to the Trident master node in a Peer-to-Peer network.

Tricon Time Synchronization

Module	Time Synchronization Options
ACM	<ul style="list-style-type: none"> • Synchronized by a Foxboro DCS. • Synchronized to the Tricon master node.
ACM or NCM	<ul style="list-style-type: none"> • Synchronized by an OPC client. For more information, see the <i>Tricon Communication Guide</i>. • Synchronized by writing aliased data to the TIMESET or TIMEADJ function blocks in a TriStation 1131 application. For assistance with the specialized programming that is required, contact Triconex Technical support. • Synchronized to the Tricon master node.
NCMG	<ul style="list-style-type: none"> • Synchronized by the GPS (Global Positioning System). • Synchronized to the Tricon master node.
SMM	<ul style="list-style-type: none"> • Synchronized by the Honeywell Universal Control Network (UCM).

Lesson 59: Using Tricon ACM to Synchronize Time

In this lesson you will learn how to use a Tricon ACM to enable time synchronization based on:

- The Foxboro I/A Series nodebus system time
- The Tricon master node in a Peer-to-Peer network

Procedure

- 1 Expand the **Configuration** tree, double-click **Configuration**, and expand **Hardware Allocation**. Double-click the **ACM** slot, and then click **Setup**.

- 2 Specify these properties on the **ACM Setup** screen.

Property	Action
Privilege	For synchronization on Net2, specify Read/Write. The default is Read.
Time Synchronization	Click On Net, to enable time synchronization with a Foxboro I/A Series DCS. Click On Net2, to enable time synchronization with the Triconex master node (controller). Click None, to not use time synchronization.

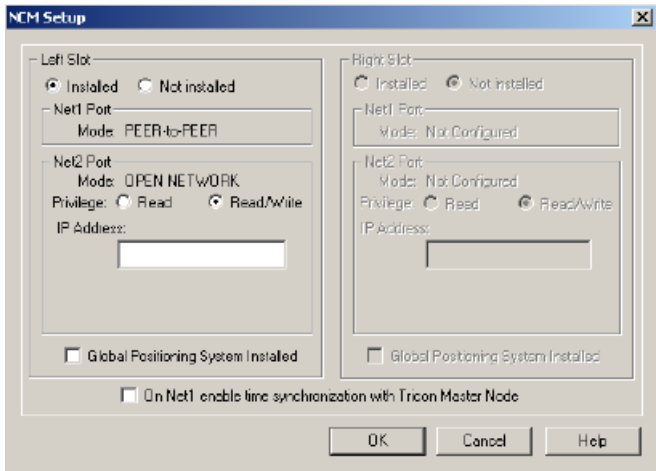
- 3 Click **OK** to save.

Lesson 60: Using a Tricon Network Communication Module to Synchronize Time

In this lesson, you will learn how to use a Tricon NCM to enable time synchronization with the Tricon master node in a Peer-to-Peer network.

Procedure

- 1 Expand the **Configuration** tree, double-click **Configuration**, and expand **Hardware Allocation**. Double-click the **NCM** slot, and then click **Setup**.



- 2 Specify this property on the **NCM Setup** screen.

Time Synchronization	Action
Time Synchronization	Click On Net1, to enable time synchronization.

- 3 Click **OK** to save.

Lesson 61: Using a Tricon NCMG to Synchronize Time

In this lesson, you will learn how to use a Tricon NCMG to enable Time synchronization through the Global Positioning System (GPS) by using the Trimble Acutime 2000 Synchronization Kit. If the Tricon NCMG is in a Peer-to-Peer network, it can be used as the master node for time synchronization of the controllers on the network.

For information on installing the kit, see the *Tricon Communication Guide*.

CAUTION: To ensure the accuracy of GPS time adjustments, the Tricon clock must be set to within 10 minutes of the correct local time.

Procedure

- 1 Expand the **Configuration** tree, double-click **Configuration**, and expand **Hardware Allocation**. Double-click the **NCMG** slot, and then click **Setup**.

NCM Setup

Left Slot

☒ Installed ☐ Not installed

Net1 Port
Mode: PEER-to-PEER

Net2 Port
Mode: OPEN NETWORK
Privilege: ☐ Read ☒ Read/Write
IP Address:
IP Subnet Mask (hex): 00000000

☒ Global Positioning System Installed

Right Slot

☐ Installed ☒ Not installed

Net1 Port
Mode: Not Configured

Net2 Port
Mode: Not Configured
Privilege: ☐ Read ☒ Read/Write
IP Address:
IP Subnet Mask (hex): 00000000

☐ Global Positioning System Installed

☒ On Net1 enable time synchronization with Tricon Master Node

OK Cancel Help

- 2 Specify these properties on the **NCM Setup** screen

Property	Action
Global Positioning System Installed	Check to have time synchronization done through the GPS. If checked, other controllers can also be synchronize to the Tricon master controller.
Time Synchronization	For a network of Triconex controllers, click ON Net1..., to enable time synchronization with the Tricon master node (controller).

- 3 Click **OK** to save.

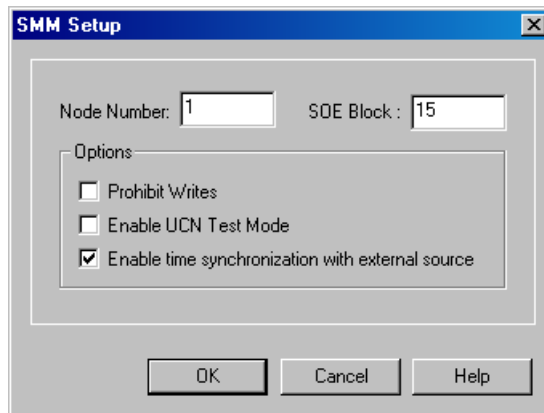
Lesson 62: Using a Tricon SMM to Synchronize Time

In this lesson, you will learn how to specify a Tricon SMM module to enable time synchronize time based on the Honeywell Universal Control Network (UCM).

For more information, see the *SMM User's Guide*.

Procedure

- 1 Expand the **Configuration** tree, double-click **Configuration**, and expand **Hardware Allocation**. Double-click the **SMM** slot, and then click **Setup**.



- 2 Specify this property on the **NCM Setup** screen.

Time Synchronization	Action
Time Synchronization	Check Enable time synchronization with external source.

- 3 Click **OK** to save, and then save the project.

Trident Controller Configuration

Overview

Physically, a basic Trident system consists of:

- Field-replaceable modules
- Baseplates on which modules are mounted
- Field wiring connections
- Programmer's workstation

Modules

Modules are field-replaceable units consisting of an electronic assembly in a protective metal housing. Each module is fully enclosed to ensure that no components or circuits are exposed—even when a module is removed from the baseplate. Offset baseplate connectors make it impossible to plug a module in upside down, and keys on that prevent the insertion of modules into incorrect slots.

Assembly

An assembly consists of a module and a baseplate. The three types of assemblies available are:

- Main Processor
- Communication
- I/O

Field Wiring Connections

Terminations for field wiring are integral to each I/O module baseplate.

Programmer's Workstation

TriStation 1131 supports Windows NT and Windows 2000.

System Configuration

After creating an application, you must configure the controller by defining the system configuration. A system configuration is created by:

- Setting operating parameters
- Allocating memory and hardware
- Configuring communication external devices
- Set up time synchronization

The system configuration is used to graphically and textually define I/O modules and memory points for each slot in the Trident controller, and assign the global variables associated with each point.

Operating Parameters

Operating parameters are settings that restrict or allow connection access and write access to a Trident controller.

These include:

- Setting password
- Restart on Power Up
- Disabling Remote Changes to Outputs
- Allowing Disabling of Points
- Use Local Time

These types of read and write access are possible:

- Input, output, and memory points can be read by any external device that can communicate with a Trident controller.
- Write access to input points is not allowed from any external device.
- Write access to a output or memory point is allowed or restricted based on the system, communication, application, and point settings.

This table describes write access to Trident points from external devices.

Trident Write Access

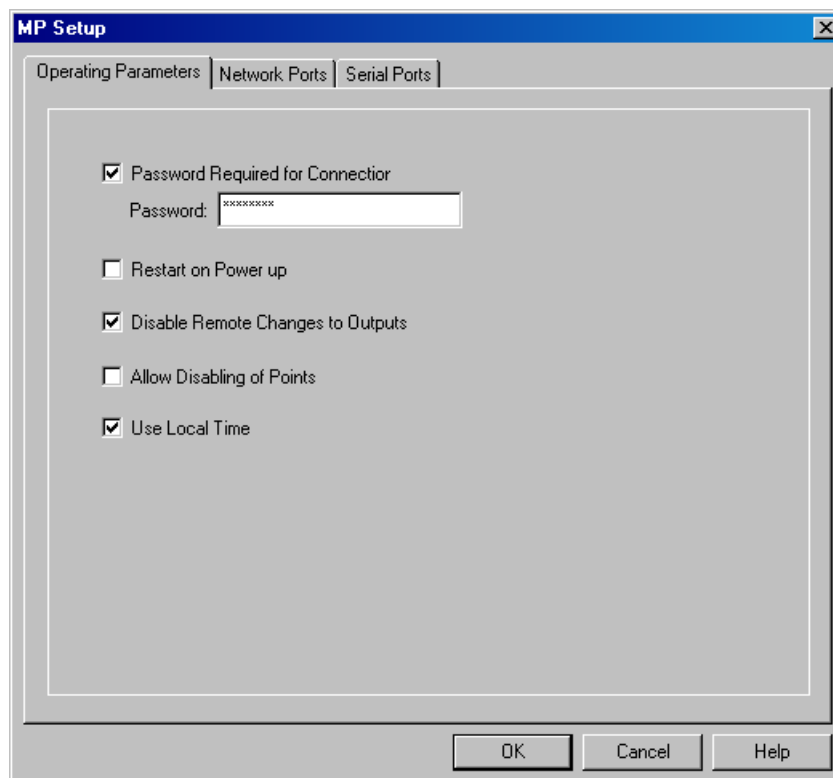
Property or Feature	Description
Disable Remote Changes to Outputs	A system setting on the MP Operating tab that determines write access to output points. When checked, external devices cannot write to outpoints.
SYS_SET_REMOTE_WRT_ENBL	A Trident function block that programmatically allows write access to output or memory read/write aliased points when used in an application.
Privilege	A Trident CM module setting that determines whether network devices using DDE, OPC, or TSAA communication have write access to output points and read/write aliased memory points. For Trident CM, the default is Read/Write. This setting does not affect Modbus access.
Point Assignment	A tagname setting that determines whether the output and memory point is assigned a Read or Read/Write alias number. For output points, all alias numbers are Read/Write. For memory points, alias numbers can be Read or Read/Write.

Lesson 63: Setting Trident Operating Parameters

In this lesson, you will learn how to specify operating parameters, which include setting to restrict access to the controller from a **TriStation 1131** PC and from remote devices.

Procedure

- 1 Expand the **Controller** tree, double-click **Configuration**, and expand **Hardware Allocation**. Double-click the **MP** module, and then click **Setup**.



- 2 Specify these properties on the **Operating Parameters** tab.

Property	Action
Password Required for Connection	Check to require a password to be used to connect to the controller. If checked, enter the password. The default is unchecked.
Restart on Power Up	Check to have the controller restart when power is returned after a failure or shutdown. The default is unchecked
Disable Remote Changes to Outputs	Check to restrict external devices, such as a DCS, to write to output tagnames in the TriStation 1131 application. The default is checked.
Allow Disabling of Points	Check to allow points to be disabled from TriStation 1131 . The default is checked.
Use Local Time	Check to use local time.

- 3 Click **OK**.

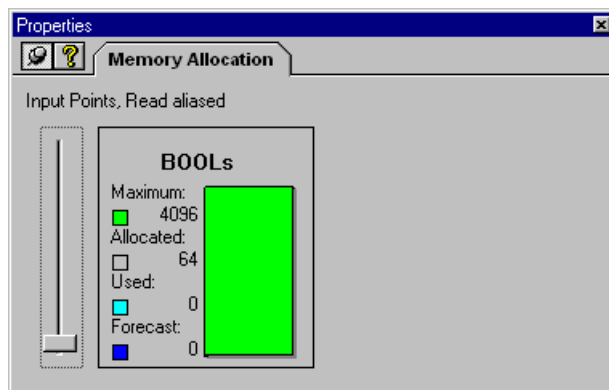
Lesson 64: Allocating Memory for Points

Initially, the amount of memory for input and output points is determined by the number and type of I/O modules configured in the **TriStation 1131** project. The amount of memory for memory points is set when a **TriStation 1131** project is created. You can change these allocations at any time before building and downloading the application.

In this lesson, you will learn how to change the memory allocation for points.

Procedure

- 1 Expand the **Controller** tree, double-click **Configuration**, and then expand **Memory Configuration**.
- 2 Double-click the type of point you want to change.



- 3 Move the slider up or down to change the memory allocated for the selected point.
- 4 Repeat for all points to be changed.

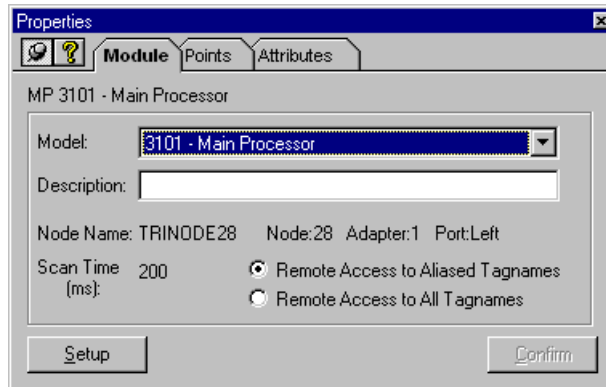
Lesson 65: Specifying Trident MP Module Properties

In this lesson, you will learn how to specify properties on the Trident MP Module tab. The properties that are displayed vary depending on the module.

Note: You cannot specify point properties for a Trident MP.

Procedure

- 1 On the **Controller** tree, expand **Hardware Allocation**, and double-click any slot.



- 2 Specify these properties on the **Module** tab.

Property	Action
Model	Select the model used in the physical system.
Description	Enter a description, if desired.
Node Name	Display the node name and number.
Scan Time	Displays the scan time, if it has been specified.
Remote Access to Aliased Tagnames	Click to have remote access only to aliased tagnames.
Remote Access to All Tagnames	Click to have remote access to all aliased tagnames.
Setup	Click the Setup button to specify properties for operating parameters, network ports, and serial ports.

- 3 Click **Confirm** to save settings.

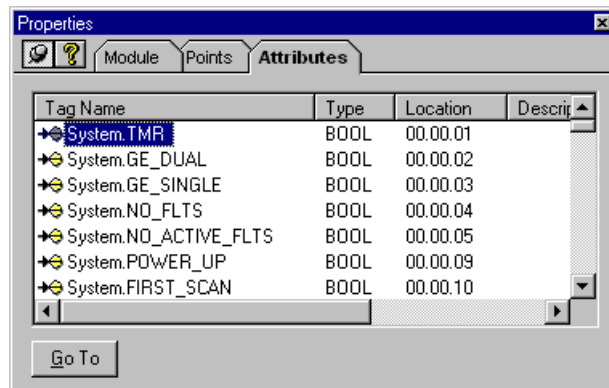
Lesson 66: Displaying Trident MP Attribute Properties

In this lesson, you will learn how to specify MP attribute properties (status and control attributes).

Note: You cannot specify point properties for a Trident MP.

Procedure

- 1 On the **Controller** tree, expand **Hardware Allocation**, and double-click the **MP** slot. Click the **Attributes** tab.



- 2 These properties are displayed on the **Attributes** tab.

Property	Action
Tagname	Name of the status or control attribute.
Data Type	Data type.
Location	Memory location for the attribute.
Description	Description of the attribute.

- 3 Click **Confirm** to save settings.

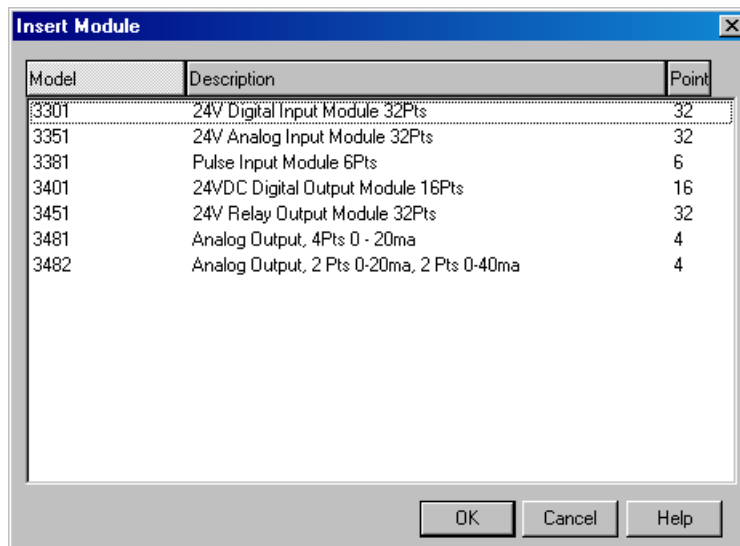
Lesson 67: Inserting Trident Modules

If a Trident Communication Module is added to a configuration after the application has been downloaded to the controller, you must use the Download All command to download the change.

In this lesson, you will learn how to insert Trident modules on a configuration.

Procedure

- 1 Expand the **Controller** tree, double-click **Configuration**, and expand **Hardware Allocation**. Double-click the **MP** slot.
- 2 Click the **Insert** button.



- 3 Click to select the type of module to add to the configuration, and then click on the configuration pane (to the right of the tree).

Lesson 68: Removing Trident Modules

If a Trident Communication Module is removed from a configuration after the application has been downloaded to the controller, you must use the Download All command to download the change.

In this lesson, you will learn how to remove modules from a Trident configuration.

Procedure

- 1 Expand the **Controller** tree, double-click **Configuration**, and expand **Hardware Allocation**. Click the icon that represents the module, and then press the **Delete** key.

Lesson 69: Configuring a Trident PI Module

The Trident Pulse Input (PI) module is used for speed and RPM inputs. Each point can be independently configured.

In this lesson, you will learn how to configure a Trident PI Module.

Procedure

- 1 Expand the **Controller** tree, double-click **Configuration**, expand **Hardware Allocation**. Double-click the **PI** slot, and then click **Setup**.
- 2 Click a tagname to be configured, click the properties to be set, and then click and configure the next tagname.
- 3 Specify these properties for each point on the **Pulse Input Setup** screen.

Property	Action
Field Power	Check Field Power Present if the installed system has field power connected to the Pulse Input Baseplate. The default is Field Power Absent.
Field Maximum Power	Enter the maximum range for field power in volts The default is 33 volts.
Field Minimum Power	Enter the minimum range for filed power in volts. The default is 0 volts.
Rate Type	Click Speed or RPM to specify the type of rate applied to pulse input signals.
Triggering Mode	Click either Rising or Falling Edge depending on the installation.
Number of Gear Teeth	Enter the number of gear teeth that are counted in each revolution. The Point Options property must be specified as RPM to enable this property.
Scale Factor	Specifies how to convert pulse input signals into engineering units. For: <ul style="list-style-type: none"> • Pulses per second, set to 0.016667. • Pulses per minute, set to 1.0 (default) • Pulses per hour, set to 60.0

- 4 After all the tagnames are specified, click **OK** to save.

Configuring Trident Communication with External Devices

To communicate with external devices, Trident modules must be configured to connect with a specific type of controller (Network or Serial) and to support specific types of port connections.

The connection setting, either Network or Serial Connection, is an initial setting and can be changed when you use the Connect command.

Lesson 70: Specifying the Trident Default Connection

In this lesson, you will learn how to specify the Trident default connection.

Procedure

- 1 Expand the **Configuration** tree, double-click **Configuration**, and then double-click **TriStation Communication**.

The screenshot shows the 'Communication' configuration window. It has a title bar with a yellow icon and the text 'Communication'. The window is divided into several sections:

- Network TCP/IP Addresses:** Contains two text boxes: 'Node Name' with the value 'TRINODE28' and 'Number' with the value '28'.
- Communication Module:** Contains a text box labeled 'TCP/IP Address' which is currently empty.
- Main Processor:** Contains a text box labeled 'Network Adapter' with the value '1'.
- Port:** Below the network adapter box, there are three radio buttons: 'Left' (selected), 'Middle', and 'Right'.
- Default Connection:** A section with two radio buttons: 'Main Processor' and 'Communication Module' (selected).
- Debug Message Options:** A section with five checkboxes, all of which are unchecked: 'Messages Sent & Received', 'Message Detail', 'State Changes', 'Errors', and 'Information'.

- 2 Specify these properties on the **TriStation Communication** screen.

Property	Action
Select Connections	To connect through the CM, click Network Connection. To connect through the MP, click Main Processor Connection.
Node Number	Enter the number for the controller.
Node Name	Enter the name for the controller.
IP Address	For a CM, enter the IP address.
Main Processor Connection Setup	For an MP connection, click the MP that is connected to the TriStation 1131 PC. The default is left.
NIC Index	Enter the number of the network interface card index in the TriStation 1131 PC. Only needed for MP connection.
Default connection	If only one Selected Connection is checked, the default connection is set based on it. If both network and serial connections are set, you must specify the default connection setting.

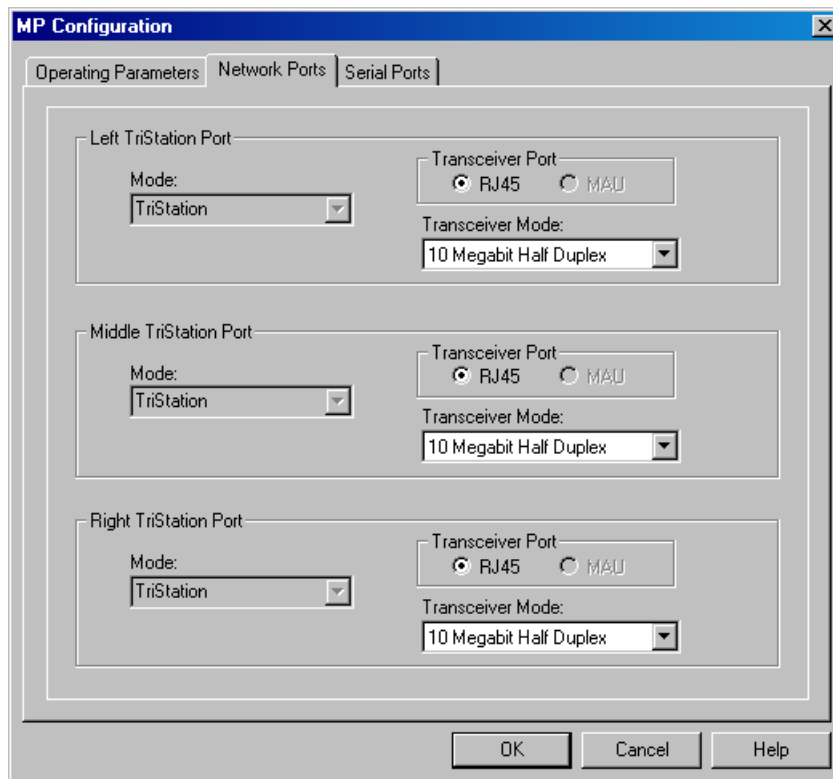
- 3 To complete the connection, you must configure a Trident MP or CM module.

Lesson 71: Configuring Trident MP Network Ports

In this lesson, you will learn how to configure network ports on a Trident MP, which support network connections to a **TriStation 1131** PC.

Procedure

- 1 Expand the **Controller** tree, double-click **Configuration**, and expand **Hardware Allocation**. Double-click the **MP** slot, click **Setup**, and then click the **Network Ports** tab.



- 2 Specify these properties on the **Network Ports** tab.

Property	Action
Mode	Set to TriStation. Cannot be changed.
Transceiver Port	Set to RJ-45. Cannot be changed.
Transceiver Mode	Select half or full duplex depending on the installation.

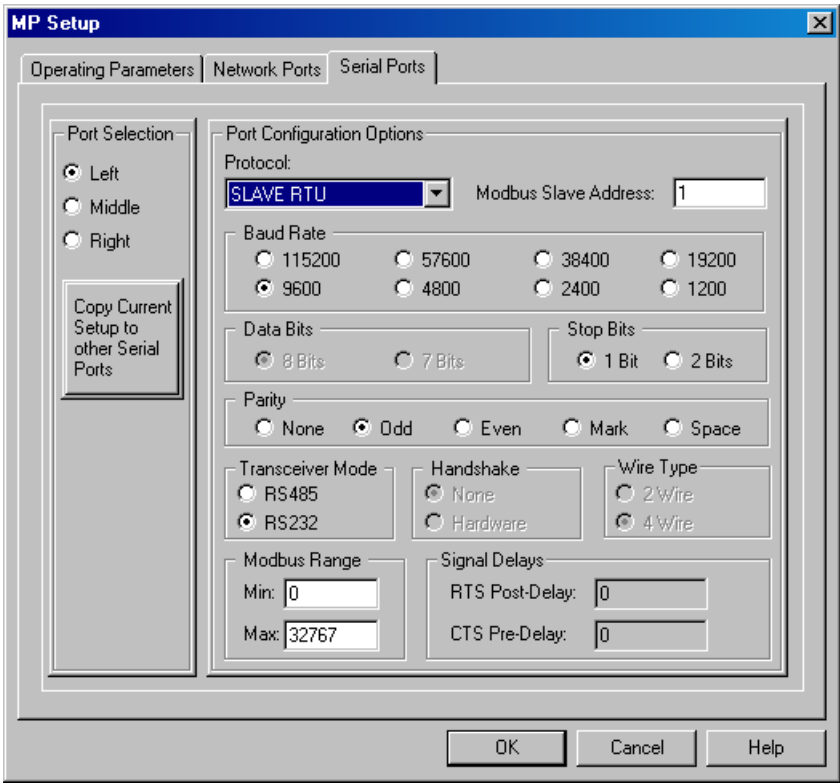
- 3 Click **OK** to save.

Lesson 72: Configuring Trident MP Serial Ports

This lesson explains how to configure serial ports on a Trident MP, which support serial connections using Modbus slave RTU protocol.

Procedure

- 1 Expand the **Controller** tree, double-click **Configuration**, and expand **Hardware Allocation**. Double-click the **MP** slot, click **Setup**, and then click the **Serial Ports** tab.



- 2 Specify these properties on the **Serial Ports** tab.

Property	Action
Port Selection	Click the port to be configured.
Copy Current Setup to Other Serial Ports Command	Click to have the settings for the selected port copied to the other ports.
Protocol	Only Modbus slave RTU is available.
Modbus Slave Address	Enter the slave address of the serial port on the MP Baseplate.
Baud Rate	Click the rate used in the installation.
Data Bits	Set at 8 bits; cannot be changed.
Stop Bits	Click 1 Bit or 2 Bits.
Parity	Click the parity option.

Property	Action
Transceiver Mode	Click RS-232 or RS-485, depending on the physical connection.
Handshake	Set to none; cannot be changed.
Wire Type	Set to 4 wire; cannot be changed.
Modbus Alias Range	Enter a minimum value (0 is default) and maximum value (32,767 is maximum). Only available with Modbus.
Signal Delays	Set to 0; cannot be changed.

- 3 Click **OK** to save.

Lesson 73: Configuring Trident CM Network Ports

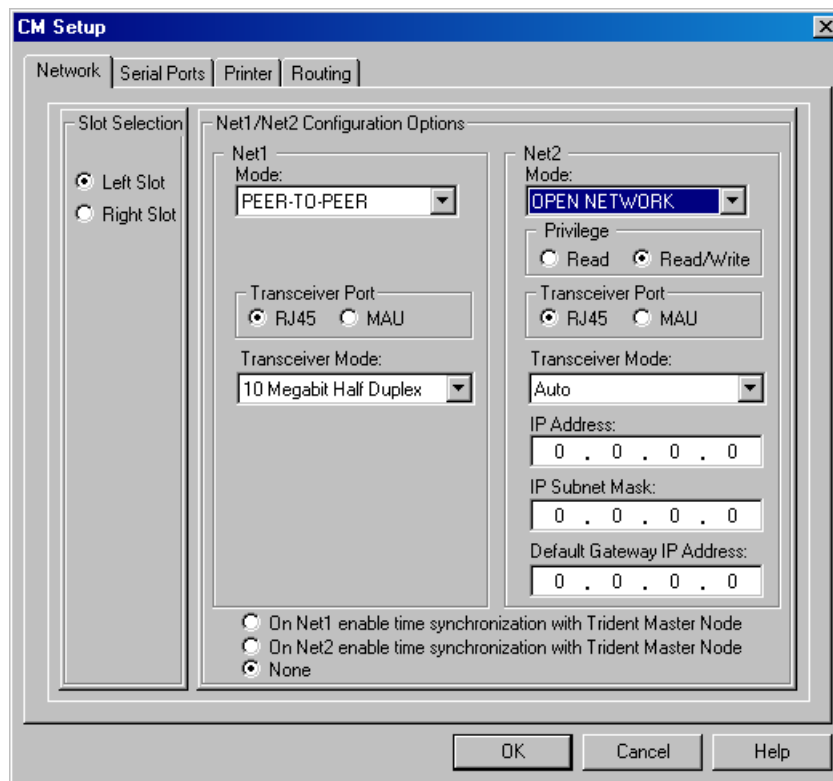
The Trident CM Network port supports these connections:

- To a **TriStation 1131** PC
- To a Peer-to-Peer network of Triconex controllers
- To an external device or network

In this lesson, you will learn how to configure network ports on a Trident CM.

Procedure

- 1 Expand the **Controller** tree, double-click **Configuration**, and expand **Hardware Allocation**. Double-click the **CM** slot, click **Setup**, and then click the **Network** tab.



- 2 Specify these properties on the **Network** tab.

Property	Action
Slot Selection	Click the slot be configured.
Mode	Select either Open Network or Peer-to-Peer for either Net1 or Net2. Cannot use the same mode on both ports.
Privilege	Click Read Only to restrict access from external devices. Not available with Peer-to-Peer. The default is Read/Write.
Transceiver Port	Click the type of port used.
Transceiver Mode	Click the mode used. Not available with Peer-to-Peer.
IP Address	Enter the IP address of the controller. Not available with Peer-to-Peer.
IP Subnet Mask	Enter information, if needed. Not available with Peer-to-Peer.
Default Gateway IP Address	Enter information, if needed. Not available with Peer-to-Peer.
Time Synchronization	Click to enable time synchronization on Net1 or Net2.

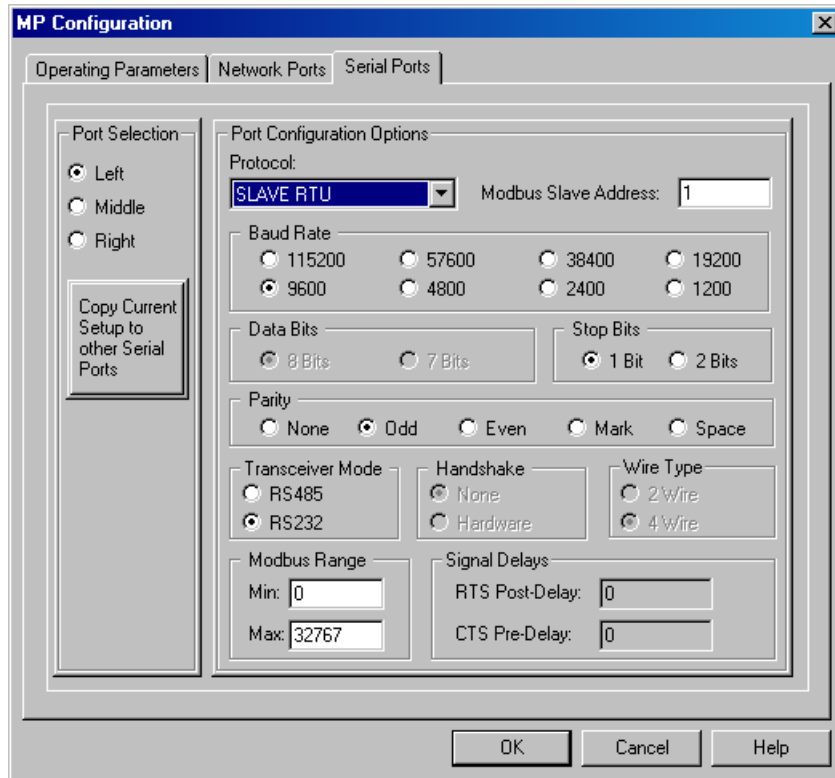
- 3 Click **OK** to save.

Lesson 74: Configuring Trident CM Serial Ports

The Trident CM supports connections using Modbus slave, master, and master/slave protocols. In this lesson, you will learn how to configure serial ports on a Trident CM.

Procedure

- 1 Expand the **Controller** tree, double-click **Configuration**, and expand **Hardware Allocation**. Double-click the **CM**, click **Setup**, and then click the **Serial Ports** tab.



- 2 Specify these properties on the **Serial Ports** tab.

Property	Action
Port Selection	Click the port to be configured.
Protocol	Select the port to use.
Modbus Slave Address	Enter the slave address of the serial port on the MP Baseplate. Not used with Master protocol.
Baud Rate	Click the rate used in the installation.
Data Bits	Click 7 or 8 bits' only available with slave ASCII protocol. Set to 8 bits for all other protocols.
Stop Bits	Click 1 bit or 2 Bits.
Parity	Click the parity option.
Transceiver Mode	Click RS-232 or RS-485, depending on the physical connection.
Handshake	Click Hardware to use signal delays to determine if the connection is valid.
Wire Type	Click 2 or 4 wire, depending on the installation.
Modbus Alias Range	Enter a minimum value (0 is default) and maximum value (32,767 is a maximum). Only available with Modbus.
Signal Delays	Enter the number of milliseconds to adjust timing of the data transmission.

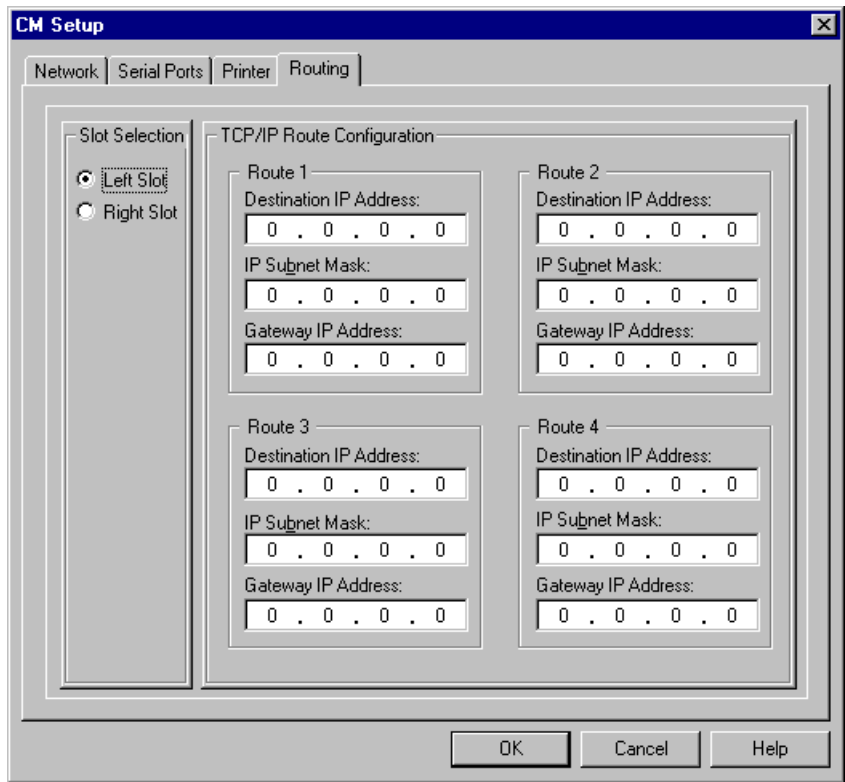
- 3 Click **OK** to save.

Lesson 75: Configuring Trident CM Routing

This lesson explains how to configure routing information for network communication on a Trident CM. This procedure is optional depending on your network configuration. For more information, see your network administrator.

Procedure

- 1 Expand the **Controller** tree, double-click **Configuration**, and expand **Hardware Allocation**. Click **CM**, double-click the **CM slot**, click **Setup**, and then click the **Serial Ports** tab.



- 2 Specify these properties on the **Routing** tab.

Property	Action
Slot Selection	Select the slot to configure.
IP Subnet Mask	Enter the address for the subnet mask.
Gateway IP Address	Enter the address for the gateway.
Destination IP Address	Enter the destination address.

- 3 Click **OK** to save.

Trident Time Synchronization

The Time Synchronization communication protocol enables a network of Triconex controllers to synchronize time with each other or with external devices.

This table summarizes the ways Trident controller time can be synchronized to an external device or to the Trident master node in a Peer-to-Peer network.

Trident Time Synchronization

Module	Time Synchronization Feature
CM	<ul style="list-style-type: none"> • Synchronized by an OPC client. For more information, see the Tricon Communication Guide. • Synchronized by writing aliased data to the TIMESET or TIMEADJ function blocks in a TriStation 1131 application. For assistance with the specialized programming that is required, contact Technical Support. • Synchronized by to the Trident master node.
MP	Synchronized by writing aliased data to the TIMESET or TIMEADJ function blocks in a TriStation 1131 application. For assistance with the specialized programming that is required, contact Technical Support.

Lesson 76: Using a Trident CM to Synchronize Time

The Trident CM is used to synchronize Trident controllers to the master node in a Peer-to-Peer network.

In this lesson, you will learn how to specify time synchronization.

Procedure

- 1 Expand the **Configuration** tree, double-click **Configuration**, and expand **Hardware Allocation**. Double-click the **CM** slot, click **Setup**, and then click the **Network** tab.
- 2 Specify this property on the **CM Setup**.

Property	Action
Time Synchronization	<ul style="list-style-type: none">• Click On Net1, to enable time synchronization with the Trident master controller.• Click On Net2, to enable time synchronization with the Trident master controller.• Click None, to not use time synchronization.

- 3 Click **OK** to save.

Lesson 77: Specifying an Alias Number for a Trident Attribute

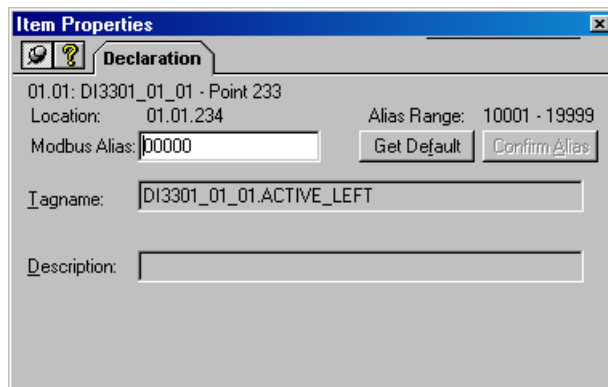
Trident system and module attributes are variables that can be used to monitor status and control Trident operations. You can add an alias number so the variable can be read or written to.

An alias is a five-digit number that can be used by an external device to read or write to an input, output, or memory point in a controller. Alias is a convention of the Modbus protocol.

In this lesson, you will learn how to specify an alias number for a system or module attribute, which allows you to read or write to the attribute.

Procedure

- 1 Expand the **Controller** tree and double-click **Configuration**. Expand **Status Attributes** and double-click the attribute to be assigned an alias.



- 2 Click **Get Default**, or enter a number and click **Confirm Alias**.

The alias number is assigned to the attribute.

TriStation 1131 Communication

TriStation 1131 Communication	278
TCM Connection	278
Tricon EICM Connection	279
Tricon ACM or NCM Connection	284
Trident MP Connection	296
Trident CM Connection	304
Tricon Printing	313
Trident Printing	317
IP Addresses	323

TriStation 1131 Communication

Communication setup can be done anytime before implementation.

This checklist includes the items that can be or should be done to set up communication between a **TriStation 1131** PC and a Triconex controller.

	Description
<input type="checkbox"/>	Connect to a TCM.
<input type="checkbox"/>	Connect to a Tricon EICM.
<input type="checkbox"/>	Connect to a Tricon ACM or NCM.
<input type="checkbox"/>	Connect to a Trident MP.
<input type="checkbox"/>	Connect to a Trident CM.
<input type="checkbox"/>	Set up printing.

TCM Connection

The Tricon Communication Module (TCM) supports these connections:

- Modbus Master using RTU protocol; Slave using ASCII or RTU protocols
- Peer-to-Peer on Net1 using DLC; on Net1 or Net2 using UDP/IP
- Printing on Net1 or Net2
- Time Synchronization on Net1 using DLC, on Net1 or Net2 using UDP/IP, on Port 1 using GPS, or SNTP
- TriStation serial connection on port 4; Ethernet connection on Net1 or Net2
- TSAA client/server connection on Net1 or Net2

Tricon EICM Connection

This section explains how to use a Tricon EICM for a direct (point-to-point) connection to a **TriStation 1131** PC. If you use port 4 for the connection, you do not need to change the switch settings on the module. If you use a different port or the switch settings have been changed from the default settings, you must identify the **TriStation 1131** port using a switch block on the side of the EICM. Then you must connect the **TriStation 1131** port to the **TriStation 1131** PC using a serial cable, and configure the connection in the project.

Lesson 78: Setting Tricon EICM Switches for the TriStation 1131 Port

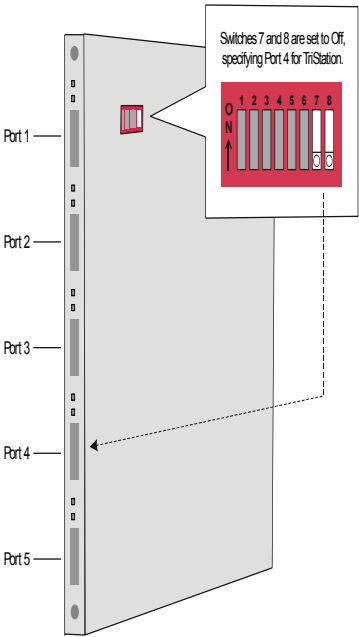
This lesson explains how to set the **TriStation 1131** port using a switch block on the EICM. Triconex strongly recommends using the default setting for **TriStation 1131**, which is port 4. If port 4 fails, you can set up port 1, 2, or 3 as the **TriStation 1131** connection.

The default (factory-configured) setting for port 4 has switches 7 and 8 set to Off.

Procedure

- 1 Remove the **EICM** from its slot in the Tricon chassis.

This figure shows the default settings for port 4.



- 2 Use this table to reset **switches 7 and 8** for the desired port.

TriStation 1131 on Port	Switch 7	Switch 8
1	Off	On
2	On	Off
3	On	On

- 3 Replace the **EICM** in its slot. You can now connect the serial cable from the EICM port to the **TriStation 1131** PC.

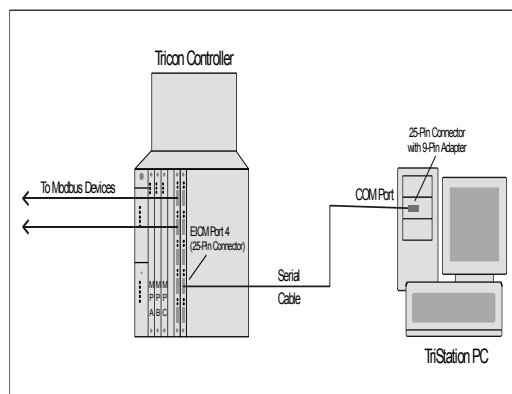
Lesson 79: Connecting a Tricon EICM to a TriStation 1131 PC

This lesson explains how to connect a Tricon EICM serial port to a **TriStation 1131** PC.

Triconex provides a serial cable that has a 25-pin connector on each end. If the COM port on the PC has a 9-pin connector, you can use a Triconex 25-pin to 9-pin adapter. If you need other parts, you can purchase them from another manufacturer.

Procedure

- 1 Connect one end of the serial cable to a serial port on the **EICM**. This is typically port 4.
- 2 Connect the 9-pin adapter to a **COM** port on the **TriStation 1131** PC. The COM port is typically numbered COM1, COM2, COM3, or COM4.
- 3 Connect the other end of the serial cable to the 9-pin adapter on the **COM** port.



Lesson 80: Configuring a Tricon EICM Connection

This lesson explains how to configure a Tricon EICM serial port that is connected to a **TriStation 1131** PC.

Procedure

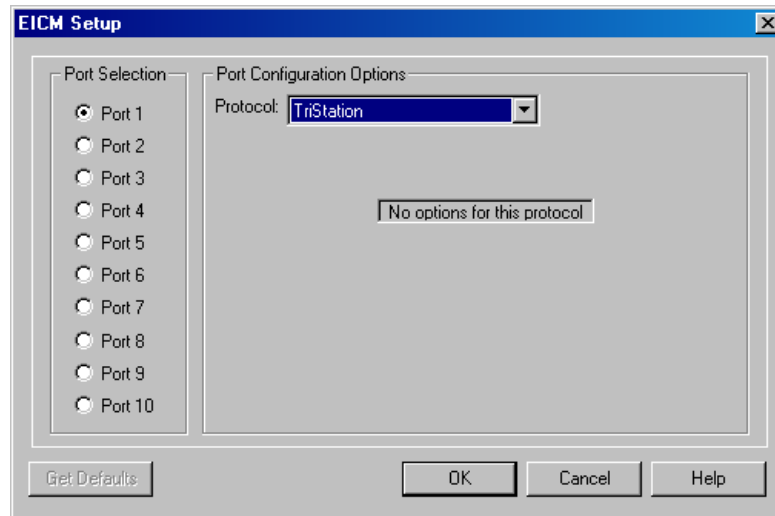
- 1 Expand the **Controller** tree, and double-click **Configuration**. On the **Configuration** tree, click **TriStation Communication**.

The screenshot shows the 'TriStation Communication' configuration window. It has a title bar with a yellow icon and the text 'TriStation Communication'. The window is divided into several sections:

- Select Connections:** A text box explains that TriStation can communicate over a network or serial port. A network connection requires a Network Communication Module, and a serial connection requires an EICM module. Below this, there are two checkboxes: 'Network Connection' (unchecked) and 'Serial Connection' (checked).
- Network Connection Setup:** This section contains three input fields: 'Node Number' (set to 6), 'IP Address' (set to 206.32.216.25), and 'Node Name' (set to TRINODE06). Below these fields, there are three lines of explanatory text: 'Node Number: The number specified by the switches on the communication module.', 'IP Address: The internet protocol address of the module (e.g. 192.168.1.1).', and 'Node Name: Any alphanumeric name up to 20 characters.'
- TriStation PC:** This section contains a 'Serial Port' dropdown menu set to 'COM1'. To the right of the dropdown, there is a text box that says 'The port on the PC that will be connected to the controller.'
- Default Connection:** This section contains a text box that says 'When you connect to the controller, which connection would you like to use as the default?'. Below this text box, there are two radio buttons: 'Network Connection' (unselected) and 'Serial Connection' (selected).

- 2 Check the **Serial Connection** check box.
- 3 For **Serial Port**, select the **COM** port on the **TriStation 1131** PC to which the serial cable is connected.

- 4 Expand the **Controller** tree, and double-click **Configuration**. Expand **Hardware Allocation**, double-click **EICM**, and then click **Setup**.



- 5 Specify these properties on the **EICM Setup** screen.

Property	Action
Port	Click the port that the TriStation 1131 PC is attached to. The default TriStation 1131 connection is port 4.
Protocol	Select TriStation.

- 6 Click **OK** to save.

Tricon ACM or NCM Connection

This section explains how to use a Tricon ACM or NCM for an Ethernet connection to a **TriStation 1131** PC. This can be a direct connection from the ACM or NCM to the PC, or a connection through a media converter.

To set up the connection, you must install a network interface card and TCP/IP protocol on the PC, set the node number of the controller, connect the PC to an Ethernet port on the ACM or NCM, and configure the connection in the **TriStation 1131** project. This section includes procedures for each of these tasks.

Lesson 81: Installing a NIC Card in a TriStation 1131 PC

This lesson explains how to install a network interface card (NIC) in a **TriStation 1131** PC to be connected to a Tricon ACM or NCM.

Procedure

- 1 Install the network interface card by following the manufacturer's instructions. *Do not change the factory default settings on the NIC card.*
- 2 If the network interface card has a BNC connector, you can connect it directly connect it to a Net2 port.

If the network interface card does not have a BNC connector, you must connect it to a media converter that is connected to a Net2 port.

- 3 Run the diagnostics provided with the network interface card according to the manufacturer's instructions.

Lesson 82: Installing TCP/IP Protocol on a TriStation 1131 PC

These lessons explain how to install TCP/IP protocol on a **TriStation 1131** PC.

Procedure (Windows NT)

- 1 On the **Start** menu, click **Settings**, then click **Control Panel**.
- 2 In the **Control Panel** window, double-click the **Network** icon.
- 3 In the **Network** dialog box, click the **Protocols** tab.
If TCP/IP is listed in Network Protocols, it means the protocol is installed and you are finished with this procedure.
- 4 For **Network Protocols**, double-click **Add**.
- 5 On the **Select Network Protocols** screen, click **TCP/IP Protocol** from the list.
- 6 If you want Windows NT to copy files from a specified location, such as a network drive, click **OK**.
- 7 If the files are on a disk, insert the disk and click **Have Disk**.

Procedure (Windows 2000)

- 1 On the **Start** menu, click **Settings**, then click **Network and Dial-up Connections**.
- 2 Right-click the network connection where you want to install TCP/IP, then click **Properties**.
- 3 On the **Networking** tab, if DLC is checked on the list of installed components, it means the protocol is installed and you are finished with this procedure.
- 4 Click **Install**, click **Protocol**, then click **Add**.
- 5 On the **Select Network Protocol** screen, click **Internet Protocol (TCP/IP)** on the **Network Protocol** list, and then click **OK**.
- 6 Verify the **Internet Protocol (TCP/IP)** check box is checked, and then click **OK**.

Lesson 83: Using Tricon ACM Switches to Set the Node Number

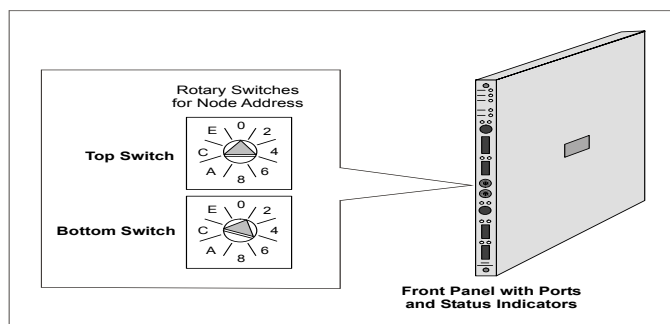
This lesson explains how to set the node number of a Tricon controller using rotary switches on the ACM. The node number uniquely identifies a controller on a network and is typically determined during network planning. The node number must be physically set on the ACM module during installation and it must match the node number that is specified in the **TriStation 1131** project.

The default (factory-configured) setting is for node number 1, which is the top switch set to 0 (zero) and the bottom switch set to 1.

Procedure

- 1 If needed, remove the module from the chassis.

This figure shows the default node number which is 1.



- 2** Set the switches to identify the port using this table. Node numbers can be 1 through 32.

If a Tricon controller includes two ACMs, you must set the switches on both modules to the same node number.

Trinode	Top Switch	Bottom Switch	Trinode	Top Switch	Bottom Switch
1	0	1	17	1	1
2	0	2	18	1	2
3	0	3	19	1	3
4	0	4	20	1	4
5	0	5	21	1	5
6	0	6	22	1	6
7	0	7	23	1	7
8	0	8	24	1	8
9	0	9	25	1	9
10	0	A	26	1	A
11	0	B	27	1	B
12	0	C	28	1	C
13	0	D	29	1	D
14	0	E	30	1	E
15	0	F	31	1	F
16	1	0			

- 3** Set the top switch and the bottom switch on the front panel of the ACM to the hexadecimal values you selected.

Lesson 84: Using Tricon NCM Switches to Set the Node Number

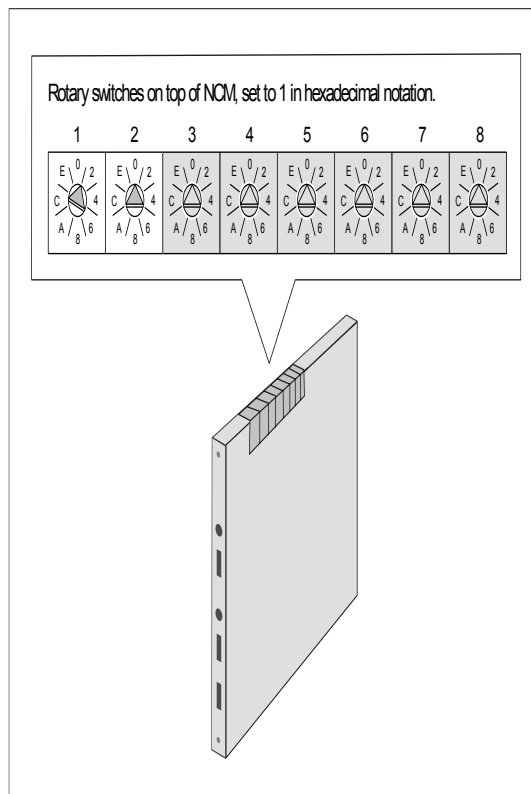
This lesson explains how to set the node number of a Tricon controller using rotary switches on an NCM. The node number uniquely identifies a controller on a network and is typically determined during network planning. The node number must be physically set on the NCM module during installation and must match the node number that is specified in the **TriStation 1131** project.

The default (factory-configured) setting is for node number 1, which is switch 1 set to 1 and switch 2 set to zero.

Procedure

- 1 If needed, remove the module from the chassis.

This figure shows the default (factory-configured) node number which is 1.



- 2** Set the switches to identify the port using this table. Node numbers can be 1 through 32.
- If a Tricon controller includes two NCMs, you must set the switches on both modules to the same node number.

Trinode	Switch 1	Switch 2	Trinode	Switch 1	Switch 2
1	1	0	17	1	1
2	2	0	18	2	1
3	3	0	19	3	1
4	4	0	20	4	1
5	5	0	21	5	1
6	6	0	22	6	1
7	7	0	23	7	1
8	8	0	24	8	1
9	9	0	25	9	1
10	A	0	26	A	1
11	B	0	27	B	1
12	C	0	28	C	1
13	D	0	29	D	1
14	E	0	30	E	1
15	F	0	31	F	1
16	0	1			

- 3** Set **switches 1** and **2** on the top of the **NCM** to the hexadecimal values you selected.
- 4** Verify that **switches 3** through **8** are set to **zero** because they are unused.

Changing Tricon ACM or NCM Node Numbers

These lessons explain how to change the node number of a Tricon ACM or NCM after the **TriStation 1131** application has been downloaded.

Typically a node number is changed only during unplanned expansion or reconfiguration of an existing Ethernet network. Changing the node number requires a shutdown of the controlled process and another Download All.

Procedure (Controller Includes EICM)

- 1 Disconnect the **TriStation 1131** cable from the Net2 port of the ACM or NCM and remove the module from its slot.
- 2 Connect the **TriStation 1131** PC to an EICM serial port using a serial cable.
- 3 In the **TriStation 1131** project, expand the **Controller** tree, and double-click **Configuration**.
- 4 On the **Configuration** tree, click **TriStation Communication**. Specify the **node name**, **node number**, and **IP address**.
- 5 On the **Configuration** tree, double-click the **Controller Panel**. On the **Commands** menu, click **Connect**. Wait for the connection to be made.
- 6 On the **Commands** menu, click **Download All**.
- 7 Change the switches on the module to match the node number you specified on the **TriStation Communication** screen.
- 8 Re-install the ACM or NCM module in its slot.
- 9 On the **Commands** menu, click **Download All**.

Procedure (Controller Does Not Include EICM)

- 1 Prepare for a complete shutdown.
- 2 Remove all three Main Processors and the ACM or NCM from their respective slots. Wait 60 seconds, then re-install the Main Processors.
- 3 Change the **node number switches** on the ACM or NCM.
- 4 Re-install the ACM or NCM.
- 5 In the **TriStation 1131** project, expand the **Controller** tree, and double-click **Configuration**.
- 6 In the **Configuration** tree, click **TriStation Communication**. Specify the **node name**, **node number**, and **IP address**.
- 7 Wait for the Pass indicators on the Main Processors and on the ACM or NCM to illuminate.
- 8 On the **Commands** menu, click **Download All**.

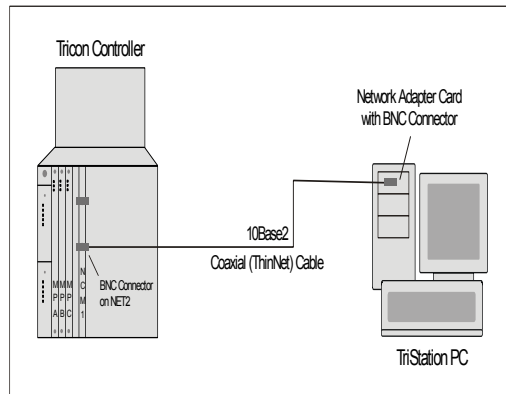
Lesson 85: Directly Connecting a Tricon ACM or NCM to a TriStation 1131 PC

This lesson explains how to directly connect a Tricon ACM or NCM to a **TriStation 1131** PC if the network interface card in the PC has a BNC connector.

The connection requires a 10Base2 coaxial cable. Triconex provides an accessory kit that includes a 10Base2 coaxial cable, BNC T-connectors, and 50-ohm terminators for unused connectors.

Procedure

- 1 To each end of a 10Base2 cable, attach a BNC T-connector and a terminator.
- 2 Attach one of the T-connectors to a BNC connector on Net2 of the ACM or NCM. An NCM is used as an example in the following diagram.



- 3 Attach the other T-connector to the BNC connector on the network interface card in the **TriStation 1131** PC.
- 4 Terminate the BNC connectors on all ACM and NCM modules that are installed in the Tricon controller.
- 5 To terminate an unused BNC connector, you can attach a T-connector with 50-ohm terminators on each end to produce a 25-ohm parallel resistance. Ask your Network Administrator for information about other termination methods.

Lesson 86: Connecting a Tricon ACM or NCM Using a Media Converter

This lesson explains how to connect a Tricon ACM or NCM to a **TriStation 1131** PC if the network interface card in the PC requires a media converter because it does not have a BNC connector.

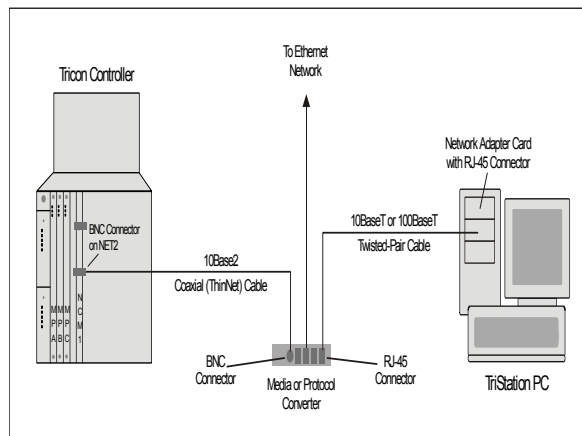
The connection requires a 10Base2 cable, a media converter, and another appropriate cable, such as a twisted-pair cable.

Procedure

- 1 To each end of a 10Base2 cable, attach a BNC T-connector and a terminator.
- 2 Attach one of the T-connectors to a BNC connector on the Net2 port.
- 3 Attach the other T-connector to a BNC connector on the media converter.

For the **TriStation 1131** PC, you can use a 10BaseT or 100BaseTX twisted-pair cable for faster communication.

- 4 Attach one end of the twisted-pair cable to an RJ-45 connector on the network interface card in the **TriStation 1131** PC.
- 5 Attach the other end of the twisted-pair cable to an RJ-45 connector on the media converter.



Lesson 87: Configuring a Tricon ACM or NCM Connection

This lesson explains how to configure a Tricon ACM or NCM connection to a **TriStation 1131** PC.

Before Starting

Before starting, you must determine which IP address to use for the ACM or NCM. Typically, you can get an IP address from your Network Administrator or Information Technology department.

Procedure

- 1 Expand the **Controller** tree, and double-click **Configuration**. On the **Configuration** tree, click **TriStation Communication**.

The screenshot shows the 'TriStation Communication' configuration window. It has a title bar with a yellow icon and the text 'TriStation Communication'. The window is divided into several sections. The first section, 'Select Connections', contains a text box explaining that a network connection requires a Network Communication Module and a serial connection requires an EICM module. Below this are two checkboxes: 'Network Connection' (checked) and 'Serial Connection' (unchecked). The second section, 'Network Connection Setup', contains three input fields: 'Node Number' (value: 6), 'IP Address' (value: 206.32.216.26), and 'Node Name' (value: TRINODE06). Below these fields are three lines of explanatory text: 'Node Number: The number specified by the switches on the communication module.', 'IP Address: The internet protocol address of the module (e.g. 192.168.1.1).', and 'Node Name: Any alphanumeric name up to 20 characters.' The third section, 'TriStation PC', contains a 'Serial Port' dropdown menu (set to COM1) and a text box explaining that this is the port on the PC that will be connected to the controller. The fourth section, 'Default Connection', contains a text box asking which connection to use as the default, with two radio buttons: 'Network Connection' (selected) and 'Serial Connection' (unselected).

- 2 Specify these properties on the **TriStation Communication** screen.

Property	Action
Network Connection	Check the Network Connection check box.
Node Number	Enter the number that you set with rotary switches on the ACM or NCM.
Node Name	Enter a name containing eight or fewer characters to identify the Tricon controller.
IP Address	Enter the physical address of the controller on an Ethernet network.

- 3 On the **Configuration** tree, click the chassis that contains the **ACM** or **NCM**. Double-click **ACM** or **NCM** to open the **Properties** screen, and then click **Setup**.

4 Specify these properties on the **ACM** or **NCM Setup** screen.

Property	Action
Installed (NCM) Used (ACM)	Click this property for the Left Slot or Right Slot, depending on which module is connected to the TriStation 1131 PC.
Privilege	Not applicable to TriStation 1131 communication.
IP Address	If using a default IP address, leave blank. If not, enter the IP address that identifies the controller on the network. This must be the same IP address entered on the TriStation Communication screen.
IP Subnet Mask	For ACM, get the subnet mask from your Network Administrator. For NCM, do not change the default setting which is eight zeroes.
Global Positioning System Installed	Not applicable to TriStation 1131 communication.
Time Synchronization	Not applicable to TriStation 1131 communication.

Trident MP Connection

This section explains how a Trident MP can be used for an Ethernet connection to a **TriStation 1131** PC which uses the DLC protocol. This can be a direct connection from the MP to the PC, or a connection through a hub on a network.

To set up the connection you must install a network interface card and DLC protocol on the PC, set the node number of the controller, connect the PC to an Ethernet port on the MP Baseplate, and configure the connection in the **TriStation 1131** project.

Lesson 88: Installing an NIC Card in a TriStation PC

This lesson explains how to install a network adapter card in a **TriStation 1131** PC that is to be connected to a Trident MP or CM.

Procedure

- 1 Install the network adapter card by following the manufacturer's instructions. *Do not change the factory default settings on the network adapter.*
- 2 Connect the network adapter card directly to an MP or CM port on the Trident controller or to an Ethernet hub.
- 3 Run the diagnostics provided with the network adapter card according to the manufacturer's instructions.

Lesson 89: Installing DLC Protocol on a TriStation 1131 PC

These lessons explain how to install DLC protocol on a **TriStation 1131** PC to be connected to a Trident MP.

Procedure (Windows NT)

- 1 On the **Start** menu, click **Settings**, then click **Control Panel**.
- 2 On the **Control Panel** screen, double-click the **Network** icon.
- 3 On the **Network** screen, click the **Protocols** tab.
If **DLC** is listed under **Network Protocols**, this means the protocol is already installed and you are finished with this procedure.
- 4 For **Network Protocols**, double-click **Add**.
- 5 On the **Select Network Protocols** screen, click **DLC Protocol**.
- 6 If you want to copy files from a specific location, such as a network drive, click **OK**.
- 7 If the required files are on a disk, insert the disk, and click **Have Disk**.

Procedure (Windows 2000)

- 1 On the **Start** menu, click **Settings**, then click **Network and Dial-up Connections**.
- 2 Right-click the network connection where you want to install **DLC**, then click **Properties**.
- 3 On the **Networking** tab, if **DLC** is checked on the list of installed components, this means the protocol is already installed and you are finished with this procedure.
- 4 To install the DLC protocol, click **Install**, click **Protocol**, and then click **Add**.
- 5 On the **Select Network Protocol** screen, click **DLC**, and then click **OK**.
- 6 Verify the **DLC** check box is checked, and then click **OK**.

Lesson 90: Setting a Trident Node Number with an Address Plug

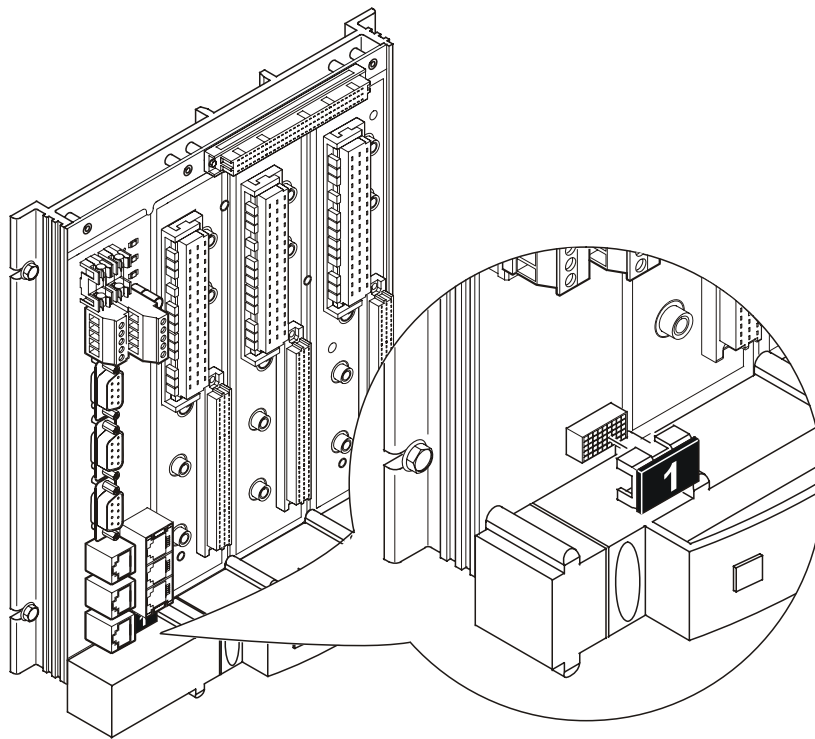
This lesson explains how to set the node number of a Trident controller using an address plug on the MP Baseplate.

The node number uniquely identifies a controller on a network. The node number must be physically set on the MP Baseplate during installation, and it must match the node number that is specified in the **TriStation 1131** project.

Procedure

- 1 In the lower-left corner of the MP Baseplate, attach a Triconex-supplied address plug which has the correct number.

To complete the setting, the node must be configured in the **TriStation 1131** project.

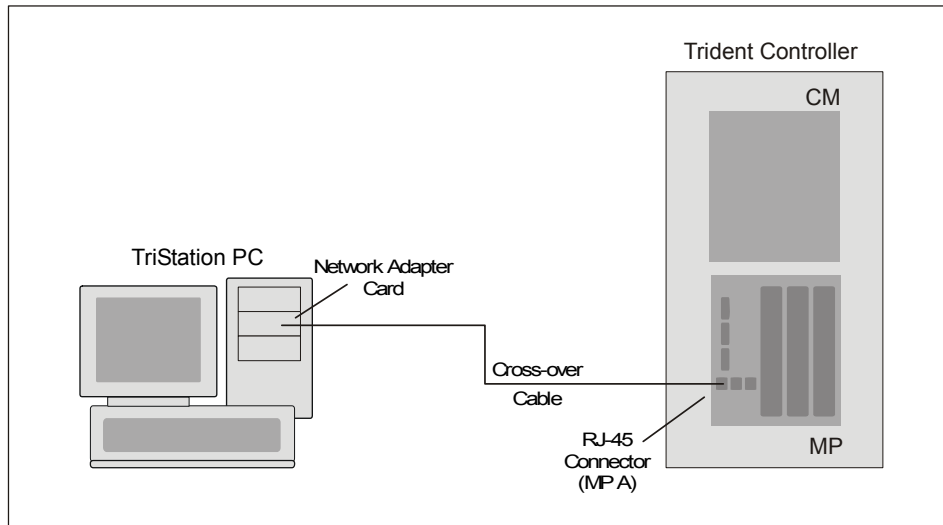


Lesson 91: Directly Connecting a Trident MP to a TriStation 1131 PC

This lesson explains how to directly connect a **TriStation 1131** PC to an Ethernet port on a Trident MP Baseplate using a 10BaseT cross-over cable.

Procedure

- 1 Attach one end of the cross-over cable to one of the RJ-45 connectors on the MP Baseplate. This is typically **MP A**, as shown in the figure.
- 2 Attach the other end of the cross-over cable to the network interface card in the PC.



Lesson 92: Connecting a Trident MP to a TriStation 1131 PC Using a Hub

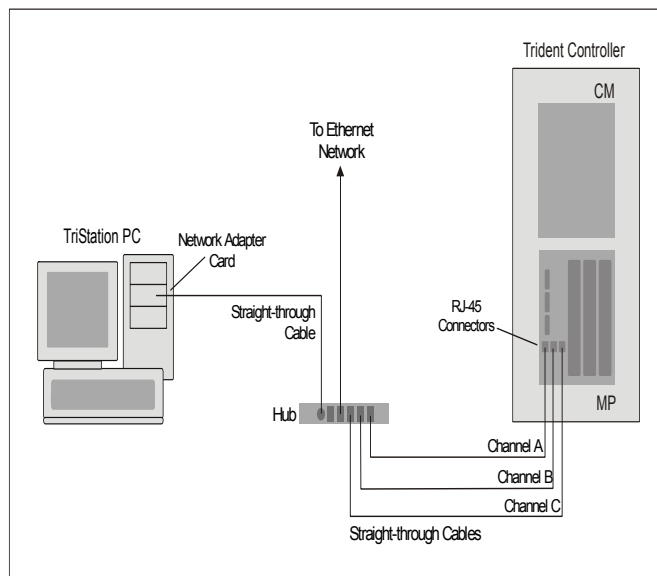
This lesson explains how to connect a Trident MP to a **TriStation 1131** PC using a 10BaseT straight-through cable and a hub.

Procedure

- 1 Attach at least one 10BaseT straight-through cable from an RJ-45 connector on an MP Baseplate to the hub.

Using more than one cable provides redundancy for the **TriStation 1131** connection. If you use only one cable during live operation, you have to unplug it and move it to another RJ-45 connector if the original connection fails.

- 2 Attach the network interface card in the **TriStation 1131** PC to the hub using another 10BaseT straight-through cable.



Lesson 93: Configuring a Trident MP Connection to a TriStation 1131 PC

This lesson explains how to configure a Trident MP connection to a **TriStation 1131** PC.

Procedure

- 1 Expand the **Controller** tree, and double-click **Configuration**. On the **Configuration** tree, and click **TriStation Communication**.

The screenshot shows the 'TriStation Communication' configuration window. It has several sections: 'Select Connections' with checkboxes for 'Network Connection' (unchecked) and 'Main Processor Connection' (checked); 'Network Connection Setup' with fields for 'Node Number' (28), 'IP Address' (206.32.216.54), and 'Node Name' (TRINODE28); 'Main Processor Connection Setup' with radio buttons for 'Left', 'Middle', and 'Right' (all unselected); 'TriStation PC' with a 'NIC Index' field (1); and 'Default Connection' with radio buttons for 'Network Connection' and 'Main Processor'.

- 2 Specify these properties on the **TriStation Communication** screen.

Property	Action
Main Processor Connection	Check the Main Processor Connection check box.
Node Number	Enter the number specified by the address plug on the MP Baseplate.
Node Name	Enter a name with eight or fewer characters to identify the Tricon controller.
Main Processor Setup	Click Left, Middle, or Right to specify which MP port is connected to the TriStation 1131 PC.
NIC Index	Enter the index position of the network interface card in the TriStation 1131 PC.

- 3 On the **Configuration** tree, expand **Hardware Allocation**, and then double-click the **Main Processors (MP/IOP1)**.

- 4 On the **Properties** screen, click **Setup**. On the **MP Setup** screen, click the **Network Ports** tab.
- 5 Select the port that is physically connected to **TriStation 1131** (Left, Middle, or Right).
- 6 Specify the **Transceiver Mode** to match the hardware installed.
- 7 Click **OK** to save.

Trident CM Connection

This section explains how to set up a CM connection to the **TriStation 1131** PC. To do so, you must install a network interface card and TCP/IP protocol on the PC, set the node number of the controller, connect the PC to an Ethernet port on the CM, and configure the connection in the **TriStation 1131** project.

Lesson 94: Installing an NIC Card in a TriStation PC

This lesson explains how to install a network adapter card in a **TriStation 1131** PC that is to be connected to a Trident MP or CM.

Procedure

- 1 Install the network adapter card by following the manufacturer's instructions. *Do not change the factory default settings on the network adapter.*
- 2 Connect the network adapter card directly to an MP or CM port on the Trident controller or to an Ethernet hub.
- 3 Run the diagnostics provided with the network adapter card according to the manufacturer's instructions.

Lesson 95: Installing TCP/IP Protocol on a TriStation 1131 PC

These lessons explain how to install TCP/IP protocol on a **TriStation 1131** PC.

Procedure (Windows NT)

- 1 On the **Start** menu, click **Settings**, then click **Control Panel**.
- 2 In the **Control Panel** window, double-click the **Network** icon.
- 3 In the **Network** dialog box, click the **Protocols** tab.
If **TCP/IP** is listed in **Network Protocols**, it means the protocol is installed and you are finished with this procedure.
- 4 For **Network Protocols**, double-click **Add**.
- 5 On the **Select Network Protocols** screen, click **TCP/IP Protocol** from the list.
- 6 If you want **Windows NT** to copy files from a specified location, such as a network drive, click **OK**.
- 7 If the files are on a disk, insert the disk and click **Have Disk**.

Procedure (Windows 2000)

- 1 On the **Start** menu, click **Settings**, then click **Network and Dial-up Connections**.
- 2 Right-click the network connection where you want to install **TCP/IP**, then click **Properties**.
- 3 On the **Networking** tab, if **DLC** is checked on the list of installed components, it means the protocol is installed and you are finished with this procedure.
- 4 Click **Install**, click **Protocol**, then click **Add**.
- 5 On the **Select Network Protocol** screen, click **Internet Protocol (TCP/IP)** on the **Network Protocol** list, and then click **OK**.
- 6 Verify the **Internet Protocol (TCP/IP)** check box is checked, and then click **OK**.

Lesson 96: Setting a Trident Node Number with an Address Plug

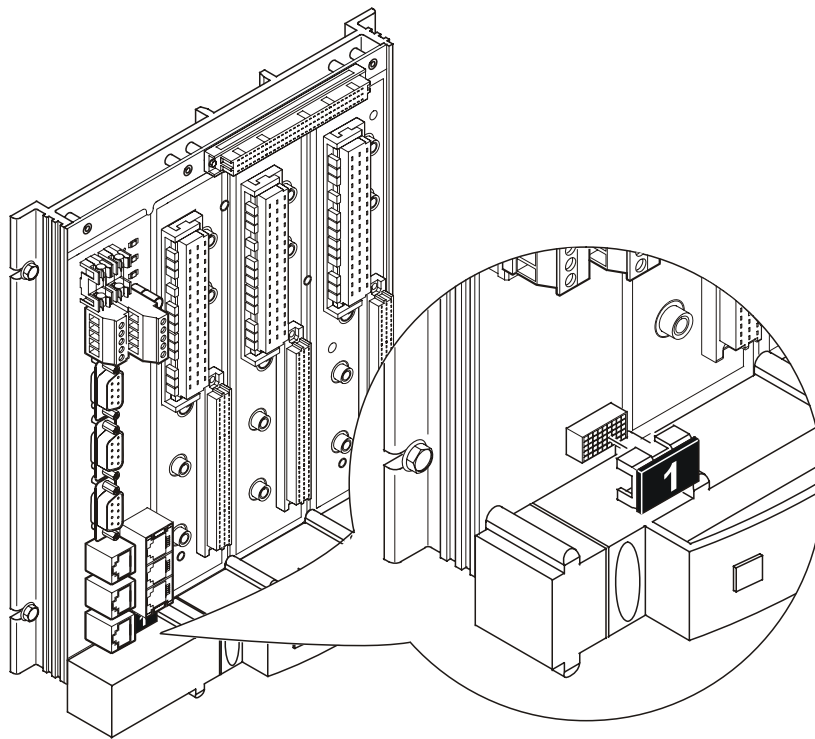
This lesson explains how to set the node number of a Trident controller using an address plug on the MP Baseplate.

The node number uniquely identifies a controller on a network. The node number must be physically set on the MP Baseplate during installation, and it must match the node number that is specified in the **TriStation 1131** project.

Procedure

- 1 In the lower-left corner of the MP Baseplate, attach a Triconex-supplied address plug which has the correct number.

To complete the setting, the node must be configured in the **TriStation 1131** project.



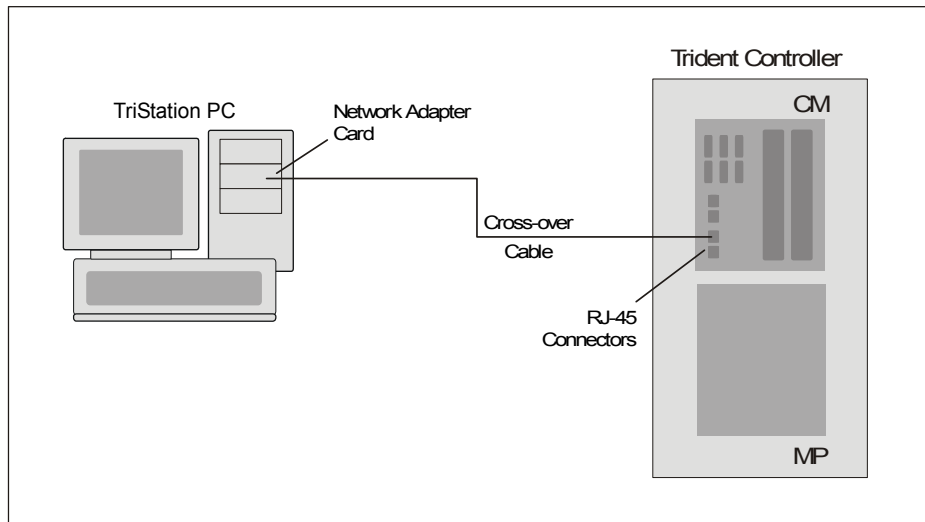
Directly Connecting a Trident CM to a TriStation 1131 PC

This lesson explains how to directly connect a Trident CM to a **TriStation 1131** PC using a cross-over cable.

For a Net1 port, you must use a 10BaseT cable. For a Net2 port, you can use either a 10BaseT or 100BaseTX cable. On the CM Baseplate, you can attach the cable to an RJ-45 connector or to a MAU. For information about MAUs, see the *Trident Communication Guide*.

Procedure

- 1 Attach one end of a cross-over cable to a Net1 or Net2 connector on the CM baseplate, as shown in this example.



- 2 Attach the other end of the cross-over cable to the network interface card in the **TriStation 1131** PC.

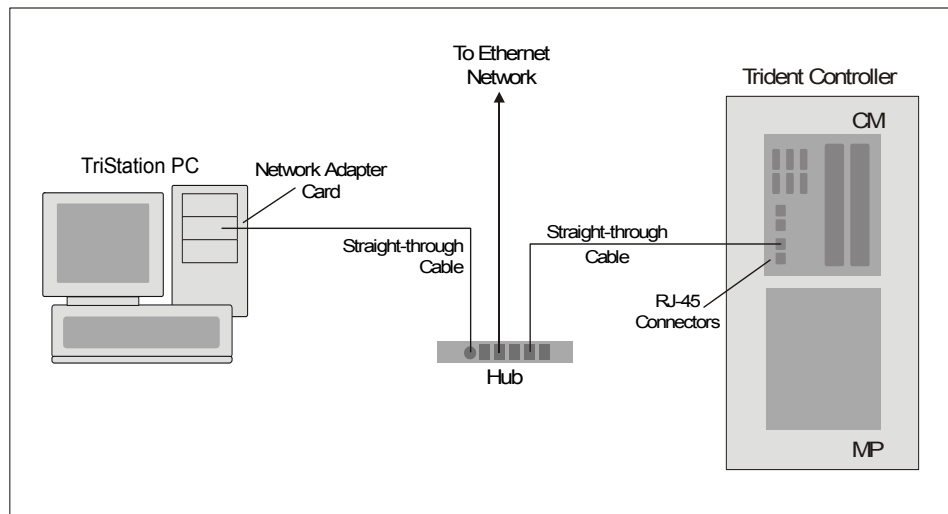
Lesson 97: Connecting a Trident CM to a TriStation 1131 PC Using a Hub

This lesson explains how to connect a Trident CM to a **TriStation 1131** PC using a straight-through cable and a hub.

For a Net1 port, you must use a 10BaseT cable. For a Net2 port, you can use either a 10BaseT or 100BaseTX cable. On the CM Baseplate, you can attach the cable to an RJ-45 connector or to a MAU. For information about MAUs, see the *Trident Communication Guide*.

Procedure

- 1 Attach one end of a straight-through cable to a Net1 or Net2 connector on the CM baseplate.
- 2 Attach the other end of the straight-through cable to an Ethernet hub, as shown in the example below.
- 3 Connect the **TriStation 1131** PC to the hub using another straight-through cable.



Lesson 98: Configuring a Trident CM Connection

This lesson explains how to configure a Trident CM connection to a **TriStation 1131** PC.

Before Starting

Before starting this procedure, you must determine the IP address to use for the CM. If the connection goes through a gateway or a router, you also need IP addresses for those devices. Typically, you can get the necessary IP addresses from your Network Administrator or Information Technology department.

Procedure

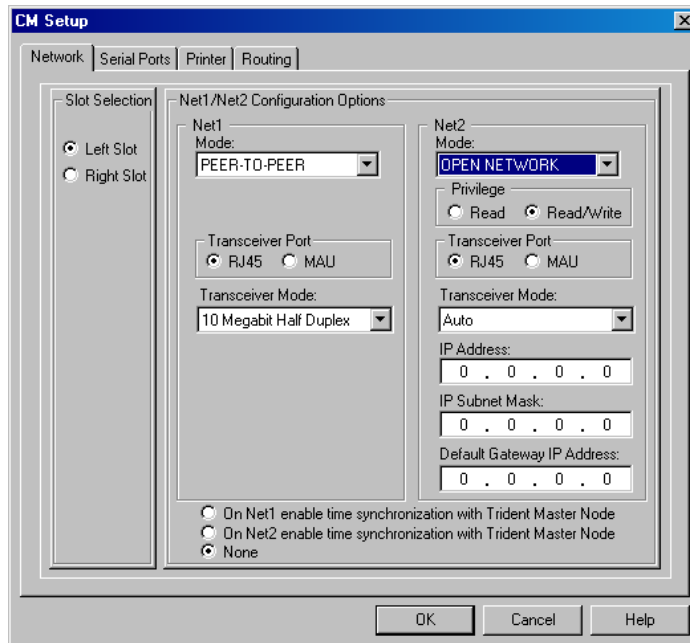
- 1 Expand the **Controller** tree, and double-click **Configuration**. On the **Configuration** tree, click **TriStation Communication**.

The screenshot shows the 'TriStation Communication' configuration window. It has a title bar with a yellow icon and the text 'TriStation Communication'. Inside, there are four sections: 1. 'Select Connections' with a text box explaining network vs. serial connections and two checkboxes: 'Network Connection' (checked) and 'Serial Connection' (unchecked). 2. 'Network Connection Setup' with fields for 'Node Number' (6), 'IP Address' (206.32.216.26), and 'Node Name' (TRINODE06). Below these are explanatory text lines for each field. 3. 'TriStation PC' with a 'Serial Port' dropdown menu set to 'COM1' and a text box explaining it's the port connected to the controller. 4. 'Default Connection' with a text box asking for the default connection and two radio buttons: 'Network Connection' (selected) and 'Serial Connection' (unselected).

- 2 Specify these properties on the **TriStation Communication** screen.

Property	Action
Network Connection	Check the Network Connection check box.
Node Number	Enter the number that you set with rotary switches on the ACM or NCM.
Node Name	Enter a name that contains eight or fewer characters to identify the Tricon controller.
IP Address	Enter the IP address.
NIC Index	Enter the index position of the network interface card in the TriStation 1131 PC.

- 3 On the **Configuration** tree, double-click the **CM (COM: CM)**. On the **Properties** screen, click **Setup**.



- 4 Specify these properties for the **Net1** or **Net2** port, depending on which is connected to the **TriStation 1131** PC.

Property	Action
Slot Selection	Click Left Slot or Right Slot, depending on which slot contains the module that is connected to the TriStation 1131 PC.
Mode	For the TriStation 1131 connection, select Open Network from the list. For each CM on a baseplate, you can select Open Network for either Net1 or Net2, but not for both of these ports.
Privilege	Click Read or Read/Write to specify access privileges for external devices on the network. A TriStation 1131 application must use the Privilege option in conjunction with the MP.REMOTE_WRT_ENBL control attribute (and possibly other write controls) to enable writes by external devices.
Transceiver Port	Click RJ-45 or MAU depending on the type of CM Baseplate port to which you have physically attached the TriStation 1131 cable.
Transceiver Mode	Select the Auto mode if the TriStation 1131 cable can auto-negotiate to either 10 or 100 megabits per second. If your cable operates at only one speed, select the appropriate speed from the list.
IP Address	If using the default node number, do not change this property (leave blank). If using a different node number, enter the IP address that identifies the controller on the network. This must be the same address you enter in step 2.

Property	Action
IP Subnet Mask	Get the subnet mask from your Network Administrator.
Default Gateway IP Address	If the CM connection to the TriStation 1131 PC goes through a default gateway, enter the IP address of the gateway.
Time Synchronization	Click None. This property does not apply to TriStation 1131 communication.

Tricon Printing

A Tricon controller can print brief ASCII text messages if a communication port is connected to a printer and the **TriStation 1131** application includes standard print function blocks.

Print messages are typically used for alarms, status, and maintenance. A sample alarm message might include the name of an analog input point, its time stamp and value, and a statement that the value is out of range. If the Tricon system includes numerous controllers or is connected to a DCS, alarms are typically displayed on an operator workstation.

To print from a Tricon controller, you must connect an EICM parallel port to a Centronics-compatible printer, configure the connection in the **TriStation 1131** project, and use print function blocks in the **TriStation 1131** application.

Effect of Printing on Scan Time

Each time a message is printed, the print function blocks in the **TriStation 1131** application are executed and the scan time increases. Typically, the print function blocks are subject to conditional execution which means they are not executed every scan. When you set the scan time in **TriStation 1131**, make sure it includes the execution time for all conditional statements in the application.

If the scan time is not long enough, the execution of all conditional statements (when the conditions are True) could result in scan-time overruns. You can minimize this problem by limiting the amount of printer output. An alternative is to use a PC event logger such as the Triconex SOE Recorder. For more information, see the *SOE Recorder User's Guide*.

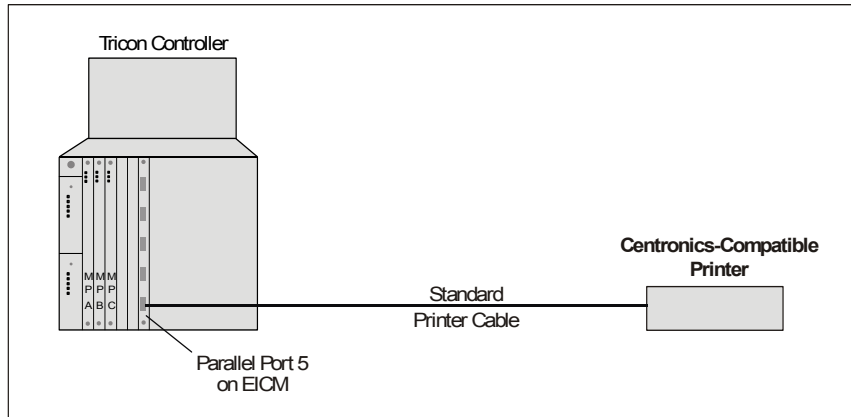
Lesson 99: Connecting a Tricon EICM Port to a Printer

This lesson explains how to set up a Centronics-compatible printer and connect it directly to a Tricon EICM parallel port.

You can use a standard PC printer cable with a maximum cable length of 5 to 6 meters (15 to 20 feet), depending on the quality of the cable

Procedure

- 1 If the printer package has an installation program, copy the program to the **TriStation 1131 PC**.
- 2 Follow the instructions, and run the diagnostic routine, if available.
You do not need the printer driver that may have come with the package.
- 3 Connect one end of the cable to the printer, and connect the other end to EICM parallel port 5 or 10. (Other EICM ports cannot be used for printing.)



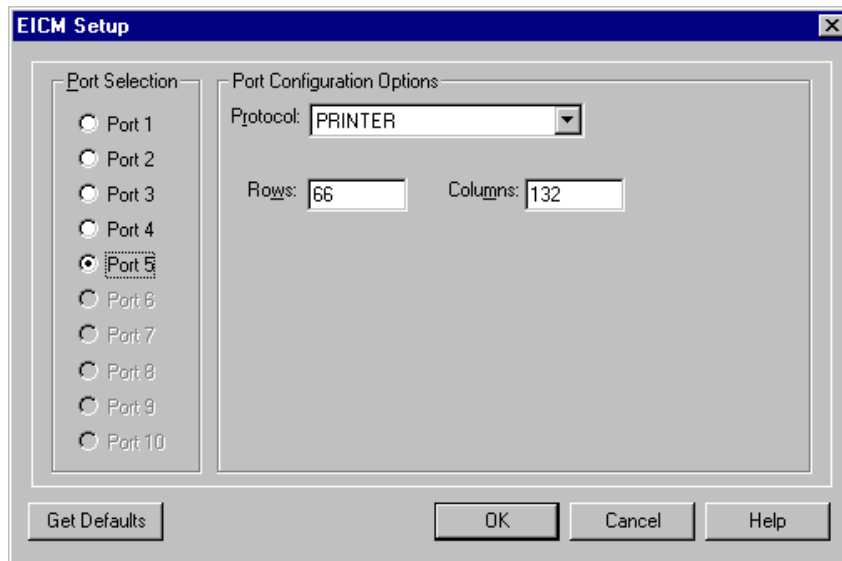
Lesson 100: Configuring a Tricon EICM Port for Printing

This lesson explains how to configure a Tricon EICM port that is connected to a Centronics-compatible printer.

You do not need the printer driver that may have come with the printer package.

Procedure

- 1 Expand the **Controller** tree, and double-click **Configuration**. On the **Configuration** tree under **Hardware Allocation**, double-click **EICM**, and then click **Setup**.



- 2 Specify these properties on the **EICM Setup** screen.

Property	Action
Port Selection	Click Port 5 or Port 10. Other ports cannot be used for printing.
Protocol	Click Printer on the Protocol list.
Rows	Enter the number of lines (rows) to be displayed on a page.
Columns	Enter the number of characters per line.

- 3 Click **OK** to save the configuration.

About Function Blocks for Tricon Printing

A **TriStation 1131** application must use print function blocks to send messages to a printer.

Each print function block has a PRINTER parameter which specifies the port number where the printer cable is connected. For a Tricon EICM port, the PRINTER parameter must be 5 for a left EICM port or 10 for a right EICM port. (Other EICM ports cannot be used for printing.)

Each time a message is printed, the print function blocks in the **TriStation 1131** application are executed and the scan time increases

This table lists the print function blocks in the Tricon Library.

Print Function Block	Purpose
PRINT_BOOL	Prints a three-character field containing either Off or On.
PRINT_CDT	Prints the current date and time.
PRINT_CRLF	Prints a new line (carriage return and line feed).
PRINT_CTOD	Prints the current time of day.
PRINT_DINT	Prints a DINT value.
PRINT_REAL	Prints a REAL value.
PRINT_STRING	Prints a string of text.
PRINTR_FLUSH	Clears the print buffer.

Trident Printing

A Trident controller can print brief ASCII text messages if a communication port is connected to a printer and the **TriStation 1131** application includes standard print function blocks.

Print messages are typically used for alarms, status and maintenance. A sample alarm message might include the name of an analog input point, its time stamp and value, and a statement that the value is out of range. If the Trident system includes numerous controllers or is connected to a DCS, alarms are typically displayed on an operator workstation.

To print from a Trident controller, you must connect a CM Ethernet port to a print server that is connected to a print server; configure these devices in the **TriStation 1131** project; and use print function blocks in the **TriStation 1131** application.

Affect of Printing on Scan Time

Each time a message is printed, the print function blocks in the **TriStation 1131** application are executed and the scan time increases. Typically, the print function blocks are subject to conditional execution which means they are not executed every scan. When you set the scan time in **TriStation 1131**, make sure it includes the execution time for all conditional statements in the application.

If the scan time is not long enough, the execution of all conditional statements (when the conditions are True) could result in scan-time overruns. You can minimize this problem by limiting the amount of printer output. An alternative is to use a PC event logger such as the Triconex SOE Recorder. For more information, see the *SOE Recorder User's Guide*.

Devices for Trident Printing

At a minimum, the printing devices you can use with a Trident controller are an HP JetDirect-compatible print server and a line printer for ASCII text. You can also use a router or a hub.

Print Server and Cables

A print server that is connected to a Trident CM must use the HP JetDirect print protocol and operate at speeds of 10 or 100 megabits per second. Standard communication cables are suitable for this connection.

You can purchase communication cables from other manufacturers. You must purchase print servers elsewhere because Triconex does not supply them. Black-box cables and Hewlett-Packard print servers are examples of dependable network printing devices.

Triconex has tested these Hewlett-Packard print servers and can recommend them:

- HP JetDirect Ex Plus
- HP JetDirect 500X Series, model J3265A

Printers

You must select a printer that is compatible with your print server. The Trident controller prints ASCII text only, which does not include formatting or graphics, so a Centronics-compatible printer is adequate. Laser printers are also suitable.

For more information, see the *Trident Communication Guide*.

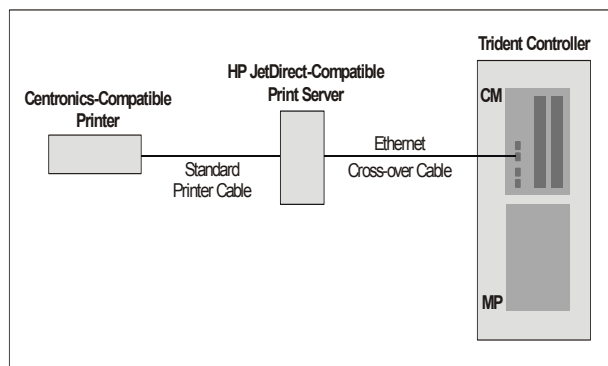
Lesson 101: Directly Connecting a Trident CM to Printing Devices

This lesson explains how to directly connect a Trident CM to an HP JetDirect-compatible print server and printer.

You can use standard communication cables for these connections.

Procedure

- 1 If the print server and printer packages have installation programs, install them on the **TriStation 1131** PC.
- 2 Follow the instructions, and run the diagnostic routines if available.
You do not need the printer drivers that came with the packages.
- 3 Connect the printer to the print server, and connect the print server to a CM Ethernet port (Net1 or Net2).



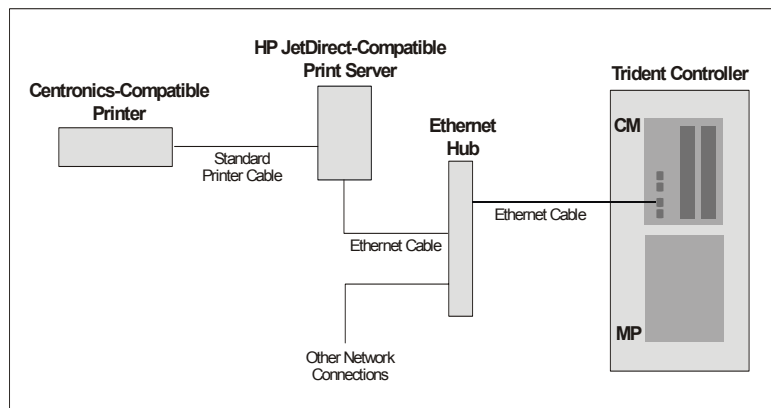
Lesson 102: Connecting a Trident CM to Printing Devices Using a Hub

This lesson explains how to connect a Trident CM to an HP JetDirect-compatible print server and printer by using a hub. You can use standard communication cables for these connections.

You do not need to install the printer drivers that may have come with the print server and printer packages.

Procedure

- 1 If the print server and printer packages have installation programs, copy the programs to the **TriStation 1131** PC.
- 2 Follow the instructions that came with the packages, and run the diagnostic routines, if available.
- 3 Connect the printer to the print server, and connect the print server to a hub. Connect the hub to a CM Ethernet port (Net1 or Net2).

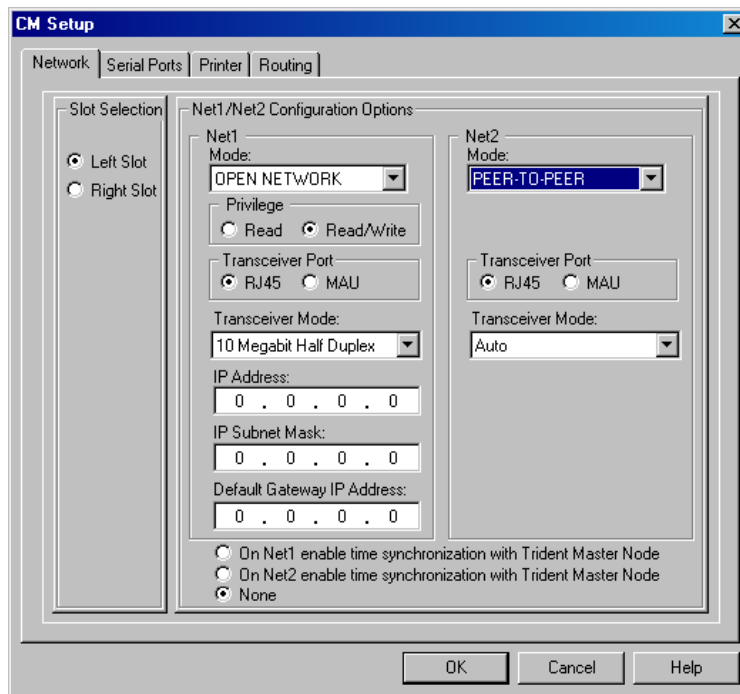


Lesson 103: Configuring a Trident CM for Printing Devices

This lesson explains how to configure a Trident CM port that is connected to a print server and printer.

Procedure

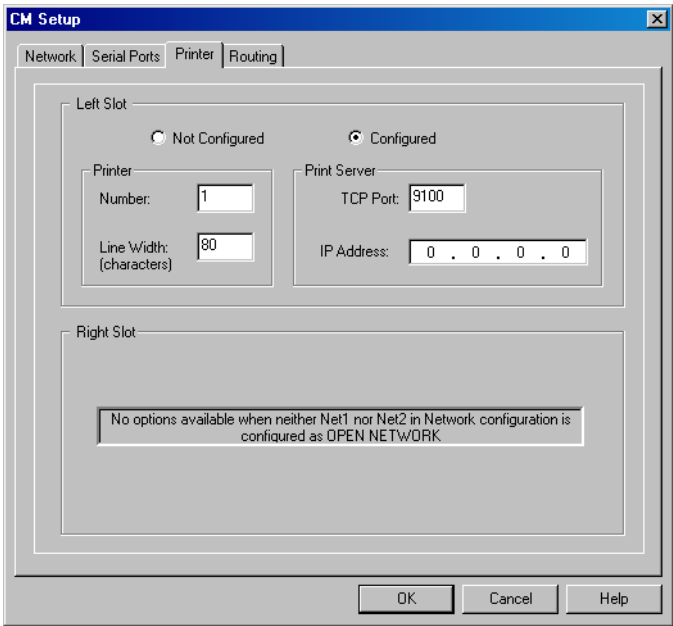
- 1 Expand the **Controller** tree, and double-click **Configuration**. On the **Configuration** tree under **Hardware Allocation**, double-click **CM (COM: CM)**, and then click **Setup**.



- 2 Specify these properties on the **CM Setup Network** tab.

Property	Action
Slot Selection	Click Left Slot or Right Slot depending on where the CM is installed on the baseplate.
Mode	For the Net1 or Net2 port, whichever is connected to the printer, click the Open Network mode.

3 Click the **Printer** tab.



4 Specify these properties on the **CM Setup Printer** tab.

Property	Action
Not Configured or Configured	Click Configured for the slot in which the CM is installed.
Printer Number	Enter a number from 1 to 10. This must be the same number that is declared for the PRINTER parameter in print function blocks.
Line Width	Enter the maximum printable line width for your printer, based on the manufacturer's specifications. The most typical line widths are 80 characters and 132 characters.
TCP Port	Enter the TCP/IP Port number that was defined by the manufacturer of the print server.
IP Address	Enter the 32-bit IP address of the print server on the network. If the print server is not on the same subnet as the controller, you must specify the destination address using the Routing tab.

5 Click **OK** to save the configuration.

IP Addresses

When communication between a **TriStation 1131** PC and a Triconex controller is done through an Ethernet network, the IP address of the controller must be specified on the communication module.

If the controller is not part of a larger network, it may be possible to use the default IP address which is set when the **TriStation 1131** project is downloaded (assuming the correct node number and default IP address are specified in the project).

If you cannot use a default address, there are other ways to set an IP address on a network. All begin with asking the Network Administrator for the intended IP addresses. The easiest way is to use a Reverse ARP (RARP) server that has been programmed in advance with the intended addresses. Other ways include temporary connection of the **TriStation 1131** PC to a non-Ethernet module during downloading.

Lesson 104: Using the Default IP Address for TriStation Communication

This lesson explains how to use the default IP address for network communication with a **TriStation 1131** PC.

Procedure

- 1 Connect the controller to the network using:
Net2 port (Tricon ACM or NCM)
Net1 or Net2 (Trident CM)
- 2 Power up the controller.
- 3 Connect the **TriStation 1131** PC to the network, or directly to a network port on the communication module.
- 4 In the **TriStation 1131** project, expand the **Controller** tree, double-click **Configuration**, and then click **TriStation Communication**.
- 5 On the **TriStation Communication** screen, verify the **IP Address** is:
192.168.1.1 (Tricon ACM or NCM)
192.168.1.1 (Trident CM Net1)
192.168.1.2 (Trident CM Net2)
- 6 On the **Controller** tree, click **Configuration**. Expand **Hardware Allocation**, click the slot where the communication module is installed and click **Setup**. If not installed, insert a communication module, and then click **Setup**.
- 7 On the **Setup** screen, enter the same **IP address** specified on the **TriStation Communication** screen.
- 8 If the controller includes a redundant communication module, enter the same **IP address** for the other slot.
- 9 On the **Controller** tree, click the **Controller Panel**. From the **Commands** menu, click **Connect**.
- 10 Wait about **40 seconds** for the module to reset and become active.
When the module is active, the **Active indicator** is green.
- 11 On the **Commands** menu, click **Download All** to download the **TriStation 1131** project to the controller.
- 12 On the **TriStation 1131** PC, from the **Start** menu, click the **MS-DOS Command Prompt**.
- 13 Enter the command *ping* followed by the **IP address** to be tested. For example, for an IP address of 206.32.216.43, enter this:

```
ping 206.32.216.43
```
- 14 If the network connection is made, the reply includes the IP address followed by byte and time information.
If the connection is not okay, the reply is "**Request timed out.**"

Lesson 105: Setting an IP Address Using a RARP Server

This procedure explains how to set the IP address of a communication module using a RARP server on the local network. To use this procedure, the Network Administrator must program the RARP server with the intended IP address for the controller. If this is not possible, use another method to set the IP address.

Procedure

- 1 Give the Network Administrator the MAC address, which is:
 40-00-00-00-00 (Tricon)
 40-00-00-00-*x*-03 (where *x* is the Trident controller node number)
- 2 Ask the Network Administrator for the IP address that is to be used for the controller.
- 3 Connect the controller to the network through a network port on the communication module. Power up the controller.
 During initialization, the communication module sends a request to the RARP server for an IP address that has been mapped to its own 48-bit MAC address.
- 4 On the **TriStation 1131** PC, from the **Start** menu, click the **MS-DOS Command Prompt**.
- 5 Enter the command *ping* followed by the **IP address** to be tested. For example, for an IP address of 206.32.216.43, enter this:
 ping 206.32.216.43
- 6 If the network connection is made, the reply includes the IP address followed by byte and time information.
 If the connection is not okay, the reply is "**Request timed out.**"
- 7 Connect the **TriStation 1131 PC** to the network, or directly to a network port on the communication module.
- 8 In the **TriStation 1131** project, expand the **Controller** tree, click **Configuration**, and then click **TriStation Communication**.
- 9 On the **TriStation Communication** screen, specify the **Node Number** of the controller and the intended **IP address**.
- 10 On the **Commands** menu, click **Connect**. Wait until the connection is made.
- 11 On the **Commands** menu, click **Download All** to download the TriStation project to the controller.

Lesson 106: Setting a Tricon IP Address Using an EICM Connection

This lesson explains how to set the IP address of a Tricon ACM or NCM by initially connecting the **TriStation 1131** PC to an EICM port, and downloading the **TriStation 1131** project. After the address is set, you can disconnect **TriStation 1131** from the EICM port, and reconnect it to a Net2 port on the ACM or NCM.

Procedure

- 1 Ask the Network Administrator for the IP address to be used for the ACM or NCM.
- 2 Connect the **TriStation 1131** PC to a serial port on the EICM.
- 3 Connect the controller to the network using a Net2 port on the ACM or NCM.
- 4 In the **TriStation 1131** project, configure the following:
 - The EICM serial port and Net2 Ethernet ports
 - The node number and node name of the controller
 - The intended IP address
- 5 Power up the controller.
- 6 On the **Controller** tree, click **Controller Panel**. On the **Command** menu, click **Connect To**.
- 7 On the **Connect To** screen, click the **Serial Port** option and the **COM** port to which the **TriStation 1131** cable is connected.
- 8 Connect to the Tricon controller and download the **TriStation 1131** project.
The ACM or NCM initializes (resets) and accepts the IP address that you specify in the **TriStation 1131** project.
- 9 On the **TriStation 1131** PC, from the **Start** menu, click the **MS-DOS Command Prompt**.
- 10 Enter the command *ping* followed by the **IP address** to be tested. For example, for an IP address of 206.32.216.43, enter this:

```
ping 206.32.216.43
```
- 11 If the network connection is made, the reply includes the IP address followed by byte and time information.
If the connection is not okay, the reply is **"Request timed out."**
- 12 If the IP address is set, you can disconnect the **TriStation 1131** PC from the EICM serial port, and connect it to a Net2 port or to the network.

Lesson 107: Setting a Trident IP Address Using an MP Connection

This lesson explains how to set the IP address of the Trident controller by initially connecting the **TriStation 1131** PC to an MP port and downloading the **TriStation 1131** project. After the address is set, you can disconnect TriStation 1131 from the MP port, and reconnect it to an Ethernet port on the CM.

Procedure

- 1 Ask the Network Administrator for the IP address to be used for the controller.
- 2 Connect the **TriStation 1131** PC to a **TriStation** port on the MP Baseplate.
- 3 Connect the controller to the network through an Ethernet port on the CM.
- 4 In the **TriStation 1131** project, configure the following:
 - The MP and CM ports
 - The node name and node number of the controller
 - The intended IP address
- 5 Power up the controller.
- 6 On the **Controller** tree, click **Controller Panel**. On the **Command** menu, click **Connect To**.
- 7 On the **Connect To** screen, click **Main Processor Module Port** and **Left**, **Middle**, or **Right** for the MP port to which the **TriStation 1131** cable is connected.
- 8 After connecting to the controller, download the **TriStation 1131** project.
The CM initializes (resets) and accepts the IP address you specified in the **TriStation 1131** project.
- 9 On the TriStation 1131 PC, from the **Start** menu, click the **MS-DOS Command Prompt**.
- 10 Enter the command *ping* followed by the **IP address** to be tested. For example, for an IP address of 206.32.216.43, enter this:


```
ping 206.32.216.43
```
- 11 If the network connection is made, the reply includes the IP address followed by byte and time information.
If the connection is not okay, the reply is **"Request timed out."**
- 12 If the IP address is set, you can disconnect the **TriStation 1131** PC from the MP port, and connect it to an Ethernet port on the CM or to the network.

Lesson 108: Setting a Trident IP Address Using a CM Connection

This lesson explains how to set the IP address for a Trident CM by temporarily configuring a default IP address for the CM, and assigning a default IP address to the **TriStation 1131** PC.

Procedure

- 1 Ask the Network Administrator for the IP address to be used for the CM.
- 2 Connect the Trident controller to the network using an Ethernet port (Net1 or Net2) on the CM.
- 3 Connect the **TriStation 1131** PC to an Ethernet port on the CM, using a direct or network connection.
- 4 On the **TriStation 1131** PC, use Windows procedures to set the **IP address** of the PC to either of the following:
 - 192.168.1.x if the PC is physically connected to a Net1 port, where x is any unused host number.
 - 192.168.2.x if the PC is physically connected to a Net2 port, where x is any unused host number.
- 5 Wait for the **TriStation 1131** PC to reset.
- 6 Open the **TriStation 1131** project. Expand the **Configuration** tree, click **Configuration**, and then click **TriStation Communication**.
- 7 Specify the **node name**, **node number**, and the **default IP address** of the controller.
- 8 Use the **Network** tab on the **CM Setup** screen to specify the intended **IP address** for the Ethernet port that is connected to the network.
- 9 Power up the Trident controller.
- 10 On the **Controller** tree, click **Controller Panel**. On the **Command** menu, click **Connect To**.
- 11 On the **Connect To** screen, click the **Serial Port** option and the **COM** port to which the **TriStation 1131** cable is connected.
- 12 Verify that **Communication Module Port** is selected and the **default IP address** is displayed.
- 13 Connect to the controller and download the **TriStation 1131** project.
- 14 Wait for the download to complete.

After the download is complete, **TriStation 1131** displays the message, “**Connection failed.**” The default IP address you specified in the node definition is invalid, and the intended IP address of the CM is set.
- 15 On the **TriStation 1131** PC, use Windows procedures to set the **IP address** of the PC to its actual address on the network.
- 16 On the **TriStation 1131** PC, from the **Start** menu, click the **MS-DOS Command Prompt**.

- 17** Enter the command *ping* followed by the **IP address** to be tested. For example, for an IP address of 206.32.216.43, enter this:

```
ping 206.32.216.43
```

- 18** If the network connection is made, the reply includes the IP address followed by byte and time information.

If the connection is not okay, the reply is **“Request timed out.”**

- 19** In the **TriStation 1131** project, change the **default IP address** to the newly set IP address of the **TriStation Communication** screen.
- 20** Use the **Controller Panel** to reconnect the **TriStation 1131** project to the controller.
- 21** After the IP address is set on the network, you must reconfigure the IP address in the TriStation 1131 project, and assign a valid IP address to the TriStation 1131 PC.

Lesson 109: Specifying a Trident CM Default Gateway

This lesson explains how to set the address of a default gateway for a controller that must communicate with devices on another network. A default gateway is a router that forwards all messages not addressed to stations within the local subnet.

Procedure

- 1 Expand the **Controller** tree and double-click **Configuration**.
- 2 On the **Configuration** tree, click **Hardware Allocation** to display the modules that are configured for this system.
- 3 Double-click the **CM** icon to open the **Properties** dialog box, and click **Setup** to display the configuration options for the CM.
- 4 On the **Network** tab, select **Left Slot** or **Right Slot** depending on which CM you are configuring.
- 5 For **Net1** or **Net2** (depending on which one is connected to an Ethernet network), select **Open Network** from the list under **Mode**.
- 6 Under **Default Gateway Address**, enter the **IP address** of the default gateway that is connected to the local subnet.

Lesson 110: Specifying a Trident CM for Network Routing

This lesson explains how to specify routes to destinations outside the local network for controllers that do not have access to a default gateway.

Each route must include an IP address for the destination, a subnet mask, and a gateway address.

Procedure

- 1 Expand the **Controller** tree and double-click **Configuration**.
- 2 On the **Configuration** tree, click **Hardware Allocation** to display the modules that are configured for this system.
- 3 Double-click the **CM** icon to open the **Properties** dialog box, and click **Setup** to display the configuration options for the CM.
- 4 Click the **Routing** tab and enter an **IP address** under **Destination Address**, **Subnet Mask**, and **Gateway Address** for each route that need to specify.

Lesson 111: Testing an Ethernet Connection

This lesson explains how to test a connection from a Triconex communication module to an Ethernet network by using the Ping command from an MS-DOS Command Prompt.

The test is performed on the **TriStation 1131** PC. Before doing the test, you must have set the IP address of the communication module on the network.

Procedure

- 1 On the **TriStation 1131** PC, from the **Start** menu, click the **MS-DOS Command Prompt**.
- 2 Enter the command *ping* followed by the IP address to be tested. For example, for an IP address of 206.32.216.43, enter this:

```
ping 206.32.216.43
```

- 3 If the network connection is made, the reply includes the IP address followed by byte and time information.

If the connection is not okay, the reply is **“Request timed out.”**

Application Building and Implementation

Application Building and Implementation 334

Controller Testing 341

Maintenance 349

Application Building and Implementation

Application Building

After creating the logic for your program, you must build the application before downloading it for testing in the Emulator or real-time execution on a controller.

If the programs in the application have not been compiled, the Build Application command compiles them, and then attempts to build the application.

Errors and warnings are displayed in the Message View. Errors must be resolved before an application can be downloaded, but warnings do not affect online execution. Typically, warnings refer to unused points in an application.

Implementation

TriStation 1131 provides two ways to test and monitor an application:

- Emulator Testing – using the Emulator Panel
- Controller Testing – using the Controller Panel

Emulator Testing

The emulator simulates the execution of program in a Tricon or Trident controller without physically connecting to a controller or field instruments. By running the logic offline, you can test and debug your application before downloading it for real-time execution.

You can test individual variables or monitor the entire program.

Although the physical connections and logical configuration do not need to be completed before emulator testing, it is good practice to do so before downloading to the emulator.

Controller Testing

Controller testing is usually performed when the controller is physically connected to field instruments, either in a test facility that simulates the system (Factory Acceptance Test) or at the site while the control process is offline (Site Acceptance Test). The **TriStation 1131** hardware must be configured and match the physical configuration before testing on a controller.

Lesson 112: Building an Application

Build Application

The Build Application command builds an application by compiling programs in the execution list that have been modified since the last time they were compiled. The command then links the object code, library routines, and configuration elements to create an executable application.

Each successive time you use the Build Application command, it compiles and links only documents and items that have changed since the last build. After using the Build Application command several times, you should use the Rebuild Application command. A rebuild compiles and links all documents and items in the application, not just the ones that have changed since the last build.

In building an application, you must select the programs to include in the application and the order the programs will be executed.

Scan Time

You can set the scan time of the application, which is the number of milliseconds anticipated for the scan. The number is requested before an application is built. After the application is built and downloaded, the controller determines an actual scan time range and uses the specified scan time if it falls within these limits.

The controller determines a valid range for the scan time based on these operations:

- Read inputs
- Executes the application
- Process messages
- Writes outputs

If the application is running, the scan time can be set to a number different from the requested number without downloading the application. To determine what the actual, requested and surplus scan times are, see Monitoring and Changing the Scan Time in the *Diagnostic Monitor Appendix*.

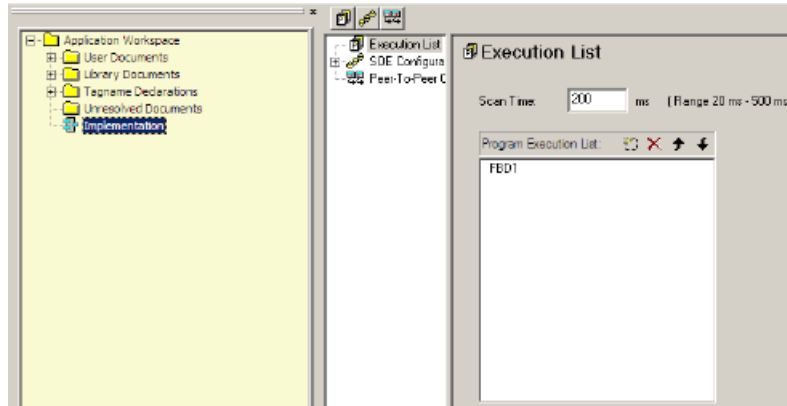
- For Tricon (3006 MP), the maximum scan time is 500 milliseconds.
- For Tricon (3008 MP), the maximum scan time is 450 milliseconds.
- For Trident, the maximum scan time is 200 milliseconds.
- For Tricon and Trident, the default is 200 milliseconds.

In this lesson, you will:



- Specify program order and scan time
- Build an application

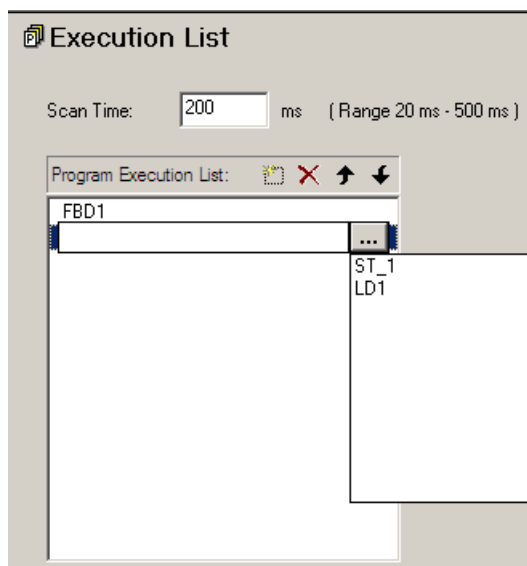
Procedure

- 1 Open the **Water_Tank_Alarm** project.
- 2 On the **Application** tree, double-click **Implementation**. On the **Implementation** tree, double-click **Execution List**.



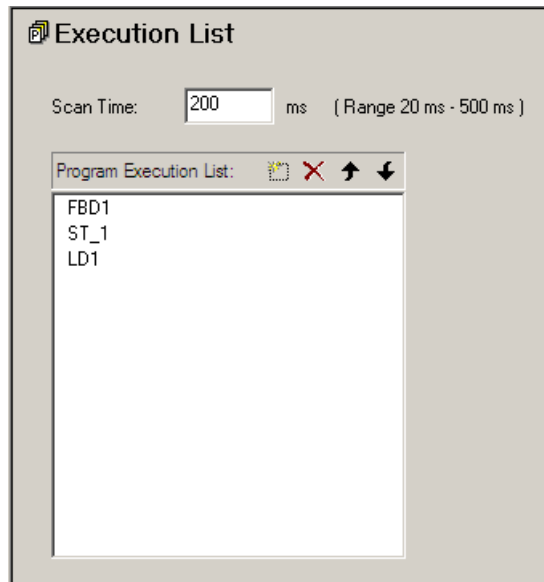
For this lesson, keep the default scan time of **200 milliseconds**.

- 3 On the **Program Execution List**, click the **New (Insert)** button) , and then click the **Browse** button  to display the available programs.



- 4 Click the **Structured Text** program and add it to the **Program Execution List**.

- 5 Click the **Ladder Diagram** program and add it to the **Program Execution List**.



- 6 On the **Project** menu, click **Build Application** or **Rebuild Application**, if you have made changes to the logic or corrected errors.
- 7 Click **OK to Save changes to Implementation**. The Message View automatically opens and displays the status of the build process.

If there are no errors, the build is successful. If there are errors, click each error message to see the location of the error. Correct the errors, compile the affected user documents, and then rebuild the application.



Lesson 113: Emulator Testing

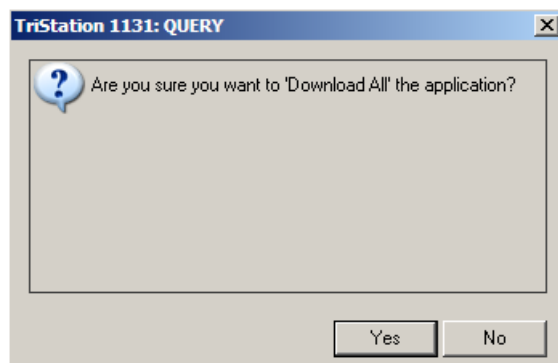
The Emulator Panel enables you to test an application offline and make changes or corrections before testing it on the controller.

You can set test values for the variables and monitor the results. You can also monitor the program execution. The program is displayed graphically on the Emulator.

In this lesson, you will download an application to the Emulator.

Procedure

- 1 Open the **Water_Tank_Alarm** project.
- 2 Click the **Controller** tab, then double-click **Emulator Panel**. The monitor sheet is displayed.
- 3 Click **Connect** .
- 4 Click **Download All** .



- 5 Click **Yes** to download the application. The Message View displays the compilation.

```
>> Building Configuration...
  ++ Initializing program 'FBD1'...
>> Creating program instances
>> Assembling Libraries for Emulator...
  > Linking for emulation...
The estimated stack size is 388 bytes.
0 ERROR(s), 0 WARNING(s)
```


Lesson 114: Monitoring the Program Execution

When testing and monitoring variables or programs in the Emulator, the three steps to remember are:

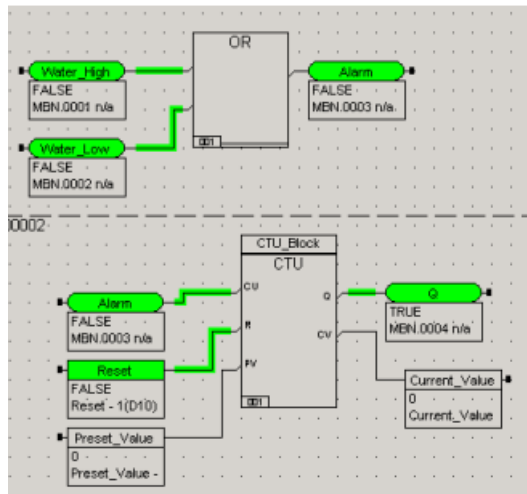
- Connect
- Download
- Run

In this lesson, you will monitor the program execution.

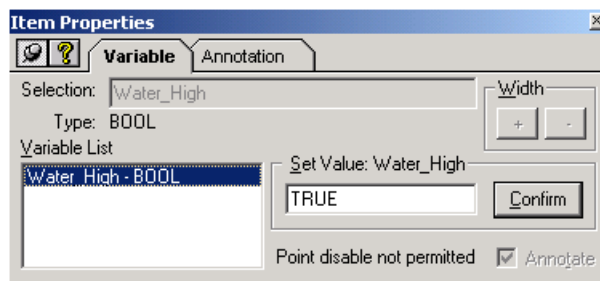
Procedure

- 1 Connect and download the **Water_Tank_Alarm** program to the Emulator.
- 2 Expand **Programs**, and click the program you want to monitor.
- 3 Click **Display Program Document**  .

The logic is displayed with the colors representing the power flow enabled. The states and values of the variables appear in the annotation boxes.



- 4 Double-click the **Water_High** variable. The **Item Properties** screen is displayed.

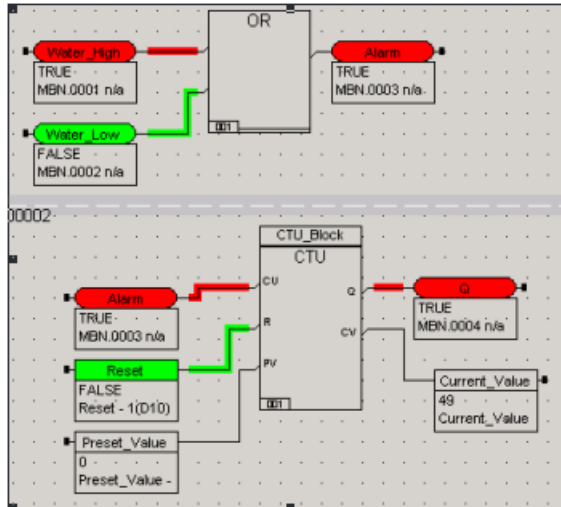





- 5 Test the logic by changing the **Set Value** to **True**. Enter **1** or **True** in the **Set Value** field, and then click **Confirm**.

A True value for that variable means that the water level is too high.

- 6 Click **Run**  to run the execution or **Single Step**  to run the execution for a single scan.

The water level is detected as being too high and sets the alarm. The alarm sets the counter, which displays the count in the Current_Value annotation box.



- 7 Click **Pause**  to pause the execution or **Halt**  to stop the program execution.
- 8 Click **Disconnect**  to stop running the program and disconnect from the Emulator.

Controller Testing

Controller testing can be performed when the controller is connected to simulation field devices or the control process is offline. There are two types of download states:

- Download All – for initial loading and execution of an application
- Download Change – for online modifications to the application

When an application is downloaded to the controller, a snapshot of the information is retained. If you make a change to any of the application components, you are advised that the change will require a Download All or a Download Change.

Download All

The Download All command is generally used:

- The first time you download the application to the Tricon or Trident controller.
- For major changes than cannot be managed by the Download Change command.
- After a planned or accidentally safety shutdown of the controlled process.

After using the Download All command, the project is automatically in a Download Change state. It is recommended that you use Download All only for the initial download to the controller.

For more information about Download All, see the Maintenance section of this chapter.

Download Change

Download Change is a way to make modifications to an off-line system during the project development phase. You must use extreme caution when using Download Change to modify a safety-critical application that is running on line because an error in the application modification or system configuration could cause a trip or unpredictable behavior.


For more information about Download Change, see the Maintenance section of this chapter.

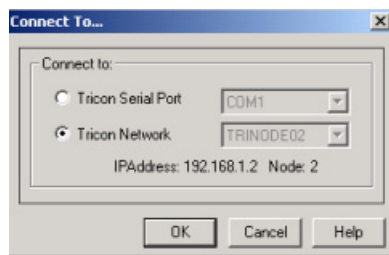
Lesson 115: Performing a Downloading All to the Controller

This lesson explains how to use the Download All command to load an application to the controller. A best practice is to rebuild the application before downloading it.

This procedure can be used for testing when the controller is connected to simulation field devices or the control process is offline.

Procedure

- 1 Test the application on the Emulator.
- 2 Ensure that the **TriStation 1131** PC is connected to a Tricon chassis. For more information, see **TriStation 1131** Communication chapter.
- 3 Ensure the keyswitch is turned to **PROGRAM**.
- 4 Expand the **Configuration** tree, and double-click the **Controller Panel**.
- 5 On the **Commands** menu, click **Connect** .

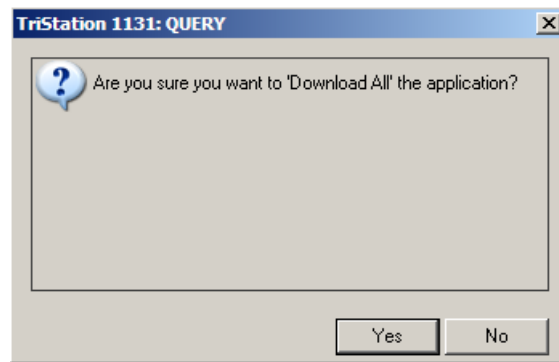


The **Connect To** screen shows the default communication setting. If needed, change the connection setting and click **OK**.

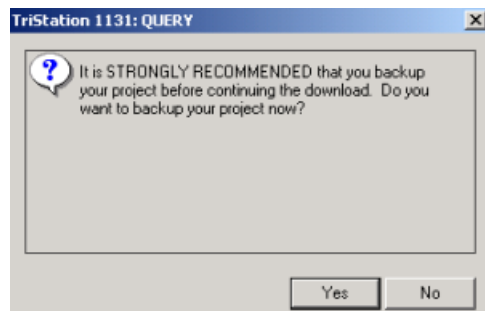
- 6 Enter the connection password if required.
- 7 For Tricon, ensure the keyswitch is turned to **PROGRAM**. This is the factory setting.
For Trident, ensure the mode is **Enable Programming and Control Operations**, by doing this:




On the **Commands** menu, click **Set Programming Mode**, and then click **Enable Programming and Control Operations**. (This is the default setting).

- 8 On the **Commands** menu, click **Download All** .



- 9 Click **Yes**.





- 10 Click **Yes** to backup your project, and then click **Save**.
The application is downloaded to the controller.
- 11 To view the execution, click **Run**  or **Single Step** .
- 12 To quit running the program, click **Disconnect** .

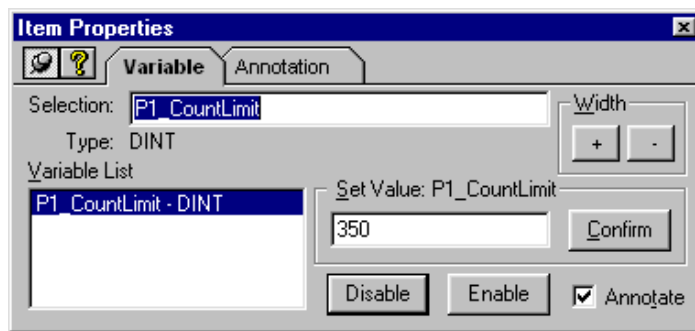
Lesson 116: Monitoring Variables on the Controller

This lesson explains how to monitor and enable or disable variables while the application is running on the controller.

CAUTION This should only be done if the controller is not connected to a live system or if maintenance is being done.

Procedure

- 1 Connect and download the application.
- 2 Drag the function blocks and variables you want to monitor to the sheet.
- 3 On the Commands menu, click **Run**  or **Single Step** .
- 4 To enable or disable a variable, double-click the variable and click **Enable** or **Disable**.




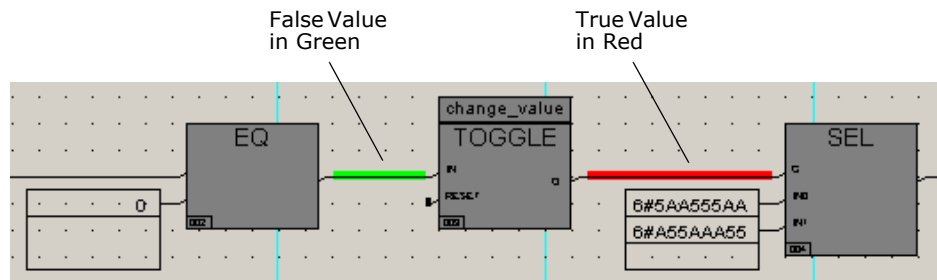
- 5 Continue testing, as needed.




Lesson 117: Monitoring the Program Execution

This lesson explains how to display the program execution, which shows the program executing on the controller.

Procedure

- 1 Connect and download the application.
- 2 On the **Controller** tree, expand **Programs**, and click the program you want to test.
- 3 Click **Display Program Document** .



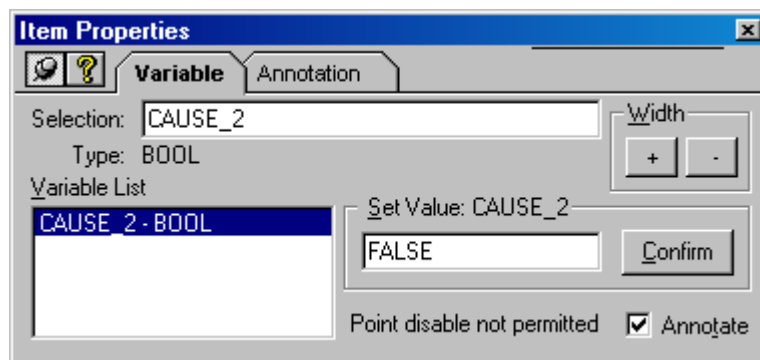
- 4 To view the program execution, click **Run**  or **Single Step** .
- 5 To quit running the program, click **Disconnect** .

Lesson 118: Adding Annotation for Variables

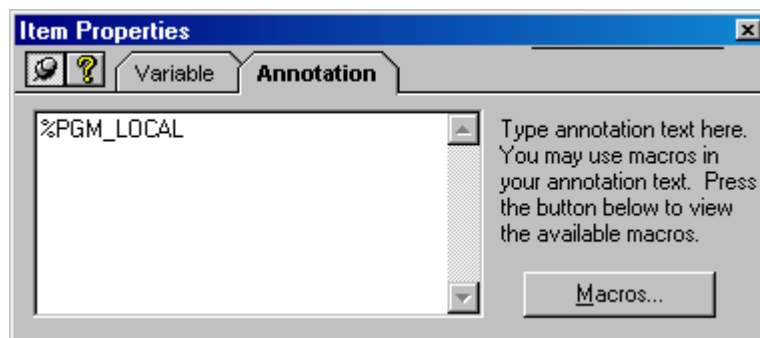
This lesson explains how to add an annotation to a variable, which allows you to specify information displayed while the program is executing. The annotation can include text and macros.

Procedure

- 1 Connect and download the application.
- 2 If needed, drag variables to be tested onto the sheet.
- 3 Double-click a variable, and check the **Annotate** check box on the Variable tab.



- 4 Click the **Annotation** tab.



- 5 Click the **Macros** button to change the macro identified with the annotation.
- 6 To copy a macro, click the macro name and press **Ctrl+c**. To paste the macro, close the **Edit Macros** screen, click in the **Annotation** field, and click **Ctrl+v**. You can also enter text with the macros.
- 7 Continue testing, as needed.

Determining the Scan Surplus

This section explains how to determine the scan surplus, which indicates whether the actual scan time required to execute the application uses less time or more time than the requested scan time in the project. When the actual time is less, the scan surplus is positive, which means the scan time setting can be decreased. When actual time is more, the scan surplus is negative,

which means the scan time should be increased to ensure that communication errors do not occur.

The Triconex Diagnostic Monitor displays information to determine scan surplus, including the requested Scan Time, Actual Scan Time, and Scan Surplus for an application. For more information, see the Help documentation for the *Diagnostic Monitor*.

Positive Scan Surplus

A positive scan surplus means the application executes in less time than the requested scan time. For example, if the requested scan time is 150 milliseconds, and the actual scan time is 100 milliseconds, there is a positive scan surplus of 50 milliseconds.

- If the surplus is 20 milliseconds or 10 percent of the actual scan time, do nothing.
- If the surplus is greater than 20 milliseconds or 10 percent of the actual scan time, decrease the number for the scan time. In this example, the scan time could be set to 130 milliseconds.

Negative Scan Surplus

A negative scan surplus means the actual scan time is greater than the requested scan time. For example, if the requested scan time is 150 milliseconds, and the actual scan time is 200 milliseconds, there is a negative scan surplus of 50 milliseconds.

- If the surplus is negative, increase the number for the requested scan time by the negative amount plus 20 milliseconds. In this example, the scan time should be set to 220 milliseconds.

Process Safety Time Requirements

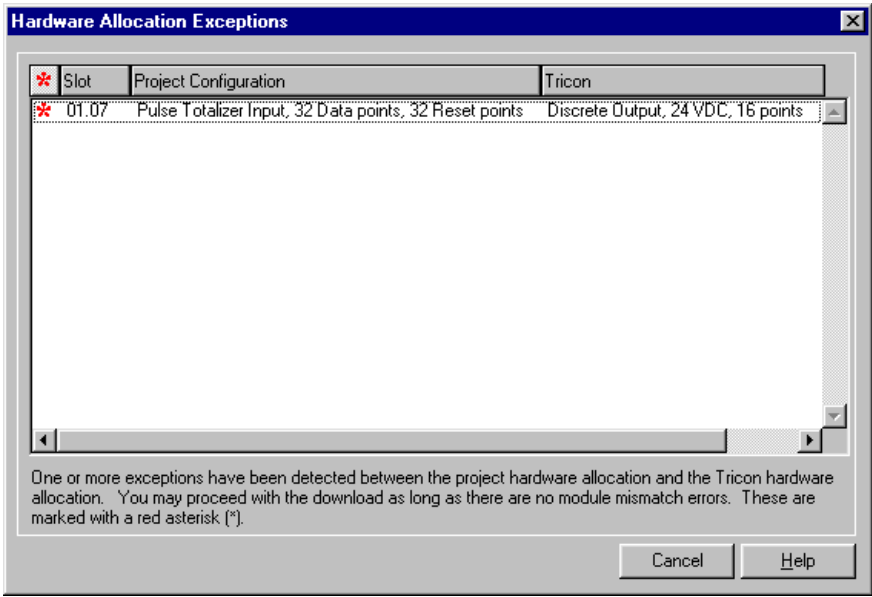
You should determine the Process Safety Time (PST) required by the application. The PST is the period of time during which the process could shift from a safe operating condition to a dangerous condition. The scan time for an application should be half the PST. For example, a burner management system has a PST of 1 second, which means the scan time should be 500 milliseconds.

Lesson 119: Displaying Hardware Allocation Exceptions

This lesson explains how to display hardware allocation exceptions, which indicate that the hardware configuration in the project does not match the physical hardware configuration.

Procedure

- 1 Download an application to the controller.
If the logical and physical configurations do not match, the **Hardware Allocation Exception** screen is displayed.



- 2 Differences are identified with an asterisk must be fixed in the project before the application can be download. Other differences may allow you to download the application.

Identifier	Description
Asterisk *	<p>A red asterisk identifies a module mismatch, which means the module specified in the hardware configuration for the project is different from the module in the physical system.</p> <p>This error must be fixed by changing the hardware configuration in the project.</p>
Empty slot	<p>An empty slot error indicates that either of these:</p> <ul style="list-style-type: none">• The hardware configuration specifies a module that is empty in the controller.• The controller contains a module that is not included in the hardware configuration. <p>The application can be downloaded.</p>

Maintenance

This section explains how to plan and manage changes to an application running on a controller which is attached to a live system.

WARNING Changing a safety-critical application that is running on a controller should be avoided because an error in the application could cause a trip or unpredictable behavior.

Steps for a Download Change

This list includes steps for making changes to an application running on a controller.

WARNING For a safety-critical application running on a live system, you must use extreme caution because a configuration error in the changed application could cause unpredictable behavior or a trip.

Step	
<input type="checkbox"/>	Verify the TriStation 1131 software is correctly installed.
<input type="checkbox"/>	Plan for the change.
<input type="checkbox"/>	Determine whether a Download All or Download Change is required.
<input type="checkbox"/>	Review the hardware configuration. If needed, correct the hardware configuration to match the physical configuration.
<input type="checkbox"/>	Compare the current project with the last downloaded.
<input type="checkbox"/>	Test on the emulator.
<input type="checkbox"/>	Ensure the scan time has a surplus
<input type="checkbox"/>	Download changed application.
<input type="checkbox"/>	Backup the project and copy it to another storage medium.

Planning and Controlling Changes

This section describes recommended procedures for planning and controlling changes to an existing application. All changes to an application should be controlled by a change control board or the equivalent, and should comply with strict control procedures.

Recommended Procedure

- 1 Generate a change request defining all changes to the application and reasons for the changes, then obtain approval for the changes from the board.
- 2 Develop a specification for changes including a test specification, then obtain approval for the specification from the board.
- 3 Make the appropriate changes to the application including those related to design, operation, or maintenance documentation.
- 4 Verify the application in the controller matches the last downloaded application. (See Verify Last Download to the Controller Command in the Help documentation.) If the applications do not match, contact Triconex Technical Support.
- 5 Print the Hardware Module Configuration report to compare the current configuration with the last one downloaded to the controller.
- 6 Print all user documents and thoroughly check all changed networks in each document to ensure the changes do not affect other parts of the application.
- 7 Test the new application on the emulator and write a test report.
- 8 Review and audit all changes and the test results with the board.
- 9 When approved by the board, download the changes to the controller.
- 10 Save the downloaded application and back up the appropriate files on a CD or other storage medium.
- 11 Archive two copies of the PT2 (project) file and all associated documentation.

Making Changes to a Downloaded Application

When an application is downloaded to the controller, a snapshot of the information is retained. If you make a change to any of the application components, you are advised that the change will require a Download Change or Download All command.

Effects of a Download Change

A Download Change command affects these features:

- IP Address Change
- I/O Module Changes

IP Address Change

When the Download Change command is used to change the IP address of a communication module, the module resets and re-initializes itself. During the reset, the Fault indicator on the module turns On, which temporarily compromises the TMR status of the controller.

CAUTION If the application has a negative Scan Surplus, do not use the Download Change command.

WARNING Contact Triconex Technical Support before making this change.

I/O Module Changes

When the Download Change command is used to add an I/O module, the TMR status of the controller is temporarily compromised for as much as 8 to 16 scans. If the application uses the following function blocks, the changed application should include logic to accommodate the behavior.

- For Tricon, the IOMAIN and IOBAD parameters of the TR_MP_STATUS function block turn off.
- For Trident, the parameters for the SYS_IO_STATUS, SYS_IOP_STATUS, and SYS_SYSTEM_STATUS function blocks turn off.

WARNING Contact Triconex Technical Support before making this change.

Effects of a Download All

A Download All command requires that the application on the controller is halted.

This table describes the type of changes that require the Download All or Download Change command.

Commands Required with Application Changes

Component	Commands
Chassis	<p>Download All: required if a chassis is added, deleted, or the type is changed.</p> <p>To avoid a Download All after the initial download, include the maximum number of chassis in the application before downloading to the controller.</p>
Functions and function blocks	<p>Download All: required if a function or function block is modified or deleted.</p> <p>Download Change: allowed if a function or function block is added.</p>
IP Address	<p>Download Change: allowed, but not advised, if the IP address of a communication module is changed. See Effects of a Download Change (page 350).</p>
Memory allocation	<p>Download All: required if memory allocation is increased.</p> <p>To avoid a Download All after the initial download, include the allocate additional memory before downloading to the controller.</p>

Commands Required with Application Changes

Component	Commands
Modules	Download All: required if modules are deleted or moved in the configuration. Download Change: allowed if a module is added and the chassis has empty slots, and there is sufficient memory allocated for the points.
Node Number	Download All: required if the address plug and node number configuration are changed.
Number of Send or Receive function blocks	Download All: required if the number of send or receive function blocks is increased or decreased.
Operating Parameters	Download All: required if the Allow Disabling of Points property is changed. Download Change: allowed if these properties are changed. <ul style="list-style-type: none">• Disable Remote Changes to Outputs• Password Required for Connection• Restart on Power-Up (Trident only)• Scan Time• Use Local Time.
Programs	Download All: required if a program is deleted from an application. Download Change: allowed if a program is added to an application and there is sufficient memory.
Tagnames	Download Change: allowed if tagnames are added, modified, or deleted. Tagnames can also be disconnected or connected to different points, if there is enough memory allocated for the required point types. Points must be enabled.
Variables	Download All: required if changes to the Data Type (BOOL, DINT, REAL) or Point Type (Unaliased, Aliases, Read/Write Aliased) require additional memory allocation. Download Change: allowed if changes to the Data Type or Point Type do <i>not</i> require additional memory allocation. Points must be enabled. Also allowed if the variable Name, Description, or Initial Value is added or changed.

Lesson 120: Disabling (Forcing) Points

This lesson explains how to disable points on an application running on a controller, which should be used with care. When a point is disabled, inputs from field instruments do not change the value of the point. Disabling points is typically used when field instruments need to be replaced or repaired.

WARNING A project should not contain disabled points unless there is a specific reason for disabling them, such as initial testing or maintenance.

WARNING Disabling points can increase the scan time.

Procedure

- 1 Expand the Configuration tree, and click Controller Panel. On the Commands menu, click the Connect command, and enter the connection password if required.
- 2 Double-click a point to be disabled, and click Disable.
Only one point can be disabled at a time.



Lesson 121: Downloading Changes to a Downloaded (Running) Application

This lesson explains how to use the Download Changes command to download changes to an application that has been downloaded and is running on a controller.

WARNING Changing a safety-critical application that is running on a controller should be avoided because an error in the application could cause a trip or unpredictable behavior.

For more information, see the Tricon or Trident *Safety Considerations Guide*.

Procedure

- 1 Expand the **Configuration** tree, and double-click the **Controller Panel**.
- 2 On the **Commands** menu, click **Connect** , and enter the connection password if required.
- 3 If a **Download Change** is allowed, on the **Commands** menu click **Download Change** .




The changes are made while the application is running.

Lesson 122: Downloading a Changed Application to the Controller

This lesson explains how to use the Download All command to download a changed application to a controller.

WARNING Using the Download All command requires the current application running on the controller to be halted.

Procedure

- 1 Expand the **Configuration** tree, and double-click the **Controller Panel**.
- 2 On the **Commands** menu, click **Connect** .
- 3 If needed, change the connection setting. Click **OK**.
- 4 If required, enter the connection password.
- 5 If needed change the state to **Download All**. On the **Commands** menu, click **Change State to Download All**.
- 6 Click **Halt** to stop the application.
- 7 For Tricon, turn the keyswitch to **PROGRAM**.
- 8 For Trident, ensure the mode is Enable Programming and Control Operations, by doing this:
On the Commands menu, click Set Programming Mode, and then click Enable Programming and Control Operations. (This is the default setting).
- 9 Click **Download All** .
- 10 Click **Run**  to start the application running on the controller.
- 11 For Tricon, turn the keyswitch to **RUN**.

Writing Custom Function Blocks

Writing Custom Function Blocks	358
Creating an UPDOWN Counter	361
Creating a Function Block Using the ST Editor	373
Creating a Move Function	376
Creating a Rising Trigger Function Block	378
Creating a Timer	380
Invoking an UPDOWN Function Block	384
Invoking an AVERAGE Function Block	386

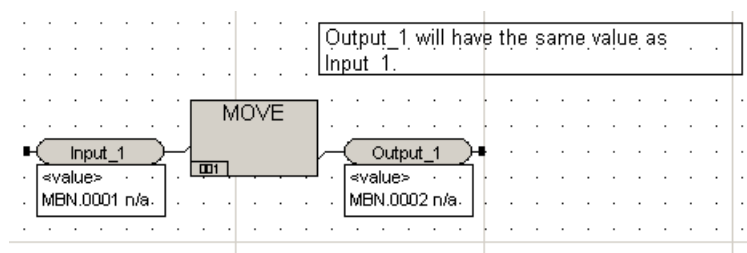
Writing Custom Function Blocks

In this chapter, you will create a project with a program that has five distinct networks. The project combines the developer's tools and techniques used in previous lessons to create custom function blocks that are then copied into the program.

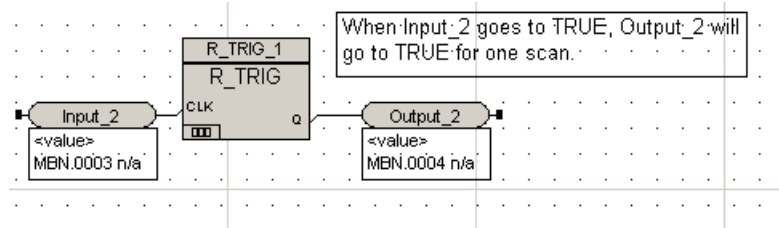
You will use the FBD and the ST editors to create user-defined function blocks that will be invoked by the program. You will then place the function blocks in a user-defined library. Finally, you will create a program with five networks, invoking the custom function blocks.

The program has five networks:

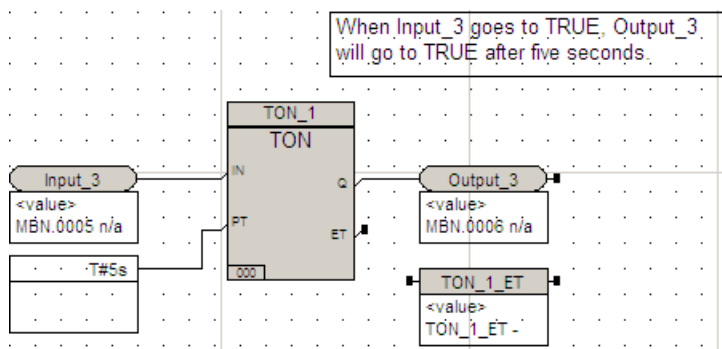
- Network 1 – Uses a MOVE function to change the value of one variable to the same value as another variable.



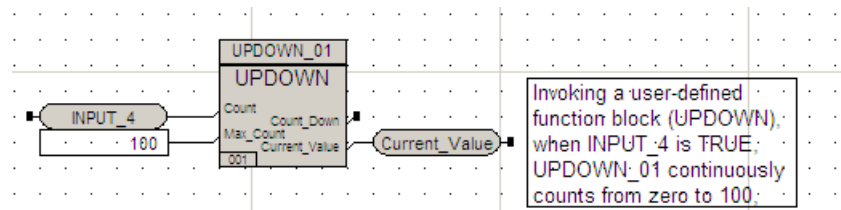
- Network 2 – Uses a Rising Trigger function block to change the status of one variable to that of another variable for one scan.



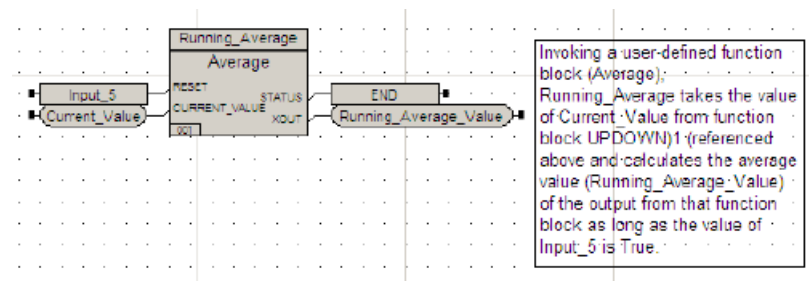
- Network 3 – Uses a Timer to change the status of one variable to that of another input variable after five seconds.



- Network 4 – Invokes a user-defined UPDOWN function block that uses an input value and counts up to 100, then counts down to zero, while displaying the current value.



- Network 5 – Invokes a user-defined AVERAGE function block that calculates the average value of the output value of UPDOWN as it counts up to 100, then counts down to zero.

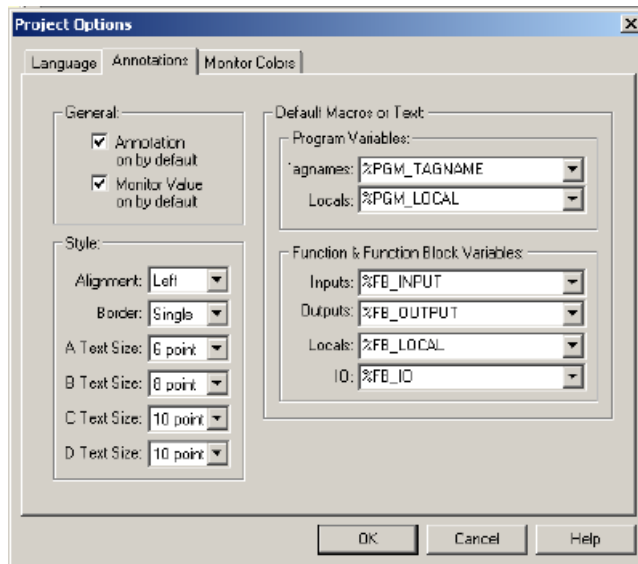


Lesson 123: Creating a TriStation 1131 Project

In this lesson, you will create a new project.

Create a TriStation 1131 Project

- 1 Create a new project using the Tricon platform.
- 2 Name the project **CUSTOM_FBD**.
- 3 Click **OK**.
- 4 Click the **Project** menu, click **Project Options**, and then click the **Annotations** tab.



- 5 Check the following:
 - **Annotation on by default**
 - **Monitor Value on by default**
- 6 Click **OK** and save the project.

Lesson 124: Creating an UPDOWN Counter

In this lesson, you will create a used-defined function block that counts from zero to 100 and then count down from 100 to zero.

For this function block, while the input count (Count) is True, the function block will count up to the maximum count (Max_Count) of 100, then count down to zero.

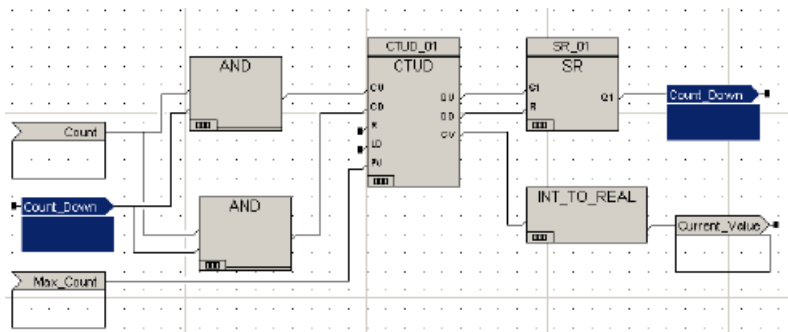
This logic has an inverted input . The value for the input is changed to the opposite value when the function block is executed. For example, if the value is True it, is changed to False.

When an input or output is inverted, a small circle is displayed on the terminal of the function.

You will:

- Place and connect function blocks on the logic sheet.
- Place and connect output variables on the logic sheet.
- Place and connect input variables on the logic sheet.
- Invert an input and an output.
- Add a comment box.

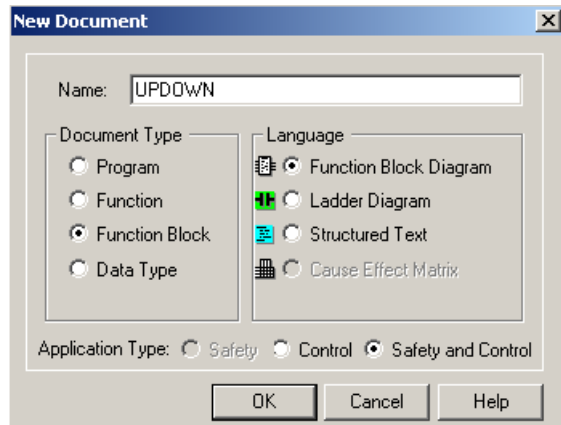
This is the completed logic:



Placing function blocks on the logic sheet

Procedure


- 1 Create a new document named **UPDOWN**.

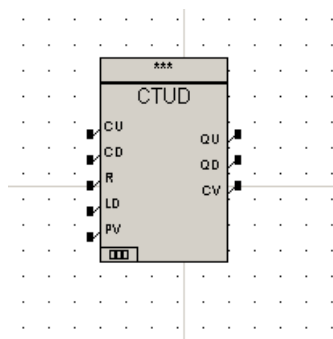


- 2 Specify the following:

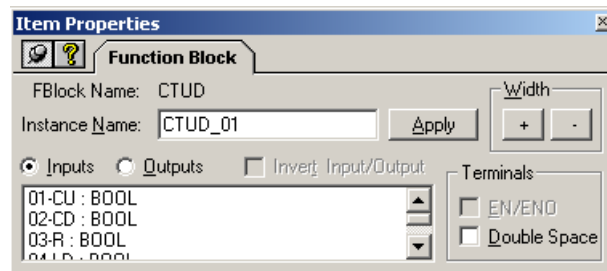
Name: Enter **UPDOWN**
Document Type: Click **Function Block**
Language: Click **Function Block Diagram**
Application Type: Click **Safety and Control**

- 3 Click **OK**.

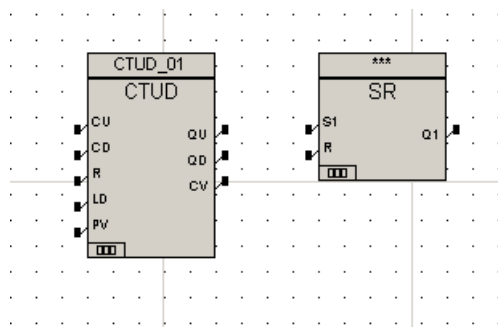
- 4 Use the **Select Function (Block)** button  and place a **Count Up/Count Down (CTUD)** function block on the logic sheet.



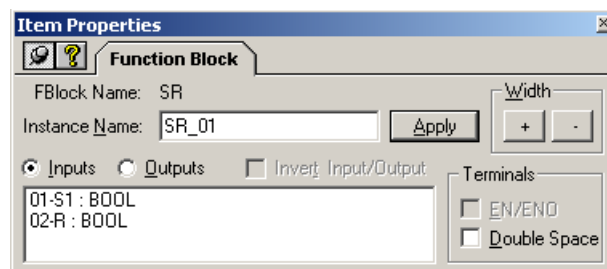
- 5 Double-click the **CTUD** function block to display the **Item Properties** screen.



- 6 Enter **CTUD_01** in the **Instance Name** field. Click **Apply** and then close the screen.
- 7 Use the **Select Function (Block)** button and place a **Set/Reset (SR)** function block on the logic sheet to the right of the **CTUD** function block.

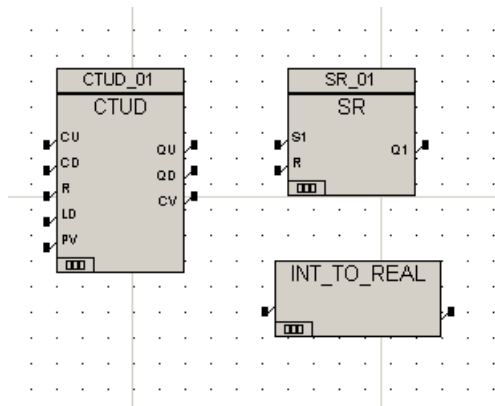


- 8 Double-click the **SR** function block to display the **Item Properties** screen.

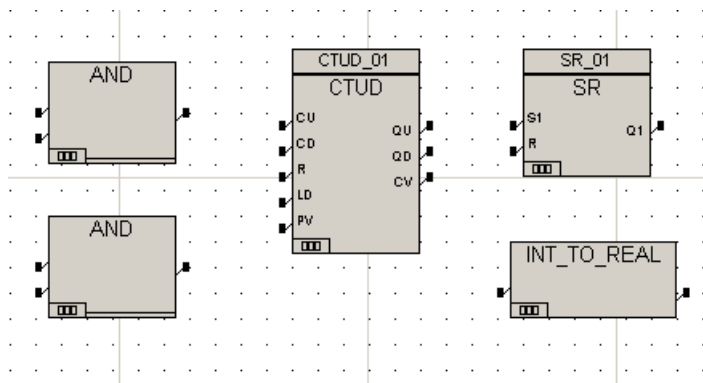


- 9 Enter **SR_01** in the **Instance Name** field. Click **Apply** and then close the screen.

- 10 Use the **Select Function (Block)** button and place an **Integer to Real** (INT_TO_REAL) function on the logic sheet below the **SR** function block.

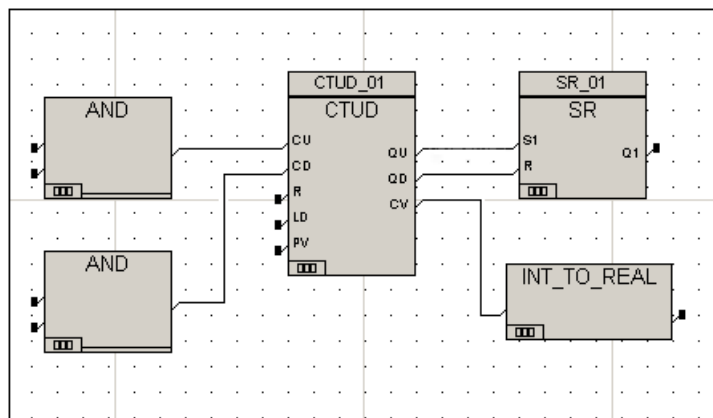


- 11 Use the **Select Function (Block)** button and place two **AND** functions to the left of the **CTUD** function block.




12 Click the Wire Tool button and draw wires from:

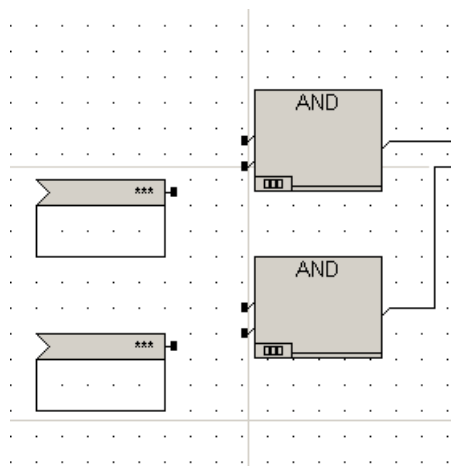
- The output terminal of the top AND function to the CU input connector on the CTUD function block.
- The output terminal of the bottom AND function to the CD input connector on the CTUD function block.
- The QU output terminal on the CTUD function block to the S1 input connector for the SR function block.
- The QD output terminal on the CTUD function block to the R (Reset) input connector for the SR function block.
- The CV output terminal on the CTUD function block to the input connector of the INT_TO_REAL function.



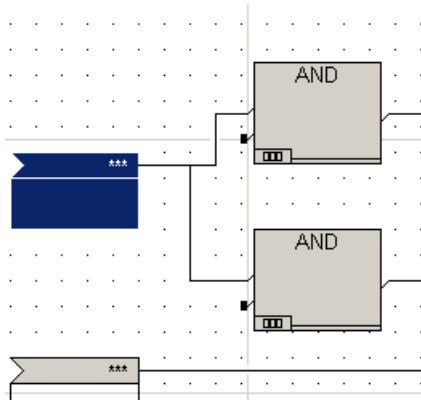
Placing input variables on the logic sheet

Procedure

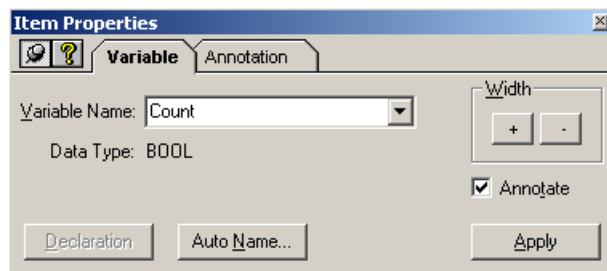
- 1 Use the **Input Variable Tool** button  and place two variables to the left of the AND functions.



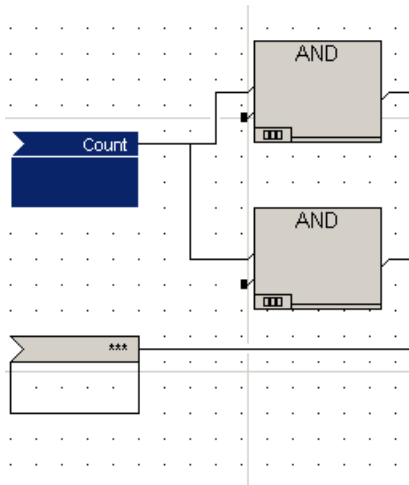
- 2 Connect the top input variable to the top terminals of each **AND** function.



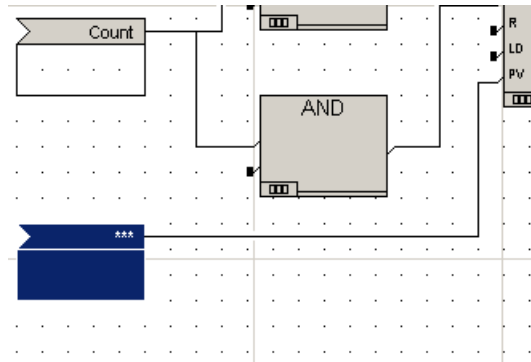
- 3 Double-click the input variable to display the **Item Properties** screen. Enter **Count** in the **Name** field. Click **Apply**.



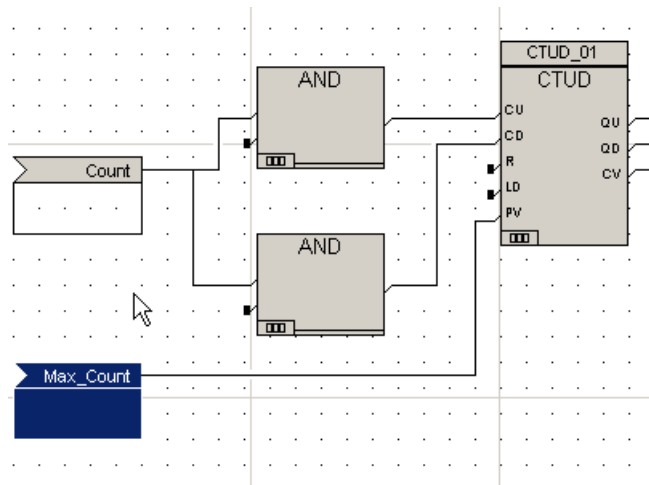
- 4 On the **Declaration** screen, confirm **BOOL** as the Data Type and **Input** as the Variable Type. Then close the screen.



- 5 Connect the bottom input variable to the **PV** (present value) input terminal on the **CTUD** function block.




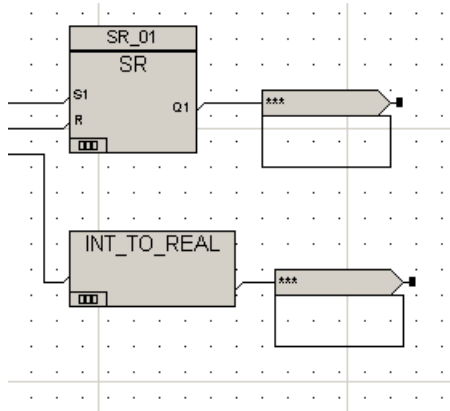
- 6 Double-click the input variable to display the **Item Properties** screen. Enter **Max_Count** in the **Name** field. Click **Apply**.
- 7 On the **Declaration** screen, select **INT** as the Data Type and confirm **Input** as the Variable Type. Click **Apply** and then close the screen.



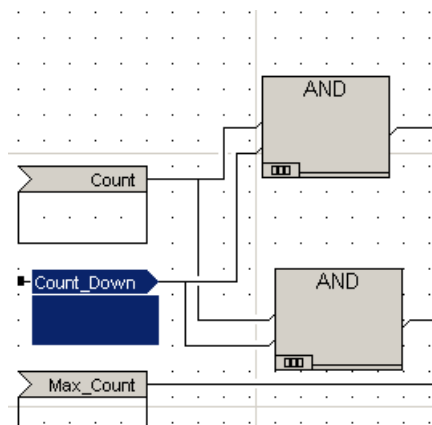
Placing output variables on the logic sheet

Procedure

- 1 Click the **Output Variable Tool** button  and connect output variables to the output terminals on the **SR** function block and the **INT_TO_REAL** function.

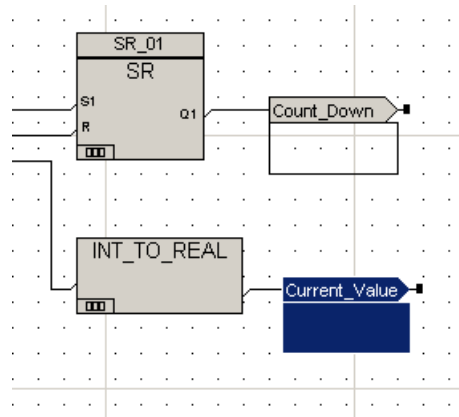



- 2 Double-click the output variable connected to the **SR** output to display the **Item Properties** screen. Enter **Count_Down** in the **Name** field and then click **Apply**.
- 3 Confirm **BOOL** as the Data Type and **Output** as the variable type, and then close the screen.

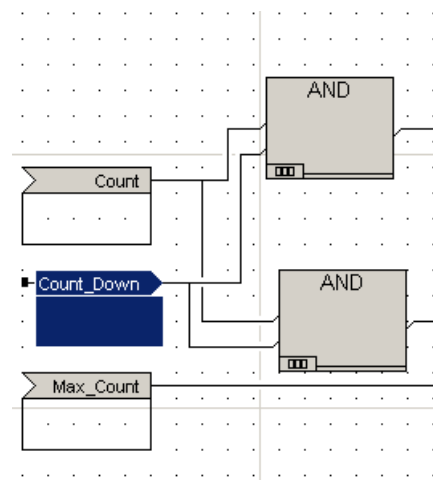


- 4 Double-click the output variable connected to the **INT_REAL** output to display the **Item Properties** screen. Enter **Current_Value** in the Name field. Click **Apply**.

- 5 Confirm **Real** as the Data Type and **Output** as the Variable Type, and then close the screen.

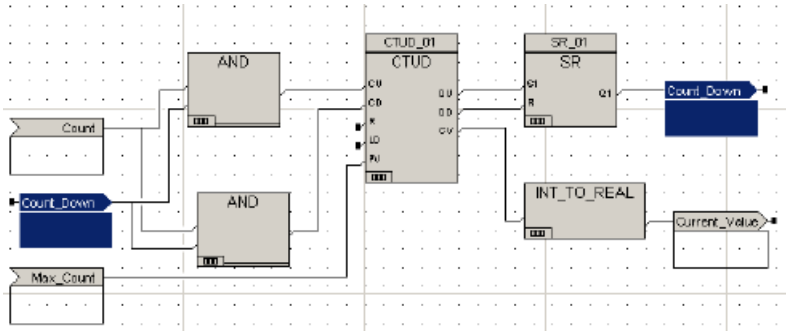


- 6 Click the **Output Variable Tool** button , place an output variable on the logic sheet and connect it to the lower input connector of each of the AND functions.



- 7 Double-click the top output variable to display the Item Properties. Enter **Count_Down** in the **Name** field. Click **Apply**.
- 8 Click the **Declaration** button. Confirm **BOOL** as the Data Type and **Output** as the Variable Type, and then close the screen.

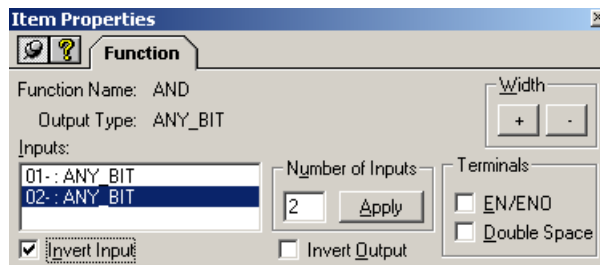
This variable now receives output from the **SR_01** function block and inputs into the **AND** function block.



Invert Input

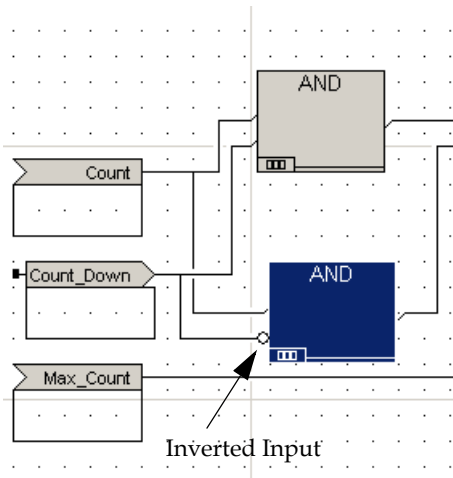
Procedure

- 1 Create a **NOT** operation for the lower **AND** function using the invert feature. Double-click the lower **AND** function. The **Item Properties** screen is displayed.

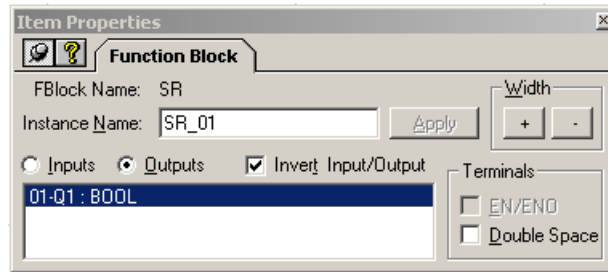


- 2 Click the **02::ANY_BIT** variable in the **Inputs** field. Check **Invert Input**. Click **Apply** and then close the screen.

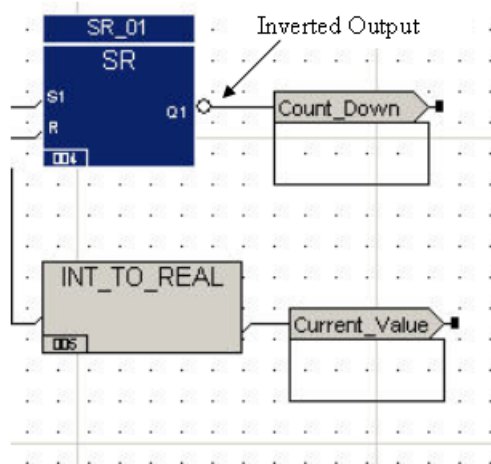
The inverted input symbol appears at the terminal.



- 3 Create a **NOT** operation for the **SR** function using the invert feature. Double-click the **SR** function. The **Item Properties** screen is displayed.



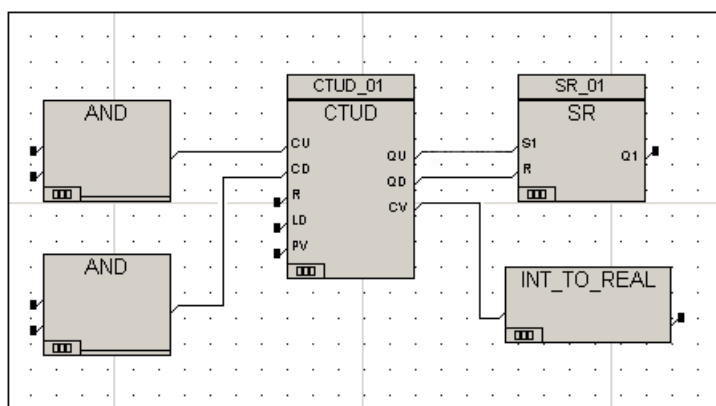
- 4 Click the **Outputs** button, and click **01-Q1:BOOL** in the field. Check **Invert Input/Output**. Click **Apply** and then close the screen.
The inverted input symbol appears at the terminal.



Adding a comment

- 1 Add a comment box to the right of the output variables.
- 2 Enter **While Count is TRUE, this function block will count up to Max_Count from zero, then count down to zero from Max_Count in the text field.**
- 3 Specify the following:

Alignment:	Left
Text Size:	10 point
Border:	Single
- 4 Close the screen. Compile the program. Enter text in the **Comments for Audit Trail** text box, and then click **OK**.
- 5 Save the project.



Lesson 125: Creating a Function Block Using the ST Editor

In this section, you will create a function block using the ST editor. This function block takes the values of the output from the user-derived function block UPDOWN and calculates the average value of the outputs as long as UPDOWN is executing.

This is the completed ST logic for the function block.

```

FUNCTION_BLOCK
(*External Interface *)
VAR_INPUT
    RESET: BOOL;                (* 1=Reset, 0=Calculate *)
    CURRENT_VALUE : REAL;        (* Current Value from UPDOWN *)
END_VAR

VAR_OUTPUT
    STATUS : BOOL;              (* State=> 1 = Resetting, 0=Calculating *)
    XOUT : REAL;                (* Averaged Output *)
END_VAR

VAR
    SAMPLE_CNT : REAL := 0.0;    (* Number of iterations of this
                                   function block *)
    SUM : REAL := 0.0 ;          (* Running Sum *)
END_VAR

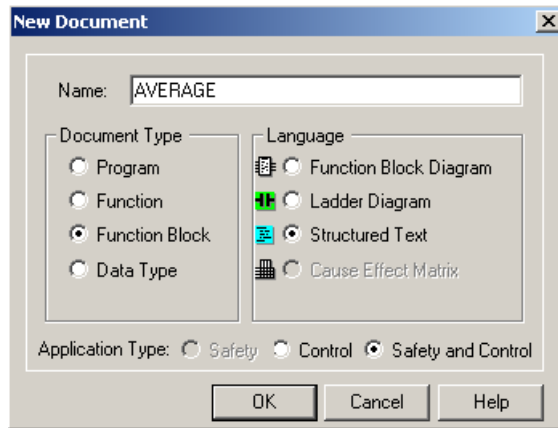
(* Function Block Body *)
IF RESET THEN
    SUM := 0.0 ;
    SAMPLE_CNT :=0.0
ELSE
    SUM := SUM + CURRENT_VALUE;
    SAMPLE_CNT := SAMPLE_CNT + 1.0 ;
    XOUT := SUM / SAMPLE_CNT;
END_IF ;
STATUS : = RESET;

END_FUNCTION_BLOCK

```

Procedure

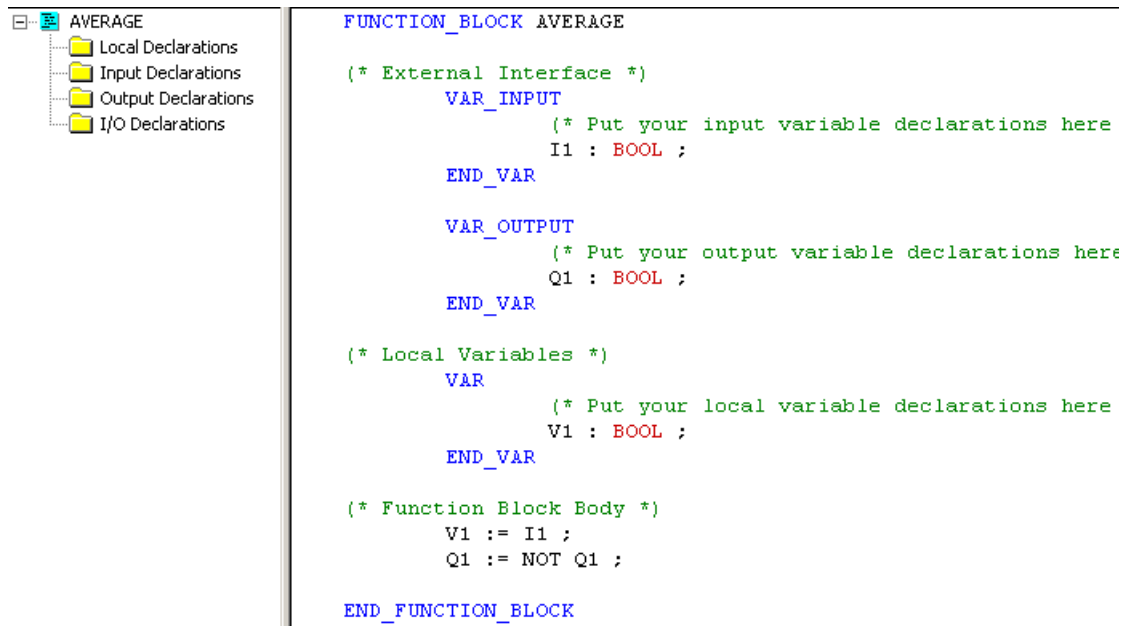
- 1 Create a new document named **Average**.



- 2 Specify the following:

Name: Enter **AVERAGE**
Document Type: Click **Function Block**
Language: Click **Structured Text**
Application Type: Click **Safety and Control**

- 3 Click **OK**. The logic sheet displays the ST template.



- 4 Use the ST editor to create and declare the input variables of the function block.

```
FUNCTION_BLOCK
(*External Interface *)
  VAR_INPUT
    RESET: BOOL;          (* 1=Reset, 0=Calculate)
    CURRENT_VALUE : REAL;  (* Current Value from UPDOWN *)
  END_VAR
```

- 5 Use the ST editor to create and declare the output variables for the function block.

```
  VAR_OUTPUT
    STATUS : BOOL;        (* State=> 1 = Resetting, 0=Calculating *)
    XOUT : REAL;          (* Averaged Output *)
  END_VAR
```


- 6 Use the ST editor to create and declare the variables for the function block in the **Function Block Body** section.

```
  VAR
    SAMPLE_CNT : REAL := 0.0;  (* Number of iterations of this
                                function block *)
    SUM : REAL := 0.0 ;        (* Running Sum *)
  END_VAR
```

- 7 Use the ST editor to create function block logic.

```
(* Function Block Body *)
  IF RESET THEN
    SUM := 0.0 ;
    SAMPLE_CNT :=0.0
  ELSE
    SUM := SUM + CURRENT_VALUE;
    SAMPLE_CNT := SAMPLE_CNT + 1.0 ;
    XOUT := SUM / SAMPLE_CNT;
  END_IF ;
  STATUS := RESET;

END_FUNCTION_BLOCK
```

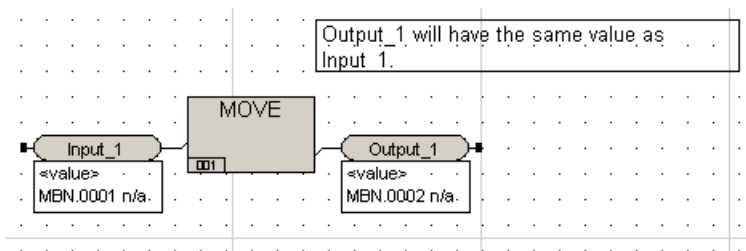
- 8 Click the **Compile** button . Click **OK** to save changes.

Lesson 126: Creating a Move Function

In this lesson, you will create the first network and a function to change the value of one variable to the same value as another variable, using a MOVE function.

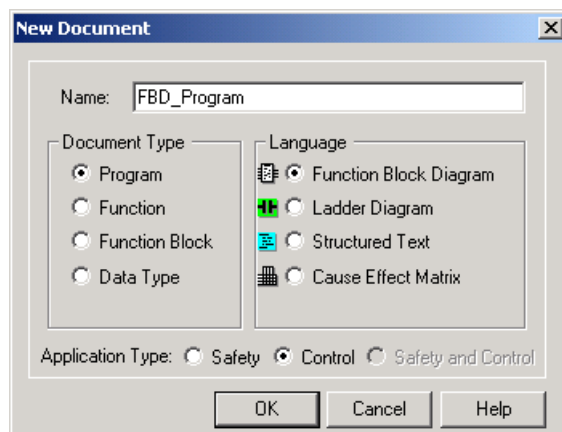
A MOVE function assigns an input value to an output value. For this function, Output_1 will have the same value as Input_1

This is the completed logic for the first network:




Procedure

- 1 Create a new document named **FBD_Program**.



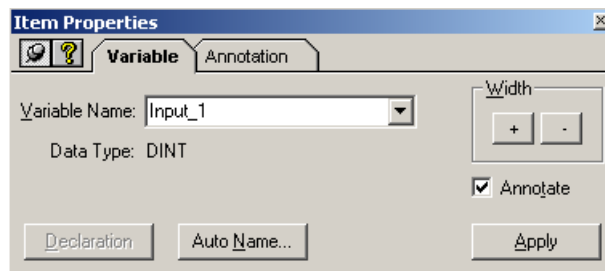
- 2 Specify the following:

Name:	Enter FBD_Program
Document Type:	Click Program
Language:	Click Function Block Diagram
Application Type:	Click Control

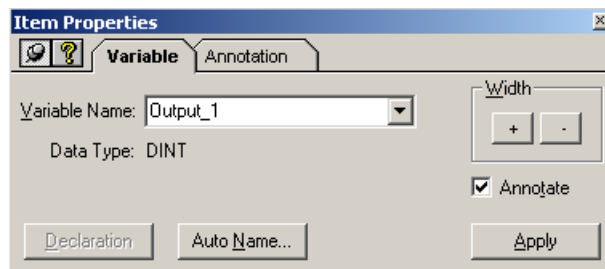
- 3 Click the **Select Function (Block)** button  and place a Move function on the logic sheet.

- 4 Click the **Tagname Tool** button  and connect an input variable to the input terminal of the Move function.

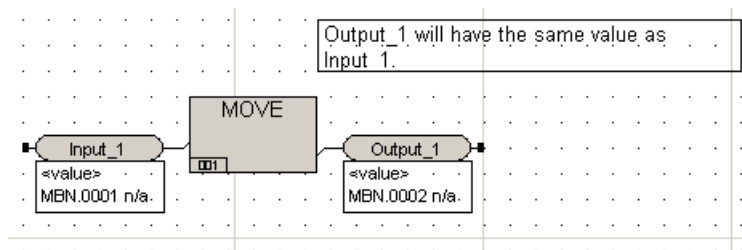
- 5 Double-click the input variable to display the **Item Properties** screen.



- 6 Enter **Input_1** in the **Variable Name** field. Click **Apply**.
- 7 Click the **Declaration** button. Select **BOOL** as the Data Type, click **Apply**, and then close the screen.
- 8 Click the **Tagname Tool** button and connect an output variable to the output terminal of the **Move** function.
- 9 Double-click the output variable to display the **Item Properties** screen.



- 10 Enter **Output_1** in the **Variable Name** field. Click **Apply**.
- 11 Click the **Declaration** button. Select **BOOL** as the Data Type, click **Apply**, and then close the screen.
- 12 Place a comment box on the screen. **Enter Output_1 will have the same value as Input_1** in the text field.
- 13 Click the **Document** menu and then click **Save Program**.



Lesson 127: Creating a Rising Trigger Function Block

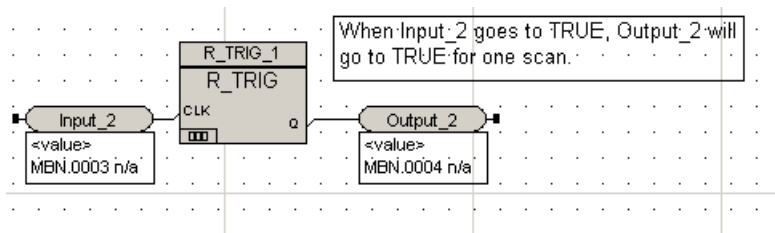
The second network uses a Rising Trigger function block to change the status of one variable to that of another variable for one scan.

The Rising Trigger function block (R_TRIG) sets the output Q on the rising edge of the CLK input. A rising edge is a change from false to true (0 to 1).

The output Q is true if the input CLK was false during the previous evaluation but true during the current evaluation of the function block instance; otherwise, the output Q is false.

For this function block, when Input_2 goes to TRUE, Output_2 will go to TRUE for one scan.



This is the completed logic:

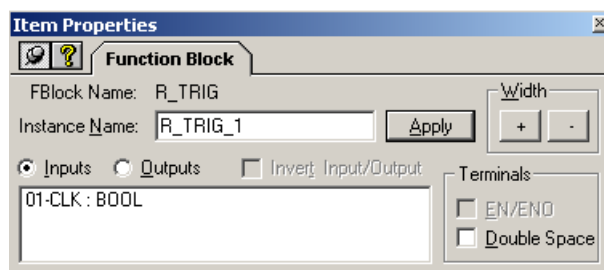


In this lesson you will:

- Add horizontal dividers to the logic sheet.
- Place a Rising Trigger function block in the second network.


Procedure

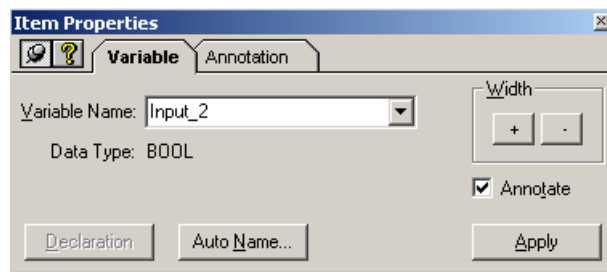
- 1 Click the **Horizontal Network Divider** button  and place a network divider on the logic sheet below the **Move** function.
- 2 Repeat the previous step to create four additional networks, leaving logic workspace in between each network.
- 3 Click the **Select Function (Block)** button  and place a **Rising Trigger** (R_TRIG) function block on the logic sheet.
- 4 Double-click the R_TRIG function block to display the **Item Properties** screen.



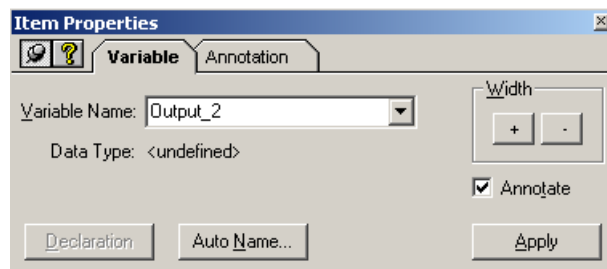
- 5 Enter R_TRIG_1 in the **Instance Name** field. Close the screen.

You can create and use multiple instances of the same function or function block in your program with each instance performing the same operations on different sets of variables.

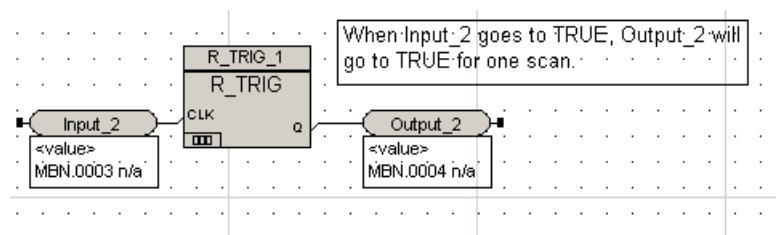
- 6 Use the **Tagname Tool** button  and connect an input variable to the input terminal of the **R_TRIG** function block.
- 7 Double-click the input variable to display the **Item Properties** screen.



- 8 Enter **Input_2** in the **Variable Name** field. Click **Apply**.
- 9 Double-click the **Declaration** tab. Confirm **BOOL** as the Data Type, and then close the screen.
- 10 Click the **Tagname Tool** button and connect an output variable to the output terminal of the **R_TRIG** function block.
- 11 Double-click the output variable to display the **Item Properties** screen.



- 12 Enter **Output_2** in the **Variable Name** field. Click **Apply**.
- 13 Click the **Declaration** tab. Confirm **BOOL** as the Data Type, and then close the screen.
- 14 Place a comment box on the logic sheet. Enter **When Input_2 goes to TRUE, Output_2 will go to TRUE for one scan** in the **Comment** text field.
- 15 Click the **Document** menu and then save the project.

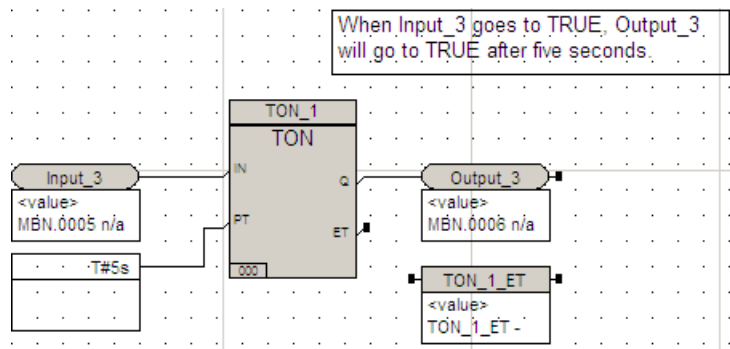


Lesson 128: Creating a Timer


In this lesson you will create a Timer (TON) function block to change the status of one variable to that of another input variable after five seconds.

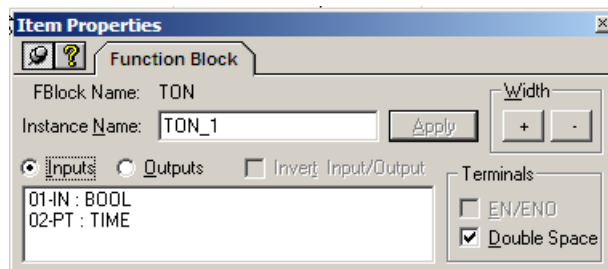
For this function block, when Input_3 goes to TRUE, Output_3 will go to TRUE after five seconds.


This is the completed logic:



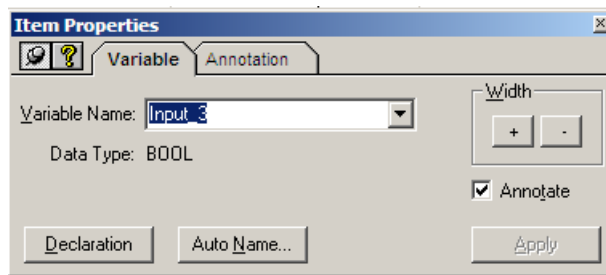
Procedure

- 1 Click the **Select Function (Block)** button  and place a **Timer On (TON)** function block on the logic sheet.
- 2 Double-click the function block. The **Item Properties** screen is displayed.

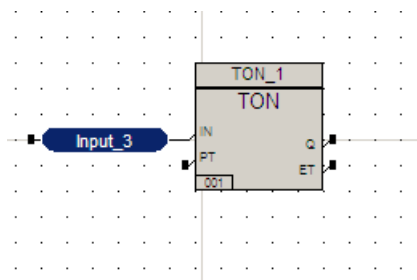


- 3 Enter **TON_1** in the **Instance Name** field. You are now using an instance of the **TON** function block called **TON_1**.
- 4 Click the **Tagname Tool** button  and connect an input variable to the input terminal of the **TON** function block.

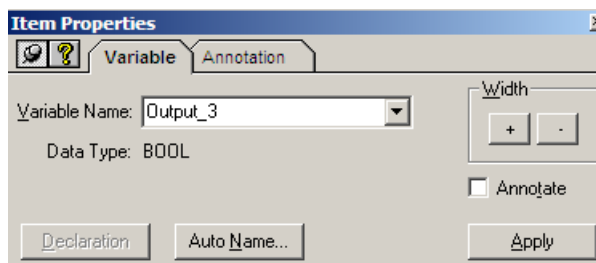
- 5 Double-click the input variable to display the **Item Properties** screen.



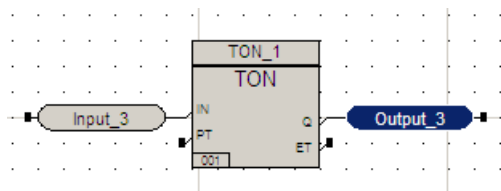
- 6 Enter **Input_3** in the **Variable Name** field and click **Apply**.
 7 Click the **Declaration** tab. Confirm **BOOL** as the **Data Type**. Close the screen.



- 8 Click the **Tagname** tool button and connect an output variable to the **Q** output terminal of the **TON** function block.
 9 Double-click output variable to display the **Item Properties** screen.

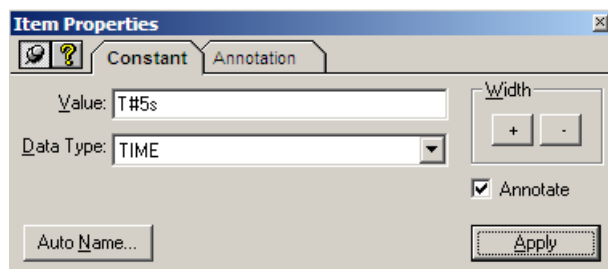


- 10 Enter **Output_3** and click **Apply**.
 11 Click the **Declaration** tab. Confirm **BOOL** as the **Data Type**. Close the screen.

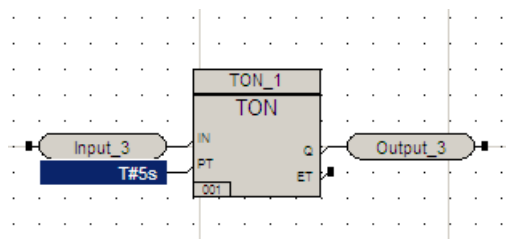



- 12 Use the **Constant** tool button  and connect a constant to the **PT** terminal of the **TON** function block.

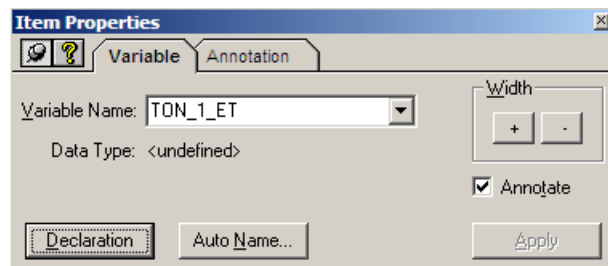
- 13 Double-click the constant variable to display the **Item Properties** screen.



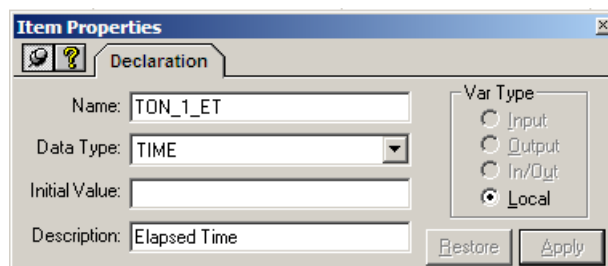
- 14 Enter **T#5s** in the **Value** field. This sets the Pulse Time of the timer to 5 second intervals. Click the Data Type down arrow and select **TIME** from the list. Click **Apply**, and then close the screen.



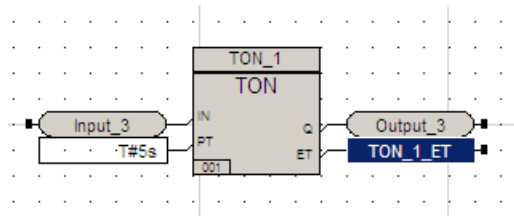
- 15 Use the **Local Variable** button  and connect a variable to the **ET** terminal of the **TON** function block.
- 16 Double-click the local variable to display the **Item Properties** screen.



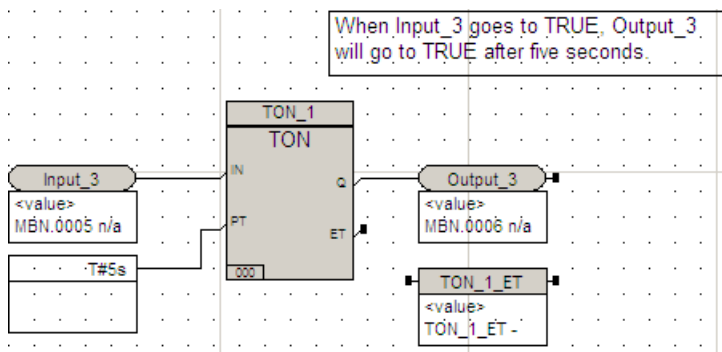
- 17 Enter **TON_1_ET** in the **Variable Name** field. Click **Apply**.



- 18 Enter **Elapsed Time** in the **Description** field. Confirm **TIME** as the **Data Type** and **Local** as the **Variable Type**. Click **Apply** and then close the **Item Properties** screen.



- 19 Place a comment box on the logic sheet. Enter **When Input_3 goes to TRUE, Output_3 will go to TRUE after five seconds** in the Comment text field.
- 20 Click the **Document** menu, click **Save Program**.
- 21 Click the **File** menu and then click **Save Project**. Enter any comments in the **Comments for Audit Trail** text box, and then click **OK**.

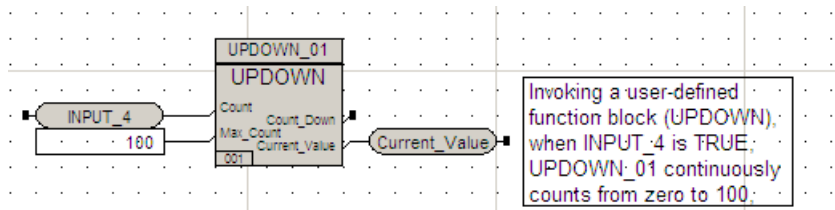


Lesson 129: Invoking an UPDOWN Function Block


In this lesson, you will create a fourth network that invokes the UPDOWN function block created earlier. UPDOWN takes an input value and counts up to 100, then counts down to zero, displaying the current value.

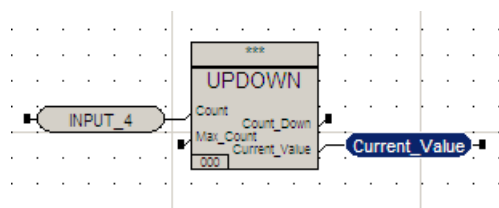
For this function block, when INPUT_4 is TRUE, UPDOWN_01 continuously counts from zero to 100, then from 100 to zero.


This is the completed logic:

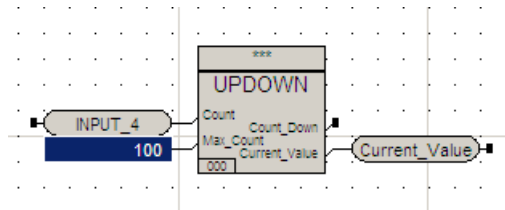


Procedure

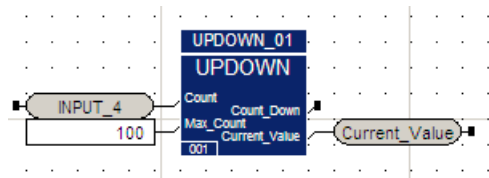
- 1 Select **UPDOWN** (user-defined function block created in a previous lesson) and place the function block in the fourth network.
- 2 Click the **Tagname** tool button  and connect a variable to the **COUNT** input terminal of the **UPDOWN** function block.
- 3 Double-click the variable. Enter **Input_4** in the **Variable Name** field. Click **Apply**.
- 4 Confirm **BOOL** as the **Data Type**, and then close the screen.
- 5 Click the **Tagname** tool button and connect a variable to the **Current Value** output terminal of the **UPDOWN** function block.
- 6 Double-click the variable. Enter **Current_Value** in the **Variable Name** field. Click **Apply**.
- 7 Confirm **REAL** as the **Data Type**, and then close the screen.



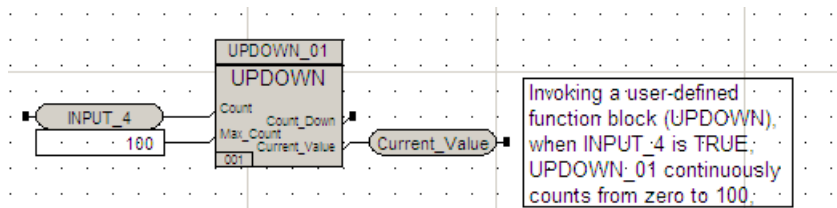
- 8 Use the **Constant** tool button  and connect a constant to the **Max_Count** terminal of the **UPDOWN** function block.
- 9 Double-click the constant to display the **Item Properties** screen.
- 10 Enter **100** in the **Value** field. Select **INT** as the **Data Type**. Click **Apply** and then close the screen.



- 11 Double-click the **UPDOWN** function block to display the **Item Properties** screen.
- 12 Enter **UPDOWN_01** in the **Instance Name** field. Click **Apply**, and then close the screen.
You are now using an instance of the user-defined **UPDOWN** function block called **UPDOWN_01**.



- 13 Add a **Comment** box to the right of the output variable. Enter **Invoking a user-derived function block (UPDOWN), when INPUT_4 is TRUE, UPDOWN_01 continuously counts from zero to 100, then from 100 to zero** in the **Comment** text field.
- 14 Close the screen.



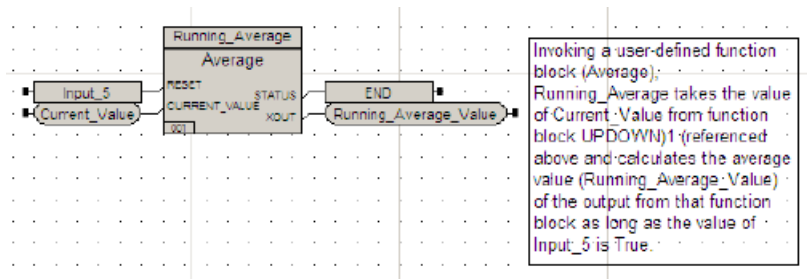
- 15 Compile the logic. Click **Yes** when asked to save the changes.
- 16 Save the project.

Lesson 130: Invoking an AVERAGE Function Block


In this lesson, you will add a network that invokes the AVERAGE function block created in a previous lesson using Structured Text. The AVERAGE function block calculates the average value of the output value of UPDOWN as it countup to 100, then counts down to zero.

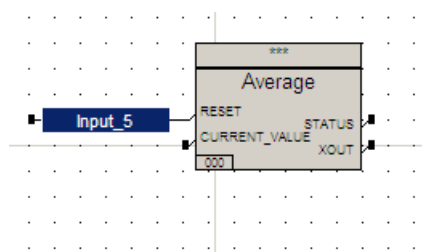
For this function block, Running_Average takes the value of Current_Value from function block UPDOWN_1 in Network 4 and calculates the average value (Running_Average_Value) of the output from that function block as long as the value of Input_5 is True.


This is the completed logic:



Procedure

- 1 Select AVERAGE (user-defined function block created in a previous lesson) and place the function block in the fifth network on the logic sheet.
- 2 Click the **Local Variable** tool button  and connect a variable to the **Reset** input terminal of **Average** function block.
- 3 Double-click the variable. Enter **Input_5** in the **Variable Name** field. Click **Apply**.
- 4 Click the **Declaration** tab. Confirm **BOOL** as the **Data Type** and **Local** as the **Variable Type**, and then close the screen.

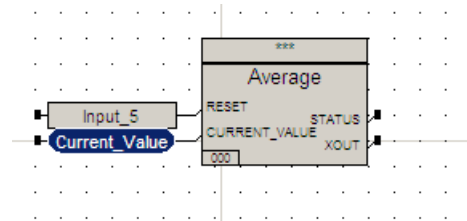


- 5 Use the **Tagname** tool button  and connect a variable to the **Current Value** input terminal of the **Average** function block.

- 6 Double-click the variable. Enter **Current_Value** in the **Variable Name** field. Click **Apply**.

Current_Value is the output from the **UPDOWN** function block in network Network Four.

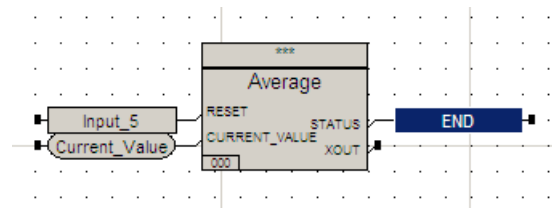
- 7 Click the **Declaration** tab. Confirm **REAL** as the **Data Type** and then close the screen.



- 8 Use the **Local Variable** tool button and connect a variable to the **Status** output terminal of the **Average** function block.

- 9 Double-click the variable. Enter **END** in the **Variable Name** field. Click **Apply**.

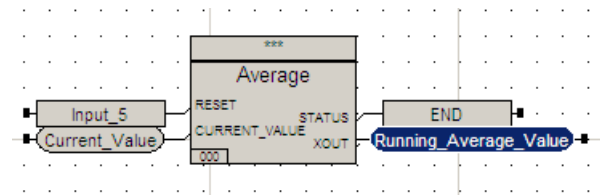
Click the **Declaration** tab. Confirm **BOOL** as the **Data Type** and **Local** as the **Variable Type**, and then close the screen.



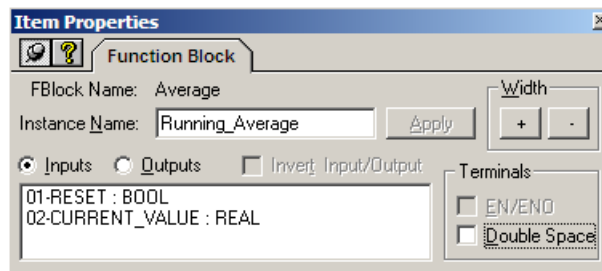
- 10 Use the **Tagname** tool button and connect a variable to the **XOUT** terminal of the **Average** function block.
- 11 Double-click the variable. Enter **Running_Average_Value** in the **Variable Name** field. Click **Apply**.

Running_Average_Value is the average value of the output from the **UPDOWN** function block in Network Four.

- 12 Click the **Declaration** tab. Confirm **REAL** as the **Data Type**, and then close the screen.



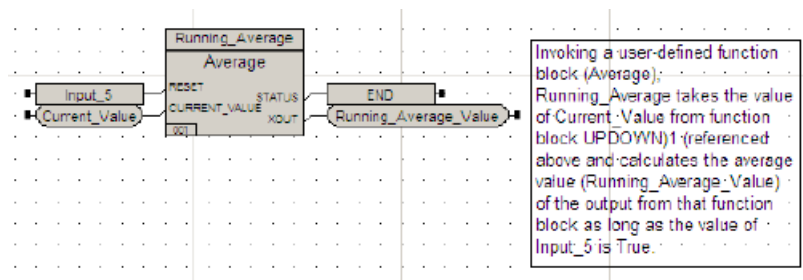
- 13 Double-click the **Average** function block to display the **Item Properties** screen.



- 14 Enter **Running_Average** in the **Instance Name** field. Click **Apply**, and then close the screen.

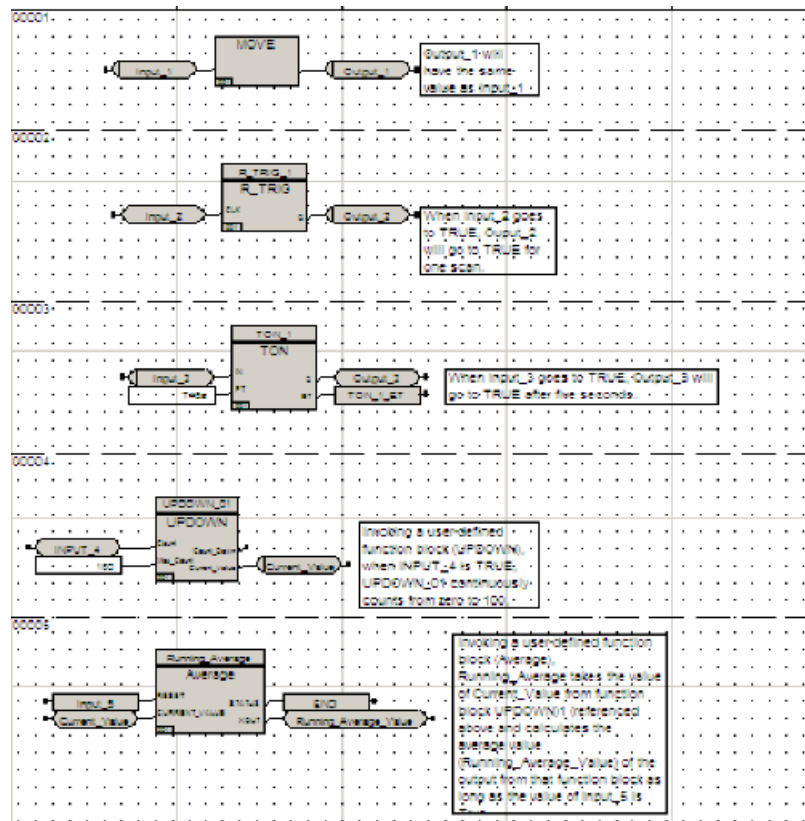
You are now using an instance of the user-defined **Average** function block called **Running_Average**.

- 15 Add a **Comment** box to the right of the output variable.
- 16 In the Comment text box, enter **Invoking a user-defined function block (Average), Running_Average takes the value of Current_Value from function block UPDOWN_1 (referenced above and calculates the average value (Running_Average_Value) of the output from that function block as long as the value of Input_5 is True.**
- 17 Close the screen.



- 18 Compile the logic. Click **Yes** to save the changes.
- 19 Save the project.

You have now completed the logic for the project.



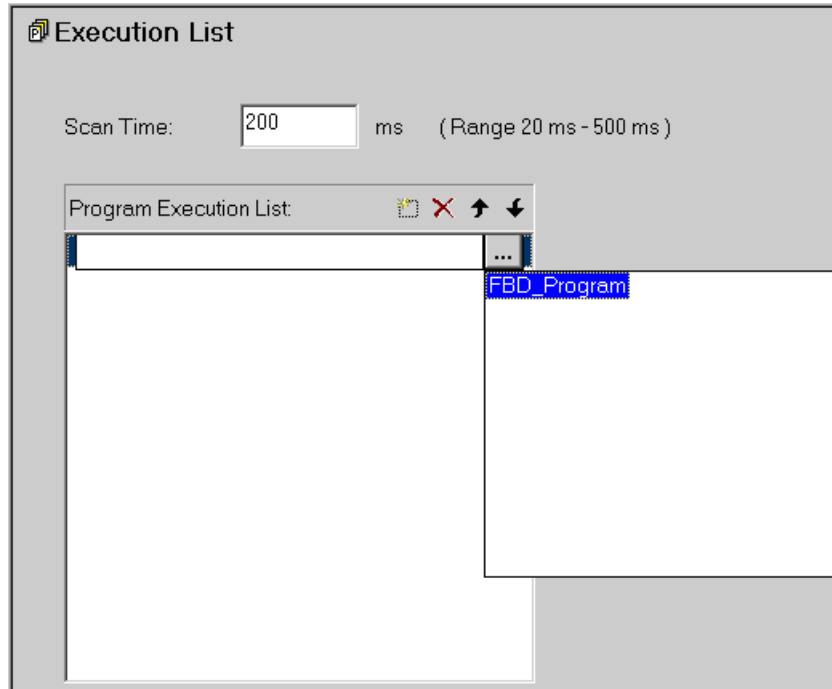
Lesson 131: Building the Application

In this lesson you will:

- Build the application.
- Specify the program order and scan time.

Procedure

- 1 On the **Application** tree, double-click **Implementation**. The **Execution List** is displayed.



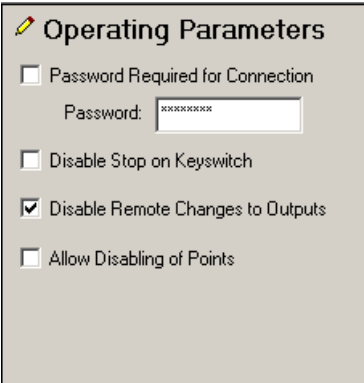
- 2 Set the scan time to **200** milliseconds.
- 3 Click the New (Insert) button .
- 4 Select the program from the Execution List.
- 5 On the **Project** menu, click **Build Application**. Click **OK** 'to save changes.
- 6 Check the Message View for errors. If there are no errors, the build is successful.
- 7 If there are errors, click each error message to see the location of the error. Correct the errors, and then rebuild the application.

Lesson 132: Configuring the Project

In this lesson, you will configure the project by setting operating parameters.

Procedure

- 1 Click the **Controller** tab to display the Controller Workspace.
- 2 Expand the **Tricon Controller** tree and double-click **Configuration** to display the **Operating Parameters**.



- 3 Select the operating parameters for your project. If you do not want to change the default settings, skip this step.

Property	Action
Password Required for Connection	Check to require a password to be used to connect from TriStation 1131 to the controller. If checked, enter a password. The default is unchecked.
Disable Stop on Keyswitch	Check to prohibit the STOP keyswitch from halting the application running on the controller. The default is unchecked.
Disable Remote Changes to Outputs	Check to restrict external devices, such as a DCS, write to output tagnames in the TriStation 1131 application. The default is unchecked.
Allow Disabling of Points	Check to allow points to be disabled from TriStation 1131 . The default is checked.

- 4 Save the project.

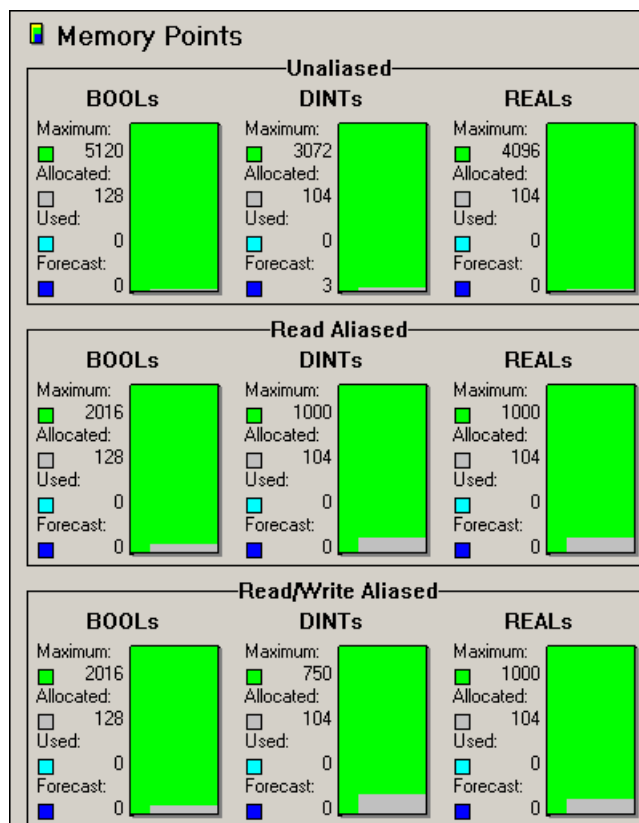
Lesson 133: Memory Allocation

With each successful build, **TriStation 1131** automatically allocates memory for each of the memory points, input points, and output points. You can change these allocations at any time before downloading the application in the Download All mode.

In this lesson, you will view the current memory allocation for your project.

Procedure

- 1 Click the **Controller** tab.
- 2 Expand the **Tricon Controller** tree, double-click **Configuration**, and then expand **Memory Allocation**. The Memory Allocation for the project is displayed.



Because the default memory allocations are adequate for the project, none of the memory allocations need to be changed.

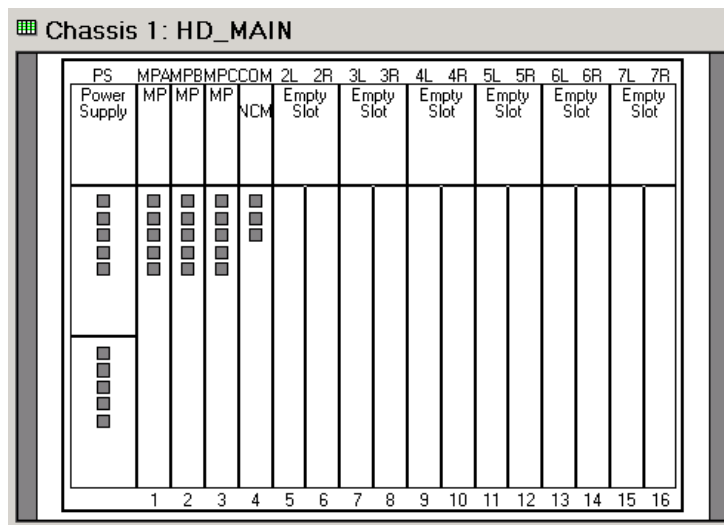
Lesson 134: Allocating Hardware

In this lesson you will:

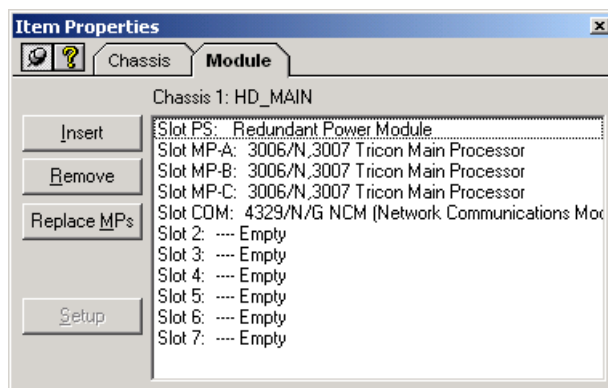
- Determine Tricon Power Usage.
- Insert Tricon Modules.

Procedure

- 1 Click the **Controller** tab.
- 2 Expand the **Controller** tree, double-click **Configuration**, and then expand **Hardware Allocation**. The **Chassis Power Usage** screen is displayed.
- 3 Verify that there is sufficient power for the project.
- 4 Click **Chassis 1: HD_Main**. The chassis screen is displayed.



- 5 Double-click **Chassis 1: HD_Main**. The **Item Properties** screen is displayed.



- 6 Click **Insert** to insert modules into the chassis as follows:

Slot 2	Model 3503E Discrete Input, 24V, 32 points
Slot 3	Model 3604 Discrete Output, 24V, 16 points



Lesson 135: Running Program Logic

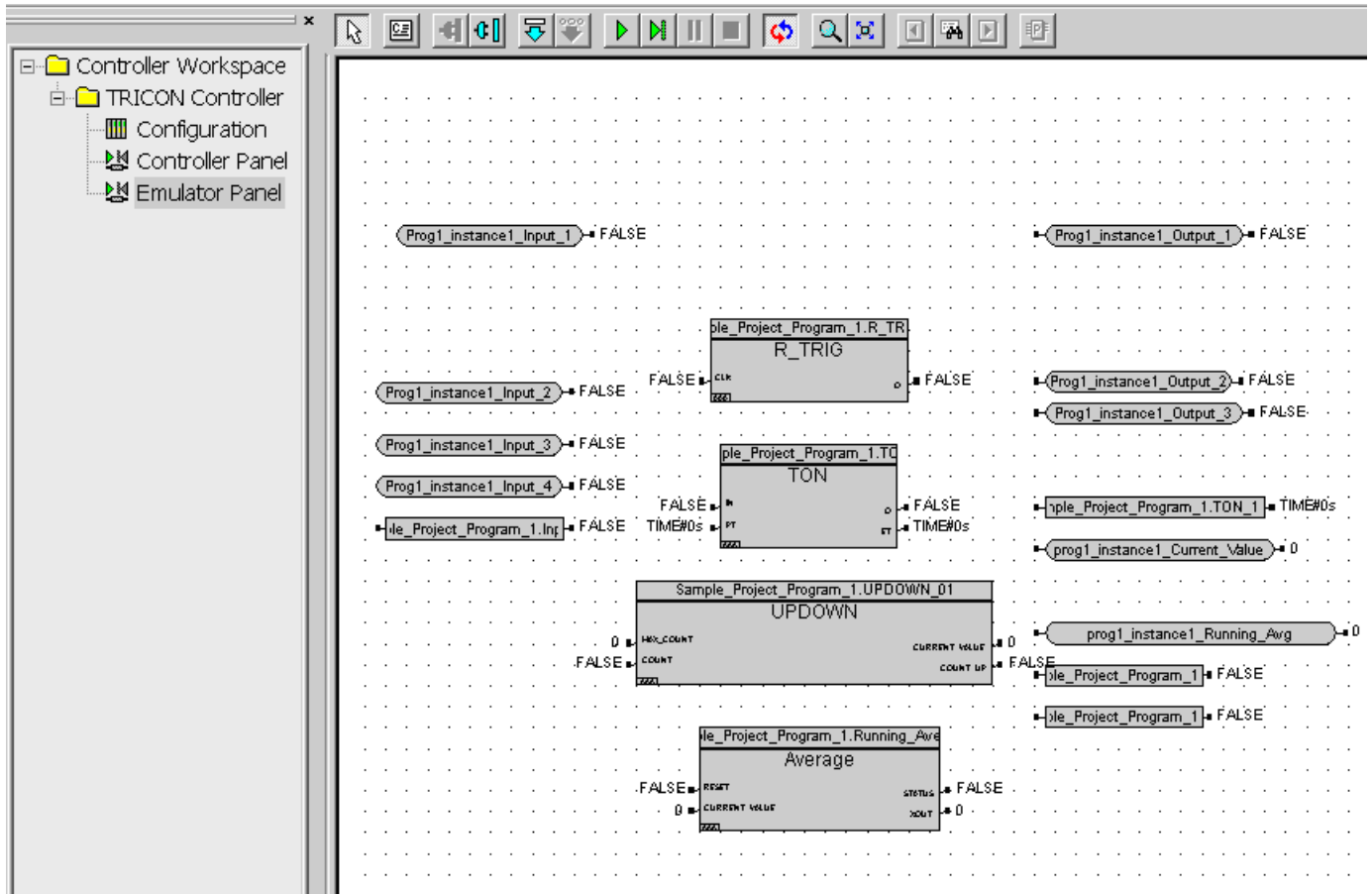
In this lesson, you will:

- Download the application to the controller.
- Run the program logic.

Procedure

- 1 Run an Install Check to verify that **TriStation 1131** is installed on the PC and no associated files are corrupted.
From the **Start** menu, select **Programs, Triconex**, and then **Install Check**.
- 2 Click **Run**, and then click **Display Details**.
- 3 Verify that the program is validated, and then close the details screen and Install Check.
- 4 Expand the **TRICON Controller** tree, and double-click **Controller Panel**.
- 5 On the **Commands** menu, click **Connect**.
- 6 Enter the connection password if required.
- 7 On the **Commands** menu, click **Download All**, and then click **Run**.
- 8 Drag the **P1_Input_1** and **P1Output_1** variables to the sheet. P1 refers to point instance.
- 9 Double-click the **PI_Input_1** input variable. The **Item Properties** screen is displayed.
- 10 Change the current value of **P1_I1.Input_1** to True by either entering **True** or **1** and clicking the **Confirm** button. This changes the current value of P1_I1.Input_1 to True.
- 11 Close the **Item Properties** screen.
- 12 Click the **Single Step** button.
The program logic executes and the value of P1_11.Output1 changes to True.
- 13 Continue testing each network by placing the logic on the sheet and changing the values, as follows:
 - Network 2: Change the current value of **P1_11.Input_2** to **True**.
 - Network 3: Change the current value of **P1_I1.Input_3** to **True**.
- 14 Double-click the **TON** function block (P1_I1.TON_1).
- 15 Select **PT** from the variable list.
- 16 Enter **TIME#5s** in the **Set Value** field
- 17 Network 4: Change the current value of **P1_I1.Input_4** to **True**.
- 18 Double-click the **UPDOWN_01** function block.
- 19 Highlight **Max_Count** in the variable list.
- 20 Change the set value to **100**.
- 21 Network 5: Change the current value of **P1_I1.Input_5** to **False**.
- 22 On the **Commands** menu, click the **Run** button. The program logic executes.

- 23 Click the **Pause** button  to pause the execution. Click the **Halt** button  to stop the execution.



11

TriStation 1131 Advanced Programming Lessons

Tank Farm 398

Lesson 136: Tank Farm

Tank Farm 140

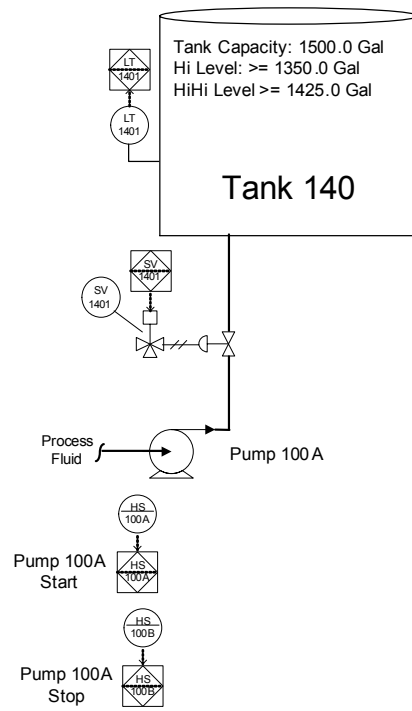
For this lesson, you will:

- Create a new project named Tank_Farm.
- Create a new FBD program named F1xx_Tanks.
- Develop logic to scale LT1401 Level Transmitter.
- Develop alarm logic for low and high tank levels.
- Develop logic to operate Pump 100A.
- Develop logic to operate Solenoid Valve SV1401.
- Place a timer function between Pump 100A and Solenoid Valve SV1401.

Tank Farm 140 Program Logic Guidelines:

- Scale Level Transmitter LT1401 input. Create logic to alarm if tank level rises above low and high levels.
- Create logic to automatically operate pump when SV1401 is energized.
- Fill tank to some level \leq low level, but prevent filling tank \geq high level.
- Utilize AIN, OR, AND, and GE functions.

Completed Logic



Lesson 137: Tank Farm

Tank Farm 150 and 160

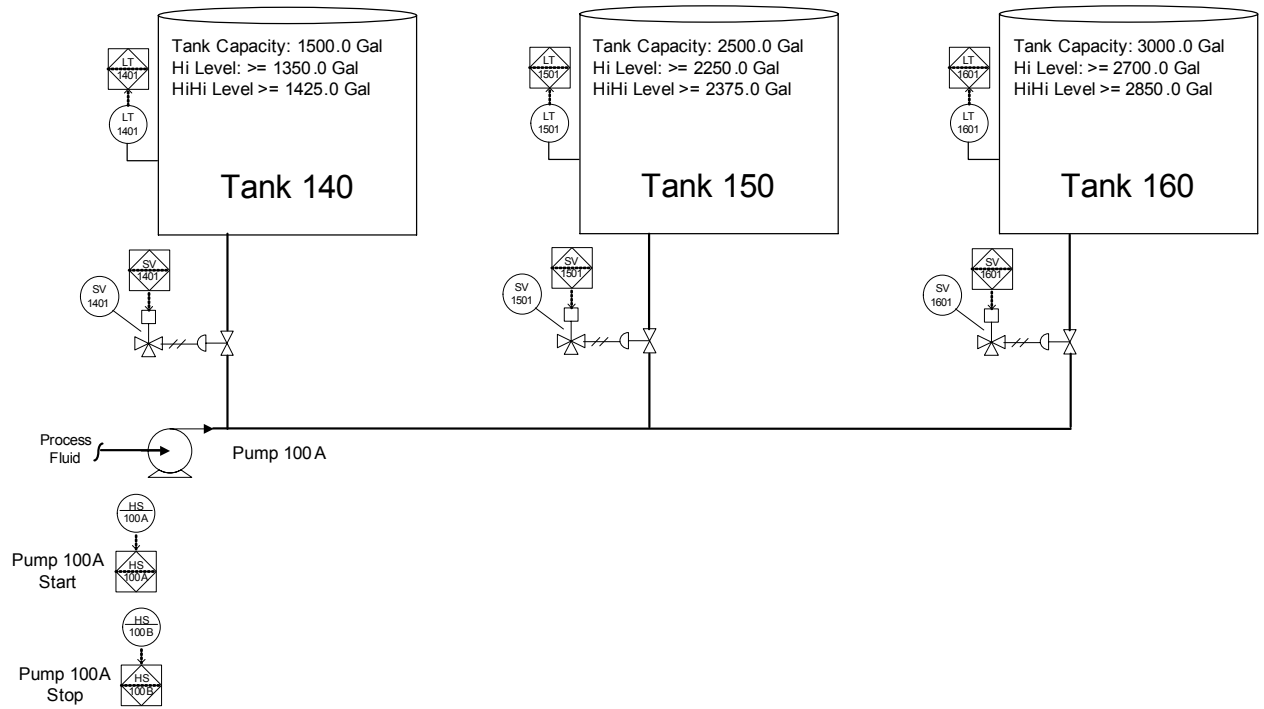
For this lesson, you will:

- Edit the Tank_Farm project.
- Create a new FBD program named F1xx_Tanks.
- Develop logic to scale LT1401 Level Transmitter.
- Develop alarm logic for low and high tank levels.
- Develop logic to operate Pump 100A.
- Develop logic to operate Solenoid Valve SV1401.

Tank Farm 150 and 160 Program Logic Guidelines:

- Create identical logic as Tank 140 for Tanks 150 and 160.
- Modify pump operation logic to permit filling only one tank at a time.
- Create logic to continuously display the average level of all three tanks.
- Utilize AIN, OR, AND, GE, ADD, and DIV functions.

Completed Logic



Lesson 138: Tank Farm

Tank Farm 170

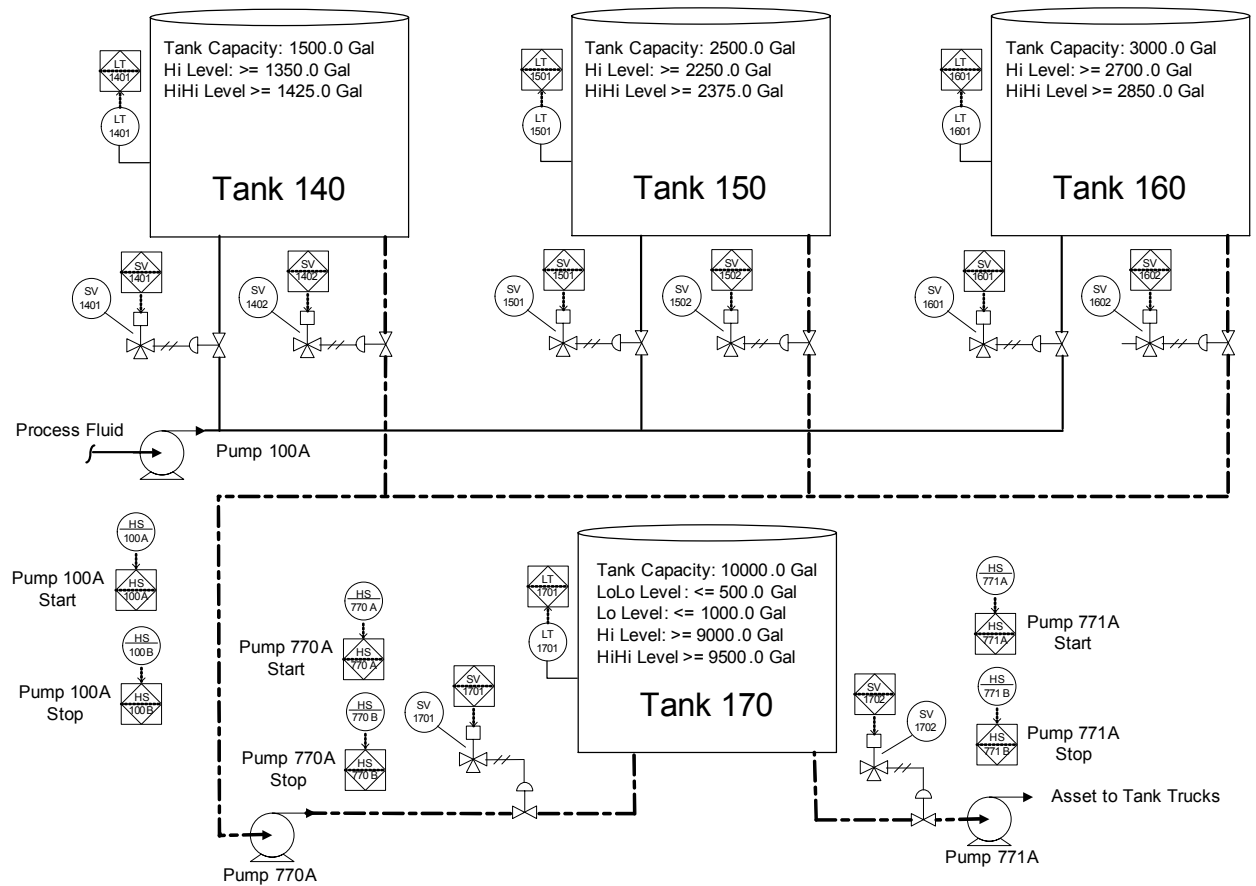
For this lesson, you will:

- Edit the Tank_Farm project.
- Edit the F1xx_Tanks program.
- Add Tank 170 level transmitter & alarm logic.
- Add Tank 170 solenoid logic.
- Add Pump 770A and 771A logic.

Tank Farm 170 Program Logic Guidelines:

- Create the same tank operation logic for Tank 170.
- Create logic to fill Tank 170 from Tank 140, 150, and 160 simultaneously. Automatically fill tank if level is \leq LoLo level. If any tank is being filled, it cannot be used to fill Tank 170.
- Automatically fill tank if level is \leq LoLo level. If any tank is being filled, it cannot be used to fill Tank 170.

Completed Logic



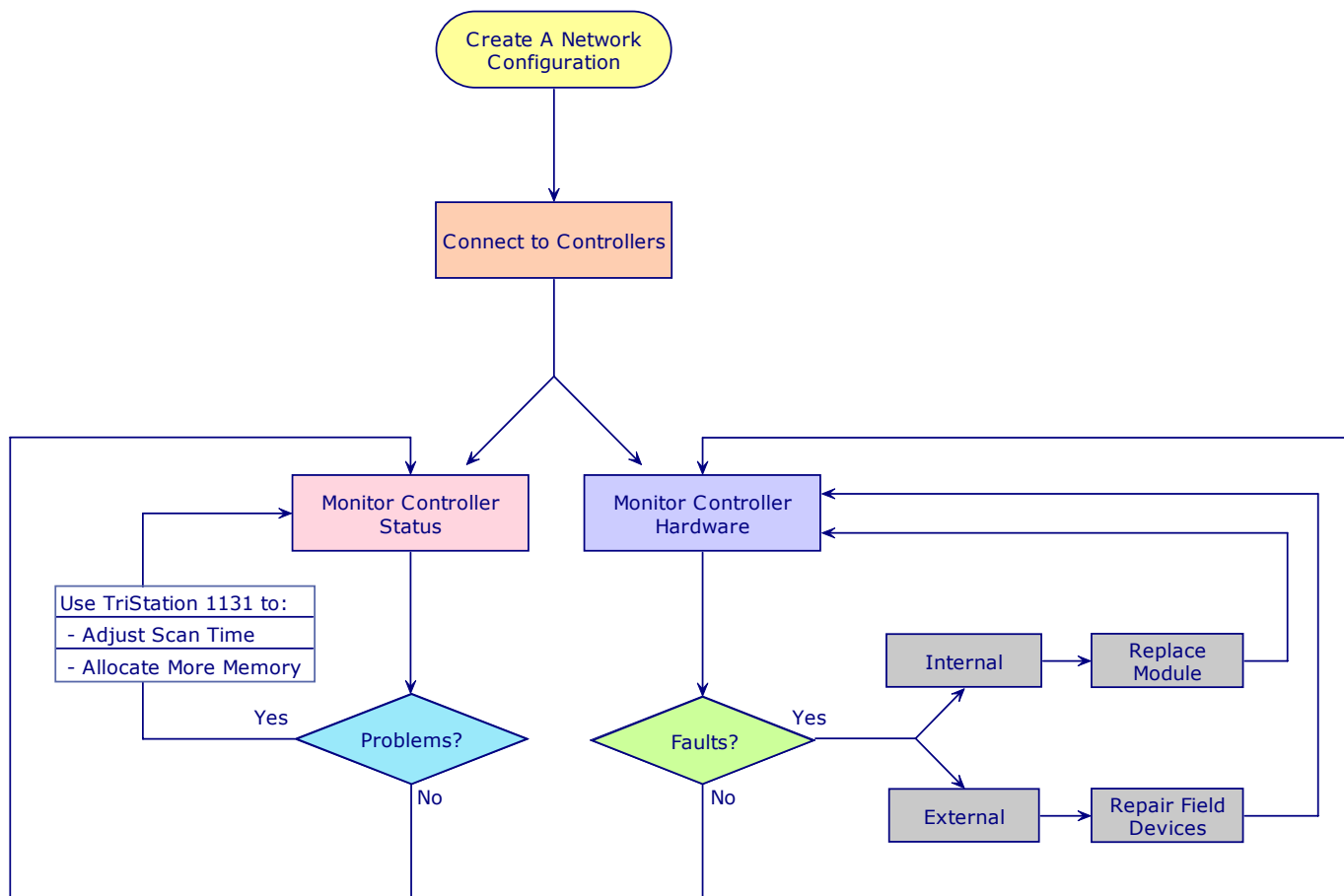


Diagnostic Monitor

Steps for Diagnostic Monitoring	406
Installing and Starting the Diagnostic Monitor	407
Setting Up a Network Configuration	408
Monitoring Controller Hardware	412
Monitoring Controller Status	425

Steps for Diagnostic Monitoring

This figure shows the main steps for using the **TriStation 1131** Diagnostic Monitor software to monitor controllers on a network.



Installing and Starting the Diagnostic Monitor

The Diagnostic Monitor is automatically installed when the **TriStation 1131** Developer's Workbench is installed on a PC. You can use the Diagnostic Monitor independently of any **TriStation 1131** project, but you must install the **TriStation** software to access it.

PC requirements for the Diagnostic Monitor, such as a Network Interface Card (NIC) and TCP/IP protocol, are the same as for **TriStation 1131**.

For installation instructions, see the *TriStation 1131™ Developer's Guide*.

To start the Diagnostic Monitor, navigate to the Triconex shortcut, and then click Diagnostic Monitor.

Setting Up a Network Configuration

A network configuration is an XML file which includes the Tricon and Trident controllers you want to monitor. You can create network configurations for Ethernet networks, and for serial links (for Tricon). You can add controllers to a network configuration, and edit their properties as often as needed. To begin monitoring controllers, you must connect the Diagnostic PC to the network and open each controller in the network configuration.

Topics include:

- [Types of Network Configurations \(page 408\)](#)
- [Creating or Changing a Network Configuration \(page 409\)](#)
- [Connecting a Node to a Network \(page 411\)](#)

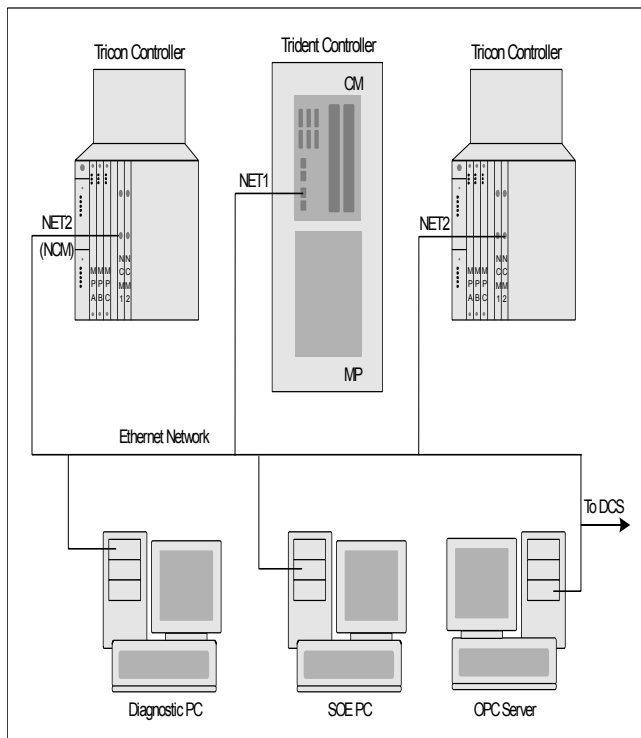
Types of Network Configurations

A network configuration can include Triconex controllers on Ethernet networks, and on serial links (for Tricon).

Ethernet Network Configurations

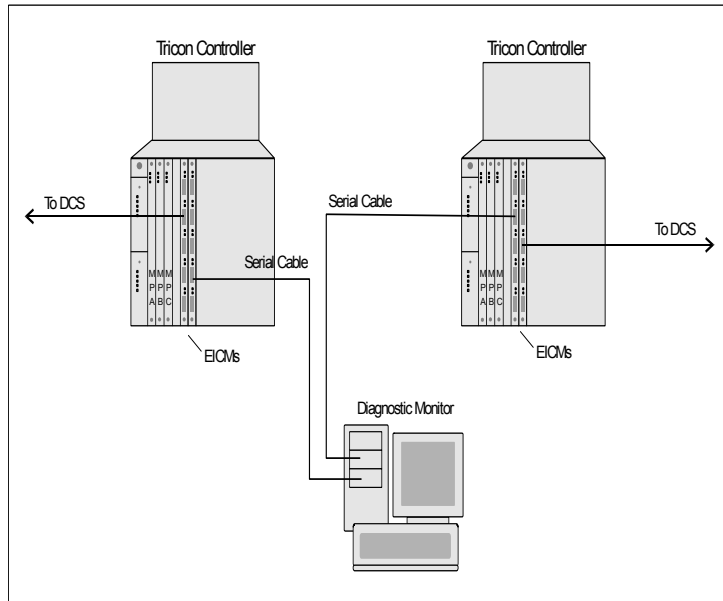
For Ethernet networks, a network configuration can include controllers on one network, or on multiple networks, assuming the networks are connected and set up for communication.

This figure shows a typical network of Triconex controllers which can be specified in a network configuration for diagnostic monitoring.



Serial Links

For Tricon, a network configuration can include controllers on serial links which use Modbus communication. This figure shows Tricon controllers on a typical multi-point serial link.



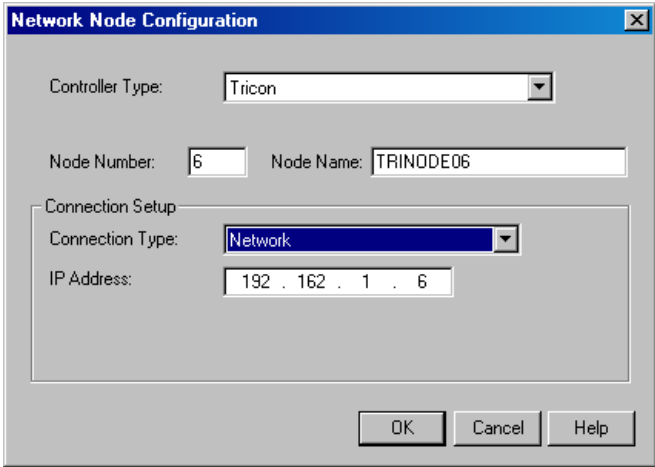
Creating or Changing a Network Configuration

This procedure explains how to create or change a network configuration of controllers to be monitored. Before starting, you should know the settings of these properties for each controller to be included in the network configuration:

- Node name and node number
- IP address (for an Ethernet connection)
- Type of **TriStation** connection

Procedure

- 1 On the **File** menu, click **New Network Configuration** or **Open Network Configuration**.
- 2 Do either of these:
 - To add a node, right-click **Network Nodes** at the top of the tree, and then click **Add**.
 - To edit a node, right-click the **node name** on the tree, and then click **Edit**.



- 3 Specify these properties on the **Network Node Configuration** screen.
- These settings must match the settings on the **TriStation Communication** screen for the application running in the controller.

Property	Action
Controller Type	Select Tricon or Trident.
Node Number	Enter the node number of the controller.
Node Name	Enter the node name of the controller.
Connection	For Tricon, click one of these options: <ul style="list-style-type: none">• Network for an Ethernet (TCP/IP) connection• Serial for a serial (Modbus) connection For Trident, click one of these options: <ul style="list-style-type: none">• Main Processor for an Ethernet (DLC) connection• Network for an Ethernet (TCP/IP) connection
IP Address	Specify the IP address of the controller to be monitored.
Serial Port	For a Tricon serial connection, specify the COM port on the Diagnostic PC which is connected to the controller.
MP Connection	For Trident, click the MP which is connected to the TriStation PC.
NIC Index	For Trident, enter the index position of the network interface card in the Diagnostic PC.

- 4 Repeat [step 2](#) and [step 3](#) for each node you want to add or change.
- 5 On the **File** menu, click **Save Network Configuration**, and save the **XML** file to a new name and destination, or the same name and destination.

Connecting a Node to a Network

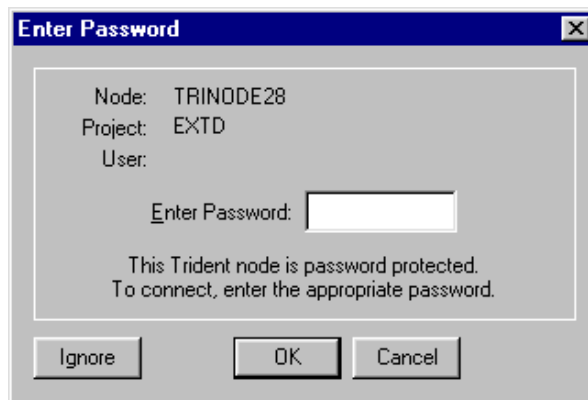
This procedure explains how to connect a node in a network configuration to a network.

Before starting, the Diagnostic PC must be connected to a network or serial link with an appropriate cable. In addition, you must have created a network configuration and added node specifications to it.

Procedure

- 1 On the **File** menu, click **Open Network Configuration** if a network configuration is not already open.
- 2 Right-click the name of a **node** on the tree, and then click **Open**.

The node connects automatically to the network, or requests a password if one was specified for the application running in the controller.



- 3 If the node cannot connect to the network, right-click it, and then click the **Edit** command. Examine the properties on the **Network Node Configuration** screen, change any that are incorrect, and then repeat [step 2](#).
- 4 If the node disconnects from the network, click the **node name** on the tree, and on the **Commands** menu, click **Connect**.

Connecting in Read-Only Mode

If you enter a password that does not match the one specified for the application running in the controller, the node connects to the network in read-only mode. These operations are not allowed:

- Clearing module faults
- Collecting system diagnostic events
- Enabling and disabling Output Voter Diagnostics (Tricon)

Monitoring Controller Hardware

The Diagnostic Monitor allows you to identify alarms on Tricon chassis and Trident IOPs, and faults on power supplies, modules, and points. After correcting faults, you can clear them on a specific module, or on all modules. In addition, you can monitor the OVD status of Tricon Digital Output modules, and display the firmware version numbers for all types of modules.

For corrective actions (such as replacing a module with a spare), see the maintenance guidelines in the *Planning and Installation Guide*.

Topics include:

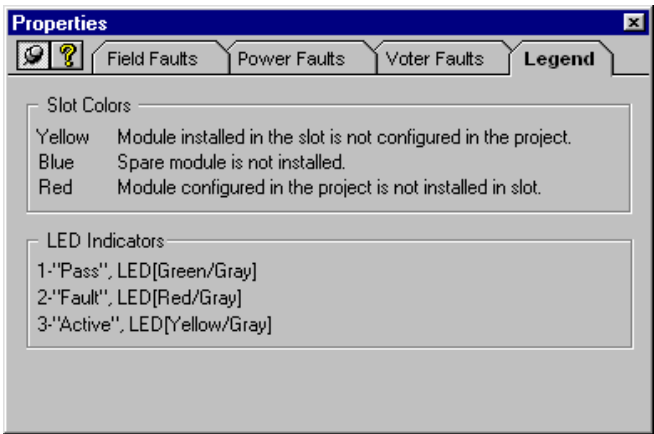
- [Tricon Overview and Chassis Windows \(page 413\)](#)
- [Tricon Module Status Screen \(page 414\)](#)
- [Trident Overview and IOP Windows \(page 415\)](#)
- [Trident Module Status Screen \(page 417\)](#)
- [Changing the View of an IOP Window \(page 418\)](#)
- [Module Indicator Behavior \(page 418\)](#)
- [Understanding External Faults \(page 419\)](#)
- [Locating and Correcting External Faults \(page 419\)](#)
- [Understanding Internal Faults \(page 420\)](#)
- [Locating and Correcting Internal Faults \(page 421\)](#)
- [Clearing Faults on All Modules \(page 422\)](#)
- [Monitoring Output Voter Diagnostics \(OVD\) \(page 423\)](#)
- [Displaying Firmware Versions \(page 424\)](#)

Module Colors

The Chassis window uses these colors to represent the current state of each module in the chassis.

Slot Color	Meaning
Yellow	Module installed in the slot is not configured in the project.
Blue	Spare module is not installed in the slot.
Red	Module configured in the project is not installed in the slot.

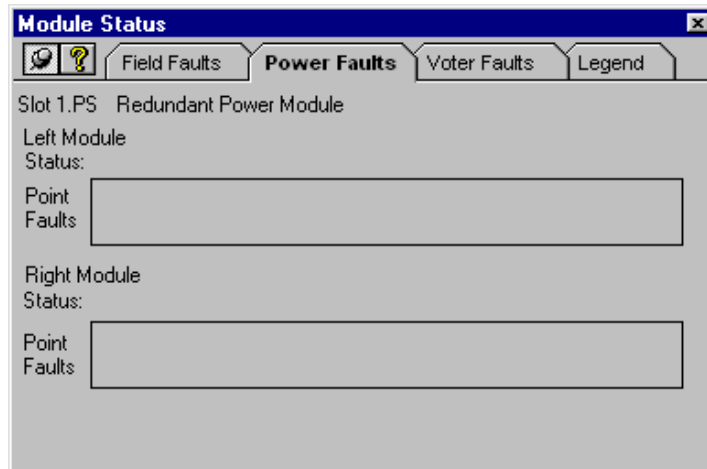
The module colors are described on the Legend tab of the Module Status screen for each module.



Tricon Module Status Screen

For each module in a Tricon controller, the Diagnostic Monitor provides a Module Status screen which includes tabs which display information about faults.

Options	Description
Field Faults	Displays load or fuse faults related to field inputs, terminations, wiring, or devices. (Field faults are not applicable to Main Processors.)
Power Faults	Displays faults related to missing field loads (power) or blown fuses; or power problems which are internal to the controller.
Voter Faults	Displays faults in the OVD (Output Voter Diagnostic) circuitry of a Digital Output module.



Fault Information Areas

Each Module Status screen includes an area for the left module and the right module in a slot. The module areas have properties which provide this information about faults.

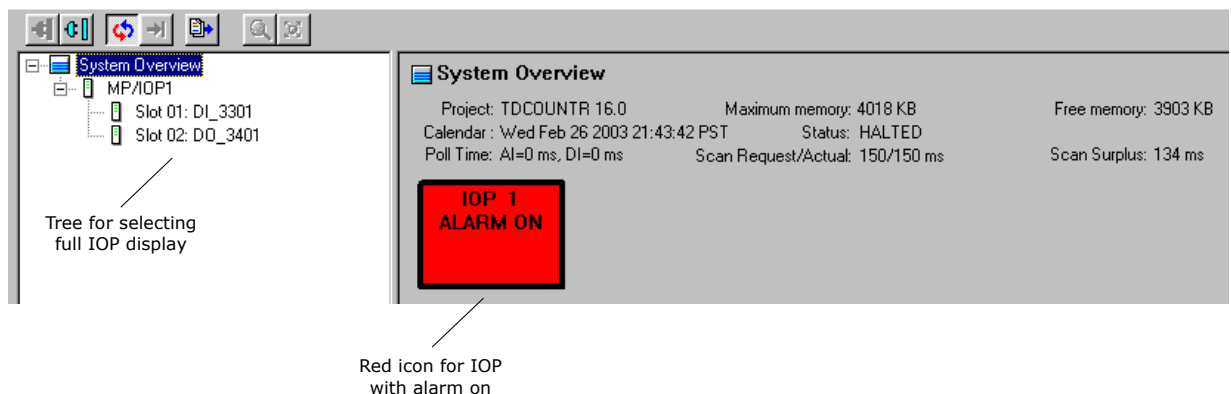
Property	Description
Status	Displays a message which describes the type of fault.
Point Faults	Displays the number of each point with a fault condition.

Trident Overview and IOP Windows

For a Trident controller, the Diagnostic Monitor includes an overview window called System Overview, and a controller configuration window called the IOP window.

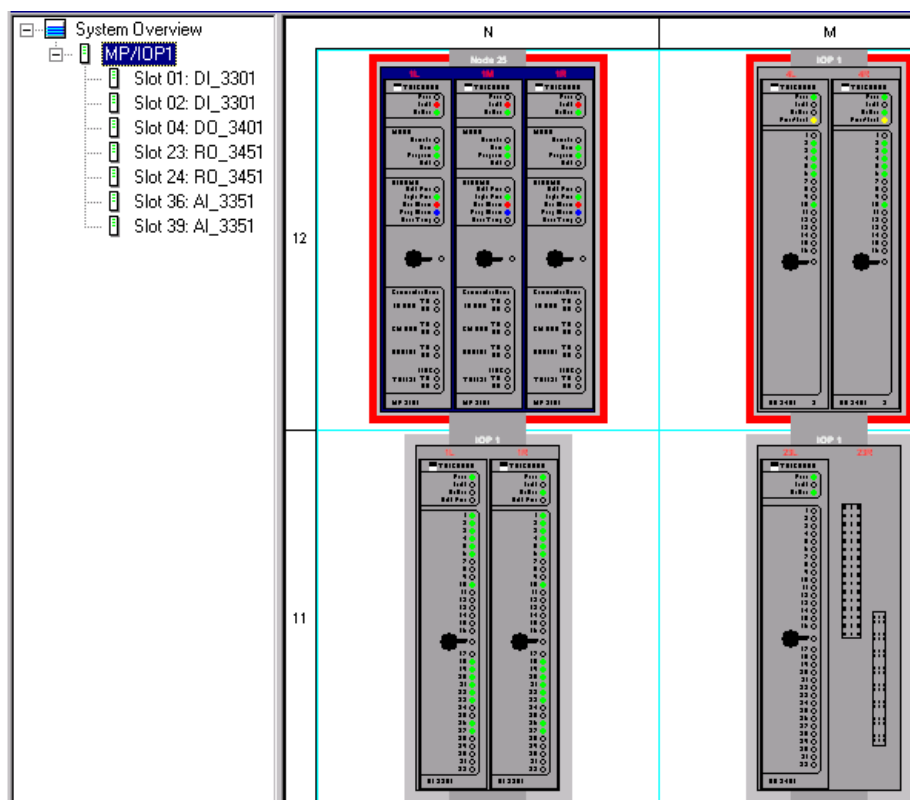
Trident System Overview Window

The Trident System Overview window displays an icon for each IOP in the configuration. Healthy IOPs are green; IOPs with faulting modules are red.



IOP Window

The IOP window, accessible through the Configuration tree, shows the setup of modules in an IOP.

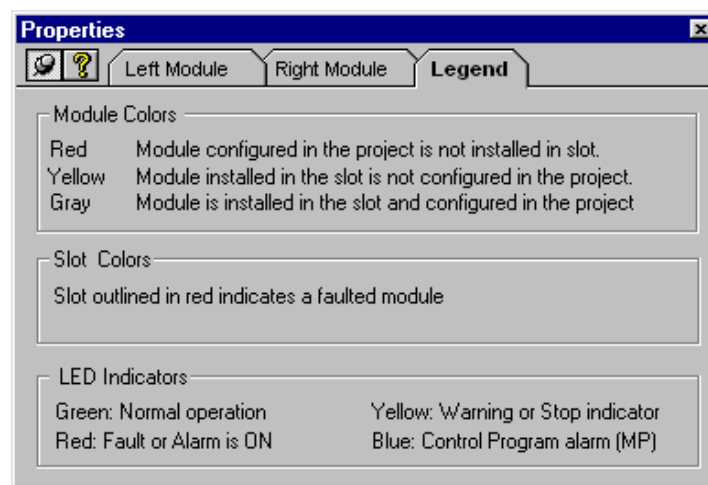


Module Colors

The IOP window uses these colors to represent the current state of each module.

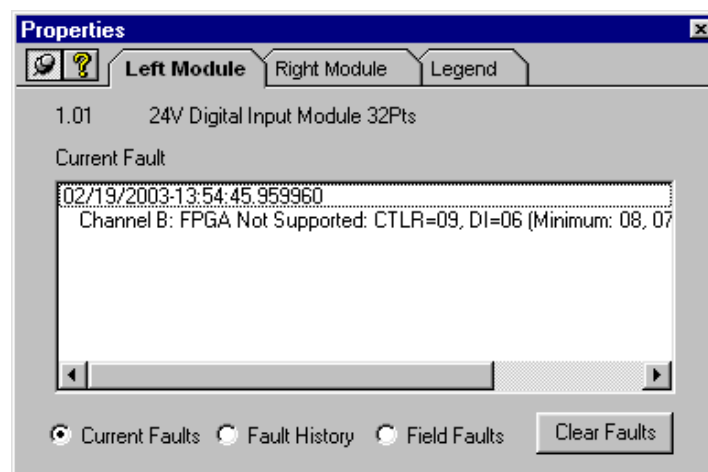
Slot Color	Meaning
Red	Module configured in the project is not installed in the slot.
Yellow	Module installed in the slot is not configured in the project.
Gray	Module is configured in the project and installed in the slot.
Blue	Slot is currently selected.

The module colors are described on the Legend tab of the Module Status screen for each module.

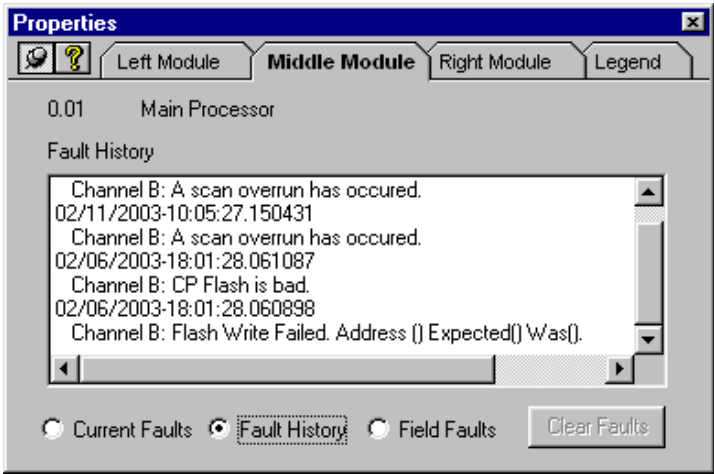


Trident Module Status Screen

For an I/O or communication module, the Module Status screen includes Left and Right Module tabs.



For the Main Processors, the **Module Status** screen includes the **Left**, **Middle**, and **Right Module** tabs.



Each tab on a **Module Status** screen provides this information about faults.

Option	Description
Current Faults	Displays the current faults on a module, which can include internal faults and field faults (external faults).
Fault History	Displays all the faults that have occurred during this session.
Field Faults	Displays load or fuse faults related to field inputs, terminations, wiring, or devices. (Field faults do not apply to Main Processors.)

Changing the View of an IOP Window

For Trident, this procedure explains how to change your view of the modules in an IOP window.

Procedure

- 1 Expand the **System Overview** tree and click a module to display the **IOP window**.
- 2 On the **View** menu, click **Zoom**, and then select one of these options:
 - Click **200**, **100**, **75**, or **50** per cent.
 - Click **Custom**, and then enter the desired percentage.
 - Click **Zoom to Fit** to size the elements to the window.

Module Indicator Behavior

Module indicators in the Diagnostic Monitor behave the same as module indicators on the controller, with these exceptions:

- During power-up of a controller, the indicators in the Diagnostic Monitor are Off (gray).
- Blinking indicators blink more slowly.
- Indicators that are blinking very quickly may appear static in the Diagnostic Monitor.

Understanding External Faults

An external fault is a problem with field inputs, field power supplies, terminations, wiring, or connected devices. Modules report external faults through these indicators:

- For Tricon, the indicator called LOAD/FUSE (if available) on the affected module turns yellow.
- For Trident, the indicator called FIELD POWER on the affected module turns yellow. If the module has a power fault, the FIELD POWER and SYSTEM ALARM indicators on the Main Processors also turn On.

If an external fault occurs, you should read the diagnostic messages on the Module Status screen and examine the field inputs, field power supplies, terminations, wiring, and connected devices.

For detailed information about faults and module indicators, see the *Planning and Installation Guide*.

A controller is subject to these types of external faults.

Fault Type	Description
Field Fault on Digital Output Point	A load or fuse problem related to field terminations on the controller, field wiring, or field devices.
Field Fault on Input Point	A faulty power supply.
Power Fault	<p>For Tricon, a power fault refers to one of these conditions:</p> <ul style="list-style-type: none"> • The field load (power) for a point is missing. • A point has a blown fuse. • One of the power supplies is turned Off. <p>For Trident, a power fault refers to the field power supplies which are connected to a specific module.</p> <ul style="list-style-type: none"> • Field Power Supply 1 has a problem. • Field Power Supply 2 has a problem. • There is a problem with field inputs, field power supplies, terminations, wiring, and connected devices.

Locating and Correcting External Faults

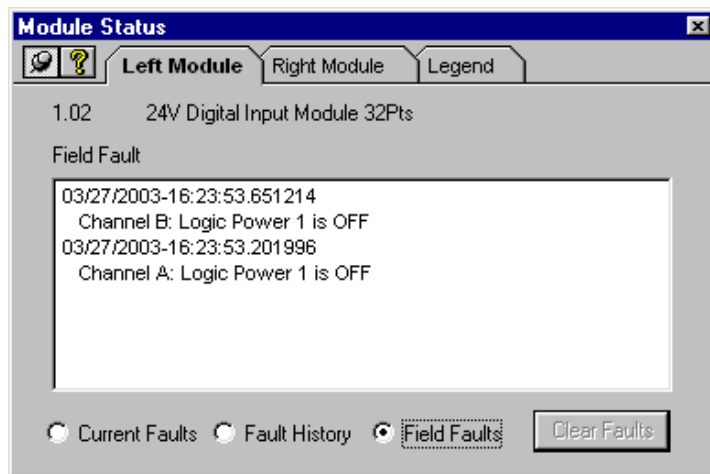
This procedure explains how to locate, correct, and clear external faults on a module.

Before starting, the Diagnostic PC must be connected to the network, and the node of interest must be open.

For details about fault conditions, see the *Planning and Installation Guide*.

Procedure

- 1 On the **System Overview** tree, click **System Overview** to view the **Chassis** or **IOP** alarm icons.
- 2 If an alarm is On (red), open the **Chassis** or **IOP** view, and then double-click a module with a yellow indicator.
- 3 On the **Module Status** screen, click these tabs to locate the fault details:
 - For Tricon, click the **Field Faults** and **Power Faults** tabs.
 - For Trident, on the **Left Module** and **Right Module** tabs, click the **Current Faults** and **Field Faults** settings.



- 4 Examine the relevant field inputs, power supplies, terminations, wiring, and connected devices, and then repair the faulty items.
- 5 On the **Module Status** screen, click **Clear Faults**, and then wait ten minutes to see if the yellow indicator turns On again. If it does, repeat [step 3](#) and [step 4](#).
- 6 When all the faults are corrected, on the **Module Status** screen, click the tab that displays the faulting points, and then click **Clear Faults**.

Understanding Internal Faults

Internal faults are failures in the internal circuitry of a module. If a module has an internal fault, its Fault indicator turns red. You should replace the module with a spare as soon as possible. For instructions, see the *Planning and Installation Guide*.

If you replace a module and the Fault indicator immediately turns red, you should read the diagnostic messages on the Module Status screen and investigate the field inputs. If the inputs are changing rapidly, they may be causing the Fault indicator to turn On.

An internal fault is usually isolated to one channel (A, B, or C) of the faulting module, which means the other two channels can maintain full control. Depending on the specific fault, the module remains in TMR mode or degrades to dual mode.

A controller is subject to these types of internal faults.

Fault Type	Description
Minor	A fault that is usually transient in nature and has no impact on system operation. An example is a CRC error in one message. (Does not cause the Fault indicator to turn red.)
Major	A fault that degrades system operation, but does not affect the correct voting of system inputs or outputs. An example is a fault that inhibits diagnostics on one or more channels.
Voter	A fault that can occur on a Digital Output module only, if the OVD circuitry becomes defective. Examples are an output switch stuck high or stuck low. Voter faults on two channels of a single point may lead to loss of control of that point.
Fatal	<p>A fault on channel A, B, or C of an I/O module which prevents the channel from reading at least one input point or controlling at least one output point. An example is loss of communication with one channel of an I/O module. The module continues to operate correctly using the remaining two channels. Depending on the specific fatal fault and module type, many modules continue to operate correctly with fatal faults on two channels.</p> <p>Triconex recommends replacing any module whose Fault indicator is red, unless the application allows the controller to run in dual or single mode for a specified time period before shutting down.</p>

Locating and Correcting Internal Faults

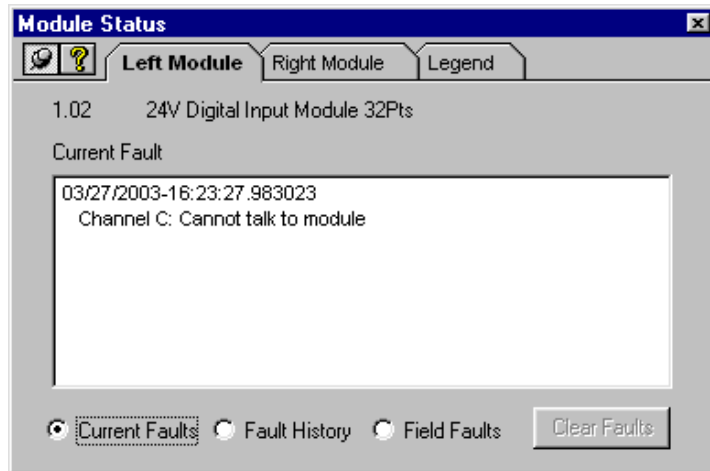
This procedure explains how to locate, correct, and clear internal faults on a module.

Before starting, the Diagnostic PC must be connected to the network, and the node of interest must be open.

Procedure

- 1 On the **System Overview** tree, click **System Overview** to view the **Chassis** or **IOP** alarm icons.
- 2 If an alarm is On (red), open the **Chassis** or **IOP** view, and then double-click a module with a red indicator.
- 3 On the **Module Status** screen, click these tabs to find the fault messages:
 - For Tricon, click the **Voter Faults** tab.
 - For Trident, on the **Left Module** and **Right Module** tabs, click the **Current Faults** setting.
- 4 On the **Module Status** screen, click **Clear Faults**, and then wait ten minutes to see if the red indicator turns On again. If it does, replace the module.

For instructions, see the *Planning and Installation Guide*.



- 5 If you replace a module and the Fault indicator immediately turns red, you should investigate the field inputs. Rapidly changing field inputs may be causing faults.
- 6 When all faults are corrected, on the **Module Status** screen, click the tab that displays the fault information, and then click **Clear Faults**.

Clearing Faults on All Modules

This procedure explains how to clear the faults on all modules in a controller.

Procedure

- 1 Ensure that all faults have been identified and corrected.
- 2 On the **Commands** menu, click **Clear Faults on All Modules**.

The Tricon chassis or Trident IOP alarm turns off, and the Fault indicators on the faulting modules turn off (gray).

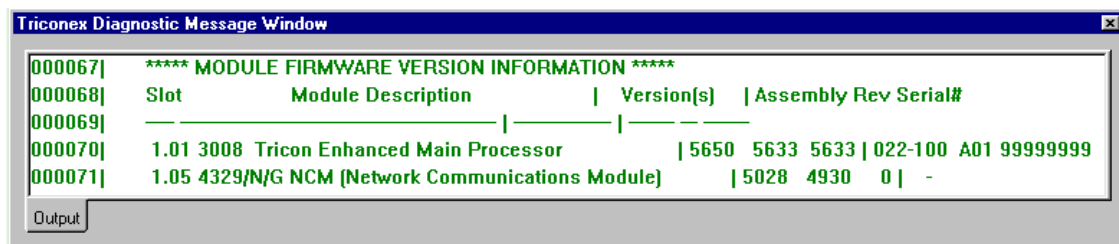
blocks in the Controller Panel of **TriStation 1131** during online execution. For instructions, see the *TriStation 1131 Libraries Reference* and the *TriStation 1131™ Developer's Guide for Trident*.

Displaying Firmware Versions

This procedure explains how to display the firmware version numbers of all modules in a Tricon chassis or Trident IOP. Typically, you only need to know the firmware version numbers if requested by a Triconex Technical Support Engineer for diagnostic purposes.

Procedure

- 1 On the **Commands** menu, click **Display Firmware Versions**.



- 2 To copy, print, or save the firmware version information, place the cursor in the **Diagnostic Message Window**, right-click the mouse, and then click the desired command.

Monitoring Controller Status

The Diagnostic Monitor allows you to monitor controller status on a system overview screen.

Before starting, you must complete these steps:

- Create a network configuration which includes node specifications for the controllers you want to monitor, or open an existing network configuration.
- Connect the Diagnostic PC to the network.
- Open the nodes you want to monitor.

Note You can also access controller performance and project information using function blocks in the **TriStation 1131** standard libraries, and for Trident, using system attributes.

Topics include:

- [Viewing Controller Status \(page 426\)](#)
- [Monitoring and Changing the Scan Time \(page 427\)](#)
- [Monitoring and Changing the Memory Allocation \(page 427\)](#)
- [Refreshing Controller Status \(page 428\)](#)
- [Viewing Program Execution Times \(page 428\)](#)
- [Collecting System Diagnostic Events \(page 428\)](#)
- [Understanding the Diagnostic Message Window \(page 429\)](#)

Viewing Controller Status

This procedure explains how to view controller status on a system overview screen.

To correct scan time problems or allocate more memory, see the *TriStation 1131™ Developer's Guide*.

Procedure

- 1 On the **Network Nodes** tree, double-click the node you want to monitor.
- 2 Click **System Overview** on the **Controller** tree.
- 3 View the performance and project information on the right pane.

Property	Action
Project	Displays the name and version number of the project (application) being monitored.
Calendar	Displays the current time of the controller being monitored in the day/date/hour/minute/second format.
Poll Time	Displays the maximum time needed by a controller to obtain data from the input modules. Ensure the poll time does not exceed the scan time.
Maximum Memory	Displays the maximum amount of memory that the controller originally made available to the TriStation project.
Key Stop/Position	For Tricon, Key Stop indicates whether the STOP position on the keyswitch is logically disabled for security purposes. Position indicates the physical setting of the keyswitch.
Scan Request/ Actual	Displays the requested scan time and the actual scan time.
Free Memory	Displays the amount of memory available for project expansion.
Status	Displays the current state of the application running in a controller.
Scan Surplus	Displays the time that remains in a scan after the control functions are completed. To avoid communication errors, ensure the scan surplus is positive.

Monitoring and Changing the Scan Time

This procedure explains how to monitor the scan time in the system overview window, and then change the scan time using **TriStation 1131**.

Procedure

- 1 On the **Network Nodes** tree, double-click the node of interest.
- 2 Click **System Overview** on the **Controller** tree.
- 3 On the right pane, view these properties, which are related to the scan time.

Property	Action
Poll Time	Displays the maximum time needed by a controller to obtain data from the input modules. You should ensure the poll time does not exceed the scan time.
Scan Request/ Actual	Displays the requested scan time and the actual scan time.
Scan Surplus	Displays the time that remains in a scan after the control functions are completed. To avoid communication errors, ensure the scan surplus is positive.

- 4 If the poll time exceeds the scan time, or the scan surplus is negative, use the **Controller Panel** in the **TriStation 1131** project to adjust the scan time.

For instructions, see the *TriStation 1131 Developer's Guide*.

Monitoring and Changing the Memory Allocation

This procedure explains how to monitor the memory allocation of a controller in the System Overview window, and then change the memory allocation using **TriStation 1131**.

Procedure

- 1 On the **Network Nodes** tree, double-click the node of interest.
- 2 Click **System Overview** on the **Controller** tree.
- 3 On the right pane, view these properties related to the memory allocation.

Property	Action
Maximum Memory	Displays the maximum amount of memory that the controller originally made available to the TriStation project.
Free Memory	Displays the amount of memory available for project expansion.

- 4 If you plan to make changes to the application that require more memory, use the **Configuration** tree in the **TriStation** project to allocate more memory, and then do a **Download All**.

For instructions, see the *TriStation 1131 Developer's Guide*.

Refreshing Controller Status

This procedure explains how to refresh the system overview and hardware configuration displays while you are monitoring a controller. You can refresh the information occasionally or continuously.

Procedure

- 1 On the **Network Nodes** tree, double-click the node (controller) of interest.
- 2 To refresh the system overview and hardware configuration once, on the **Commands** menu, click **Refresh Panel**.
- 3 To continuously refresh the system overview and hardware configuration, on the **Commands** menu, click **Continuous Refresh**.

For Tricon, a beep sounds once when a chassis alarm turns on.

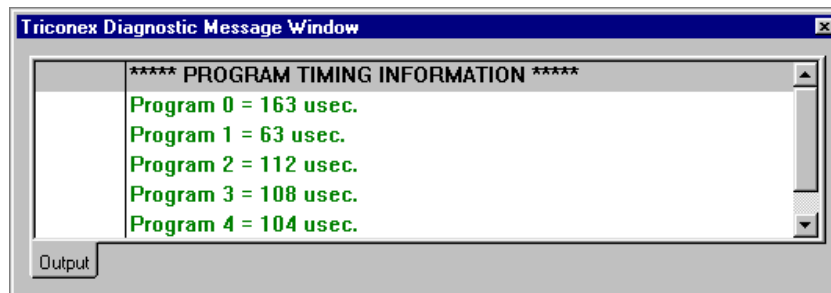
Viewing Program Execution Times

This procedure explains how to view the execution time of each program in an application that is running in a controller.

The programs are identified as Program 0, Program1, and so on. The first and the last programs are provided by **TriStation 1131** to initialize and terminate the scan. The other programs are the programs specified in the program execution list in the **TriStation 1131** project.

Procedure

- 1 On the **Network Nodes** tree, double-click the node (controller) of interest.
- 2 On the **Commands** menu, click **Display Execution Times**, and then view the execution time for each program in the **Diagnostic Message Window**.



- 3 To copy, print, or save the program execution information, place the cursor in the **Diagnostic Message Window**, right-click the mouse, and then click the desired commands.

Collecting System Diagnostic Events

This procedure explains how to collect system diagnostic events in an event log file for troubleshooting problems. Typically, this procedure is only done at the request of a Technical Support representative.

You should send the event log file to a Customer Support Center, using one of these methods:

- Put the file on a zip disk or CD and send it by postal mail.
- Zip the file and e-mail it to Technical Support.

Procedure

- 1 On the **Network Nodes** tree, double-click the node (controller) of interest.
- 2 For Tricon, on the **Commands** menu, click **Collect System Events**.
- 3 On the **Save As** screen, enter a file name for the event log file, and then click **Save**.
The correct extension is automatically appended if you do not type it in. For Tricon, the extension is TEC. For Trident, the extension is TDE.
- 4 Click **OK**, and then allow the Diagnostic Monitor to collect the historical events.
The Diagnostic Monitor collects a maximum of 500,000 events.
- 5 To stop event collection at any time, on the **Collecting System Events** screen, click **Stop**.

Understanding the Diagnostic Message Window

The Diagnostic Message Window displays messages about diagnostic operations. To show or hide the message window, click Messages command on the View menu.

Managing Data in the Message Window

You can manage the data in the Diagnostic Message Window through these commands, which are accessible by right-clicking anywhere in the window.

Command Name	Operation
Copy Command	Copies all messages, and places them on the clipboard.
Cut Command	Cuts all messages, and places them on the clipboard, erasing the previous content.
Find Command	Finds specified text.
Print Command	Prints the contents on a specified printer.
Save As Command	Saves messages in a LOG file.
Clear All Command	Clears all messages.
Select All Command	Selects all text, in preparation for a Copy or Cut operation.
Show Line Numbers Command	Shows the number of each line of text.

Typical Diagnostic Messages

These commands report messages or results in the Diagnostic Message Window.

Command Name	Operation
Collect System Events Command	Collects system diagnostic events in an event log file for troubleshooting problems.
Display Execution Times Command	Allows you to view the execution time of each program in an application that is running in a controller.
Display Firmware Versions Command	Displays the firmware version numbers of all modules in a Tricon chassis or Trident IOP.

B

Triconex OPC Server

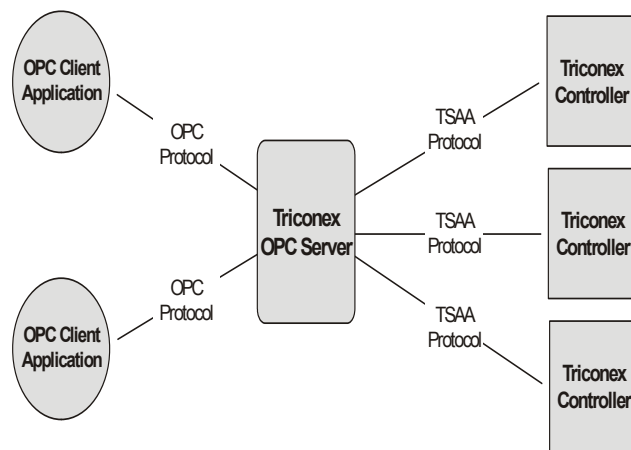
Overview	432
Configuring the OPC Server	433
Other OPC Products	437

Overview

The Triconex OPC Server is a Windows application which allows OPC clients to have read and write access to Triconex program variables. OPC stands for *OLE for Process Control*, which is a standard set of non-proprietary interfaces used to develop client/server programs.

The OPC Server workstation must be connected to an Ethernet port on a Triconex controller (Tricon or Trident). For Tricon, the Net2 port on the ACM or NCM must be used. For Trident, the Net1 or Net2 port on the Communication Module must be used.

OPC Server is configured by exporting an XML configuration file from a **TriStation** project and opening that file in OPC Server. After OPC Server is configured, an OPC client can ask OPC Server to get data from a Triconex controller.



You can include **TriStation** configurations for multiple networked controllers in one XML file by using the same file name when exporting each configuration. The information from each **TriStation** configuration is appended to the file.

In OPC Server, you can edit the properties of aliases and tagnames and other aspects of the configuration. If you change the name of the configuration or alias, a new entry is created in the XML configuration file. If you change properties related to the entry, but do not change the configuration or alias name, those properties are changed for the entry.

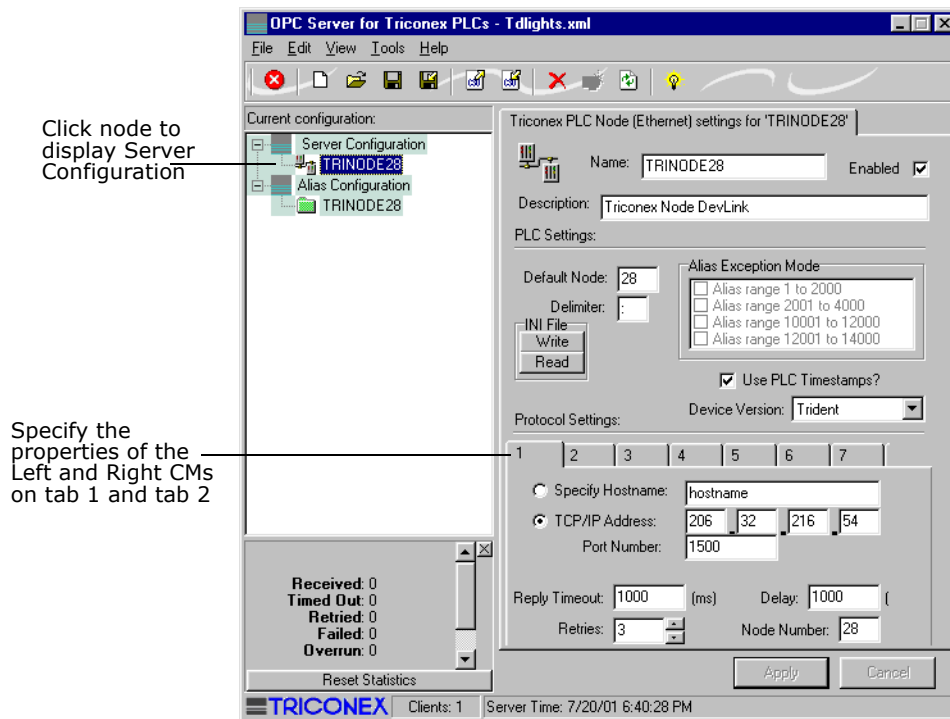
The Triconex OPC Server is available from Triconex and Matrikon. For more information on the Triconex OPC Server and OPC client applications, see the Matrikon Web site at www.matrikon.com.

Configuring the OPC Server

This procedure explains how to configure the OPC Server with alias information from a **TriStation** project. With **TriStation 1131** v3.1 and later, alias information can be exported to an XML file and then imported to the OPC Server. To use OPC Server with **TriStation** versions earlier than v3.1, alias information must be manually entered to the OPC Server.

Procedure

- 1 In **TriStation**, assign all aliases to be accessed by OPC Server.
- 2 To allow an OPC client to change the values of Triconex variables, you must enable write access in the **TriStation** application.
- 3 In **TriStation**, complete the project and download it to the controller.
- 4 Export the XML configuration file by exporting the tagnames (points).
- 5 Select the file type as Matrikon OPC XML Data Files(*.XML). Name the file using the XML extension, using a maximum of eight characters. If you include multiple configurations, use the same file name each time you use the export command.
- 6 If not already done, install OPC Server and start it. When OPC Server is loaded, a gray Triconex icon is displayed on the status bar.
- 7 To open the XML configuration file, right-click the Triconex icon from the status bar and select Configure.
- 8 The OPC Server for Triconex PLCs window is displayed.
- 9 From the File menu, select Open, then select the XML file you exported from **TriStation**. As the file loads, statistics are displayed. When finished, you can display Server Configuration and Alias configuration information.
- 10 To display Server Configuration information, select a node from the Current Configuration pane. This screen shows the Server Configuration information. You can make changes to these properties by entering the changes and pressing the Apply button.

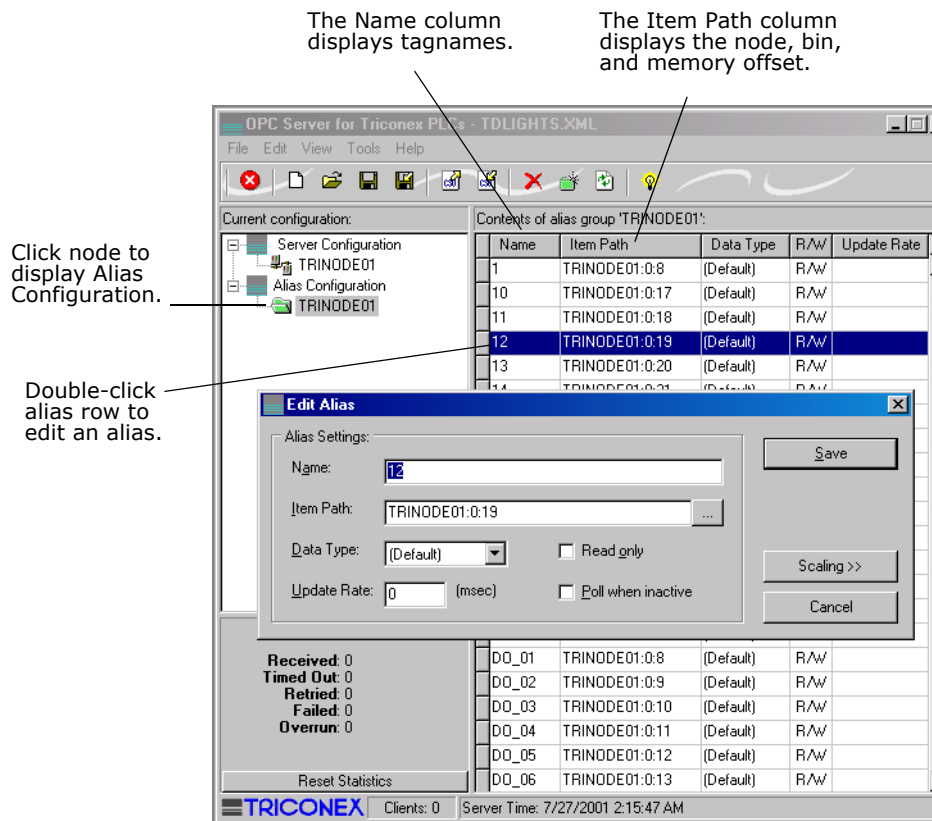


- 11 Under Protocol Settings, specify the TCP/IP address and other properties of the Triconex communication module on tab 1.

If the controller includes two communication modules, specify the properties of the left module and the right module on tabs 1 and 2.

Do not use tabs 3 through 7.

- 12 To display alias information, select a node under Alias Configuration in the Current Configuration pane. To make changes to an alias, double-click the alias row. An Edit Alias screen is displayed, as shown in this screen.



- 13 Repeat steps 8 through 11 for each node configuration included in the XML file.
- 14 If you made changes to any of the configurations and want to keep them, save the configuration file.
- 15 To use OPC Server to get data from a Triconex controller, install an OPC client application. (Matrikon sells OPC client applications.)
- 16 In the OPC client application, you can specify the tagnames or aliases of the data to be accessed.

A sample tagname is DO_02, as shown in the previous screen.

The location of the data is described as node: bin: offset in the Item Path column for the Alias Configuration.

Redundant Configuration

OPC Server can be configured for dual redundancy by using two OPC Server workstations. Each workstation must include two Ethernet adapter cards, which must be connected to a Triconex communication module on the primary network and a Triconex communication module on the redundant network. You must specify the properties of the redundant Triconex communication modules on the OPC Server Configuration screen.

Adjusting System Time

An OPC client can use the Device Clock tagname to read or write the system time of a Triconex controller. The Device Clock tagname is derived from Triconex status information in the OPC Server Configuration. For more information, see the documentation for the OPC client software.

Before you can use the Device Clock tagname to adjust the system time of a Triconex controller, you must configure the **TriStation** project to allow writes by external devices on an open network.

Other OPC Products

For users of OPC Server, two additional OPC products are available from Triconex and Matrikon: the OPC Data Manager and the OPC Redundancy Broker.

OPC Data Manager

The OPC Data Manager (ODM) is an application that transfers data from one OPC server to another. ODM is useful for sharing data between two or more control systems such as a Triconex controller and a DCS. Traditional OPC-enabled systems share data by implementing one application as an OPC client, and another as an OPC server. If two applications are servers instead of clients, they cannot exchange data. ODM solves this problem by acting as a *double-headed* or *thin* OPC client to both servers. It requests data from one OPC server and immediately sends it to the other OPC server.

ODM includes these features:

- Support for both COM and DCOM architectures
- Support for DDE and OPC message protocols
- Operation as a Windows service or a normal application
- Real-time data monitoring
- Extensive error tracking and management

For more information, see the *Matrikon OPC Data Manager User's Manual*.

OPC Redundancy Broker

The OPC Redundancy Broker (ORB) is a messaging application designed for systems that must use redundant devices to ensure high reliability. ORB constantly monitors the primary OPC server and redirects communication to the standby OPC server when a failure is detected. ORB can integrate with any OPC compliant client/server configuration and can be retrofitted to existing configurations.

ORB includes these features:

- Intuitive configuration and monitoring features
- Choice of hot, cold, or warm fail-over for each OPC Server
- Automatic fail-over notification by e-mail, fax, log file, or pager
- Extensive error tracking and diagnostic capabilities

For more information, see the *Matrikon OPC Redundancy Broker User's Manual*.

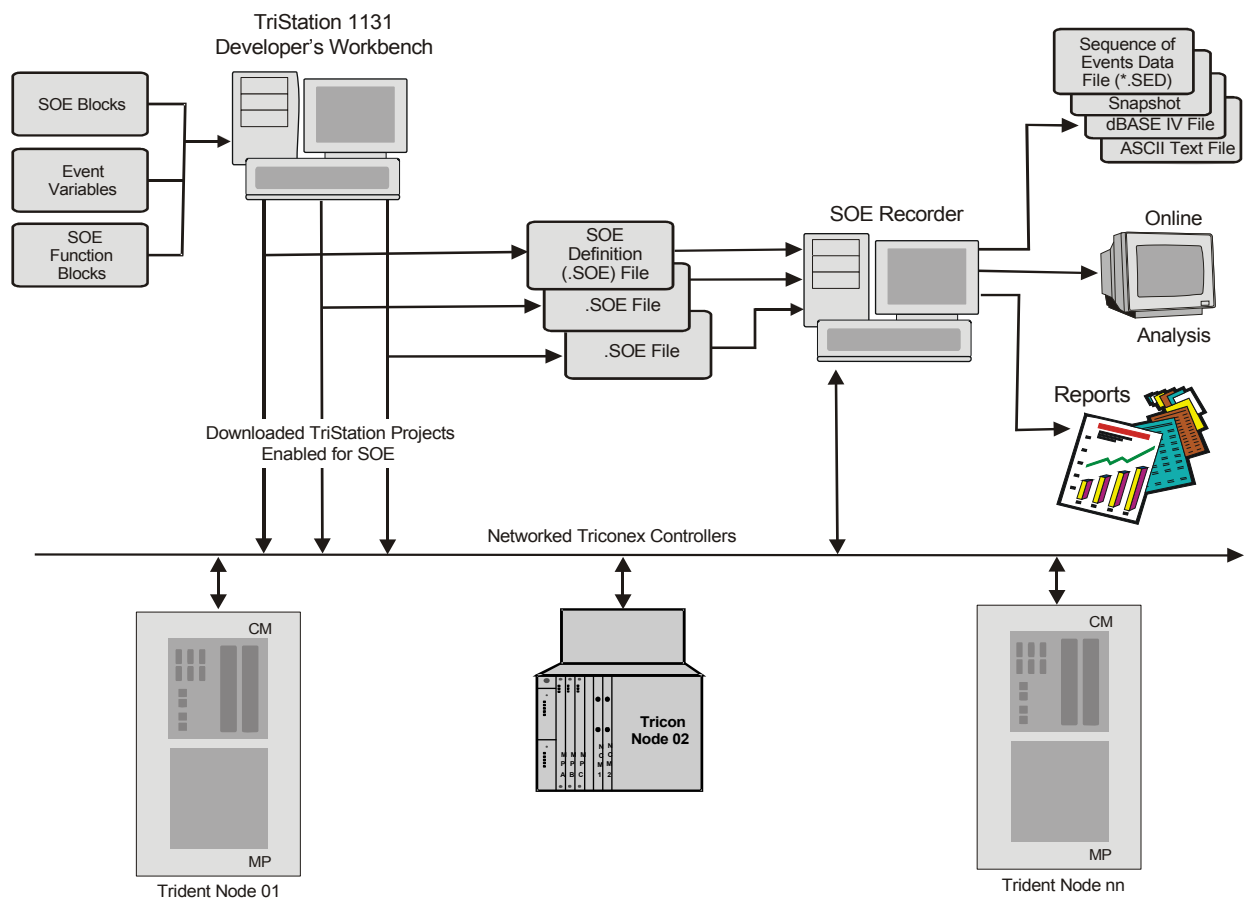
Sequence of Events Recorder

Overview of Functionality	440
SOE Setup in TriStation 1131	442
SOE Development in TriStation 1131	447
SOE Recorder Setup	452
SOE Recorder Event Retrieval	461

Overview of Functionality

The Triconex Sequence of Events (SOE) Recorder software allows you to retrieve event data from as many as 32 networked Triconex controllers. This event data can be used to identify the causes of shutdowns, define corrective actions and procedures for preventive maintenance, and solve other process-control problems.

Understanding the sequence of events that leads to unsafe process conditions and possible shutdowns may be essential to your safety application. For Triconex controllers, an event is the state change of a Boolean variable from True to False or from False to True. A Triconex controller can collect events and record the date, time, state, and variable name of each event in SOE blocks that reside in the controller memory. This figure shows how Triconex controllers, **TriStation 1131**, and SOE Recorder work together to provide sequence-of-events functionality.



To enable event data to be detected by the controller, you must define the event variables and SOE blocks in the **TriStation** application. In addition, the application must include SOE function blocks that start event collection. After you download an SOE-enabled project to the controller, **TriStation** creates an SOE definition file that contains the SOE block definitions.

To retrieve events with SOE Recorder, you must connect the SOE workstation to a communication port on a Triconex controller. (You can also retrieve events from a Triconex controller with an SOE program on a non-Triconex external device.) You must use **TriStation** to set the IP address of the controller and you must specify the address in SOE Recorder.

SOE Recorder can simultaneously retrieve event data from as many as 32 networked controllers, assuming that IP addressing has been done correctly. SOE Recorder queries all the controllers on the network to determine which **TriStation** applications include SOE blocks. If a project includes one or more SOE blocks, SOE Recorder opens the appropriate SOE definition file and begins retrieving events from the associated controller.

SOE Recorder accumulates events in an event file until a trip occurs or the maximum allowable number of events occurs. SOE Recorder can also save *snapshots* for periods that you specify, such as snapshots that reflect each shift of a plant's operation. During event retrieval, you can manually save an intermediate snapshot of event data at any time.

SOE Recorder also allows you to export event data, either manually or automatically, to dBASEIV or ASCII text files. A report engine and standard report are also included.

While the **TriStation** application is running, SOE Recorder allows you to analyze events online as it retrieves them from the controllers. To analyze the event data, SOE Recorder includes tools for:

- Finding events and copying them to other Windows applications
- Filtering and sorting saved event data,
- Specifying the display of point properties for event data
- Viewing the properties of individual events.

You can also save snapshots of events that cover specific periods of time before or after trips have occurred. In addition, a report engine and a standard report are included so you can generate reports and export event data, either manually or automatically, to dBASEIV or ASCII text files.

SOE Setup in TriStation 1131

This section explains how to set up sequence of events collection in a **TriStation 1131** application. Events can be retrieved from a Triconex controller by using the SOE Recorder software.

For more information, see the *Sequence of Events Recorder™ User's Guide*.

Topics include:

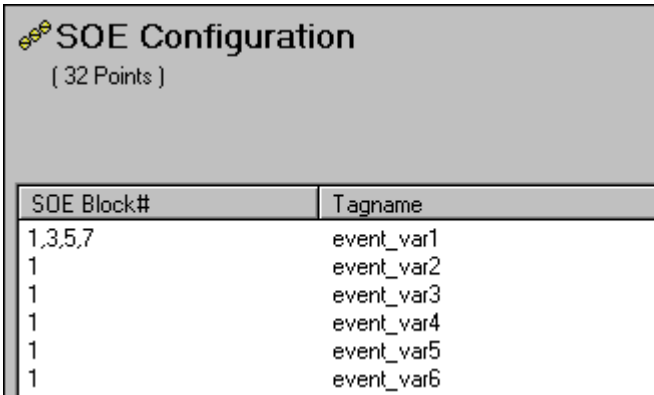
- [SOE Configuration \(page 442\)](#)
- [Defining SOE Block Properties \(page 442\)](#)
- [Assigning Event Variables to SOE Blocks \(page 444\)](#)
- [Specifying a Trip Variable \(page 445\)](#)

SOE Configuration

This procedure explains how to display the SOE Configuration screen, which displays the SOE blocks that have been configured.

Procedure

- 1 Expand the **Application** tree, double-click **Implementation**, and click **SOE Configuration**.



The screenshot shows the 'SOE Configuration' window with a title bar and a subtitle '(32 Points)'. Below the title bar is a table with two columns: 'SOE Block#' and 'Tagname'. The table contains six rows of data.

SOE Block#	Tagname
1,3,5,7	event_var1
1	event_var2
1	event_var3
1	event_var4
1	event_var5
1	event_var6

- 2 To change the SOE block properties, see [Defining SOE Block Properties \(page 442\)](#) and [Assigning Event Variables to SOE Blocks \(page 444\)](#).

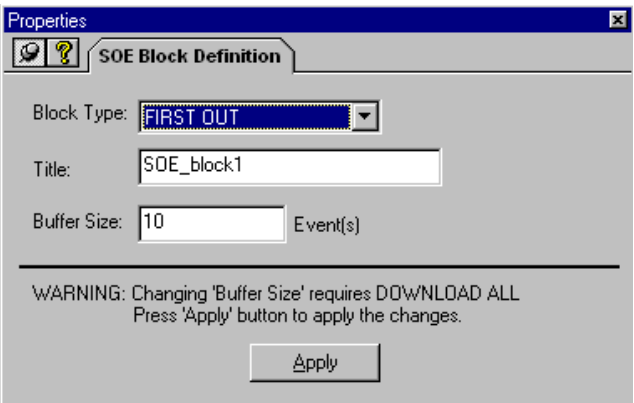
Defining SOE Block Properties

This procedure explains how to define the properties of an SOE block, which is required if you are using sequence of events in an application.

Defining SOE block properties is related to the general task of controller configuration and can be completed at the same time. You can define SOE blocks to provide continuous event information to external devices on a network, or you can define them for limited operation when your controller is not on a network.

Procedure

- 1 Click the **Application** tree, and double-click **Implementation**. Expand the **SOE Configuration** tree, and double-click a block number.



- 2 Specify these settings on the **SOE Block Definition** tab.

Property	Action
SOE Block Type	<p>Select the Block Type.</p> <p>The default is unassigned.</p> <p>Historical: The Historical Block Type property is typically used to monitor current events in SOE Recorder application. Blocks 1 through 14 may be configured with any combination of History, First Out, and External block types. When the buffer is full, the MPs overwrite the oldest event entries.</p> <p>First Out: The First Out Block Type property is used to analyze events that led to a process upset. Blocks 1 through 14 may be configured with any combination of History, First Out, and External block types. When the buffer is full, the MPs change the block’s status from collecting to stopped and discard new events.</p> <p>External: The External Block Type property is typically used by external devices, such as a Foxboro or Honeywell DCS. When the MPs write an event to the block, they notify the external device. The external device requests the data from the MPs and acknowledges the receipt. Blocks 1 through 14 may be configured with any combination of History, First Out, and External block types. When the MPs receive the acknowledgment, they clear the data from the block. When the buffer is full, the MPs discard new event entries.</p> <p>Modified External: The Modified External Block Type property applies to Blocks 15 and 16 in an SOE Recorder application and are used with the SMM only. Blocks 1 through 14 cannot be designated as Modified External blocks.</p> <p>When the MPs write an event to Block 15 or 16, they notify the SMM. The SMM requests the data from the MPs and acknowledges the receipt. When the MPs receive the acknowledgment, they clear the data from the block. When the buffer is full, the MPs discard new event entries.</p>
SOE Block Name	<p>Enter a title for the block.</p>

Property	Action												
SOE Buffer Size	<p>The SOE Buffer Size property, expressed in multiples of 16 bytes, is the amount of memory that the MPs reserve for recording event entries. An event entry consists of an eight-byte time stamp and one or more eight-byte data entries. Each data entry contains the changed variable alias and state. This figure shows how the MPs store data in an SOE block.</p> <p>The maximum block size you can define is 20,000 events (320,000 bytes). A maximum size of 60,000 events (1 megabyte) is allowed across all blocks.</p> <div><table><tr><th>Scan Number</th><th>Number of Events</th><th>Block Contents</th></tr><tr><td>1</td><td>1</td><td><div><div>time stamp</div><div>data entry</div></div></td></tr><tr><td>2</td><td>2</td><td><div><div>time stamp</div><div>data entry</div><div>data entry</div></div></td></tr><tr><td>3</td><td>1</td><td><div><div>time stamp</div><div>data entry</div></div></td></tr></table></div>	Scan Number	Number of Events	Block Contents	1	1	<div><div>time stamp</div><div>data entry</div></div>	2	2	<div><div>time stamp</div><div>data entry</div><div>data entry</div></div>	3	1	<div><div>time stamp</div><div>data entry</div></div>
Scan Number	Number of Events	Block Contents											
1	1	<div><div>time stamp</div><div>data entry</div></div>											
2	2	<div><div>time stamp</div><div>data entry</div><div>data entry</div></div>											
3	1	<div><div>time stamp</div><div>data entry</div></div>											

- 3

Click **Apply** to have the settings saved.
- 4

Repeat steps 2–5 for all the blocks to be configured.

Assigning Event Variables to SOE Blocks

This procedure explains how to assign event variables to an SOE block, which is required if using sequence of events in an application.

Event variables must be of type BOOL and their states can be displayed with names and colors that you define. You can designate one variable in a **TriStation 1131** project as the trip variable that notifies the operator when a trip occurs.

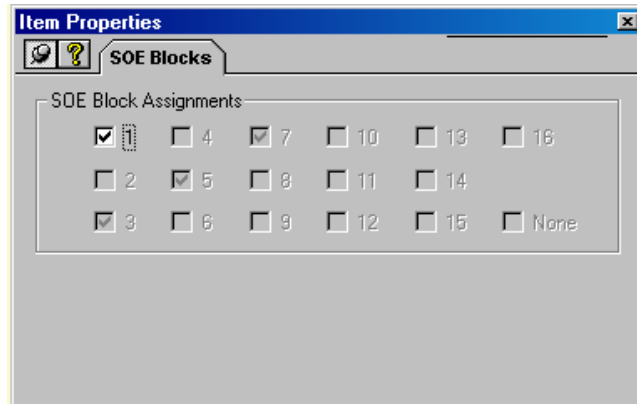
For Tricon, if you define a block for use with the Advanced Communication Module (ACM), the Foxboro I/A Series system assigns the event variables. The only additional configuration you can do is to specify a type of External and a buffer size. For more information, see the *ACM User's Guide*.

Before Starting

Before you can assign event variables, you must define SOE block properties.

Procedure

- 1 Click the **Application** tree, and double-click **Implementation**. Expand the **SOE Configuration** icon, and click a block number.
- 2 Double-click the **Tagname** (event variable) to be assigned.
- 3 Check the block numbers to assign event variables.



- 4 To assign other event variables to SOE blocks, repeat Steps 2 and 3.
- 5 After assigning all the event variables, save the project so the tagnames are available when specifying a trip variable.

Specifying a Trip Variable

This procedure explains how to designate a trip variable, which is optional if you are using sequence of events in an application.

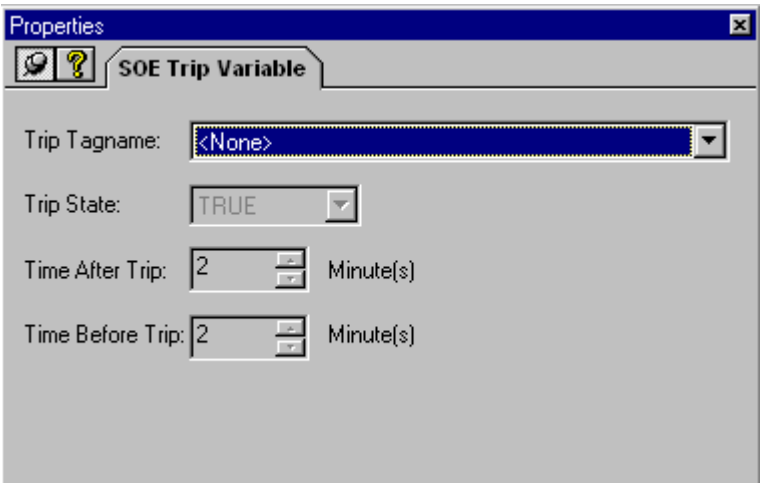
In an application used for safety shutdown, a trip variable is an aliased tagname whose state transition causes SOE Recorder to automatically create a trip snapshot. An application can have one trip variable, but it can apply to all blocks. If an application requires several variables related to trip conditions, these variables must be evaluated in combination to determine the final state of the trip variable.

Before Starting

You must define at least one SOE block and assign an event variable to the block.

Procedure

- 1 Click the **Application** tree, and double-click **Implementation**. Double-click **SOE Configuration**.



- 2 Specify these settings on the **SOE Trip Variable** tab.

Property	Action
Trip Tagname	Click the Trip Tagname from the list of event variable names, and then select a Trip State of True or False.
Trip State	Click True or False.
Time After Trip	Set the Time After Trip in minutes. The minimum time is two minutes; the maximum time is ten minutes.
Time Before Trip	Set the Time Before Trip in minutes. The minimum time is two minutes; the maximum time is ten minutes.

SOE Development in TriStation 1131

This section describes function blocks that are used in an application to control and verify event collection for SOE blocks. The start function block (SOESTRT) must be used to identify the SOE blocks to be used for event collection. The other SOE function blocks are optional.

SOESTRT (SOE Start)	Starts event collection for a specified SOE block (Required)
SOESTOP (SOE Stop)	Stops event collection for a specified SOE block (Optional)
SOECLR (SOE Clear)	Checks status of a specified SOE block (Optional)
SOESTAT (SOE Statistics)	Clears a specified SOE block (Optional)

Programming Notes

To properly execute the SOESTRT, SOESTOP and SOECLR function blocks, a program should turn the CI input On for only one scan. If you leave CI on for more than one scan, SOE Recorder generates another event for every scan. For more information, see the sample projects on the **TriStation** CD.

If you are programming with the Structured Text (ST) language, you must use conditional statements to execute SOESTRT, SOESTOP and SOECLR. For more information, see the *TriStation 1131 Libraries Reference*.

SOESTRT (SOE Start)

The SOESTRT function block starts event collection for the designated block. The MPs write an SOESTRT time-stamp entry to the buffer and change the state of the designated block from *not started* or *stopped* to *collecting*. The MPs use these initial values as the basis for comparison during subsequent scans. If the SOESTRT operation is successful, the returned status is 1, which indicates the block is collecting data.

This function block must be executed for each SOE block from which you want to collect events.

Note The SOESTRT function block is executed for only one scan, which means the resulting STAT output is only valid for that scan.



Input Parameters

Name	Data Type	Description
CI	BOOL	Enables SOESTRT.
BLOCK	DINT	The block number (1-16).

Output Parameters

Name	Data Type	Description
CO	BOOL	True if SOESTRT executes successfully.
STAT	DINT	Status: 0 = The block is not started or not configured with data type and size. 1 = The block is collecting data. 2 = The block is stopped or cleared. 3 = The block is full.

SOESTOP (SOE Stop)

The SOESTOP function block stops event collection for the designated block, writes an SOESTOP time stamp into the buffer, and changes the state of the block from *collecting* to *stopped*. No further events are collected until the next SOESTRT function block is executed. If the SOESTOP operation is successful, the returned status is 2, indicating that event collection is stopped.

This function block is not required, but is recommended following each start function block.

Note The SOESTOP function block is executed for only one scan, which means the resulting STAT output is only valid for that scan.



Input Parameters

Name	Data Type	Description
CI	BOOL	Enables SOESTOP.
BLOCK	DINT	The block number (1-16).

Output Parameters

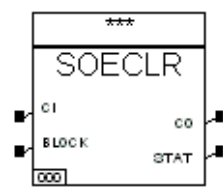
Name	Data Type	Description
CO	BOOL	True if SOESTOP executes successfully.
STAT	DINT	Status: 0 = The block is not started or not configured with data type and size. 1 = The block is collecting data. 2 = The block is stopped or cleared. 3 = The block is full.

SOECLR (SOE Clear)

The SOECLR function block clears the buffer of the designated SOE block, removes all entries from the block, and writes an SOECLR time stamp into the buffer. You must stop event collection using the stop function block before clearing the block. If the SOECLR operation is successful, the status is 2, indicating the block has been cleared. If event collection was not stopped before the SOECLR operation was attempted, the status is 0, 1 or 3.

This function block is optional.

Note The SOECLR function block is executed for only one scan, which means the resulting STAT output is only valid for that scan.



Input Parameters

Name	Data Type	Description
CI	BOOL	Enables SOECLR.
BLOCK	DINT	The block number (1-16).

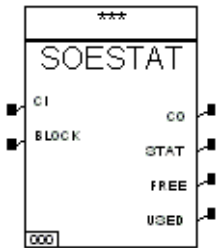
Output Parameters

Name	Data Type	Description
CO	BOOL	True if SOECLR executes successfully.
STAT	DINT	Status: 0 = The block is not started or not configured with data type and size. 1 = The block is collecting data. 2 = The block is stopped or cleared. 3 = The block is full.

SOESTAT (SOE Statistics)

The SOESTAT function block returns the status of the designated SOE block, the number of eight-byte entries that are used, and the number of eight-byte entries that are free.

This function block is optional.



Input Parameters

Name	Data Type	Description
CI	BOOL	Enables SOESTAT.
BLOCK	DINT	The block number (1-16).

Output Parameters

Name	Data Type	Description
CO	BOOL	True if SOESTAT executes successfully.
STAT	DINT	Status: 0 = The block is not started or not configured with data type and size. 1 = The block is collecting data. 2 = The block is stopped or cleared. 3 = The block is full.
FREE	DINT	The number of unused 8-byte entries in the block.
USED	DINT	The number of unused 8-byte entries in the block.

SOE Recorder Setup

This section explains the steps required to set up SOE Recorder so you can retrieve and view event information. You can use the default settings for event retrieval or customize the options that are listed in this table.

Options	Description
Display options for event files	Specifies the column display settings.
Periodic snapshots	Specifies a snapshot based on a time period.
Auto snapshots	Specifies a snapshot based on a trip event or a maximum number of events.
Event File Directory	Specifies where to store the event files.
Auto export to a journal file	Automatically exports events to a file.
TCP/IP network addresses	Specifies TCP/IP addresses for controllers.

Topics include:

- [Installing SOE Recorder \(page 452\)](#)
- [Uninstalling SOE Recorder \(page 453\)](#)
- [Copying the SOE Definition File \(page 454\)](#)
- [Specifying Display Options \(page 454\)](#)
- [Specifying Periodic Snapshots \(page 455\)](#)
- [Specifying Shift Snapshots \(page 456\)](#)
- [Specifying Auto Snapshot Options \(page 456\)](#)
- [Specifying an Event File Folder \(page 457\)](#)
- [Specifying Controllers for Retrieval \(page 458\)](#)
- [Specifying Automatic Export of Event Data \(page 459\)](#)

Installing SOE Recorder

This procedure explains how to install SOE Recorder.

Before Starting

You must be logged on to NT as an administrator or a user with administrative privileges. You can install and run SOE Recorder on the same PC as **TriStation** or on a different PC.

Procedure

- 1 Exit any open applications.
- 2 Place the SOE Recorder CD in the CD ROM drive.
- 3 From the **Start** menu, click **Settings**.

- 4 Click the **Control Panel** icon.
- 5 Double-click the **Add/Remove Programs** icon.
The **Add/Remove Programs Properties** dialog box is displayed.
- 6 Click the **Install/Uninstall** tab.
- 7 Click **Install**.
- 8 Follow the instructions given in the Installation Wizard.
Triconex recommends installing SOE Recorder in the default destination directory, which is c:\My Documents\Triconex\TcxSoe.
- 9 To restart your computer when the installation has finished, click **Yes**. To restart your computer sometime later, click **No**.
- 10 To complete the installation of SOE Recorder, click **OK**.

SOE Recorder is now installed on your computer. To start it, click Triconex Sequence of Events from the Triconex shortcut of your Programs menu.

Uninstalling SOE Recorder

This procedure explains how to uninstall SOE Recorder. If you have installed a previous version of SOE Recorder, you must uninstall it before installing a new version.

Procedure

- 1 From the **Start** menu, click **Settings**.
- 2 Click the **Control Panel** icon.
- 3 Double-click the **Add/Remove Programs** icon.
The **Add/Remove Programs Properties** dialog box is displayed.
- 4 Click the **Install/Uninstall** tab.
- 5 Click **Triconex Sequence of Events**, and then click the **Add/Remove**.
The **Confirm File Deletion** dialog box is displayed, asking you to confirm deletion of the selected application and all its components.
- 6 To delete, click **Yes** or press **Y**.
The **Remove Programs from Your Computer** dialog box is displayed while the previous version is uninstalled.
- 7 When the message “**Uninstall successfully completed**” is displayed, click **OK**. (If this message is not displayed, contact your system administrator for help in uninstalling the program.)
- 8 To close the **Add/Remove Programs Properties** dialog box, click **OK** when the uninstall is finished.

Copying the SOE Definition File

When you download an SOE-enabled **TriStation** project, **TriStation** creates an SOE definition file in a **TriStation** folder. The SOE definition file contains SOE block definitions and point properties for the event variables assigned to the SOE blocks. Before you can use SOE Recorder to retrieve events, you must copy the SOE definition file from the **TriStation** project folder to the SoeConfig folder for each controller from which you want to retrieve data. The **TriStation** version of an application must match the SOE Recorder version.

The **TriStation** naming convention for the SOE definition file includes the first eight characters of the **TriStation** project name, the major version number, minor version number, and time code. An example is:

```
SOE_EX_9_0_33F4CE47.SOE
```

CAUTION Do not rename the SOE definition file under any circumstances. Renaming it prevents SOE Recorder from retrieving event data.

Specifying Display Options

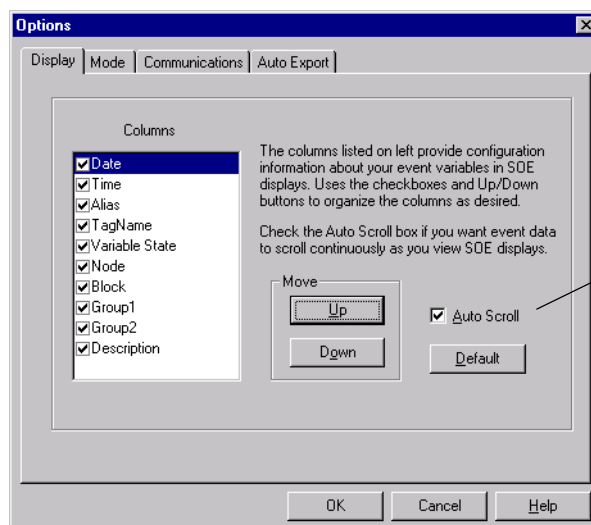
This procedure explains how to specify the columns to be displayed in an event file. The columns in an event file organize retrieval information. You can:

- Dynamically reconfigure the columns displayed during live event retrieval.
- Specify column display options for each event file.

When you create an event file, the column display settings specified for that file are automatically applied to the next file you create unless you change the settings. Adjustments you make to column widths while an event file is open are saved when you close the file.

Procedure

- 1 From the **Tools** menu, click **Options**.
- 2 On the **Options** dialog box, click the Display tab.



Property/Command	Description
Column	Check the column name to include a column in the display. Uncheck the column name to remove a column from the display.
Move Command	Check the column name to be moved, and then click Up or Down to organize the columns.
Auto Scroll	Check the Auto Scroll option if you want event data to scroll continuously during live event retrieval. This option is only available during live event retrieval.

Specifying Periodic Snapshots

This procedure explains how to set the Periodic Snapshot property, which allows you to specify whether a file of events is generated during a specific time period.

Periodic snapshot filenames include the date and time of the snapshot and one of the following file extensions:

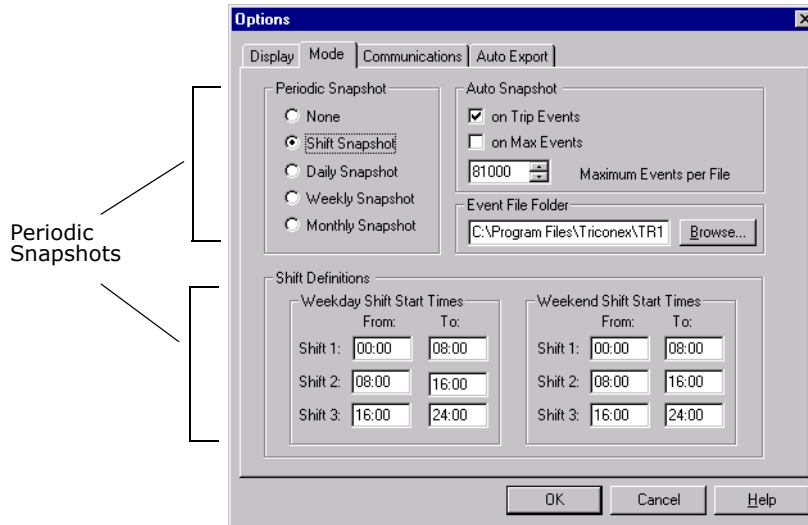
- *_D.SED* for daily snapshot
- *_W.SED* for weekly snapshot
- *_M.SED* for monthly snapshot
- *_S.SED* for shift snapshot

A sample file name for a weekly snapshot is Node7_19970613_10:20:15_W.SED. The parts of the file name have the following meanings:

Node7_	19970613_10:20:15_	W.SED
User-defined name of event file (.SED file)	Date and time of snapshot	Weekly extension

Procedure

- 1 From the **Tools** menu, click **Options**, and then click the **Mode** tab.



- 2 Click the desired period for snapshots. If you select Shift Snapshot, see [Specifying Shift Snapshots](#).

Specifying Shift Snapshots

This procedure explains how to set up a shift snapshot. A shift snapshot retrieves events based on time periods you specify, which can include up to three weekday shifts and three weekend shifts.

The name that SOE Recorder gives to a shift snapshot is the SED file name, concatenated with the date and time of the snapshot, and the extension “_S.SED.”

Procedure

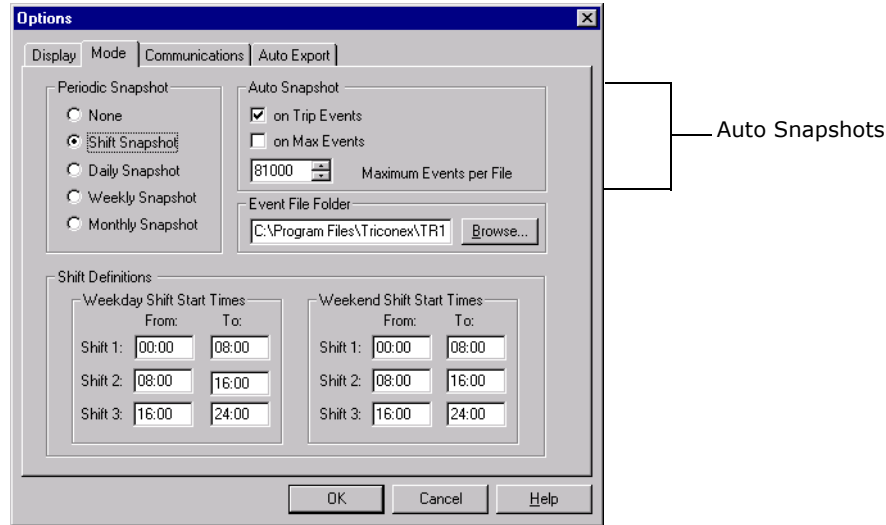
- 1 From the **Tools** menu, click **Options**, and then click the **Mode** tab.
- 2 Click the **Shift Snapshot** option, and then enter the **start** and **stop times** for **Shift Definitions**.

Specifying Auto Snapshot Options

The Auto Snapshot property automatically saves a snapshot when a trip occurs or when a specified number of events occurs. You can have snapshots taken when either or both situations occur.

Procedure

- 1 From the **Tools** menu, click the **Options** command, and then click the **Mode** tab.



- 2 Specify these properties on the **Mode** tab.

Property	Description
On Trip Events	Check Trip Events to save a snapshot when a trip occurs.
On Max Events	Check on Max Events to save a snapshot when a specified number of events occurs.
Maximum Events Per File	If you select On Max Events, enter the Maximum Number of Events (up to 100,000) to retrieve.

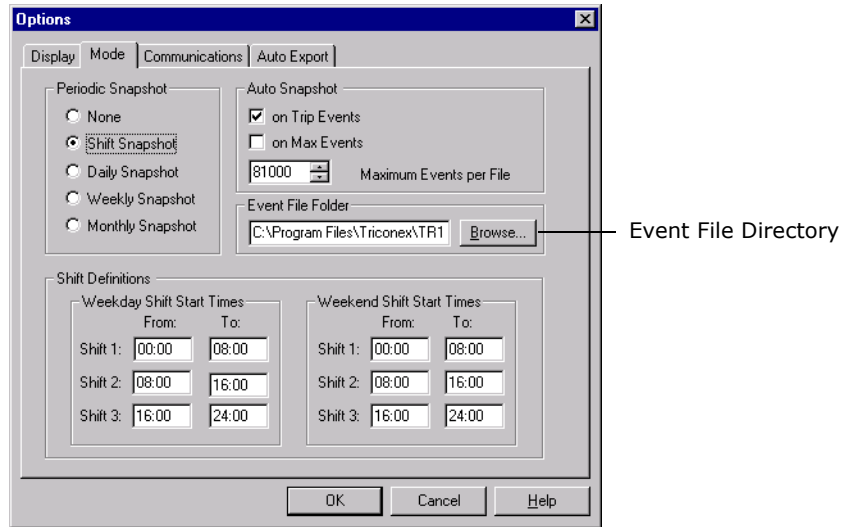
Specifying an Event File Folder

The Event File Folder property allows you to specify the folder used to store event files. The default Event File Directory is:

```
<drive letter>:\My Documents\Triconex\TR1SOE\SoeData
```

Procedure

- 1 From the **Tools** menu, click the **Options** command, and then click the **Mode** tab.



- 2 Browse to the desired folder, and then click **OK**.

Specifying Controllers for Retrieval

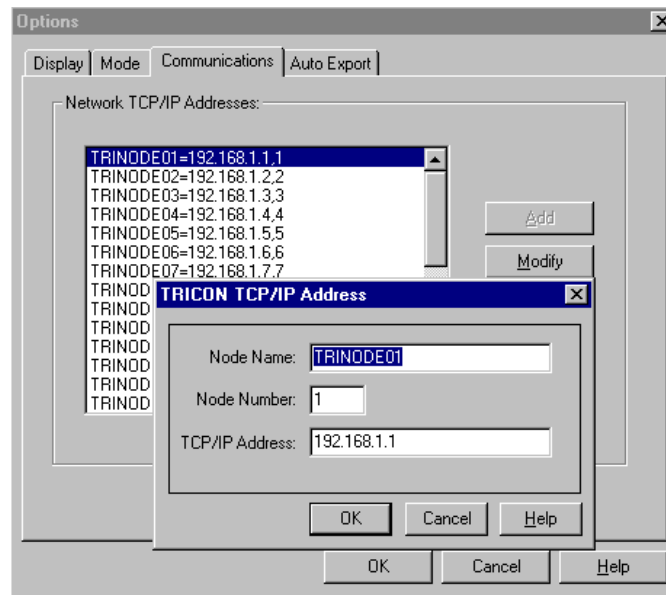
This procedure explains how to specify IP addresses. You must specify the IP address of each Triconex controller from which you want to retrieve events. If you are using one Triconex controller and one workstation running SOE Recorder, specifying a default address supplied by **TriStation** and SOE Recorder is sufficient. If you want SOE Recorder to retrieve events from multiple controllers on a network, you must specify the IP addresses that are used by the controllers on the network.

Before Starting

In SOE Recorder, close any open event files. The addresses you specify apply to the next event file you create.

Procedure

- 1 On the **Tools** menu, click **Options**, and then click the **Communications** tab.
- 2 Click the **Trinode number** for the desired controller, and then click **Modify**.



- 3 Change the **Node Name** if desired, enter the **TCP/IP Address**, and then click **OK**.

Specifying Automatic Export of Event Data

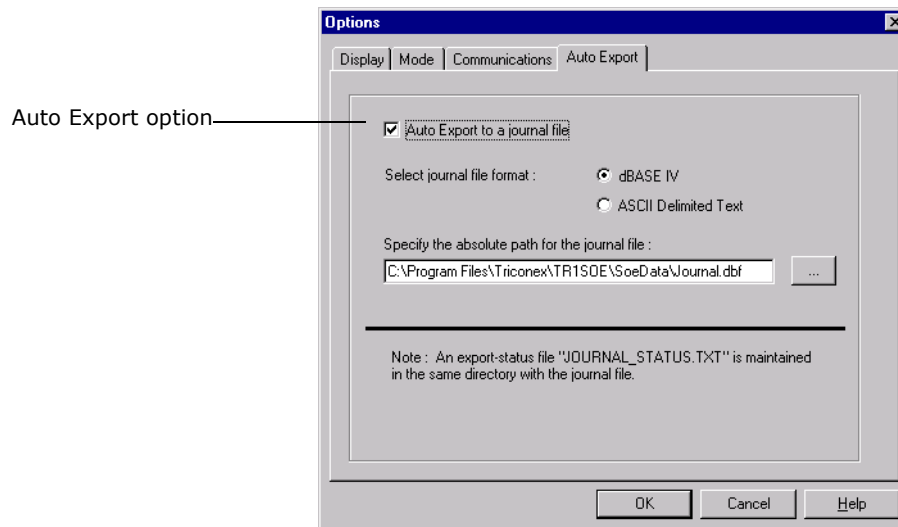
This procedure explains how to specify the automatic export of event data to a dBASE IV or ASCII journal file. You can enable this option before or after you begin retrieving events. Event data are exported to the specified file every 30 seconds.

To export events, SOE Recorder must have exclusive access to the journal file. If another application accesses the file for several minutes, SOE Recorder waits until the file is free to resume the export activity.

If you don't specify a file name, SOE Recorder uses the name JOURNAL. If you don't specify a path, SOE Recorder uses the SoeData directory under the directory where you installed the SOE Recorder executable. If you move a journal file from the original directory, SOE Recorder automatically creates a new journal file in the directory of origin.

Procedure

- 1 From the **Tools** menu, click the **Options** command, and then click the **Auto Export** tab.
- 2 To enable the option, check **Auto Export** to a journal file.



3 Specify these properties on the **Auto Export** tab.

Property	Description
Auto Export to a Journal File	Check to automatically export event data to a dBASE IV or ASCII journal file.
Journal File Format	Click the dBASE IV or ASCII Delimited Text setting.
Specify the Absolute Path for a Journal File	Specify the absolute path for your journal file, including a file name. Use Browse if needed. To ensure that any dBASE IV application can open this file, you should limit the file name to eight characters.

4 Click **OK**.

SOE Recorder Event Retrieval

This section describes how to retrieve events from one or more controllers. When you create a new event file, SOE Recorder looks for controllers that are connected to the network. If the controller is connected and the SOE definition file matches the SOE definition in the SoeConfig directory, you can retrieve events from it.

Topics include:

- [Setting Up Event Retrieval \(page 461\)](#)
- [Retrieving Events \(page 462\)](#)
- [Turning Off Event Retrieval \(page 463\)](#)

Setting Up Event Retrieval

This procedure explains how to set up event retrieval.

Before Starting

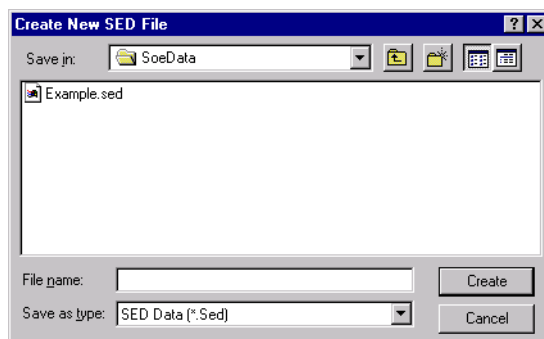
Verify that the following tasks are done:

- The SOE definition file is copied from the **TriStation** application folder to the SoeConfig folder.
- IP addresses are specified for the controllers from which you are going to retrieve events.
- The options for event retrieval are set.

Procedure

- 1 To create an event file, click the **New** command from the **File** menu, or click the **New** icon from the **Toolbar**.

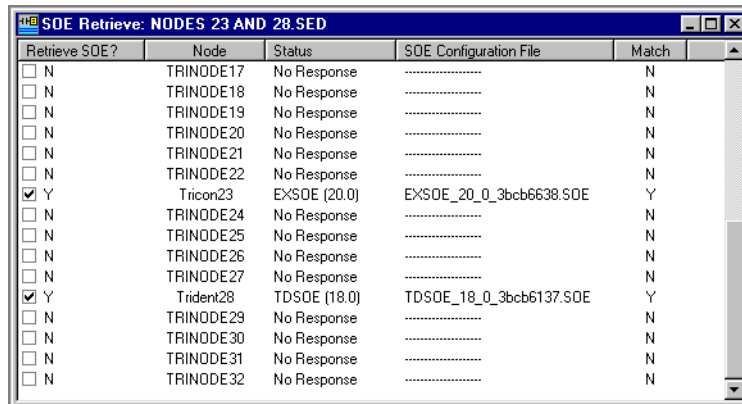
The **Create New SED File** dialog box is displayed.



- 2 In the **File** name field, enter a name for the sequence of events data file, and then click **Create**.

If you do not enter **SED** as the file extension, it is automatically appended to the file name.

The **SOE Retrieve** dialog box is displayed, which lists the names of controllers (nodes) that are connected to your network.



If a controller is not on the list, it might not be connected to the network or the SOE definition file might be missing from the SoeConfig directory.

- To specify a controller for event retrieval, click the check box next to the node name. You can select only those nodes that have a file name in the **SOE Configuration File** column.

Note If the status is “No response” in the **Status** column, it means SOE Recorder could not connect to the node. If the status is “No SOE Data,” it means SOE Recorder is connected to the node, but no SOE blocks have been configured.

Retrieving Events

This procedure explains how to retrieve events.

Procedure

- Click **Begin Event Retrieval** on the **Command** menu, or click **Begin Event Retrieval**.

SOE Recorder displays events in the event file as they are retrieved from the controller. If the Auto Scroll option is checked, the list automatically scrolls as events occur.

Date	Time	Alias	TagName	Variable State	Node	Block
10/15/2001	15:45:34.799	02008	EVENT_VAR8	ON	23 - Tricon23	01 - soe_block_1
10/15/2001	15:45:34.799	02007	EVENT_VAR7	OFF	23 - Tricon23	01 - soe_block_1
10/15/2001	15:45:34.799	02006	EVENT_VAR6	ON	23 - Tricon23	01 - soe_block_1
10/15/2001	15:45:34.799	02005	EVENT_VAR5	OFF	23 - Tricon23	01 - soe_block_1
10/15/2001	15:45:34.799	02004	EVENT_VAR4	ON	23 - Tricon23	01 - soe_block_1
10/15/2001	15:45:34.799	02003	EVENT_VAR3	OFF	23 - Tricon23	01 - soe_block_1
10/15/2001	15:45:34.799	02002	EVENT_VAR2	ON	23 - Tricon23	01 - soe_block_1
10/15/2001	15:45:34.799	02001	EVENT_VAR1	OFF	23 - Tricon23	01 - soe_block_1
10/15/2001	15:37:23.203	05001	EVENT_VAR2	UNSAFE	28 - Trident28	01 - soe_block_1
10/15/2001	15:37:23.203	05002	EVENT_VAR3	SAFE	28 - Trident28	01 - soe_block_1
10/15/2001	15:37:23.203	05003	EVENT_VAR4	UNSAFE	28 - Trident28	01 - soe_block_1
10/15/2001	15:37:23.203	05004	EVENT_VAR5	SAFE	28 - Trident28	01 - soe_block_1
10/15/2001	15:37:23.203	05005	EVENT_VAR6	UNSAFE	28 - Trident28	01 - soe_block_1
10/15/2001	15:37:23.203	05006	EVENT_VAR7	SAFE	28 - Trident28	01 - soe_block_1
10/15/2001	15:37:23.203	05007	EVENT_VAR8	UNSAFE	28 - Trident28	01 - soe_block_1
10/15/2001	15:37:23.203	05008	EVENT_VAR9	UNSAFE	28 - Trident28	01 - soe_block_1

Event retrieval continues until you select the End Event Retrieval command or at the end of a snapshot period.

Turning Off Event Retrieval

This procedure explains how to turn off event retrieval.

Procedure

- 1 Click **End Event Retrieval** on the **Command** menu, or click the **End Event Retrieval** icon.

D

Triconex DDE Server

Overview	466
Configuring the DDE Server Application	466
Configuring Triconex Host Information	467
Configuring Server Properties for 802.2 Protocol (Tricon Only)	470
Testing a TCP/IP Connection	471
Configuration Requirements for DDE Network Redundancy	471
Requesting Data with a DDE Client Application	473
Requesting Network Status	473
Monitoring Responses from the Controller	474
DDE Menu Commands	474

Overview

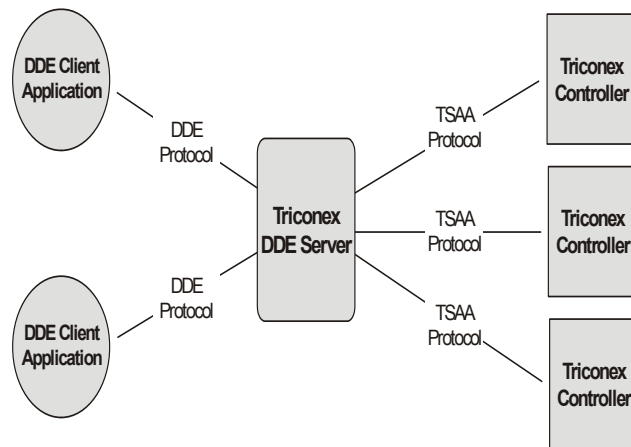
Triconex DDE Server is a Windows (NT and 2000) application that enables DDE-compliant clients to request data and, if allowed, to change data in a Triconex (Tricon or Trident) control program. A client can request data about input and output variables, memory variables, and system attributes. Triconex DDE Server is based on a client/server model in which a *client* requests information from a server and a *server* sends information to a client.

Client applications use DDE (Dynamic Data Exchange) protocol to communicate with a DDE Server. Any Windows application that supports DDE protocol—such as Microsoft Excel—can use Triconex DDE Server.

Triconex DDE Server communicates with one or more Triconex controllers through TSAA (Triconex System Access Application) protocol. To return data to clients, the DDE Server uses DDE protocol.

The DDE Server workstation must be connected to an Ethernet port on a Triconex controller. For Tricon, the Net2 port on the ACM (Advanced Communication Module) or NCM (Network Communication Module) must be used. For Trident, the Net1 or Net2 port on the CM (Communication Module) must be used.

This figure depicts multiple clients communicating with multiple Triconex controllers through a Triconex DDE Server.



These sections explain how to use Triconex DDE Server.

Configuring the DDE Server Application

When you configure the DDE Server application, you specify communication properties used by a Triconex controller (also called a *host or node*) to communicate with DDE clients. These properties allow DDE clients to identify which controller to communicate with and what communication protocol to use.

If you plan to use a redundant DDE network, you need to use DDE Server PCs and install redundant communication modules in the controller. For more information, see [Configuration Requirements for DDE Network Redundancy](#) (page 471).

You can also modify or delete the configuration of a Triconex controller. Before modifying a configuration, make sure it is not being used by a DDE client. If you delete a configuration, the associated controller can no longer be accessed by a DDE client.

To allow a DDE client to change the values of Triconex variables, you must enable write access in one of these ways:

- For Tricon, physically set the keyswitch to the Remote position.
- For Trident, set the MP.SET_REMOTE_WRIT_ENBL control attribute to True in the **TriStation** application.

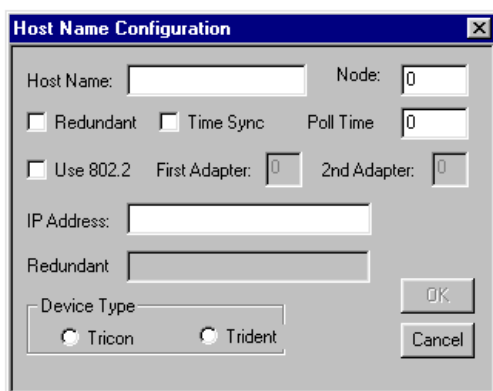
Configuring Triconex Host Information

This procedure explains how to specify host information for the Triconex controller, which must be done before a DDE client can access data from the controller.

Procedure

- 1 Start DDE Server from the Start menu by selecting Programs, then Triconex DDE Server. The DDE Server main window is displayed.
- 2 From the File menu, click Configure. The Configuring Host Information screen is displayed.
- 3 Either select an existing node and click Modify or click Add to add a host.

Note The default button clears all previous configuration settings for all of the hosts. The Host Name Configuration is displayed.



- 4 Specify these properties in the Host Name Configuration screen, as needed.

2nd Adapter (Tricon Only)

The 2nd Adapter property specifies the number of the second network adapter card in your redundant PC. This property is enabled only if the Use 802.2 property and Redundant property are selected.

The second adapter number is usually one (1).

This property is only available for Tricon.

Device Type

The Device Type property specifies whether the host is a Tricon or a Trident controller.

First Adapter (Tricon Only)

The First Adapter property specifies the number of the first network adapter card in your primary PC. This property is enabled only if the Use 802.2 property is selected.

You can have multiple Ethernet adapters in your DDE PC. One is typical; two are needed for redundancy.

The first adapter number is usually zero (0).

This property is only available for Tricon.

Host Name

The Host Name property specifies the user-defined name for a controller which must be unique for each controller. (This name is used by the DDE client application to request data from the controller.)

The default values are:

- For Tricon, TRINODE01 (for node 1) through TRINODE16 (for node 16)
- For Trident, TRINODE01 (for node 1) through TRINODE32 (for node 32)

IP Address

The IP address property specifies the unique 32-bit network address of the primary communication module in the Triconex controller.

For a Tricon controller, you must specify this property if the Use 802.2 property is not selected.

Node Number

The Node Number property specifies the node number which must be unique for each controller.

For Tricon, the node number must match the physical switch settings on the NCM or ACM and the node number specified in the **TriStation** project. Default values are 1 to 16.

For Trident, the node number must match the address plug on the MP Baseplate and the node number specified in the **TriStation** project. Default values are from 1 to 32.

Poll Time

The Poll Time property specifies how often the Triconex controller refreshes the data stored as aliases. The polling interval must be greater than the scan time of the controller.

The default is 1,000 milliseconds (one second).

Redundant

The Redundant property specifies whether there are redundant paths to the controller. If the physical configuration is redundant, select this property.

This means that two network adapter cards must be connected to network ports on two communication modules, as follows:

- For Tricon, the Net2 port on two ACMs or two NCMs
- For Trident, the Net1 or Net2 port on two CMs

For more information, see [Configuration Requirements for DDE Network Redundancy \(page 471\)](#).

The default is not redundant.

Redundant (IP Address)

The Redundant (IP Address) property specifies the IP address of the redundant communication module in the Triconex controller.

The redundant module must have the same IP address that is specified here.

For a Tricon controller, you cannot specify a redundant IP address if the Use 802.2 property is selected.

Time Sync

The Time Sync property specifies whether a Triconex node (host) is synchronized with the clock on the DDE Server PC. If there is more than one Triconex controller in a network, you should select the master node for synchronization with the DDE Server PC clock. The master node can then synchronize the time of the other Triconex controllers.

For time-critical applications, Triconex does not recommend selecting the Time Sync property because PCs are not generally a reliable source for time synchronization.

Note Do not use the Time Sync property if any other type of synchronization is being used for the Triconex controllers.

The default is not synchronized.

Use 802.2 (Tricon Only)

The Use 802.2 property specifies whether 802.2 (DLC) protocol or TCP/IP protocol is used to communicate with the DDE client. To use this property, you must install DLC protocol on the DDE PC.

If you configure a node to use DLC protocol, you must also configure the server parameters. To do so, see the instructions in the section called [Configuring Server Properties for 802.2 Protocol \(Tricon Only\) \(page 470\)](#).

The default is not selected, which means that TCP/IP protocol is used.

This property is only available for Tricon.

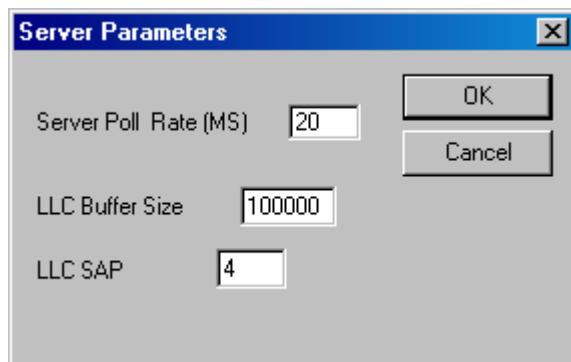
Configuring Server Properties for 802.2 Protocol (Tricon Only)

This procedure explains how to configure the server properties, which must be specified if you select the Use 802.2 property for any of the hosts.

The LLC Buffer Size property must be have 50,000 bytes specified for each Triconex host using the 802.2 protocol. For example, if three hosts use 802.2 protocol, the buffer must be set to 150,000 bytes.

Procedure

- 1 If not already done, start DDE Server from the Start menu by selecting Programs, then Triconex DDE Server. The DDE Server main window is displayed.
- 2 Click the Server button. The Server Parameters screen is displayed.



- 3 Specify these properties, as needed.

Server Poll Rate (MS)

The Server Poll Rate property specifies the rate in milliseconds at which DDE server updates clients such as Microsoft Excel or Wonderware InTouch applications. The server poll rate must be greater than 20 milliseconds and less than 1,000 milliseconds.

LLC Buffer Size

The LLC Buffer Size property specifies the size of the buffer (in bytes), which depends on the number of Triconex controllers using 802.2 protocol. For each host using 802.2 protocol, you must specify a minimum of 50,000 bytes for each host.

The default is 100,000.

LLC SAP

The LLC SAP property specifies the address for the DDE Server on the PC. This number must be a unique address. The default is 4.

Testing a TCP/IP Connection

This procedure explains how to test a TCP/IP connection, which is useful after completing the configuration of Triconex hosts. You might want to test the IP addresses of these devices:

- Network adapter card in the client PC
- Triconex communication modules

Procedure

- 1 From the Start menu, select the MS-DOS Command Prompt.
- 2 Type the word *ping* followed by the IP address to be tested. For example, for an IP address of 206.32.216.43, you would enter this:

```
ping 206.32.216.43
```
- 3 If the network connection is valid, the reply includes the IP address followed by byte and time information.
If the connection is not okay, the reply is "Request timed out."

Configuration Requirements for DDE Network Redundancy

A redundant network of Tricon controllers can be configured using either TCP/IP protocol or 802.2 protocol.

A redundant network of Trident controllers must be configured using TCP/IP protocol.

Typically, hardware setup is done before software configuration. To configure a redundant DDE network, this hardware is required:

- For Tricon, two ACM or NCM modules in one or more Tricon controllers.
- For Trident, two CMs in one or more Trident controllers.
- For the DDE PC, two network adapter cards.

Using TCP/IP Protocol

This procedure explains how to configure DDE network redundancy with TCP/IP protocol.

The configuration procedure involves setting IP addresses. If the network configuration permits, use the Triconex default addresses. If not, get the IP addresses from your Network Administrator.

For Trident only, if a DDE Server PC is not on the same subnet as the Trident controller, you must specify the destination address during Ethernet port configuration. If necessary, get help from your Network Administrator.

Procedure

- 1 Install two network adapter cards and the TCP/IP protocol on the DDE Server PC.

- 2 On the DDE Server PC, use Windows procedures to set the IP addresses of the network adapter cards. A sample IP address is:
206.32.216.x (where x = 1 to 254)
- 3 Connect the network adapter cards on the DDE Server PC to Ethernet ports on the primary and redundant Triconex communication modules
- 4 In **TriStation**, set the IP addresses for the primary and redundant communication modules.
A sample IP address is 206.32.64.y where y is the node number. The node number is set as follows:
 - For Tricon, it is set with physical switches on the NCM or ACM.
 - For Trident, it is set with the address plug on the MP Baseplate.
- 5 From the DDE server application, configure each Triconex node with a host name. You must use the same IP address for the node configuration in DDE Server that is used in Step 5 above. For instructions, see [Configuring Triconex Host Information \(page 467\)](#).

Using 802.2 Protocol (Tricon Only)

This procedure explains how to configure redundant Tricon controllers which use 802.2 protocol.

Procedure

- 1 Install two network adapter cards and the DLC protocol in the DDE PC.
- 2 Connect the first network adapter card in the DDE PC to the left NCM or ACM.
- 3 Connect the second network adapter card to the right NCM or ACM.
- 4 In the DDE Server application, select the Redundant and Use 802.2 properties. (When these properties are selected, it is not necessary to configure IP addresses.)
- 5 Set the First Adapter property to 0 (zero) and the 2nd Adapter property to 1 (one).

The 802.2 protocol is only available for Tricon controllers.

Requesting Data with a DDE Client Application

When you use a DDE client application to request data, you identify the DDE Server application to use, the Triconex controller to be accessed, and the data to be accessed. This information is referred to as the DDE address. Each DDE client application uses a three-part DDE address format, but might use slightly different syntax.

The DDE address format includes these parts:

Application + Topic + Item

DDE Address	Description
Application	Specifies the Triconex DDE Server application name which is TR1DDE.
Topic	Specifies the node name for a Triconex controller as configured in the DDE Server application. For Tricon, the default node names for controllers 1–16 are TRINODE01 through TRINODE16. For Trident, the default node names for controllers 1–32 are TRINODE01 through TRINODE32. For more information on defining nodes, see Configuring Triconex Host Information (page 467) .
Item	Specifies the alias number for the requested Triconex variable. You can identify one or more items.

To begin DDE Server operation, save the address in the DDE client application and start the DDE Server application. Both the client and server applications must be running concurrently to request or exchange data. The DDE Server sends the request to the Triconex controller, then returns the data to the DDE client application.

As an example, this address could be entered in a blank cell of a Microsoft Excel worksheet to request the value for alias 40001 in TRINODE02:

```
=TR1DDE|TRINODE02!'40001'
```

Although you can run only one DDE Server application at a time, you can run as many DDE client applications as allowed by the virtual memory available on your PC.

Requesting Network Status

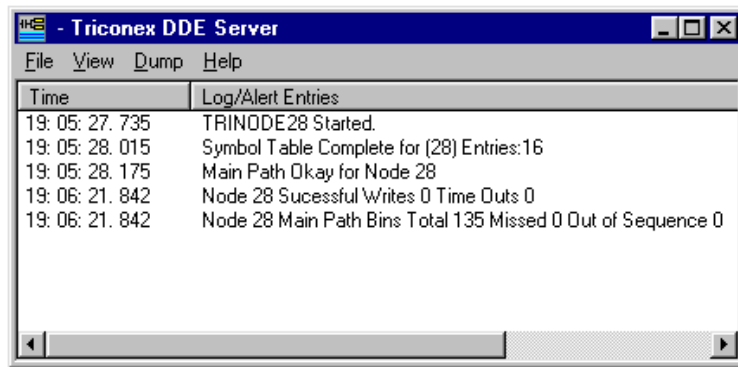
To find out whether the network ports on a Triconex controller are receiving data, enter either of these commands in any client application using this format.

=tr1dde TRINODE01!STATUS	Reads network status
=tr1dde TRINODE01!RSTATUS	Reads redundant network status

For details on syntax for the DDE address, see the user's manual for the client application you are using.

Monitoring Responses from the Controller

The Triconex DDE Server allows you to monitor responses from the Triconex controller which can include alert entries as well as log entries that indicate a successful response. The entries are logged in the order in which they occur. To view the most current entries, scroll to the bottom of the list. If you select the Stats command on the Dump menu, older entries might appear at the bottom of the list, as shown in this screen.



Changing View Options

To keep the DDE Server main window on top of all other windows, select the Always on Top command on the View menu. A check mark next to the command means it is selected.

DDE Menu Commands

The DDE Server includes these menus and commands.

File Menu

Command	Description
Configure...	Opens the Configure Host Information screen and allows you to configure up to 16 Tricon or 32 Trident controllers for use with the DDE Server application.
Exit	Closes the DDE Server application.

View Menu

Command	Description
Always on Top	Keeps the DDE Server main window on top of other windows.

Dump Menu

Command	Description
Stats	Displays statistics for all Triconex controllers.

Help Menu

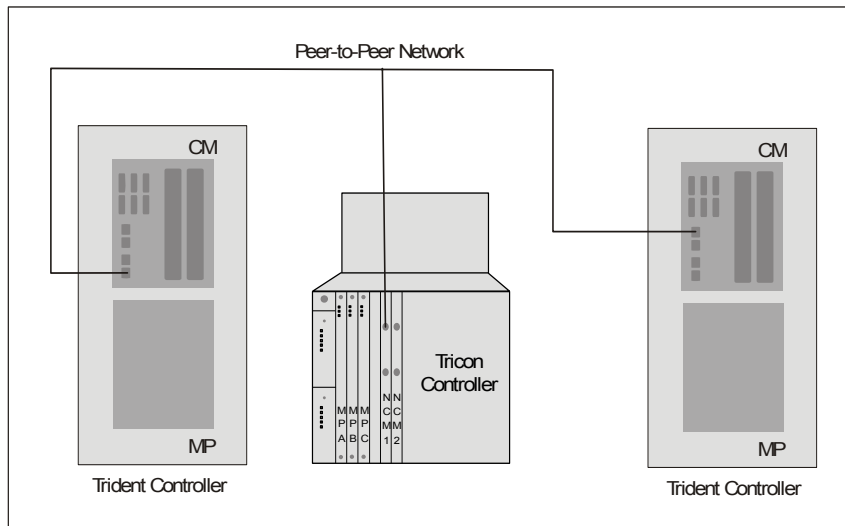
Command	Description
Triconex DDE Server Help	Opens the Help documentation.
About Triconex DDE Server	Displays the current version number of the DDE Server application and registered owner information.

Peer-to-Peer Communication

Overview	478
Peer-to-Peer Data Transfer Time	480
Configuring Peer-to-Peer Ports for Tricon	482
Configuring Peer-to-Peer Ports for Trident	483
Allocating Peer-to-Peer Memory	485
Using Send and Receive Function Blocks	486
Restrictions on Data Transmission Speed	488
Monitoring Peer-to-Peer Communication	489
Sample Peer-to-Peer Programs	492

Overview

Triconex Peer-to-Peer protocol is designed to allow multiple Tricon and Trident controllers in a closed network to exchange safety-critical data. (If you plan to implement a complex Peer-to-Peer network, please contact Triconex Technical Support.) To enable Peer-to-Peer communication, you must connect each controller to an Ethernet network by using a Net1 (Ethernet) port on the Tricon NCM or an Ethernet port on the Trident CM. The controllers exchange data by using Send and Receive function blocks in their **TriStation** applications.



To configure a **TriStation** application for Peer-to-Peer communication, you must do these tasks:

- Configure the physical port connection for Peer-to-Peer mode
- Allocate memory for Send and Receive function blocks
- Add Send and Receive function blocks to the programs
- Observe restrictions on data transmission speed

In addition, Triconex recommends that you calculate the data transfer time to determine whether the control algorithms will operate correctly.

A **TriStation** application must use a specific Send function block to send data to a matching Receive function block in another **TriStation** application. Each Send function block has a parameter that identifies the Receive function block to which it sends data. Each Receive function block has a parameter that identifies the Send function block from which it receives data.

Tricon controllers do Peer-to-Peer communication at 10 megabits per second, however, Trident controllers can do Peer-to-Peer communication at 10 or 100 megabits per second. If the network includes both types of controllers, you can run the entire network at 10 megabits per second, or you can use a hub that converts messages from 10 to 100 megabits per second when they are transferred from a Tricon to a Trident. For more information, see [Restrictions on Data Transmission Speed \(page 488\)](#).

For monitoring Peer-to-Peer data exchange, **TriStation** provides function blocks and Tricon system aliases or Trident attributes to track network communication paths and verify whether the Ethernet ports are receiving data from other controllers.

The sample programs described in this appendix on the **TriStation** CD. These programs show how to send data at high speed and under controlled conditions, and how to measure the maximum data transfer time.

Peer-to-Peer Data Transfer Time

In a Peer-to-Peer application, data transfer time includes the time required to initiate a send operation, send the message over the network, and have the message read by the receiving node. Additional time (at least two scans) is required for a sending node to get an acknowledgment from the MPs that the message has been acted on.

These time periods are a function of the following parameters of the sending and receiving controllers:

- Scan time
- Configuration size
- Number of bytes for aliased variables
- Number of Send function blocks, Receive function blocks, printing function blocks, and Modbus master function blocks
- Number of controllers on the Peer-to-Peer network

Send function blocks require multiple scans to transfer data from the sending controller to the receiving controller. The number of send operations initiated in a scan is limited to 5. The number of pending send operations is limited to 10.

A typical data transfer time (based on a typical scan time) is 1 to 2 seconds, and the time-out limit for a Peer-to-Peer send (including 3 retries) is 5 seconds. Consequently, the process-tolerance time of the receiving controller must be greater than 5 seconds. Process-tolerance time is the maximum length of time that can elapse before your control algorithms fail to operate correctly. If these limitations are not acceptable, further analysis of your process is required.

Estimating Memory for Peer-to-Peer Data Transfer Time

This procedure explains how to estimate memory for Peer-to-Peer data transfer time between a pair of Triconex controllers. The more memory allocated for aliased points the slower the transfer time.

Procedure

- 1 Open the **TriStation** project used for the *sending* controller and go to the Memory Allocation screen.
- 2 Find the bytes allocated for BOOL, DINT, and REAL points.
 - Add the number of bytes allocated for all BOOL input, output, and aliased memory points.
 - Repeat these steps for DINT and REAL points.
- 3 Open the **TriStation** project used for the receiving controller and go to the Memory Allocation screen.
- 4 Find the bytes allocated for BOOL, DINT, and REAL points.
 - Add the number of bytes allocated for all BOOL input, output, and aliased memory points.

- Repeat these steps for DINT and REAL points.
- 5 Use this worksheet to estimate the transfer time.

Steps	Point Type	Allocated Bytes	Operation	Result
1. Enter the number of bytes for each point type on the <i>sending</i> controller and divide or multiply as indicated. Add the results.	BOOL	_____	÷ 8 =	_____
	DINT	_____	• 4 =	_____
	REAL	_____	• 4 =	_____
	System Variables		+ 376	
	Total bytes of aliased points TBS =			_____
2. Multiple total bytes sending TBS (step 1) by .01			TS =	_____
3. Enter the number of bytes for each point type on the <i>receiving</i> controller and divide or multiply as indicated. Add the results.	BOOL	_____	÷ 8 =	_____
	DINT	_____	• 4 =	_____
	REAL	_____	• 4 =	_____
	System Variables		+ 376	
	Total bytes of aliased points TBR =			_____
4. Multiple total bytes receiving TBR (step 3) by .01			TR =	_____
5. Get the scan time of sending node in milliseconds by viewing the Scan Time in the diagnostic screen or by using a function block to get the value. For Tricon, get the SCANDELTA parameter in the TR_SCAN_STATUS; for Trident, get the ACTUAL_SCAN_TIME parameter in the SYS_MP_EXT_STATUS function block.			SS =	_____
6. Get the scan time of receiving node in milliseconds by viewing the Scan Period in the Execution List.			SR =	_____
7. Multiply the larger of TS or SS by 2.				_____
8. Multiply the larger of TR or SR by 2.				_____
9. Add the results of step 7 and 8 to get the data transfer time = DT				_____
10. If the number of pending send requests in the application is greater than 10, divide the number of send requests by 10.				_____
11. Multiply the results of steps 9 and 10 to get the adjusted data transfer time.			Adjusted DT	_____
12. Compare the adjusted DT to the process-tolerance time to determine if it is acceptable.				

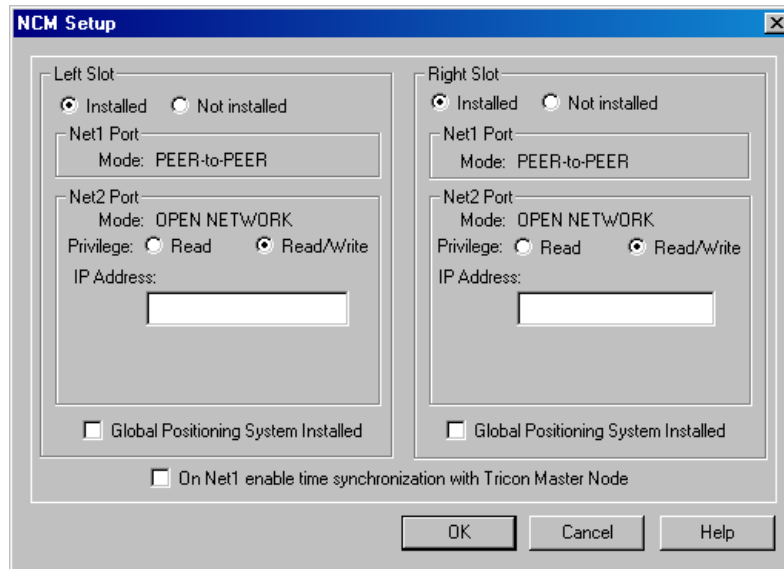
Configuring Peer-to-Peer Ports for Tricon

This procedure explains how to configure an Ethernet port on the Tricon NCM for communication with other Triconex controllers on a Peer-to-Peer network.

Procedure

- 1 Open the **TriStation** project and go to the NCM Setup screen.

This is the NCM Setup screen in **TriStation** v4.0.



- 2 Specify these properties, as needed.

Installed/Not Installed (Slots)

Specify the slots that are installed. The Net1 port on an NCM is pre-configured for Peer-to-Peer communication, and operates at 10 megabits per second.

Global Positioning System Installed

For an NCMG in a Peer-to-Peer network, check this box only if the NCMG acts as the master node.

Net1 Enable Time Synchronization

To synchronize this controller with the master node, select the option called *On Net1 enable time synchronization*. If this controller is the master node, you should still select this option.

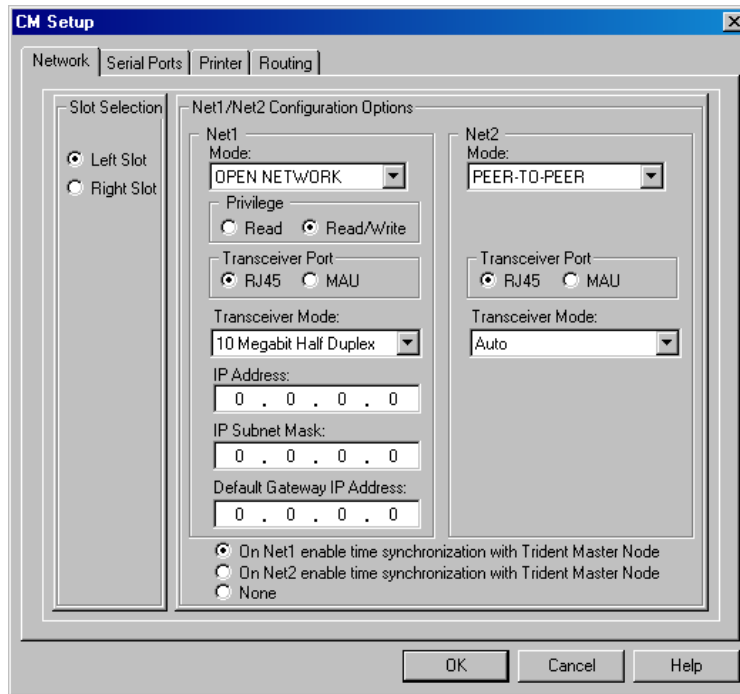
- 3 Click OK to save the setting.

Configuring Peer-to-Peer Ports for Trident

This procedure explains how to configure an Ethernet port on the Trident CM for communication with other Triconex controllers on a Peer-to-Peer network.

Procedure

- 1 Open the **TriStation** project and go to the Network tab on the CM Setup screen. This is the CM Setup screen in **TriStation** v4.0.



- 2 Specify these properties, as needed.

Mode

Select Peer-to-Peer for Net1 or Net2 Mode.

For each CM on a baseplate, you can select Peer-to-Peer for one Ethernet port. If you are using two CMs, you must select Peer-to-Peer for the same Ethernet port (Net1 or Net2) on each CM.

Transceiver Port

This option describes the physical network connection. Select RJ-45 (the default) if you have attached your communication cable to an RJ-45 connector on the CM Baseplate.

Select the MAU option if you have attached an MII MAU to a 40-pin subminiature D connector or an AUI MAU to a DB-15 connector on the CM Baseplate. You might have to set the physical address of a MAU before attaching it to a CM port. You must purchase MAUs from a third-party manufacturer.

Transceiver Mode

This option specifies whether messages are transmitted and received simultaneously (10 Megabit Full Duplex mode) or in one direction at a time (10 Megabit Half Duplex mode). Select a mode that is compatible with the available hardware.

Time Synchronization

This option specifies whether to use time synchronization in your network. Select one of the following:

Select one of the following options:

- On Net1: Synchronizes the time with the master node on the network that is connected to Net1
- On Net2: Synchronizes the time with the master node on the network that is connected to Net2
- None: Time synchronization is not performed by a Trident controller, but can be performed by an external device (such as an OPC client)

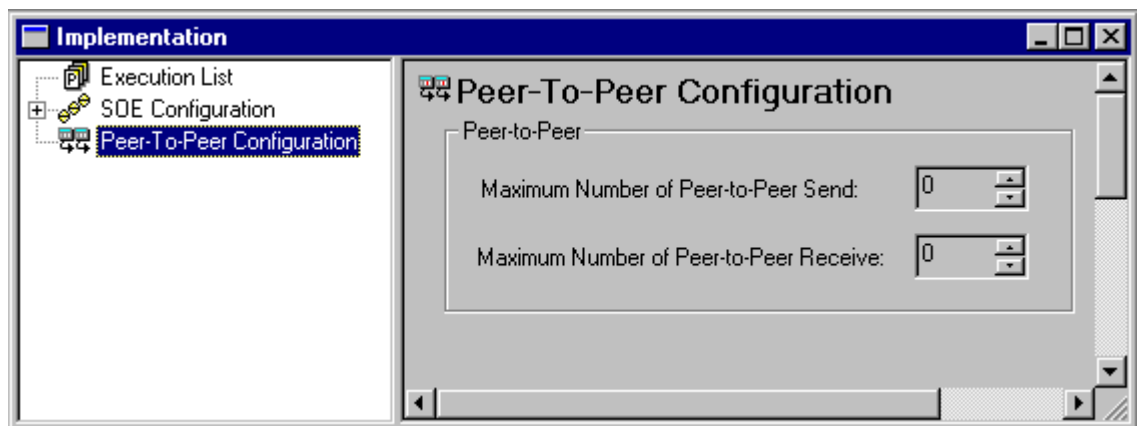
Allocating Peer-to-Peer Memory

This procedure explains how to allocate memory for Peer-to-Peer functions, which is based on the maximum number of Send and Receive numbers you specify. To save memory and minimize scan time, you should use the lowest possible numbers.

The maximum number does not have to be the same for Sends and Receives. For example, a **TriStation** application might need to send messages to three applications running on other controllers, but need to receive messages from only one application.

Procedure

- 1 Expand the Application tree, double-click Implementation, and then click Peer-to-Peer Configuration.



- 2 Set these properties by clicking the up and down arrows.
 - Maximum Number of Peer-to-Peer Receives (function blocks)
 - Maximum Number of Peer-to-Peer Sends (function blocks)
- 3 If you want to change the settings for an application running on the controller, you must build the application and do a Download All.

Using Send and Receive Function Blocks

A **TriStation** application must use a specific Send function block to send data of a certain type to a matching Receive function block in another **TriStation** application. Each Send function block has a parameter that identifies the Receive function block to which it sends data. Each Receive function block has a parameter that identifies the Send function block from which it receives data.

For more on information on function blocks, see the *TriStation 1131 Libraries Reference*.

Send and Receive Function Blocks

The Send and Receive function blocks that you can include in a **TriStation** application have data types of BOOL, DINT, and REAL. These function blocks are available.

Send and Receive Function Blocks

Send Function Blocks	Receive Function Blocks
TR_USEND_BOOL	TR_URCV_BOOL
TR_USEND_DINT	TR_URCV_DINT
TR_USEND_REAL	TR_URCV_REAL
TR_USEND_BOOL_32	TR_URCV_BOOL_32
TR_USEND_DINT_32	TR_URCV_DINT_32
TR_USEND_REAL_32	TR_URCV_REAL_32

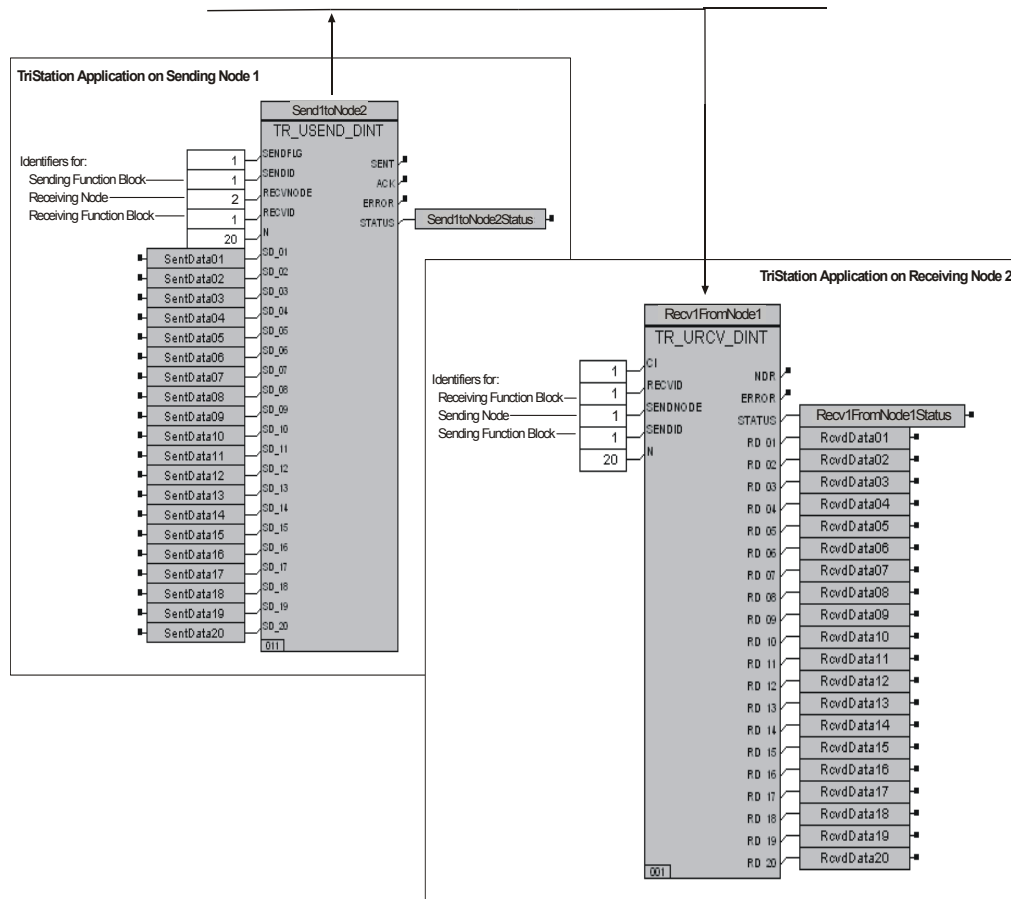
The _32 ending means that the function block can send 32 data values. Function block names that do not include the _32 ending can send 20 data values.

All Send function blocks – and all Receive function blocks – have the same parameters, except for the data transfer parameters which are BOOL, DINT, or REAL.

Sample Send and Receive Pair

This figure shows a sample pair of Send and Receive function blocks. A Send function block in one **TriStation** application is sending input values from the field over a Peer-to-Peer network to a matching Receive function block in another **TriStation** application. The Recvid and Sendid parameters are used to cross-reference the Send and Receive function blocks. The Recvnode and Sendnode parameters are used to cross-reference the sending and receiving nodes (**TriStation** applications).

For more information, see [Sample Peer-to-Peer Programs \(page 492\)](#).



Restrictions on Data Transmission Speed

Tricon controllers do Peer-to-Peer communication at 10 megabits per second, however, Trident controllers can do Peer-to-Peer communication at 10 or 100 megabits per second. Triconex suggests using the Net1 port on both Tricon and Trident communication modules, because 10 megabits per second is the only speed available on Net1. With this setup, Net2 is available for faster communication with external devices on an Ethernet network.

If the network includes both types of controllers, you can choose either of these solutions.

Solution	Description
Run the entire network at 10 megabits	Data exchange among Triconex controllers can be effectively done at a rate of 10 megabits per second. Triconex suggests using the Net1 port on both Trident and Tricon communication modules, because 10 megabits per second is the only speed available on Net1. With this setup, Net2 is available for faster communication with external devices on an Ethernet network.
Convert messages from 10 to 100 megabits	The data rate can be converted when messages are transferred from a Tricon controller to a Trident controller. A typical method is to connect the Tricon and Trident controllers to a hub which can convert from 10 to 100 megabits. For Trident controllers, another method is to connect MAUs, which can convert from 10 to 100 megabits, to CM ports.

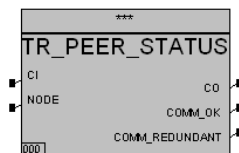
Monitoring Peer-to-Peer Communication

TriStation provides Tricon function blocks and Trident system attributes for monitoring the status of Peer-to-Peer communication paths (routes between Triconex communication modules on the network) and the status of communication ports. For detailed information, see the *TriStation 1131 Libraries Reference*.

Status of Communication Paths

A Peer-to-Peer network can communicate over one or two paths, depending on whether each controller contains one or two Tricon NCMs or Trident CMs. If there are two paths (two communication modules), then both are used simultaneously to exchange Peer-to-Peer data. The failure of one path does not affect Peer-to-Peer communication. To monitor the paths, use the TR_PEER_STATUS function block in the **TriStation** application. Path status is updated every 30 seconds.

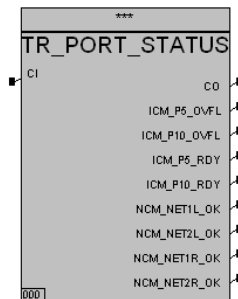
This figure shows the FBD representation of a TR_PEER_STATUS function block.



Status of Net1 Ports for Tricon

For Tricon controllers, you can determine whether the Net1 ports on an ACM or an NCM are receiving Peer-to-Peer data by using the TR_PORT_STATUS function block in the **TriStation** application.

This figure shows the FBD representation of a TR_PORT_STATUS function block.



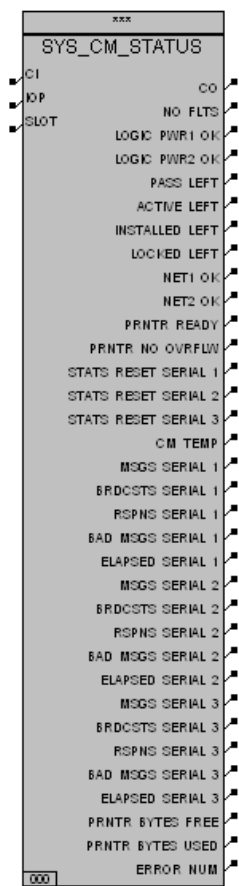
Status of Ethernet (Net1 and Net2) Ports for Trident

For Trident controllers, you can determine whether the Net1 and Net2 ports are receiving Peer-to-Peer data by using the SYS_CM_STATUS function block or CM_ system attributes in the **TriStation** application.

Using the SYS_CM_STATUS Function Block

The SYS_CM_STATUS function block is used to monitor the status of the Net1 and Net2 ports, serial ports, and the print buffer on the CM. The NET1OK or NET2OK parameter is True if a message was received by the Net1 or Net2 port in the last 15 seconds. (Most of the other parameters in the function block are for serial ports.) The network port status is updated every 30 seconds.

This figure shows the FBD representation of a SYS_CM_STATUS function block.



Using the Trident CM System Attributes

For Trident controllers, system attributes are points which allow you to monitor the status of system components and conditions, and control several system operations. The following system attributes for the CM can be used to verify whether the Net1 and Net2 ports have received messages in the last 15 seconds from other controllers in a Peer-to-Peer network.

System Attribute Name	Port on the CM
CM_L.NET1_OK	NET1 port on the left CM
CM_L.NET2_OK	NET2 port on the left CM
CM_R.NET1_OK	NET1 port on the right CM

System Attribute Name	Port on the CM
CM_R.NET2_OK	NET2 port on the right CM

Sample Peer-to-Peer Programs

The sample programs described in this chapter are on the **TriStation** CD. These programs show how to send data at high speed and under controlled conditions, and how to measure the maximum data transfer time.

Topics include:

- [Fast Send to One Controller \(page 492\)](#)
- [Sending Data Every Second to One Controller \(page 492\)](#)
- [Sending Data Only When Requested \(page 492\)](#)
- [Fast Send of Safety-Critical Data \(page 492\)](#)

Fast Send to One Controller

These programs show how to send data as fast as possible from Node 2 to Node 3. This technique can be used with a scan time as low as 100 milliseconds.

- PEER_EX1_SEND_FBD (for sending Node 2)
- PEER_EX1_RCV_FBD (for receiving Node 3)

Sending Data Every Second to One Controller

These programs show how to send data every second from Node 2 to Node 3. This technique can be used with a scan time as low as 100 milliseconds.

- PEER_EX2_SEND_FBD (for sending Node 2)
- PEER_EX2_RCV_FBD (for receiving Node 3)

Sending Data Only When Requested

These programs show how to use Send and Receive function blocks in a controlled way. The programs send data only when an acknowledgment for the last send operation is received and new data is available.

- PEER_EX3_SEND_FBD (for sending Node 2)
- PEER_EX3_RCV_FBD (for receiving Node 3)

Fast Send of Safety-Critical Data

These programs show how to transfer a small amount of safety-critical data between two **TriStation** applications as fast as possible. The programs also show how to measure the actual maximum time for transferring data from the sending node to the receiving node.

- PEER_EX4_SEND_FBD (for sending Node 1)
- PEER_EX4_RCV_FBD (for receiving Node 3)

Because safety-critical data is being transferred, each controller must have two communication modules (Tricon NCMs or Trident CMs) which are connected to redundant Peer-to-Peer networks.

The data transfer time described in this section is calculated using this equation.

$$DT = (2 \cdot \text{<Larger of TS or SS>}) + (2 \cdot \text{<Larger of TR or SR>})$$

Parameter	Description
DT	Data transfer time in milliseconds.
TBS	Total bytes of aliased variables in the sending node.
TS	Time for sending controller to transfer aliased data over the communication bus in milliseconds. $TS = (TBS \div 100,000) \cdot 1000$
SS	Scan time of sending node in milliseconds.
TBR	Total bytes of aliased variables in the receiving node.
TR	Time for receiving controller to transfer aliased data over the communication bus in milliseconds. $TR = (TBR \div 100,000) \cdot 1000$
SR	Scan time of receiving node in milliseconds.

Sample Timing Calculations

SS = 150 milliseconds

TBS = 2000 bytes

TS = $(2000/100,000) \cdot 1000 = 20$ milliseconds

SR = 250 milliseconds

TBR = 5000 bytes

TR = $(5,000 \div 100,000) \cdot 1000 = 50$ milliseconds

DT = $2 \cdot 150 + 2 \cdot 250 = 800$ milliseconds

Process tolerance time = 4 seconds

The PEER-EX4_SEND_FBD program packs 32 BOOL values into a DINT (for Tricon) or DWORD (for Trident) and sends the value with a diagnostic variable to a receiving node as fast as possible by permanently setting the SENDFLG parameter to 1. The diagnostic variable is incremented every time a new send operation is initiated. The receiving node verifies that the diagnostic variable has changed from the previous value received. The receiving node also verifies that it has received at least one sample of new data within the process tolerance time. If not, the receiving program takes an appropriate action, such as using the last data received or using default data to make safety-critical decisions.

If the sending controller does not receive acknowledgment from the receiving controller in 1 second, it automatically retries the last Send message a second time. Due to network collisions,

communication bus loading, or other problems, the sending controller occasionally has to retry the send a third time. This is why the general rule for data transfer time is 1 to 2 seconds, even though the estimated time (DT) is 800 milliseconds.

The application running in the receiving controller includes a network that measures the actual time so that you can validate the assumed 2-second maximum transfer time. Since the process tolerance time of the receiving node is 4 seconds, the maximum time-out limit is set to 2 seconds (half the process tolerance time). The receiving node should get at least one sample of new data within the maximum time-out limit. Using this criteria satisfies the basic requirement for using Peer-to-Peer to transfer safety critical data.

Modbus Protocol

Overview	496
Message Response Time	497
Modbus Messages	499
Modbus Functions	505
Modbus Alias Range	516
Transmission Errors and Exception Conditions	518

Overview

This section provides detailed information about Modbus protocol, which is a communication protocol used with serial ports to transmit data between a Modbus master and slave. Modbus protocol includes functions which define the message format for the query and response.

Query-Response Sessions

Modbus communication is a query-response session, in which the Modbus master initiates a query and a Modbus slave responds. In Modbus communication, a serial link transmits data in both directions, but in only one direction at a time.

A query-response session consist of these actions:

- The master sends a query to a slave.
- The master starts a fail-safe timer while it waits for the slave response. Typical slave response time is in hundreds of milliseconds.
- The slave returns a response to the master.
- The master waits until it has received the response from the slave before sending another query.
- If there is a slave response time-out, the master will retry the query. The number of retries and the time-out interval is configured by the MBCTRL function block.

For information on configuring Modbus communication, see the *Tricon* and *Trident Communication Guides*.

Message Response Time

This section explains how to estimate the message response time, which is the total time for preparing, transmitting, receiving, and processing a Modbus query. Function blocks that are the least and most affected by scan time increases are also identified in this section.

Topics include:

- [Determining Message Response Time \(page 497\)](#)
- [Modbus Functions and Scan Time \(page 498\)](#)

Determining Message Response Time

This table explains how to estimate the number of milliseconds required for the message response time on a Triconex controller acting as a Modbus slave.

Modbus Operation	Equation or Constraints
Prepare Query (master)	Varies depending on the specific Modbus function (message) and any other program processing
Transmit Query (master)	$(1000 \div \text{Baud Rate}) \times \text{Bits per Characters} \times \text{Number of Characters}$
Receive and Process Query	<p>Tricon EICM slave:</p> <p>Writes: 3 x Scan Time</p> <p>Reads: 10 milliseconds</p> <p>Trident MP slave:</p> <p>Writes: 3 x Scan Time</p> <p>Reads: 2 x Scan Time</p> <p>Trident CM slave:</p> <p>Writes: 3 x Scan Time</p> <p>Reads: 10 milliseconds</p>
Transmit Response (slave) (in milliseconds)	$(1000 \div \text{Baud Rate}) \times \text{Bits per Characters} \times \text{Number of Characters}$
Process Response (master) (in milliseconds)	Depends on customer provided equipment performance.
Time-Out and Retry Values	<p>Varies depending on settings for the MBCTRL function block, which determines the time-out and retry values which can increase the message time.</p> <p>Message Response Time = the sum of all the results.</p>

Modbus Functions and Scan Time

Modbus performance degrades slightly as the scan time of the controller increases.

When the controller acts as a slave, the functions most affected by scan time increases are:

- [Force Single Coil \(Function Code 05\) \(page 510\)](#)
- [Preset Single Register \(Function Code 06\) \(page 511\)](#)
- [Force Multiple Coils \(Function Code 15\) \(page 514\)](#)
- [Preset Multiple Registers \(Function Code 16\) \(page 515\)](#)

The functions least affected by scan time increases are:

- [Read Coil Status Function \(Function 01\) \(page 506\)](#)
- [Read Input Status \(Function 02\) \(page 507\)](#)
- [Read Holding Registers \(Function Code 03\) \(page 508\)](#)
- [Read Input Registers \(Function Code 04\) \(page 509\)](#)

Modbus Messages

This section describes the Modbus messages (query and response functions) supported by Triconex communication modules. The serial ports on Triconex communication modules support several Modbus message formats and functions (queries and responses).

Topics include:

- [Communication Modes \(page 499\)](#)
- [Function Names and Aliases \(page 500\)](#)
- [Modbus Message Formats \(page 501\)](#)
- [Sample Query and Response Messages \(page 503\)](#)
- [Modbus Message Lengths \(page 504\)](#)

Communication Modes

A Modbus serial link must use either the Remote Terminal Unit (RTU) or ASCII mode of communication. If both modes are available, you should choose RTU because it is more efficient and robust than ASCII. Each serial port can use a different communication mode, assuming that each port is connected to a separate Modbus master or slave device. If you configure a port for combination Modbus master and slave operation, you must use RTU mode.

RTU Mode

In RTU mode, data is sent in 8-bit binary characters. Gaps between characters cannot exceed three character times (the time it takes to send a character). RTU mode uses a 16-bit cyclic redundancy check (CRC) to detect transmission errors.

ASCII Mode

In ASCII mode, data is transmitted in pairs of ASCII characters. The first character is the ASCII representation of the most significant 4 bits of the corresponding RTU character. The second character is the ASCII representation of the least significant 4 bits of the corresponding RTU character. For example, the RTU character 01001111_2 ($4F_{16}$) is sent as the two ASCII characters 4 and F (34_{16} and 46_{16}). Each ASCII message has a colon at the beginning and a carriage return and line feed at the end. Gaps between characters in an ASCII message are not significant.

Function Names and Aliases

The starting address field of a Modbus message ranges from 0 to one less than the number of coils or registers available.

A Trident CM or MP serial port maps the Modbus starting address field to an alias by adding a constant determined by the function code, as shown in this table.

Function Name	Code	Coil or Register	Constant
Read Coil Status	01	Coil	1
Read Input Status	02	Coil	10001
Read Holding Registers	03	Register	40001
Read Input Registers	04	Register	30001
Force Single Coil	05	Coil	1
Preset Single Register	06	Register	40001
Read Exception Status	07	Coil	n/a
Loop Back Diagnostic Test	08	Register	n/a
Force Multiple Coils	15	Coil	1
Preset Multiple Registers	16	Register	40001

Modbus Message Formats

For each Modbus function, the message formats for RTU and ASCII modes are depicted as shown in the following figures.

RTU Mode

Bytes	1	2	3	4	5	6	7	8
	Station Address	Function Code	Data		Data		CRC	

ASCII Mode

Bytes																
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
:	Station Address		Function Code		Data				Data				LRC		CR	LF

Message Header Field (ASCII Only)

The Message Header in ASCII mode is a colon (:) and is required. There is no message header in RTU mode.

Station Address Field

The Station Address field identifies the station to which a query is directed or the station that is sending a response. In RTU mode, the station address has one character (8 bits). In ASCII mode, the station address has 2 characters.

The range for station addresses is 1 through 247. Each station connected to a Modbus serial link must have a unique address. Station address 0 (zero) is the broadcast address and addresses all slaves. When a slave receives a query with the broadcast address, the slave processes the query but does not send a response.

Function Code Field

The Function Code field identifies the operation to be performed (the query), or the operation that was performed (the response). If the most significant bit of the function code in a response is 1, the response is an exception response.

Data Fields

The Data fields contain information that is specific to the query or response. The length of the data varies, depending on the function code.

Checksum Field (CRC or LRC)

The Checksum field is a 16-bit word which is a CRC in RTU mode or an LRC in ASCII mode. The error check is performed by both the transmitting and the receiving units to detect transmission errors. These sections describe the error check calculations that are performed for CRC and LRC.

CRC Error Check – RTU Mode

During a CRC error check, the CRC-16 polynomial is used to compute a checksum for the entire message. The CRC-16 polynomial is:

$$x^{16} + x^{15} + x^2 + 1$$

The CRC is computed across the station address, the function code, and the data and appended to the end of the message.

LRC Error Check – ASCII Mode

The LRC checksum is an 8-bit binary number represented and transmitted as 2 ASCII hexadecimal characters. To calculate the LRC, do this:

- Add the hex characters for the message content. (In the example, this includes the address, function code, starting address, and number of points fields.) The colon, carriage return, and line feed are ignored.
- Take the two's complement.

This table shows how to calculate the LRC for a sample message.

LRC Checksum Sample Calculation

Message Field	Message Content
Address	0x30
	0x32
Function Code	0x30
	0x31
Starting Address (high order)	0x30
	0x30
Starting Address (low order)	0x31
	0x33
Number of Points (high order)	0x30
	0x30
Number of Points (low order)	0x32
	<u>0x35</u>
Sum of message content.	0x4E
Take the two's complement.	0xB2
The resulting Error Check (LRC)	0x42 ('B')
	0x32 ('2')

CR Field and LF Field (ASCII Only)

The CR field contains an ASCII carriage return and the LF field contains an ASCII line feed.

Sample Query and Response Messages

This table shows the content of a sample query and response in RTU and ASCII modes. The query is a Read Input Status (Function 02) requesting 37 (25_{16}) points starting at point 20 ($13_{16} + 1$). The response packs the 37 points into five 8-bit bytes, and clears the 3 high-order bits of the last byte.

Query Message	RTU	ASCII
Header	None	:
Station Address	0000 0010	0 2
Function Code	0000 0001	0 1
Starting Address (High Order)	0000 0000	0 0
Starting Address (Low Order)	0001 0011	1 3
Number of Points (High Order)	0000 0000	0 0
Number of Points (Low Order)	0010 0101	2 5
Error Check	0000 1100 0010 0111	C 5
Trailer	None	CR LF

Response Message	RTU	ASCII
Header	None	:
Station Address	0000 0010	0 2
Function Code	0000 0001	0 1
Byte Count	0000 0101	0 5
Data Byte 1	<u>1</u> 100 ¹ 110 <u>1</u> ²	C D
Data Byte 2	0110 1011	6 B
Data Byte 3	1011 0010	B 2
Data Byte 4	0000 1110	0 E
Data Byte 5	0001 1011	1 B
Error Check	0000 0100 1111 1111	E 5
Trailer	None	CR LF

1. The underscored digit indicates that Coil #27 is in the On state.
2. The underscored digit indicates that Coil #20 is in the On state.

Modbus Message Lengths

The length of a Modbus message depends on the function being used and whether the message is a query or a response.

Function Code	Query	Number of RTU Characters	Number of ASCII Characters
01	Read Coil Status	8	17
02	Read Input Status	8	17
03	Read Holding Registers	8	17
04	Read Input Registers	8	17
05	Force Single Coil	8	17
06	Preset Single Register	8	17
15	Force Multiple Coils	9 + (1 per 8 coils)	19 + (2 per 8 coils)
16	Preset Multiple Registers	9 + (2 per register)	19 + (4 per register)

Function Code	Response	Number of RTU Characters	Number of ASCII Characters
01	Read Coil Status	5 + (1 per 8 coils)	11 + (2 per 8 coils)
02	Read Input Status	5 + (1 per 8 coils)	11 + (2 per 8 coils)
03	Read Holding Registers	5 + (2 per register)	11 + (4 per register)
04	Read Input Register	5 + (2 per register)	11 + (4 per register)
05	Force Single Coil	8	17
06	Preset Single Register	8	17
15	Force Multiple Coils	8	17
16	Preset Multiple Registers	8	17

Modbus Functions

This section includes details on Modbus functions.

Functions include:

- [Read Coil Status Function \(Function 01\) \(page 506\)](#)
- [Read Input Status \(Function 02\) \(page 507\)](#)
- [Read Holding Registers \(Function Code 03\) \(page 508\)](#)
- [Read Input Registers \(Function Code 04\) \(page 509\)](#)
- [Force Single Coil \(Function Code 05\) \(page 510\)](#)
- [Preset Single Register \(Function Code 06\) \(page 511\)](#)
- [Read Exception Status \(Function Code 07\) \(page 512\)](#)
- [Loop-Back Diagnostic Test \(Function 08\) \(page 513\)](#)
- [Force Multiple Coils \(Function Code 15\) \(page 514\)](#)
- [Preset Multiple Registers \(Function Code 16\) \(page 515\)](#)

Read Coil Status Function (Function 01)

Query Format

The Read Coil Status query requests the On/Off status of a group of logic coils from a station. You can request the status for as many as 2,000 coils with each query, but some Modbus devices have lower limits. The coils are numbered beginning at 0. For example, coil 0 is alias 1, coil 1 is alias 2, and so forth.

The Read Coil Status query is also known as the Read Output Status query.

RTU Mode

Bytes

1	2	3	4	5	6	7	8
Station Address	0000 0001	Starting Address		Number of Coils		CRC	

ASCII Mode

Bytes

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
.	Station Address	0	1	Starting Address		Number of Coils		LRC	CR	LF						

Response Format

The Read Coil Status response data is packed with one bit for each coil, where 1=On, and 0=Off. The low-order bit of the first RTU character contains the status of the first coil. For coil quantities that are not even multiples of eight, the last RTU character is zero-filled at the high-order end.

RTU Mode

Bytes

1	2	3	4			n	n+1	n+2
Station Address	0000 0001	Data Length	Data			Data		CRC

ASCII Mode

Bytes

1	2	3	4	5	6	7	8			n	n+1	n+2	n+3	n+4
.	Station Address	0	1	Data Length	Data					Data	LRC	CR	LF	

Read Input Status (Function 02)

Query Format

The Read Input Status function operates in the same manner as Read Coil Status (Function Code 01), except that the status of digital inputs is obtained. Inputs are also numbered beginning at 0. For example, input status 0 is alias 10001, input status 1 is alias 10002, and so forth. You can request the status of as many as 2,000 coils with each query, but some Modbus devices have lower limits.

RTU Mode

Bytes							
1	2	3	4	5	6	7	8
Station Address	0000 0010		Starting Address	Number of Input Points		CRC	

ASCII Mode

Bytes																
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
:	Station Address	0	2	Starting Addresses		Number of Input Points		LRC		CR	LF					

Response Format

RTU Mode

Bytes							
1	2	3	4	n		n+1	n+2
Station Address	0000 0010		Data Length	Data		Data	CRC

ASCII Mode

Bytes																
1	2	3	4	5	6	7	8	n		n+1	n+2	n+3	n+4			
:	Station Address	0	2	Data Length		Data		Data		LRC	CR	LF				

Read Holding Registers (Function Code 03)

Query Format

The Read Holding Registers query requests the binary content of holding registers from a station. You can request the status of as many as 125 registers with each query, but some Modbus devices have lower limits. The registers are numbered beginning at 0. For example, register 0 is alias 40001, register 1 is alias 40002, and so forth.

The Read Holding Registers query is also known as the Read Output Registers query.

RTU Mode

Bytes

1	2	3	4	5	6	7	8
Station Address	0000 0011	Starting Address		Number of Registers		CRC	

ASCII Mode

Bytes

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
•	Station Address	0	3	Starting Address		Number of Registers		LRC		CR		LF				

Response Format

The Read Holding Registers response data consists of 2 bytes for each register queried, with the binary content right-justified. The leftmost character includes the high-order bits, and the rightmost character includes the low-order bits.

RTU Mode

Bytes

1	2	3	4			n	n+1	n+2
Station Address	0000 0011	Data Length	Data			Data	CRC	

ASCII Mode

Bytes

1	2	3	4	5	6	7	8			n	n+1	n+2	n+3	n+4
•	Station Address	0	3	Data Length	Data			Data	LRC	CR	LF			

Read Input Registers (Function Code 04)

Query Format

The Read Input Registers function operates in the same manner as the Read Holding Registers query (Function Code 03), except that it obtains the status of input registers. You can request the status of as many as 125 registers with each query, but some Modbus devices have lower limits. The registers are numbered beginning at 0. For example, register 0 is alias 30001, register 1 is alias 30002, and so forth.

RTU Mode

Bytes							
1	2	3	4	5	6	7	8
Station Address	0000 0100		Starting Address		Number of Registers		CRC

ASCII Mode

Bytes																
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
:	Station Address	0	4	Starting Address				Number of Registers				LRC		CR	LF	

Response Format

RTU Mode

Bytes							
1	2	3	4	n		n+1	n+2
Station Address	0000 0100		Data Length	Data		Data	CRC

ASCII Mode

Bytes													
1	2	3	4	5	6	7	8						
:	Station Address		0 4		Data Length		Data						
								n	n+1	n+2	n+3	n+4	
								Data	LRC		CR	LF	

Force Single Coil (Function Code 05)

Query Format

The Force Single Coil function turns a single coil On or Off, depending on its current state. Because the slave is actively scanning, it can also alter the state of the coil (unless the coil is disabled). Coils are numbered beginning at 0. For example, coil 0 is alias 1, coil 1 is alias 2, and so forth.

A coil value of 65,280 (FF00_{16}) turns the coil On, and a coil value of zero (0000_{16}) turns the coil Off. All other values are illegal and do not affect the coil. If the query contains legal values, the slave responds after the coil state has been altered.

RTU Mode

Bytes

1	2	3	4	5	6	7	8
Station Address	0000 0101	Address to Modify		Coil Value		CRC	

ASCII Mode

Bytes

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
:	Station Address	0	5	Address to Modify		Coil Value		LRC		CR	LF					

Response Format

RTU Mode

Bytes

1	2	3	4	5	6	7	8
Station Address	0000 0101	Address Modified		Coil Value		CRC	

ASCII Mode

Bytes

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
:	Station Address	0	5	Address Modified		Coil Value		LRC		CR	LF					

Preset Single Register (Function Code 06)

The Preset Single Register function modifies the content of one holding register. Because the slave is actively scanning, it can also alter the content of the register. Register values are 16 bits. Holding registers are numbered beginning at 0. For example, register 0 is alias 40001, register 1 is alias 40002.

Query Format

RTU Mode

Bytes	1	2	3	4	5	6	7	8
Station Address	0000 0101	Address to Modify		Register Value		CRC		

ASCII Mode

Bytes	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
: Station Address	0	5	Address to Modify		Register Value		LRC		CR	LF							

Response Format

RTU Mode

Bytes	1	2	3	4	5	6	7	8
Station Address	0000 0110	Address to Modify		Register Value		CRC		

ASCII Mode

Bytes	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
: Station Address	0	6	Address to Modify		Register Value		LRC		CR	LF							

Read Exception Status (Function Code 07)

Query Format

The Read Exception Status function returns the status of eight coils from the slave application running in the controller. Which coils and what they represent depends on the slave. When a serial port, configured as a slave, responds to this query, it sends the status of the first eight coils (aliases 00001 through 00008) defined in the application. Coils are numbered beginning at 0. For example, coil 0 is alias 1, coil 1 is alias 2, and so forth. The status of each coil is packed in the data field, one bit for each coil (1=On, 0=Off). You can program these coils to hold any type of information; for example, machine on or off, heads retracted, safeties satisfied, and receipt-in-process error conditions.

Note A serial port configured as a Modbus master cannot use the Read Exception Status function.

RTU Mode

Bytes	1	2	3	4
	Station Address	0000 0111	CRC	

ASCII Mode

Bytes	1	2	3	4	5	6	7	8	9
	:	Station Address	0	7	LRC	CR	LF		

Response Format

RTU Mode

Bytes	1	2	3	4	5
	Station Address	0000 0111	Coil Data	CRC	

ASCII Mode

Bytes	1	2	3	4	5	6	7	8	9	10	11
	:	Station Address	0	7	Coil Data	LRC	CR	LF			

Loop-Back Diagnostic Test (Function 08)

Query Format

The Loop-Back Diagnostics Test query tests the communications link between the Modbus master and slave. This query does not affect point values in the slave. When the serial port acting as a slave receives this query, it re-transmits the query as the response.

RTU Mode

Bytes

1	2	3	4	5	6	7	8
Station Address	0000 1000	Data				CRC	

ASCII Mode

Bytes

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
:	Station Address	0	8	Data										LRC	CR	LF

Response Format

RTU Mode

Bytes

1	2	3	4	5	6	7	8
Station Address	0000 1000	Data				CRC	

ASCII Mode

Bytes

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
:	Station Address	0	8	Data										LRC	CR	LF

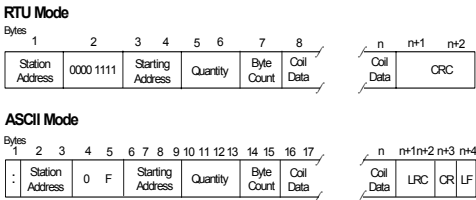
Note A serial port configured as a Modbus Master cannot use the Loop-Back Diagnostic Test function.

Force Multiple Coils (Function Code 15)

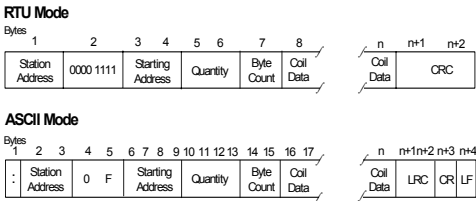
Query Format

The Force Multiple Coils query sets each coil in a consecutive block of coils to the specified state (On or Off) regardless of whether the coils are enabled or disabled. Because the slave is actively scanning, it can also alter the state of a coil (unless it is disabled). Coils are numbered beginning at 0. For example, coil 0 is alias 1, coil 1 is alias 2, and so forth. The status of each coil is packed in the data field, one bit for each coil (1=On, 0=Off).

A single Force Multiple Coils query can set a maximum of 128 coils. The query-response time required by some Modbus masters might require a much smaller quantity.



Response Format



Preset Multiple Registers (Function Code 16)

Query Format

The Preset Multiple Registers query can change the contents of a maximum of 60 consecutive holding registers, however, some Modbus devices have lower limits. Because the slave is actively scanning, it can also alter the state of the registers (unless they are disabled). The values are provided in binary code up to the maximum valid register value of the controller (16-bit for Trident). Unused high-order bits must be set to 0. The registers are numbered beginning at 0. For example, register 0 is alias 40001, register 1 is alias 40002, and so forth.

RTU Mode

Bytes	1	2	3	4	5	6	7	8		n	n+1	n+2
	Station Address	0001 0000	Starting Address	Quantity	Byte Count	Register Data				Register Data		CRC

ASCII Mode

Bytes	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17		n	n+1	n+2	n+3	n+4
	:	Station Address	1	0	Starting Address	Quantity	Byte Count	Register Data											Register Data	LRC	CR	LF	

Response Format

RTU Mode

Bytes	1	2	3	4	5	6	7	8
	Station Address	0001 0000	Starting Address	Quantity				CRC

ASCII Mode

Bytes	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
	:	Station Address	1	0	Starting Address	Quantity								LRC	CR	LF	

Modbus Alias Range

The Modbus Alias Range identifies the range of five-digit numbers that can be assigned to the point. The leftmost digit identifies the data type and the other four digits identify the hardware address in the controller.

Ranges include:

- The minimum value is -32,768; the default is 0.
- The maximum value is 32,767; the default is 32,767.
- Honeywell DHP uses 0 to 9,999.

The Modbus Range operates in conjunction with the Minimum and Maximum values on the Tricon EICM Setup, Trident MP, and Trident CM Serial Ports tab to scale values of REAL points.

Tricon Alias Ranges

Point Type	Data Type	Message Type	Available Points	Aliased Points ¹	Alias Range
Input	Discrete	Read Only	4096	2000	10001 - 12000
Input	Integer	Read Only	1024	1000	30001 - 31000
Input	Real	Read Only	120	120	32001 - 32120
Output	Discrete	Read/Write	2048	2000	00001 - 02000
Output	Integer	Read/Write	512	250	40001 - 40250
Memory	Discrete	Read/Write	2016	2000	02001 - 04000
Memory	Discrete	Read Only	2016	2000	12001 - 14000
Memory	Integer	Read/Write	750	750	40251 - 41000
Memory	Integer	Read Only	1000	1000	31001 - 32000
Memory	Real	Read/Write	1000	1000	41001 - 42000
Memory	Real	Read Only	1000	1000	33001 - 34000

1. For certain point types, some of the available points have no aliases. These unaliased points can be used in your application, but cannot be accessed by external hosts.

Trident Alias Ranges

Point Type	Data Type	Message Type	Maximum Points	Default Alias Range	Allowable Alias Range
Input	Discrete	Read Only	4096	10001 - 14999	10001 - 19999
Input	Integer	Read Only	1024	30001 - 32499	30000 - 39999
Input	Real	Read Only	120	35000 - 37499	30000 - 39999
Output	Discrete	Read/Write	2048	00001 - 04999	00001 - 09999
Output	Integer	Read/Write	512	40000 - 42499	40000 - 49999
Memory	Discrete	Read/Write	2016	05000 - 09999	00001 - 09999

Trident Alias Ranges *(continued)*

Point Type	Data Type	Message Type	Maximum Points	Default Alias Range	Allowable Alias Range
Memory	Discrete	Read Only	2016	15000 - 19999	10001 - 19999
Memory	Integer	Read/Write	750	42500 - 44999	40000 - 49999
Memory	Integer	Read Only	1000	32500 - 34999	30000 - 39999
Memory	Real	Read/Write	1000	45000 - 49999	40000 - 49999
Memory	Real	Read Only	1000	37500 - 39999	30000 - 39999

Transmission Errors and Exception Conditions

During Modbus communication, transmission errors and exception conditions can occur. Transmission errors do not cause exception conditions and are not acknowledged by Modbus slaves. Programming and operation errors do cause exception conditions which elicit exception responses from slaves.

Topics include:

- [Transmission Errors \(page 518\)](#)
- [Exception Conditions \(page 519\)](#)
- [Exception Responses \(page 520\)](#)
- [Exception Response Codes \(page 521\)](#)

Transmission Errors

The most frequent cause of transmission errors is noise. Noise sources include improperly installed or broken connectors, damaged cables, electrical equipment such as generators and elevators, and lightning. Transmission errors can be detected through the use of character framing, parity checking, and redundancy checking.

When a slave detects a transmission error, it does not act on or respond to the message. The master assumes a communication error has occurred if there is no response within a specified time, usually 3 seconds.

Parity checking helps detect single-bit transmission errors. However, if there are two errors within a single character, parity checking cannot detect a change. For example, if 1100 0100 is distorted to 1111 0100, the number of 1 bits in the data is still odd.

Modbus protocol provides several levels of error checking to ensure the accuracy of data transmission. To detect multiple bit errors, the system uses cyclic redundancy check (CRC) for RTU mode, or longitudinal redundancy check (LRC) for ASCII mode.

Related Topics

[Checksum Field \(CRC or LRC\) \(page 501\)](#)

Exception Conditions

If a master detects an exception in a response to a query or does not receive a response, it takes appropriate actions which usually include re-transmitting the query. This table lists exception conditions that are returned by the slave if a programming or operation error causes a master to send an incorrect query.

Exception Condition	Description
Query Message CRC or LRC Error	The slave does not respond, because the error could be in the station address. The master uses its response fail-safe timer to recover.
Query Function Code Error	The slave sends an Illegal Function (01) response code when it detects an error in the function code field.
Query Address Error	The slave sends an Illegal Data Address (02) response code when it detects an error in the starting address field.
Query Data Error	The slave sends an Illegal Data Value (03) response code when it detects an error in the data field.
Main Processors Not Communicating	<p>This exception applies only to serial ports which are configured as slaves.</p> <p>If the slave port receives a query requiring a data exchange and it cannot communicate with the Main Processors, it sends a Busy, Reject Message (06) response code and turns off the Active indicator on the communication module.</p>
Remote Write Disabled	<p>The slave port sends a Busy, Reject Message (06) response code if a master sends one of these queries and the slave port is not enabled for remote (external) writes:</p> <ul style="list-style-type: none"> Force Single Coil (Function Code 05) Preset Single Register (Function Code 06) Force Multiple Coils (Function Code 15) Preset Multiple Registers (Function Code 16)

Exception Responses

When a slave detects an exception condition, it sends a response message to the master consisting of the slave’s station address, function code, error code, and error-check fields. To indicate that the message is an exception response, the slave sets the high-order bit of the function code to 1. This example shows an exception response to a Preset Multiple Registers query.

Sample Query

RTU Mode

Bytes																
1	2	3	4	5	6	7	8									
Station Address	0001 0000			Starting Address		Quantity		Byte Count	Register Data							
								n		n+1		n+2				
								Register Data		CRC						

ASCII Mode

Bytes																
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
:	Station Address	1	0	Starting Address		Quantity		Byte Count	Register Data							
								n		n+1		n+2		n+3		n+4
								Register Data		LRC		CR		LF		

Sample Exception Response

RTU Mode											
Bytes											
1	2	3	4	5	6						
Station Address		1001 0000		Exception Code		CRC					

ASCII Mode													
Bytes													
1	2	3	4	5	6	7	8	9	10	11	12	13	
:	Station Address		9	0	Exception Code		LRC		CR	LF			

Exception Response Codes

This table lists exception response codes which are sent by the slave after an invalid query.

Code	Name	Description
01	Illegal Function	The requested function is not in the slave's repertoire.
02	Illegal Data Address	The alias in the query does not exist in the slave.
03	Illegal Data Value	The value is not in the range allowed for the alias.
04	Failure in Associated Device	The slave failed to respond to a message or an error that occurred in the controller. When a master receives this response code, it must issue a supervisory alert.
05	Acknowledge	A slave port does not send this exception response code.
06	Busy, Rejected Message	The query was received without error, but the slave cannot comply.
07	Negative Acknowledge	A slave port does not send this exception response code.
08	Memory Parity Error	A slave port does not send this exception response code.



Triconex Trilogger Event

Overview	524
Product Overview	525
Typical TriLogger Installations	527
Theory of Operation	529
Configuration	532
Operation	552

Overview

This document applies to the Triconex TriLogger Event software package which provides user configurable event recording for Tricon based ESD and Critical Control systems.

About This Manual

The TriLogger Event manual is a guide to the installation and operation of the TriLogger Event software package.

Since this document is relatively small, an index is not included. Instead, a detailed table of contents is provided. This manual contains the following sections:

- Product Overview
 - Presents an overview of the product as a whole, including typical configurations.
- Installation
 - Describes the minimum PC requirements, other required software and the details necessary to install the TriLogger Event package
- Theory of Operation
 - Explains how a Tricon stores data internally and gives advise on maximizing performance.
- Configuration
 - Explains how to export a Tricon configuration from TS1131 or from the MSW programming packages and generate a configuration file for the TriLogger package.
- Operation
 - Describes how to operate the TriLogger package to obtain the maximum benefit from the software.

Reference Documents

Tricon DDE Server Users Guide

TriLogger Remote Installation and Operations Guide, Revision 2.

TriLogger Playback Installation and Operations Guide, Revision 2.

Product Overview

The Triconex TriLogger Event software allows user-definable events to trigger high-speed analog and discrete data recording using an Ethernet communications with the Tricon Network Communications Module (NCM). Multiple instances of the program can be run on a single PC to increase point counts in a single Tricon, or to record events in multiple Tricons connected together on an Ethernet communications network.

The Triconex TriLogger Event software uses the Tricon DDE Server package. The DDE Server must be installed and functional before installing the Triconex TriLogger Event Software. Installation details for the Tricon DDE Server package are in the Tricon DDE Server Users Guide, included with this package.

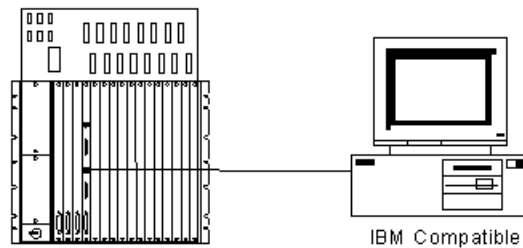
TriLogger Event Features

- High point capacity (up to 4000 points total)
 - Up to 2000 analog tags, physical I/O or calculated values
 - Up to 2000 discrete tags, physical I/O or internal flags
- User Configurable Event Times
 - Event recording time user configurable from 1 to 30 minutes
 - Adjustable trigger "fence" stores data from 0 to 30 minutes before of after the event - user configurable
- Multiple Event Triggers
 - Analog event triggers (high, low or both)
 - Discrete triggers on rising/falling edge or change of state
 - Every point can trigger an event
- High Speed
 - 250 analog and 250 discrete I/O mix scanned in approximately 100 milliseconds
 - 1000 discrete I/O scanned in approximately 50 milliseconds, or Tricon scan time whichever is greater
- Tricon driven time base
 - Multiple time synced Tricons will have synchronized event files
 - Remote connections (from anywhere in the world) will see the local Tricon time
- Heartbeat feature allows the Tricon to alarm on TriLogger Event failure
- Scan multiple Tricons (using multiple program instances in the same PC)
- Simple no hassle configuration using TS1131 or MSW export files
- Configuration files saved as Comma separated Variable (CSV) files, simplifying viewing, editing, and grouping of tags
- Configuration modifiable online
 - Triggers can be added or removed while TriLogger is running
 - Analog Ranges can be modified

- Trigger limits can be modified
- Multiple event file data storage formats
 - Excel workbook
 - Native data format - high speed playback and smaller file size
- Remote Connectivity
 - Up to 3 concurrent remote connections
 - Real time graphical data displays available from remote locations
 - Manually trigger events from remote
- Local data
 - Analog and discrete data viewable
 - Event list
 - Manually trigger events locally
- Log File Generation
 - Logging start/stop times
 - Remote connects/disconnects times
 - Events triggers and times
 - Loss/recovery of communication times

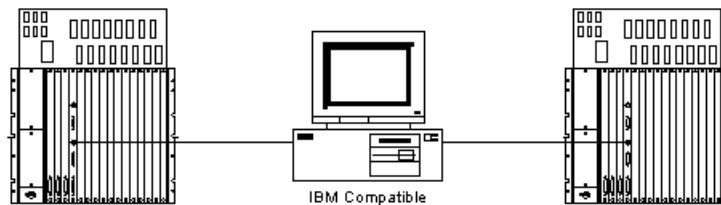
Typical TriLogger Installations

There many ways to configure an Ethernet network with PCs connected to one or more Tricons. Here are a few examples:



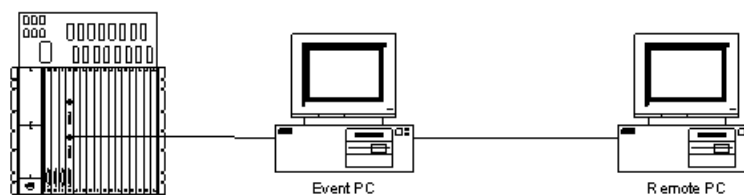
Simple Single Ethernet Network
Figure 1

In Figure 1 there is a single Ethernet connection between the Tricon and a PC. The TriLogger Event package runs on this PC and data captured is reviewed locally, either using Excel or TriLogger Playback.



Single PC Networked to Two Tricons
Figure 2

In this example a single PC is running two instances of the TriLogger monitoring two Tricons. If the Tricons are configured to use the Net 1 connection of the NCM card, the time clocks of the Tricons will be synchronized. Events captured by the PC will have the same time base.



A Local and Remote PC Connected to a Tricon
Figure 3

In Figure 3, the Event Logger PC has two Ethernet cards installed. One card handles communications to the Tricon, while the second Ethernet card allows remote access. Dual Ethernet cards are recommended in TriLogger PCs to minimize extraneous network traffic on the link connecting the Tricon and the PC.

Installation

TriLogger Event is distributed on CDRom. To install TriLogger simply run the Startup program on the CD and follow the instructions. The installation will take several minutes and will restart the computer several times during the install process. During this process The Triconex DDE Server, Wonderware NetDDE, and Triconex TriLogger will be installed.

Minimum Hardware Requirements

TriLogger Event is a high-speed communications program which can stress a PC that does not have adequate resources. As a minimum a PC should have the following

- 500 MHz CPU clock rate. Higher clock speeds are definitely better.
- 256 MB RAM
- 20 GB hard disk drive
- CD ROM drive
- Removable storage media such as a CDRW, ZIP or Jazz drives to copy event files
- VGA display with 800 by 600 minimum resolution
- Ethernet communications board that supports BNC type (10 MB/sec) connection

Performance is significantly enhanced with a 1 GHz or higher CPU clock rate.

If it is desired to connect the PC to a local area network, a second Ethernet communications card is highly recommended.

Minimum Software Requirements

TriLogger Event has been tested with the following software.

- Microsoft Windows 2000
- Triconex DDE Server version 3.1 or higher

Theory of Operation

TriLogger Event uses the Triconex DDE Server as a basis for collecting data from a Tricon using a Network Communications Module (NCM). The NCM supports Ethernet communication at 10 megabytes per second using TCP/IP or TCP/UDP protocols. The speed at which TriLogger Event can collect data from a Tricon depends on the number and type of data points and the program scan rate of the Tricon.

Tricon Bins

The Triconex DDE Server requests data from the Tricon that TriLogger Event has been configured to acquire. Data is stored in the Tricon within bins. Bins are divided into categories by point type, each point type requiring differing amounts of storage space. Boolean variables require a single bit, while DINT (double integer) and real types require 32 bytes each. Consequently the time required to transmit real and DINT variables require significantly more time than Boolean variables.

A breakdown of the bin storage is as follows:

Bin	Type	Point	Alias Range	Access
0	Boolean	Outputs	1-2000	Read/Write
1	Boolean	Memory	2001-4000	Read/Write
2	Boolean	Inputs	10001-12000	Read Only
3	Boolean	Memory	12001-14000	Read Only
4	DINT	Inputs	30001-31000	Read Only
5	DINT	Memory	31001-31382	Read Only
6	Real	Input	32001-32120	Read Only
7	Real	Memory	33001-34000	Read Only
8	Boolean	System Status	14001-19999	Read Only
9	DINT	System Status	39631-39999	Read Only
10	DINT	Outputs	40001-40250	Read/Write
11	DINT	Memory	40251-40632	Read/Write
12	Real	Memory	41001-42000	Read/Write

Any aliased variable, with the exception of the System Status variables (bins 8 and 9), can be scanned by TriLogger Event. System variables do not show up in the point Tricon Point Connections and consequently will not appear in an export of the Tricon database.

When a variable is requested the Tricon NCM assembles the requested data into packets that are transmitted to TriLogger Event. The NCM requires time to assemble the requested data into packets for transmission. This overhead also affects how quickly the NCM can accomplish a data transfer.

Performance

TriLogger Event performance is dependent on several factors. Scan rate of the Tricon, types and number of points requested, and the number of different bins accessed.

While absolute numbers for scan time cannot be given there are several general rules which affect the TriLogger Event scan rate:

- TriLogger Event cannot collect data faster than the Tricon scan rate
 - This should be obvious since data cannot change until the Tricon processors update the NCM
- It takes longer to transmit analog points than discretes.
 - Analogs are transmitted as 32 bit numbers while discretes occupy a single bit
- Requesting variables from multiple bins increases NCM overhead
 - Additional bin processing can add 10 to 20 milliseconds to scan time
- Increasing number of points increases the scan time

In general, the performance of TriLogger Event is so fast that there may not be an issue with a particular installation. These guidelines may improve performance significantly and help obtain the highest possible data transfer rates:

- 4 Don't log unused points. Often spare points are interspersed with active data points. Carrying a large number of spare points increases packet size and increases the size of the data files generated while providing no useful information.
- 5 Don't log raw analog points (bin 4). Raw analog inputs are DINT variables which are 0 to 4096 counts. Counts do not normally mean much to an observer. These values are converted to engineering units in the Tricon program and typically assigned to bin 7 (alias 33001-34000). Accessing these variables in bin 7 or 12 reduces the number of bins that must be transmitted.
- 6 Don't log raw analog outputs (bin 10). Analog outputs are DINT variables which are 0 to 4096 counts but are used within the program as real numbers. This means they can be mapped into bin 7 or 12 and brought across with other variables.
- 7 Decide which variables are really needed. Many variables are simply for indication or used as a startup permissive, they may not be needed for event diagnostics.
- 8 Don't log points more than once unless change of state triggers are desired.

Events

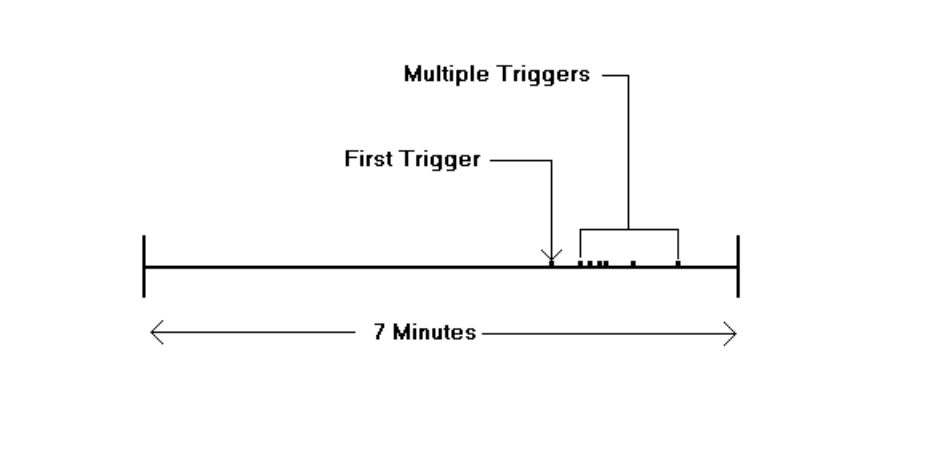
An event occurs whenever an analog or discrete exceeds a trigger threshold, causing TriLogger Event to save up to 30 minutes of data to a file. A point can trigger an event only if it is configured by the user to do so. Trigger thresholds, also user configurable, are defined for analog and discrete points.

When an event is triggered by any configured point exceeding a trigger limit, all selected data is recorded, not just the trigger point.

For an analog point both low and high trigger limits can be configured. If the analog value exceeds either limit, an event will be triggered. For a discrete point a state change (i.e. from true to false) can trigger an event. When an event is triggered TriLogger Event will continue to collect data samples from all selected points until the user defined time period expires, at which point the data will be written to disk.

Multiple events are not possible. When any point exceeds its trigger limit an event is triggered. This point is the first out for the event. All point values are recorded, but the first out point is saved since it has particular importance in problem analysis.

In this figure, an event is triggered by the first out, the point that exceeded its trigger limit first.



The tag name of the first trigger is saved with the data file as the first out. Any subsequent points exceeding their trigger limits during the event window are recorded, but they do not cause an additional file to be recorded since an event is already active. TriLogger Event latches all triggers after the first occurrence and does not re-enable them until after they return to normal.

During a process trip it is common for subsequent alarms to occur which are a result of the first alarm. Consequently TriLogger Event only records an event on the first trigger. The first and any subsequent triggers will be recorded, but they are prevented from forcing additional event files to be generated. This prevents multiple event logs being generated when equipment or process shutdowns occur and many points are in an alarm state.

Once the process is restarted and the process returns to normal the triggers are unlatched and they can again trigger an event. In the example above if an unrelated trigger occurs after the event window a data file is written to disk another event will be triggered, and the data files will overlap as shown below.

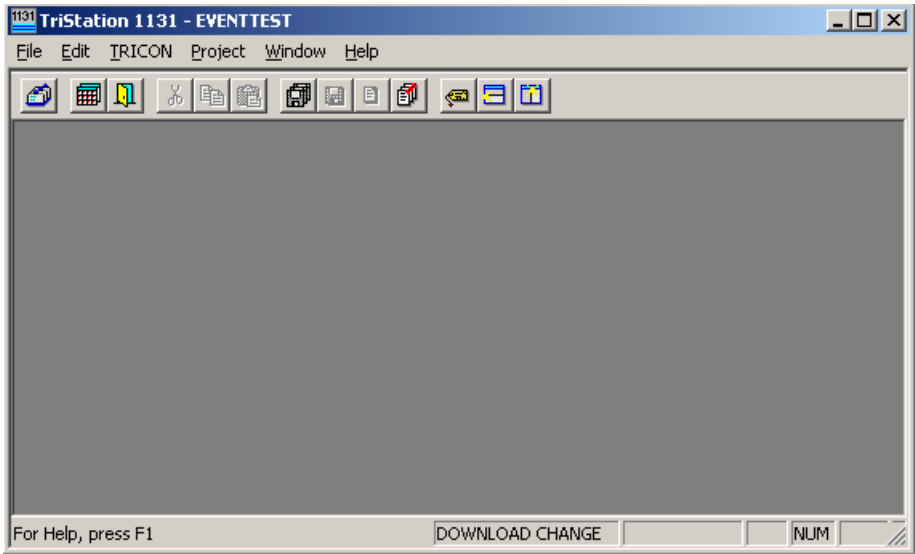
The triggers from the first event are still latched and cannot trigger again until they return to their normal state. If a new, unlatched trigger occurs after the event file is written to disk, a new event file will be generated.

TriLogger Event continues collecting data throughout an event and in this example the data files generated will overlap each other and some data will be contained in both event logs.

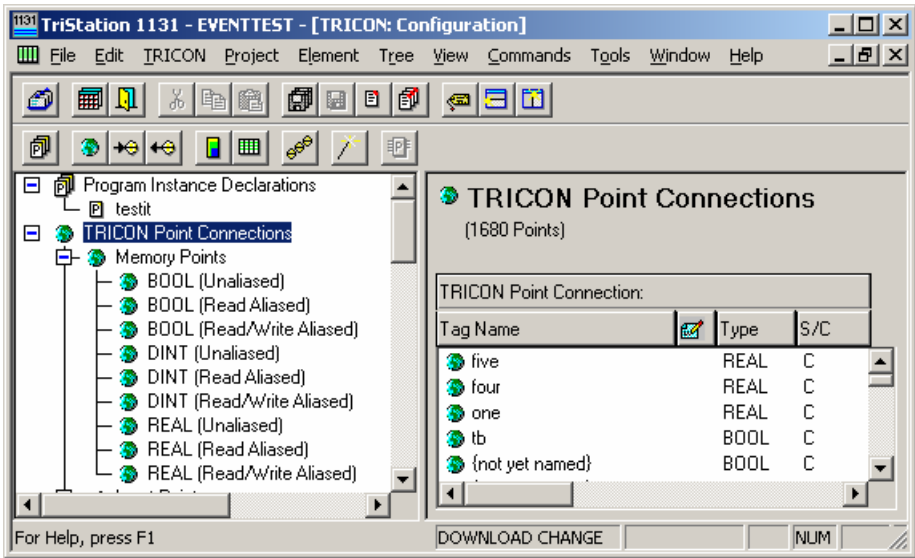
Configuration

Creating an Export File

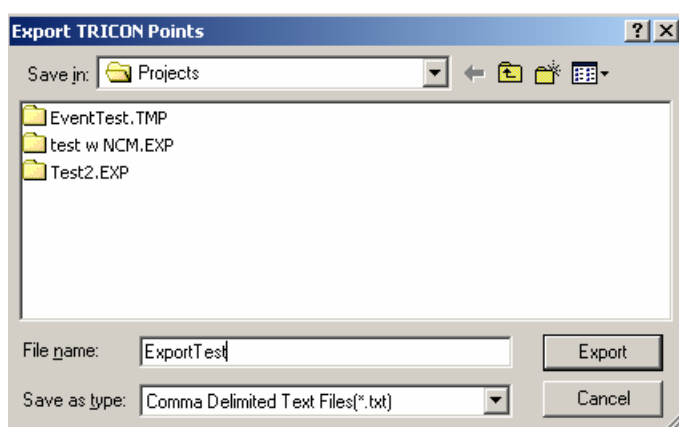
For TriLogger to know what points are available, an export of the Tricon database must be made for TriLogger to use. With TS1131 running and the project file open, the main screen of TS1131 looks like this:



- 1 Select Tricon > Edit Configuration and the following screen displays:



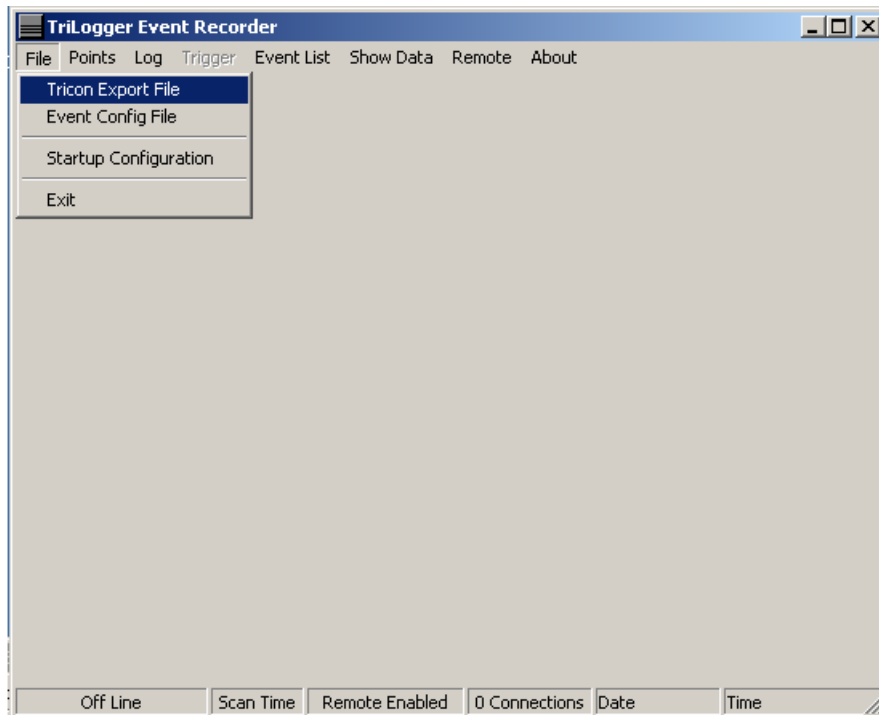
- 2 Select Edit > Export TRICON Points and the following screen is displayed:



- 3 Enter the name that you wish for the export file and select the file type as Comma Delimited Text (*.txt) format.
- 4 Press the Export button and TS1131 will export all aliased variables.

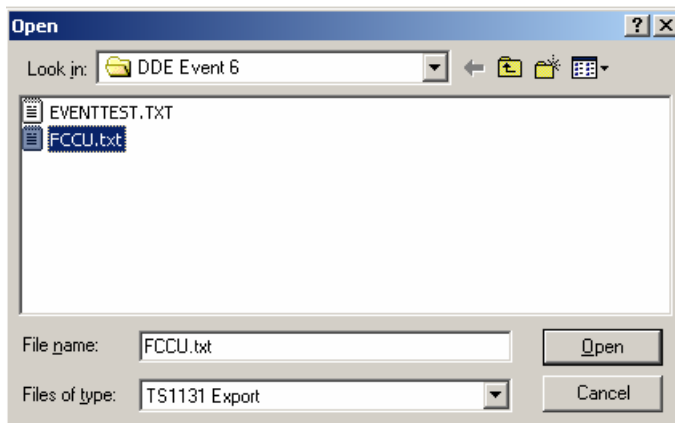
Reading the Export File

When TriLogger Event is started the screen will look like this:



- 1 To import a Tricon export file, click File on the main menu. Select Tricon Export File from the following choices.

A new form will appear that will allow the export file generated to be selected for import.



The Tricon export file has the 'TXT' file extension. If an attempt is made to load a file that is not a Tricon export file an error will occur. If this happens simply locate the correct Tricon export file and reload the database.

If the Tricon was programmed using MSW the export file extension will be 'ASC' (ASCII text). Select the MSW export file type from the TriLogger Open File Dialog as shown below.

Selecting Points to be Logged

Once the file is selected, TriLogger Event will read the file and generate a form which allows points to be selected for event logging.

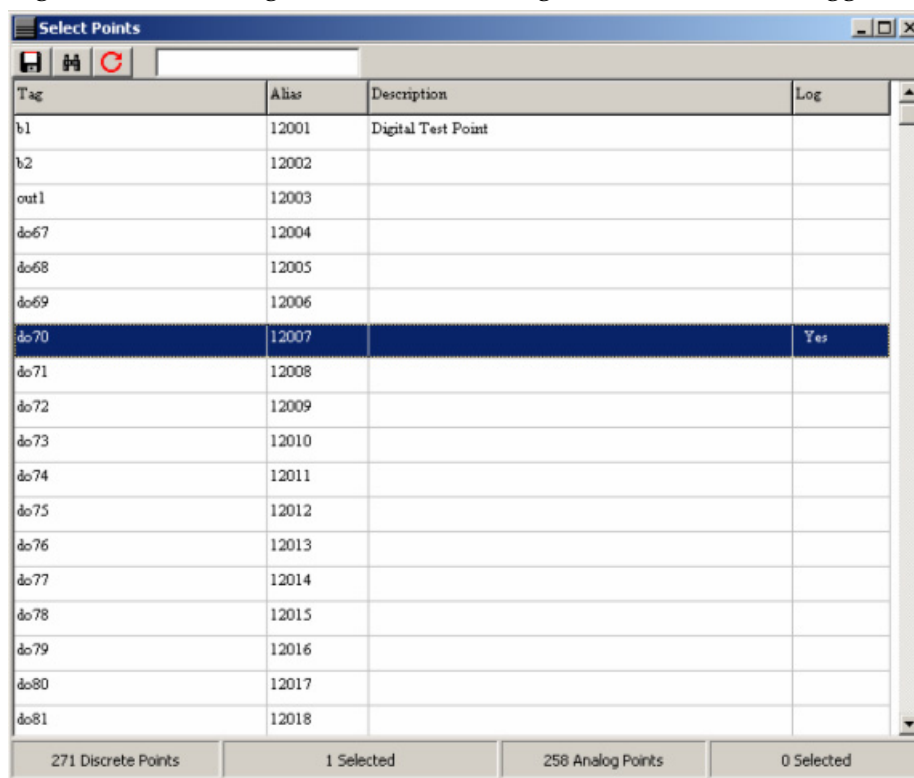
Tag	Alias	Description	Log
b1	12001	Digital Test Point	
b2	12002		
out1	12003		
do67	12004		
do68	12005		
do69	12006		
do70	12007		
do71	12008		
do72	12009		
do73	12010		
do74	12011		
do75	12012		
do76	12013		
do77	12014		
do78	12015		
do79	12016		
do80	12017		
do81	12018		

271 Discrete Points 0 Selected 258 Analog Points 0 Selected

This form contains a list of all aliased tags within the Tricon database. Tags can be selected individually or in groups.

- 1 To select points individually, left click on the row containing the tag desired. The selected row is then highlighted.

- 2 Right click on the tag selected and the Log Point column will toggle to "Yes."

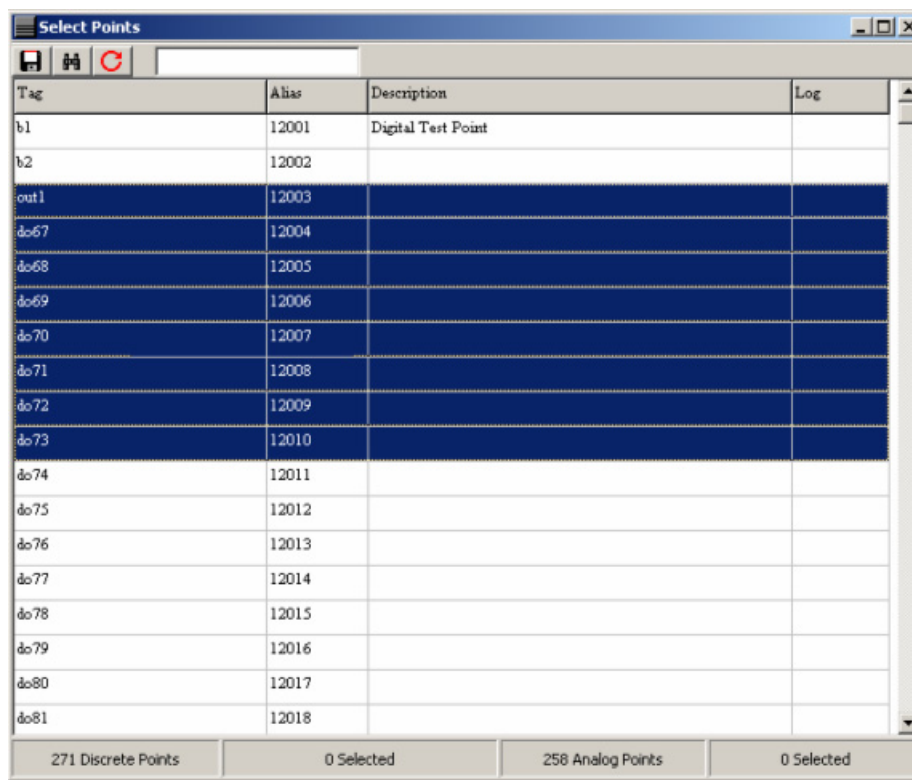


Tag	Alias	Description	Log
b1	12001	Digital Test Point	
b2	12002		
out1	12003		
do67	12004		
do68	12005		
do69	12006		
do70	12007		Yes
do71	12008		
do72	12009		
do73	12010		
do74	12011		
do75	12012		
do76	12013		
do77	12014		
do78	12015		
do79	12016		
do80	12017		
do81	12018		

271 Discrete Points 1 Selected 258 Analog Points 0 Selected

When a point is selected the counter at the bottom of the screen will increment to reflect the number of points (analog or discrete) that have been selected. To deselect the tag simply right click on the highlighted tag again; the "Yes" will disappear, and the counter at the bottom of the screen will decrement, indicating that the tag will not be scanned by TriLogger.

- 3 To quickly select groups of tags, hold down the left mouse button while dragging the cursor along the rows of tags. All rows passed in this manner will be highlighted.



- 4 To select all the highlighted tags right click on the mouse and the Log Point column will display "Yes" and the counter at the bottom of the screen will increase, indicating that all the highlighted tags will be scanned by TriLogger.

Groups of points can also be selected by left clicking on the desired starting tag, moving the mouse to the desired end position, holding the shift key and left clicking again. This will highlight and select the desired points in one simple operation.

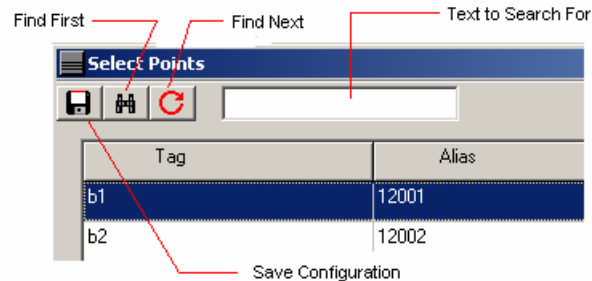
Tag	Alias	Description	Log
b1	12001	Digital Test Point	
b2	12002		
out1	12003		Yes
do67	12004		Yes
do68	12005		Yes
do69	12006		Yes
do70	12007		Yes
do71	12008		Yes
do72	12009		Yes
do73	12010		Yes
do74	12011		
do75	12012		
do76	12013		
do77	12014		
do78	12015		
do79	12016		
do80	12017		
do81	12018		


271 Discrete Points 8 Selected 256 Analog Points 0 Selected

To deselect the highlighted points simply right click the mouse again and the "Yes" will disappear.

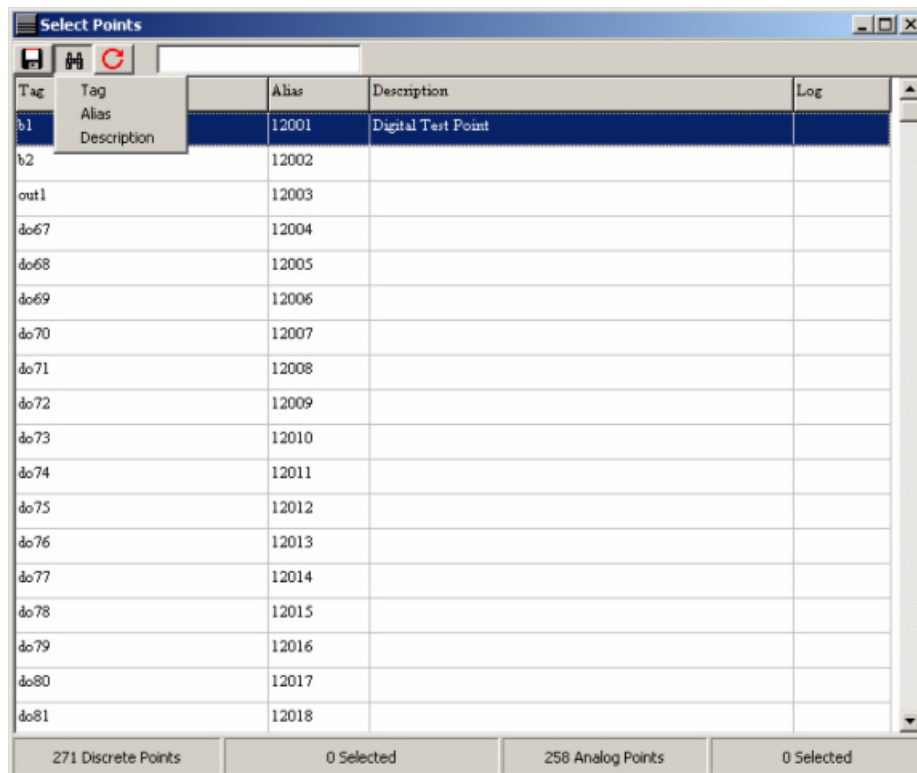
Searching for Specific Tags

With large numbers of tags in a Tricon database it can be difficult to locate a specific tag. The toolbar across the top of the screen has three icons and a text entry window located on it.




- 1 To search for a particular tag, point description, or alias type the text string into the text box and click on the binocular  button.

A drop down menu will appear which allows a choice of search field. Searches can be made on tag name, alias number, or point description. A point description may or may not be present depending on whether or not the Tricon programmer entered a description in the original TS1131 program. An example of a search is shown below:

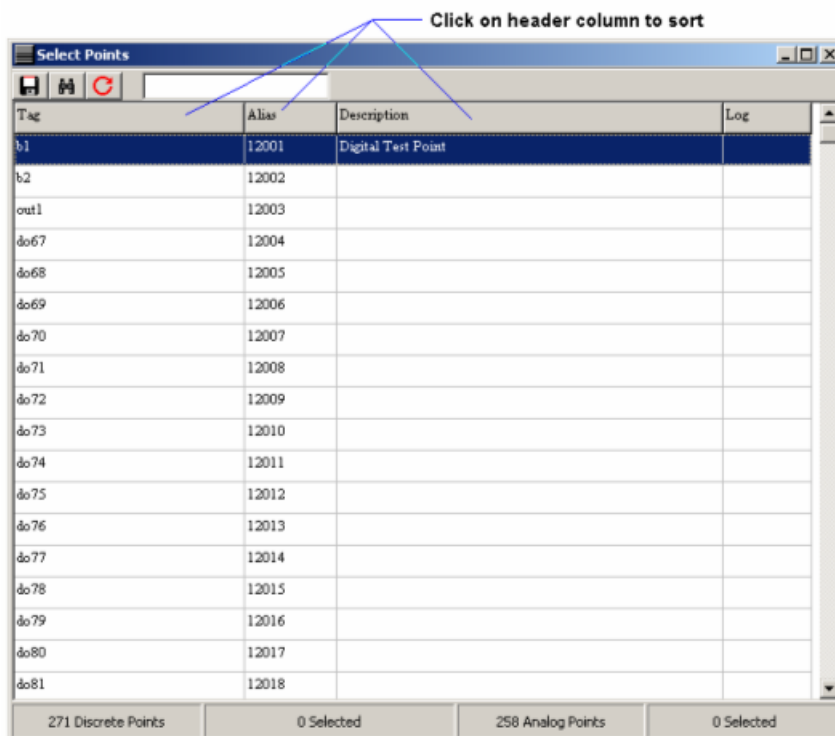


This particular search was for the tag name. It could just have well been for an alias number or a description. The search is not case sensitive and it does not have to be an exact match. For example searching for an alias number of 33 would find the first alias number that contained the number 33.


To find the second occurrence of a search string click on the find next  button. This will find the next occurrence in the list of the text contained in the database. When no further incidences of a search string can be found an error dialog will appear stating that the search string could not be found.

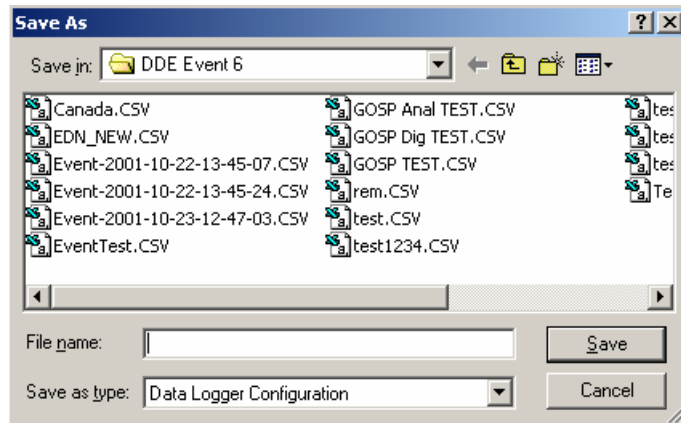
Sorting the Database

A left click on the top column will sort the database in ascending order (A to Z). Clicking on the column a second time will invert the database sort (Z to A). The database may be sorted on any column.



Saving the Configuration

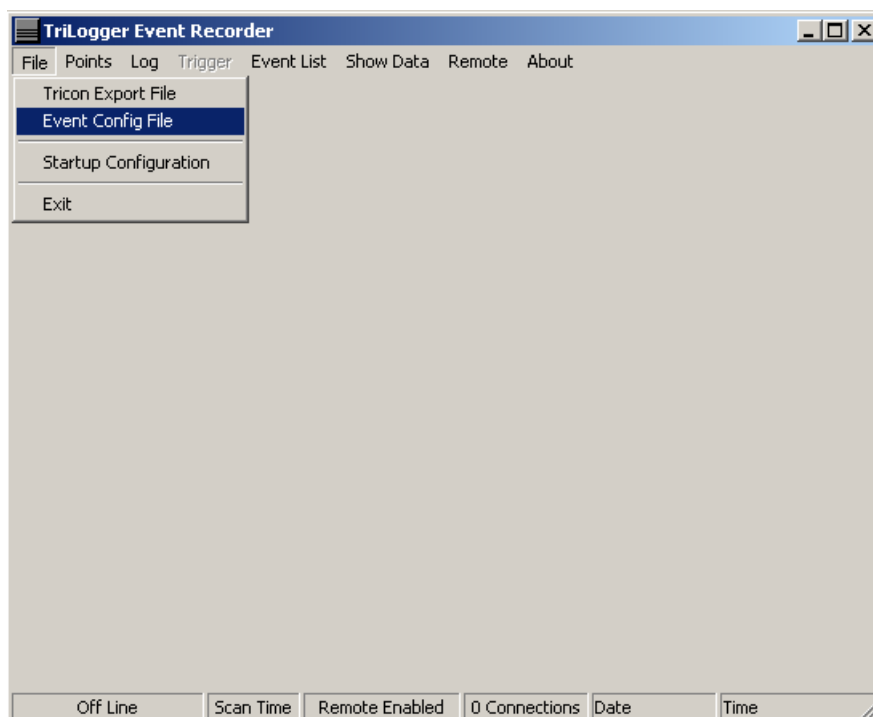
Clicking the save  button will bring up a save file dialog that allows the user to save the configuration to a new file, or to append an existing configuration. Appending to an existing configuration is how points are added. The file is again stored in an Excel readable format (*.CSV to differentiate it from the Tricon 1131 export file 'txt' and the MSW export file 'ASC').



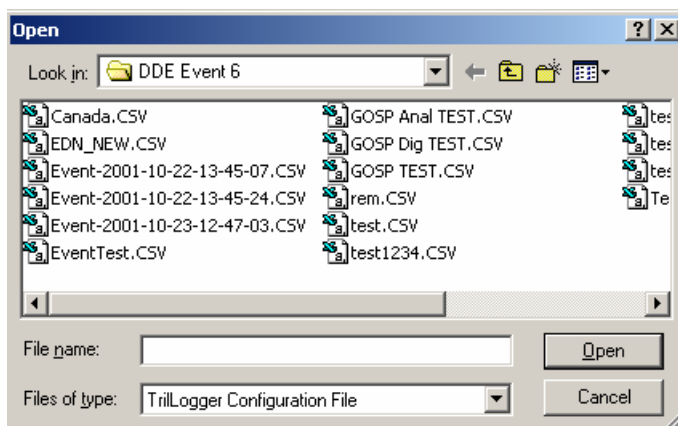
The saved file contains the tags that will be recorded during an event. The next step is to define which tags and what conditions that will trigger the event.

Reading Saved Configuration

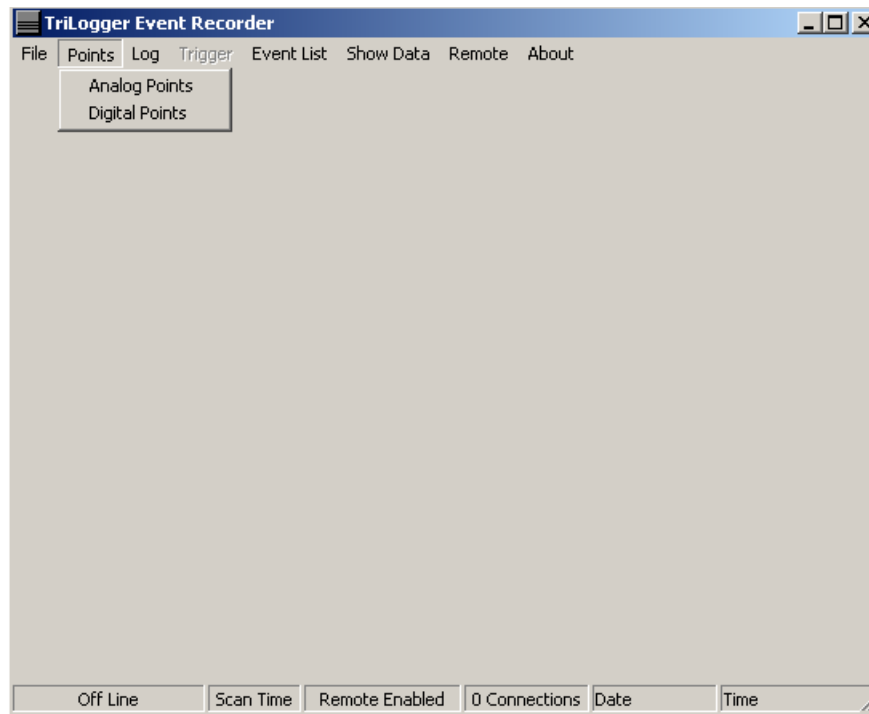
The points selected above can be configured directly, or a previously saved Event Configuration File as shown below:



- 1 Clicking on the Event Config File selection will bring up a file selection dialog as shown below:



Once the file is selected information about the points selected will be read from the file. These points are configured by opening the Configure selection from the main menu as shown below. Analog or Discrete points are selected on separate screens since the data required for analog triggers differs from that required for analog triggers.



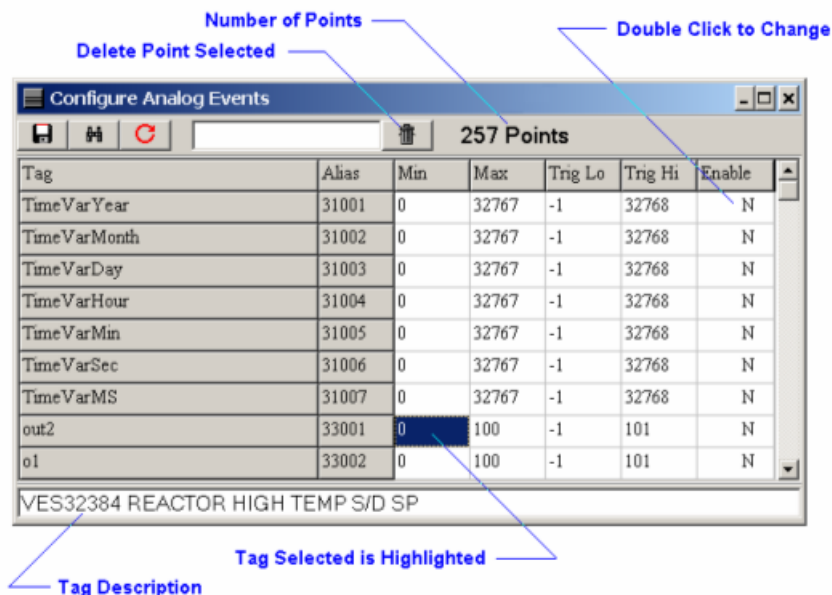
Configuring Event Triggers

The configuration file saved in the step above is not complete. At least one selected point must be configured to trigger an event (there is no upper limit on triggers, every point selected can trigger an event if so desired). This can be a discrete point turning on or off, or an analog point moving beyond a trigger threshold (high, low or both).

While the points selected above can be configured without saving, it is recommended that a base file be saved prior to defining the event triggers and thresholds. These selected but unconfigured points can then serve as a basis for multiple configuration files. Multiple configuration files may be useful for triggering event logs based on different root causes, or special operating conditions such as plant start up. Multiple configuration files can be easily loaded on the fly as long TriLogger is communicating with the same Tricon.

Configuring Analog Points

- 1 Selecting Analog Points on the screen above will bring up the analog configuration menu.



This form holds all the information about every analog point selected from the Tricon database. This is the information contained in the Tricon. All the information may be edited by the end user except the tag name and the alias. Tag name and alias cannot be edited because these are the links that TriLogger uses to read the data and they must match the Tricon database.

Min and Max Engineering ranges defined in the Tricon are primarily intended to allow proper scaling of analog points for Modbus communications. They have no direct purpose in an Ethernet communications link since scaling is not required, but they are necessary to properly display graphical trend data in the TriLogger Playback and TriLogger Remote programs. If a point has not been specifically given min and max engineering ranges in the Tricon database they should be assigned here so they can be properly displayed.

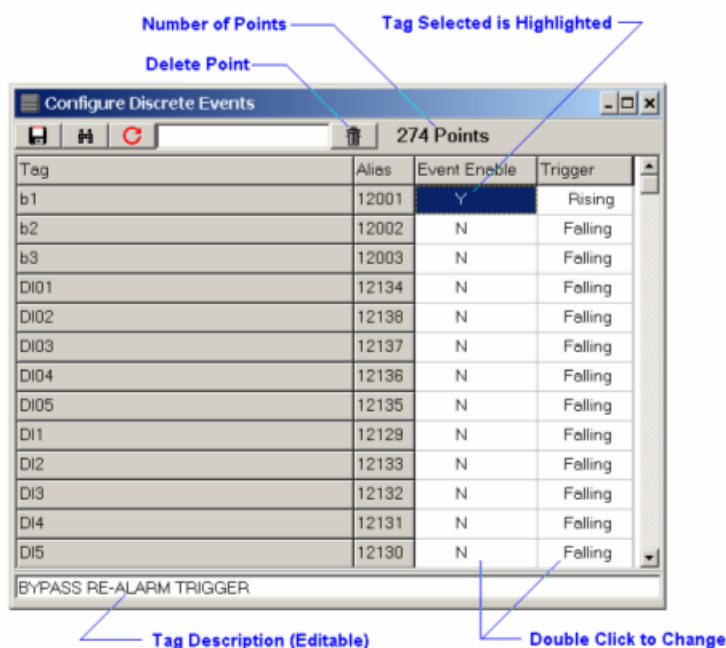
A selected tag can be deleted from the configuration by clicking on the trashcan icon. A dialog will open asking for confirmation prior to deleting the point. A point cannot be added or deleted while TriLogger Event is collecting data. To delete a point from a running TriLogger Event, simply stop logging, delete the point, and start logging again.

If it is desired to trigger an event from an analog value, a trigger setpoint must be defined (high, low or both) and the Enable Event flag must be set to true. Event flags can be toggled on or off by double clicking the Event Enable column in the row of the desired tag.

Specific tags and alias' can be located using the search functions defined in the previous Point Selection section.

Configuring Discrete Points

- 1 Selecting Discrete Points will bring up the following form:

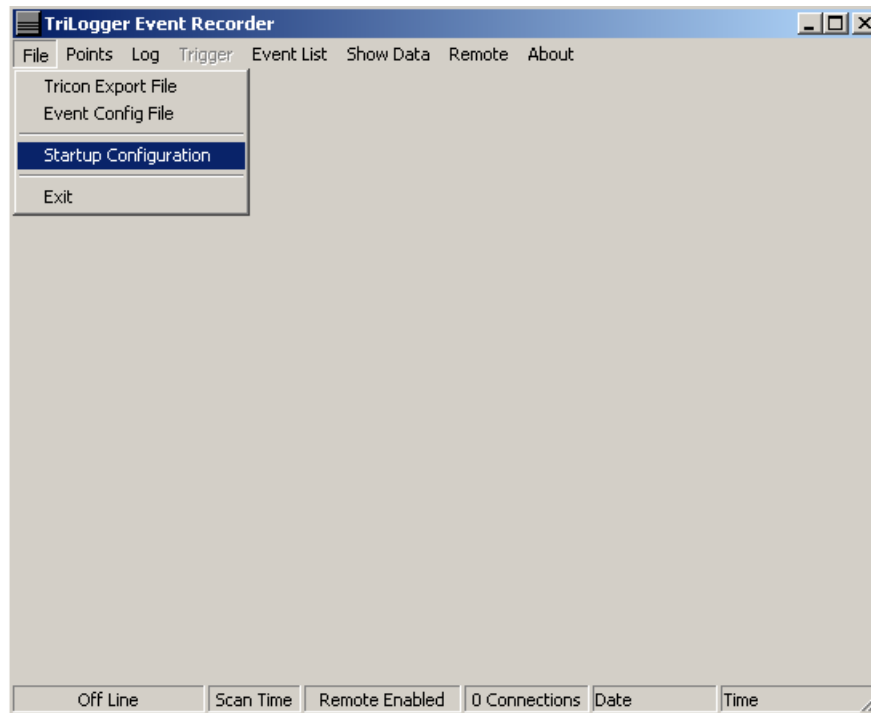


As with the analog configuration the tag and alias numbers cannot be edited. To enable an event to be recorded double click on the Event Enable column of the selected tag. This will toggle the Event Enable flag on or off. In addition the trigger itself must be defined as rising or falling edge. A rising edge is a transition of a discrete point from a false to a true state. In other words from a 0 to a 1. In this case an event will be triggered if the Enable Event flag is true, and the discrete changes from false to true (or 0 to 1). A falling edge would trigger an event if the point went from true to false (or 1 to 0).

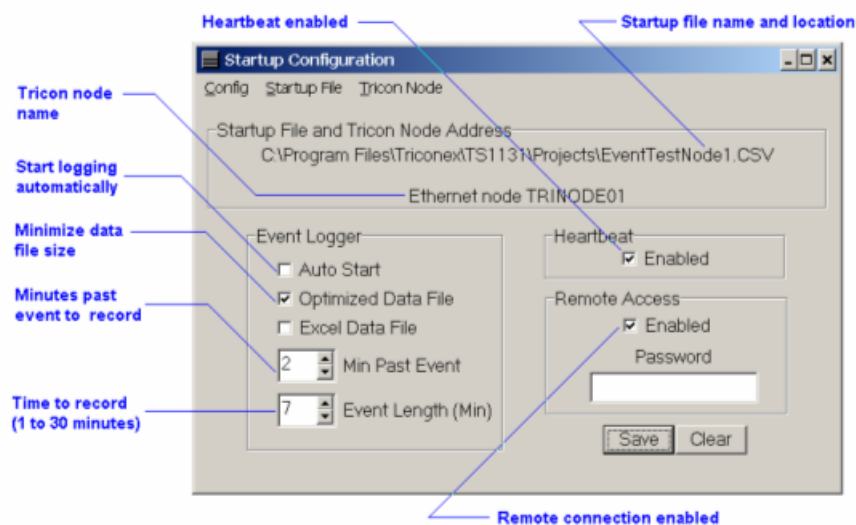
The TriLogger does not specifically allow an event trigger on a change of state, however this can be accomplished by appending a point to an existing configuration so that it appears twice. This duplication of a point allows setting one trigger as rising and the other as falling. In this special case an event will be triggered on a discrete change of state.

TriLogger Startup Configuration

- 1 The TriLogger startup defines the configuration file used, the event file storage format, the amount of time before and after the event to save data, whether or not remote logins are allowed, and if so defines the login password.



- 2 Selecting the Startup option from the file menu displays the Startup configuration form.



Before TriLogger can begin event recording certain configuration items must be determined:

- Startup File and Tricon Node Address -
 - Selecting the Startup File menu option opens a file dialog that allows an Event Configuration file defining the selected points and the triggers for the event file
 - Selecting the Tricon Node menu option allows the node name defined in the DDE Server configuration to be specified. This node name must match at least one Tricon node defined in the DDE Server configuration.

- Event Logger

- Select Auto Start to have the TriLogger begin communicating with the Tricon immediately after starting. If the DDE Server is not running, TriLogger will auto launch the DDE Server application.

For TriLogger to launch the DDE Server automatically both programs must reside in the same directory. The installation program will place both programs in the same directory, however if multiple TriLogger Event instances are run on the same PC they should have different working directories to allow individual configurations and even different Tricons to be logged. As events are saved to the hard disk they will be stored in the working directory defined above. No matter how many instances of TriLogger Event are running only one copy of the program is required.

- Select Optimized File Format is required for the TriLogger Playback. It also reduces the event file size and the time required to write the event file to disk after an event occurs.
- Select Excel File Format to save event data in an Excel Workbook format. This format cannot be used by TriLogger Playback and will be over 4 times larger than the Optimized File Format. Both file formats can be selected and TriLogger will generate both file formats after an event. If TriLogger Playback is used for data analysis it can export an Excel Workbook from the Optimized data file.

Due to the large size of an Excel file generated by TriLogger Event, a maximum of 500 analog and 500 discrete points can be collected using the Excel data file format. Optimized file format can record up to 2000 analog and 2000 discrete points. TriLogger Playback can only use the optimized data format but can export data to Excel regardless of point size.

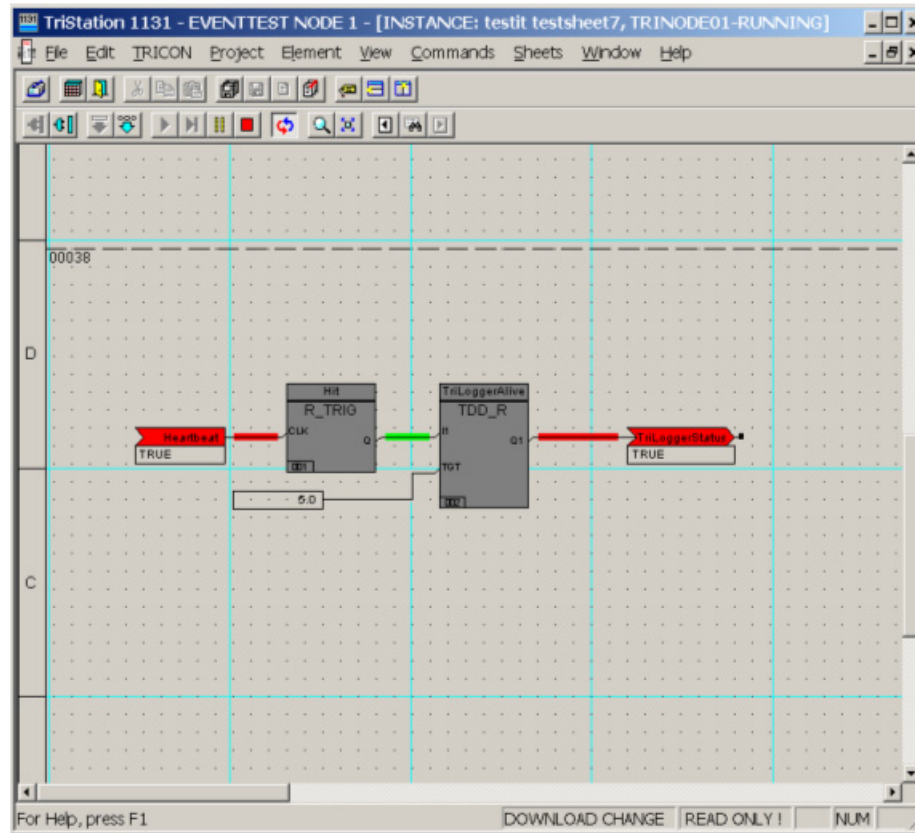
- Define the "Event Fence", the minutes past the event that TriLogger should continue to collect data before writing the event file to disk in the selected format. For example an Event Fence of 2 would mean that TriLogger will continue to record data for 2 minutes past the event itself. Valid Event Fence times range from 0 to the configured event length.
- Event Length is the time that TriLogger will record data. From 1 to 30 minute event lengths can be configured. Longer event lengths generate longer data files and require more computer memory. Event length cannot be changed while TriLogger Event is collecting data. To change the event length simply stop logging, change the event length and restart.

- Heartbeat

- TriLogger Event can toggle a discrete point on and off in the Tricon. The Tricon can then monitor whether or not TriLogger is active and generate an alarm if TriLogger Event stops toggling the bit indicating a failure.
- A red LED will turn on and off on the main form as the heartbeat is toggled on and off within the Tricon. This is a visual indication that the heartbeat function is enabled.
- To utilize the Heartbeat function a tag containing the word Heartbeat must exist within the Tricon database as a discrete memory point (this means the point can be written to). If the Heartbeat tag does not exist then enabling heartbeat will have no effect.
- Remote Access
 - Select Enabled to allow remote access by TriLogger Remote.
 - Define a password that a remote user must have to log onto TriLogger. Without this password a remote user cannot successfully log on and view data from a remote location. If left blank no password is required.

Configuring the Heartbeat in Tricon

To use the Heartbeat function, Tricon must be configured to recognize when TriLogger Event is active. In addition to creating a tag name as a discrete memory point an alarm must be configured to detect the toggling of the heartbeat. The example below is one way an alarm can be generated when the heartbeat stops toggling.



The code above is just a suggestion, there are many ways this could be programmed. In this example the heartbeat toggle is connected to an R_TRIG function block. This function (Rising Edge Trigger) sets its output true when a rising edge is detected on its input. The output from R_TRIG is connected to TDD_R (Time Delay to De-Energize). This function will hold its output true for the time (in this case 5 seconds) connected to the TGT input. As the heartbeat toggles on and off the R_TRIG function sets the time delay to de-energize the alarm contact. If the heartbeat stops toggling for more than 5 seconds the TriLoggerStatus alarm tag will become false, generating an alarm that TriLogger Event has stopped and corrective action needs to be taken.

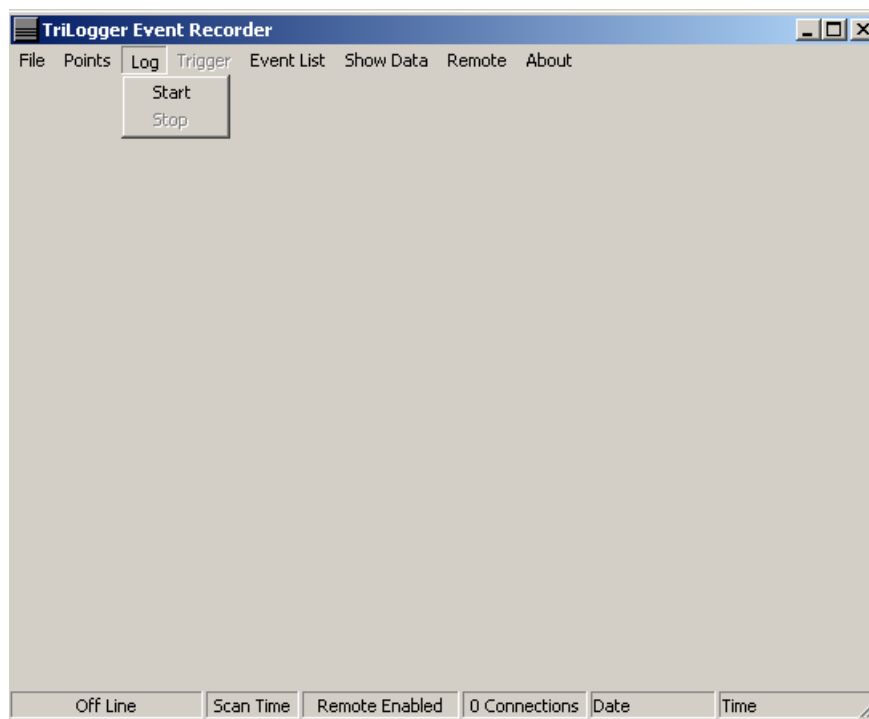
At first glance it may appear that the R_TRIG function is not necessary and the Heartbeat could be connected directly to the TDD_R. The problem with this approach is that the Heartbeat flag can fail either on or off. If the Heartbeat fails in the off state connecting to TDD_R would work fine. If Heartbeat fails in the on state TDD_R will not activate The TriLoggerStatus alarm.

Operation

TriLogger is designed to be simple to operate. From the main menu there are a series of options that are available:

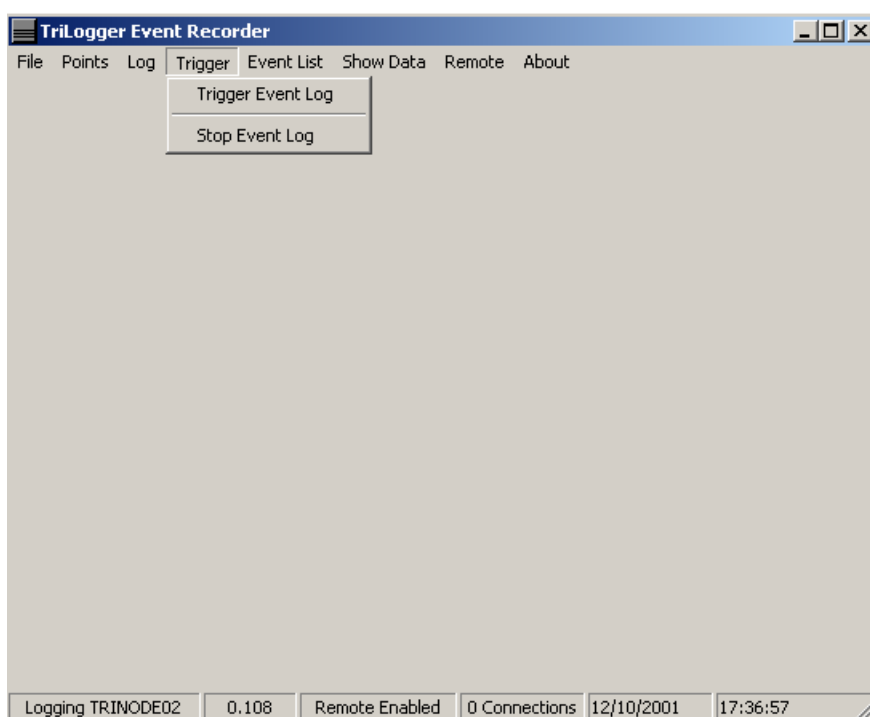
Log

- Log Start - starts logging data if TriLogger has not been configured to automatically start logging. The Start option is grayed out and cannot be selected if TriLogger Event is already collecting data.
- Log Stop - stops TriLogger from logging events. The Stop option is also grayed out if TriLogger Event is not collecting data.



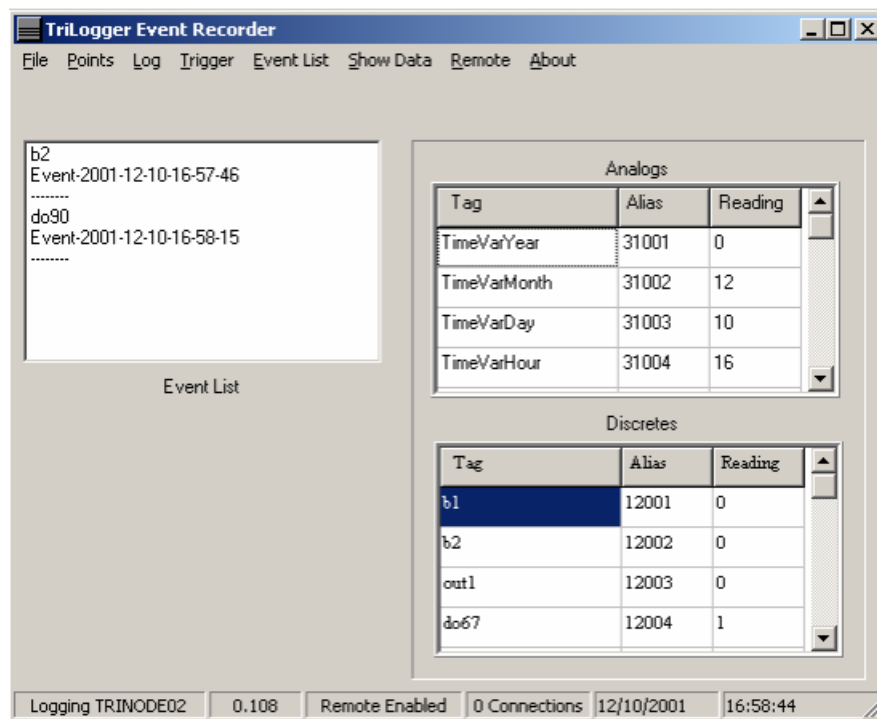
Trigger

- Trigger Event Log - manually forces an event log to be written to disk even though no configured trigger point has exceeded a trigger limit.
 - If an event has been triggered by a configured point and TriLogger is still collecting data prior to writing to disk; a manual trigger from TriLogger will immediately flush all data to the disk. Manual triggers, local within TriLogger Event, or remote from TriLogger Remote will over ride a configured trigger and immediately flush data to disk. Manual operator actions always take precedence over configured trigger points.
- Stop Event Log - will prevent an event log from being written to disk if it was initiated by a trigger limit. This is useful for debugging to prevent recording of event files that have minimal value.



Event List

The Event List option opens a window that shows the file names of the recent events that have occurred. The most recent 5 events, time of occurrence and trigger point are displayed in the Event List window. If the event was triggered manually there is no trigger point, but the source of the manual trigger is displayed. In the example below one of the listed events was triggered from TriLogger Remote. In this example the Show Data option has been selected as well and real time data is displayed in the data window.



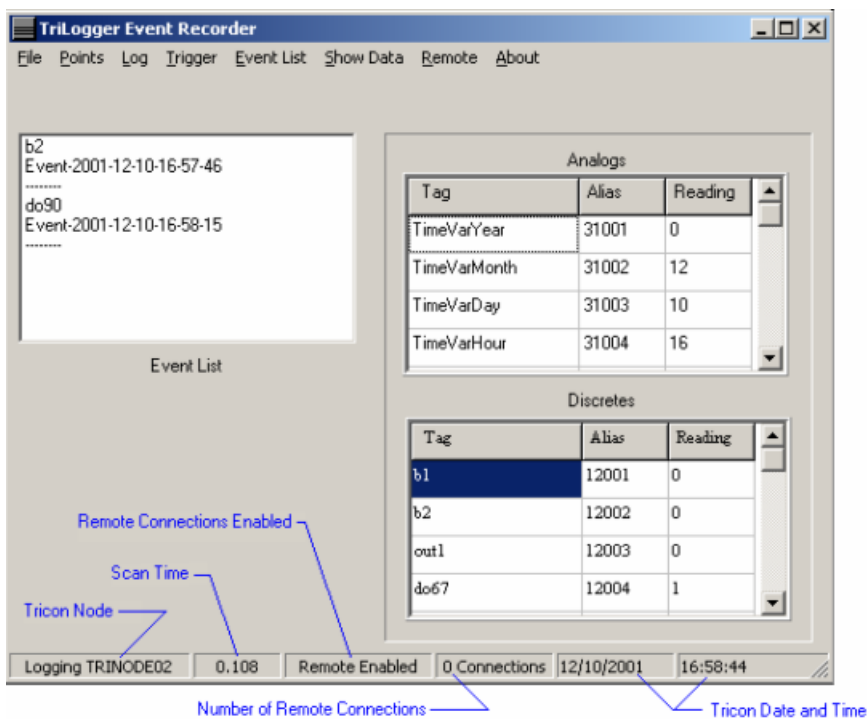
Show Data

Show Data opens a window that allows all the data being scanned by TriLogger to be viewed in real time. Analog and discrete data is updated once per second in the data windows. Discrete data is represented by a 1 (on) or a 0 (off).

Scan Time

The scan time shown above located in the status bar across the bottom of the screen represents the time required for TriLogger Event to update the points request from the Tricon. Scan time can be affected by many things and a detailed discussion is contained in Chapter 4, Theory of Operations. The scan time cannot be less than the Tricon scan time, and may be greater depending on the number and type of points configured to be scanned.

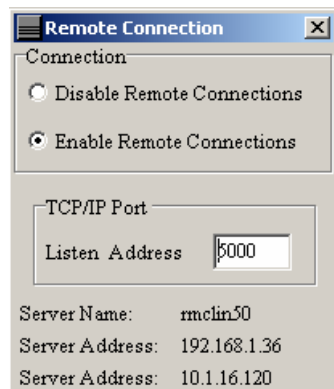
If a transient can occur and clear faster than the TriLogger can gather data it is recommended that the Tricon SOE product be used by itself or in conjunction with TriLogger.



Remote

The Remote option allows TriLogger Remote users to connect to TriLogger over a local area network and collect real time data. TriLogger Remote can also be run on the same PC as TriLogger Event. This allows local graphical trending of data.

Running multiple instances of TriLogger Event, TriLogger Remote and TriLogger Playback on the same PC is allowable, however the user should be aware that this can cause degraded scan times. No errors will occur, however the scan time shown in the status window below will increase.



Remote connection options

- **Enable/Disable** - this option allows TriLogger Remote programs in other locations to connect to the TriLogger Event PC over an Ethernet link. It is recommended that an Ethernet link be reserved for the TriLogger Event to Tricon communications and a second Ethernet link be used for communications tasks (such as TriLogger Remote and TriLogger Playback file access).
- **TCP/IP listen address.** All TCP/IP communications must have a unique listen address. Many listen addresses are reserved for network applications and should not be selected for use by TriLogger (low numbers such as 23 or 80 are reserved for applications such as email). TriLogger Remote must be configured to match the TriLogger Event listen address.
- **Server Name** - this is the name of the PC running the TriLogger Event software. If this PC is connected to a local area network that supports Domain Name Service (DNS), TriLogger Remote can locate the TriLogger Event by name on the network. Names are easier to remember than numeric TCP/IP addresses.
- **Server Address** - Up to two addresses are shown from Ethernet boards installed in the TriLogger Event PC for convenience. TriLogger Remote can use connect using the numeric TCP/IP address if Domain Name Service is not available.

Log Files

Log files are automatically generated by TriLogger Event to keep a running record of events and maintenance information. The log is saved in the directory with TriLogger Event under the file name EVENT.LOG. TriLogger Event log file is a simple text format that can be viewed with any editor or word processor that supports ASCII text, such as NOTEPAD.EXE that comes with Microsoft Windows.

If the log file is deleted TriLogger Event will generate a new file on the next log event occurrence. Items recorded in the log file, together with the time that they occurred are:

- Log start time and Tricon node being logged
- Log stop time
- Triggered events, including the tag name and time of the trigger
- Remote enable/disable and time
- Remote login/logoff and time
- Loss of communication errors and time
- Re-establishment of communications and time

Below is an example of the information contained in a TriLogger Event log file.

```
06/03/2002 11:26:18 Logging TRINODE01
06/03/2002 11:26:19 1 Connections
06/03/2002 11:26:19 Logon Successful
06/03/2002 11:26:37 0 Connections
06/03/2002 11:26:56 1 Connections
06/03/2002 11:26:57 Logon Successful
06/03/2002 11:27:20 0 Connections
06/03/2002 11:38:51 1 Connections
06/03/2002 11:38:53 Logon Successful
06/03/2002 11:40:47 2 Connections
06/03/2002 11:40:48 Logon Successful
06/03/2002 11:41:16 3 Connections
06/03/2002 11:41:17 Logon Successful
06/03/2002 11:42:31 b2 Triggered Event
06/03/2002 11:48:00 Remote Trigger
06/03/2002 11:48:00 b2 Triggered Event
06/03/2002 11:48:07 Remote Trigger
06/03/2002 11:51:28 b2 Triggered Event
```

06/03/2002 11:51:59 Event Log Stopped

06/03/2002 11:52:00 2 Connections

06/03/2002 11:52:00 1 Connections

06/03/2002 11:52:00 0 Connections

06/03/2002 11:52:00 Logging Stopped



Invensys® Wonderware® InTouch®

Dynamic Data Exchange (DDE)	560
InTouch I/O Addressing	561
InTouch Access Names	562
Defining an I/O Item in InTouch	566

Dynamic Data Exchange (DDE)

Dynamic Data Exchange (DDE) is a communication protocol developed by Microsoft to allow applications in the Windows environment to send/receive data and instructions to/from each other. It implements a client-server relationship between two concurrently running applications.

The server application provides the data and accepts requests from any other application interested in its data. Requesting applications are called clients. Some applications such as InTouch® and Microsoft Excel can simultaneously be both a client and a server.

Note: NetDDE, used for network communication to non-Wonderware I/O sources, is supported on Windows XP and Windows 2000, but not on Windows 2003 Server. For more information on NetDDE, please see the InTouch User documents.

Wonderware SuiteLink™

Wonderware SuiteLink uses a TCP/IP-based protocol. SuiteLink is designed specifically to meet industrial needs, such as data integrity, high-throughput, and easier diagnostics. This protocol standard is supported for both Windows® 2000 and Windows XP.

SuiteLink is Not a Replacement for DDE. Wonderware recommends that DDE be used for internal client communication, and SuiteLink for communication over the network.

Each connection between a client and a server depends on your network situation.

SuiteLink provides the following benefits:

- Consistent high data volumes can be maintained between applications, regardless of whether the applications are on a single node or distributed over a large node count.
- Value Time Quality (VTQ) places a time stamp and quality indicator on all data values delivered to VTQ-aware clients.
- Extensive diagnostics of the data throughput, server loading, computer resource consumption, and network transport are made accessible through the Microsoft Windows NT® operating system performance monitor. This feature is critical for the scheme and maintenance of distributed industrial networks.
- The network transport protocol is TCP/IP using Microsoft's standard Winsock interface.

To use the SuiteLink Communication Protocol, the following conditions must be satisfied:

- You must have Microsoft TCP/IP configured and working properly.
- You must use computer names (Node Names) of no more than 15 characters.
 - For more information on installing and configuring Microsoft TCP/IP, see your Microsoft Windows operating system's documentation.
- Wonderware SuiteLink must be running as a service. If for some reason SuiteLink has been stopped, you will need to start it again. (SuiteLink is automatically installed as a Common Component when you install InTouch. It is configured to startup automatically as a Windows Service).

InTouch I/O Addressing

InTouch identifies an element of data in an I/O Server program by using a three-part naming convention that includes the application name, topic name and item name. To obtain data from another application, the client program (InTouch) opens a channel to the server program by specifying these three items.

In order for InTouch to acquire a data value from another application, it must also know the name of the application providing the data value, the name of the topic within the application that contains the data value, and the name of the specific item within the topic. In addition, InTouch needs to know the data's type: discrete, integer, real (floating point), or message (string).

This information determines the I/O type for the gagman when it is defined in the InTouch database. When WindowViewer is running, it will automatically perform all of the actions required to acquire and maintain the value of this item.

For example, in the case of Excel, the application name is "Excel," the topic name is the name of the specific spreadsheet that contains the data and the item name is the identification of the cell on the spreadsheet to/from which the data is to be read/written.

When another Windows application requests a data value from InTouch, it also must know the three I/O address items.

The following describes the I/O address convention for InTouch:

- **VIEW** (application name) identifies the InTouch runtime program that contains the data element.
- **TAGNAME** (topic name) is the word always used when reading/writing to a tagname in the InTouch database.
- **ActualTagname** (item name) is the actual tagname defined for the item in the InTouch Tagname Dictionary.

For example, to access a data value in InTouch from Excel (running on the same node), a DDE Remote Reference formula would be entered in the cell into which the data value is to be written:

=VIEW|TAGNAME!'ActualTagname'

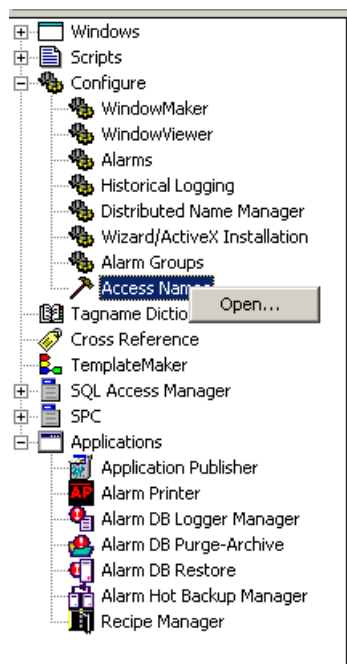
InTouch Access Names

When you create I/O-type tags or remote tagname references, they must be associated with an Access Name. Access Names contain the information that is used to communicate with other I/O data sources including the node name, application name and topic name.

Create an Access Name

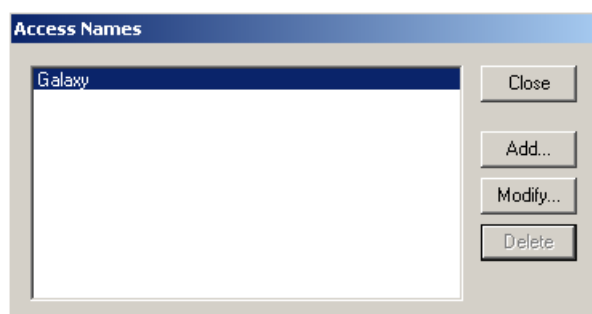
Note: The Access Name configured in the following steps is used in the subsequent lab.

- 1 Expand **Configure** in the **Application Explorer**.
- 2 Right-click **Access Names** and select **Open**.



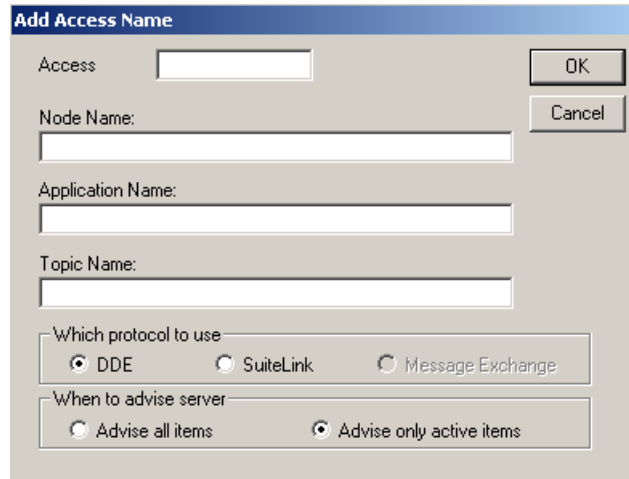
Note: **Access Names** can also be created while you are defining an I/O-type tag in the Tagname dictionary.

The **Access Name** dialog box appears:



- 3 Click **Add**.

The **Add Access Name** dialog box opens:



Access Name: Enter the new Access Name (For simplicity, use the same name that you will use for the topic name.).

InTouch uses Access Names to reference real-time I/O data. Each Access Name equates to an I/O address, which can contain a Node, Application, and Topic. In a distributed application, I/O references can be set up as global addresses to a network I/O Server or local addresses to a local I/O Server.

Node Name: Used when the data is from a remote I/O Server over the network.

Application Name: Enter the actual program name for the I/O Server program from which the data value will be acquired. If the value is coming from a Wonderware Modbus I/O Server, MODBUS is used. Do not enter the .exe extension portion of the program name.

Topic Name: Enter the topic name to access. The Topic Name is an application-specific sub-group of data elements. In the case of data coming from a Wonderware I/O Server program, the topic name must be the exact same name given to the spreadsheet when it was saved. For example, Book1.xls.

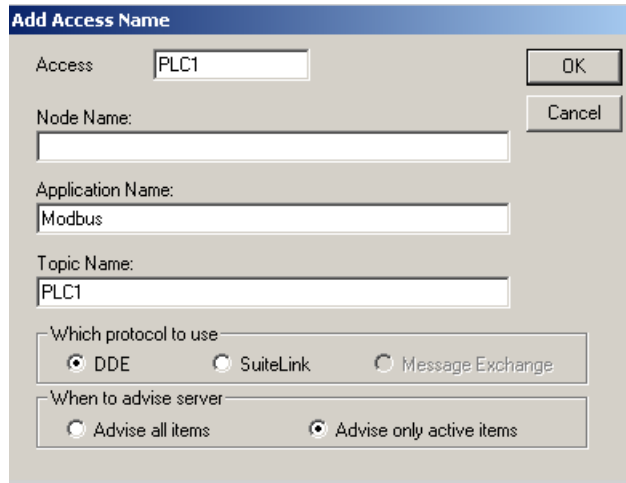
Which Protocol to use: Select DDE (internal) or SuiteLink (network).

When to advise server:

Advise all items: Select if the server program is to poll for all data whether or not it is in visible windows, alarmed, logged, trended or used in a script. Selecting this option will impact performance and is not recommended.

Advise only active items: Select if the server program is to poll only points in visible windows and points that are alarmed, logged, trended or used in any script. A Touch Pushbuttons action script will not be polled unless it opens in a visible window.

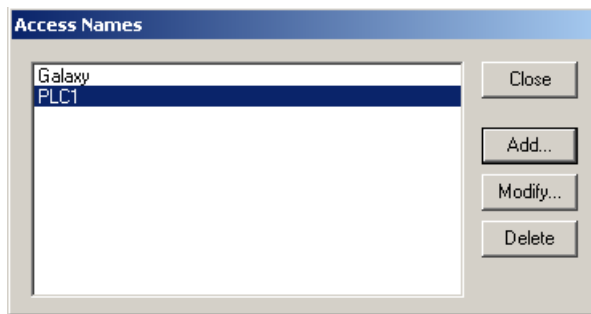
The following figure shows an example of a completed access name (if the I/O server is on the local node, the Node Name: can be left blank as shown in the following figure):



The **Add Access Name** dialog box contains the following fields and options:

- Access:** Text box containing "PLC1".
- Node Name:** Empty text box.
- Application Name:** Text box containing "Modbus".
- Topic Name:** Text box containing "PLC1".
- Which protocol to use:** Radio buttons for ☒ DDE, ☐ SuiteLink, and ☐ Message Exchange.
- When to advise server:** Radio buttons for ☐ Advise all items and ☒ Advise only active items.
- Buttons:** OK and Cancel.

- 4 Click **OK** to accept the new **Access Name**.



The **Access Names** dialog box displays a list of access names:

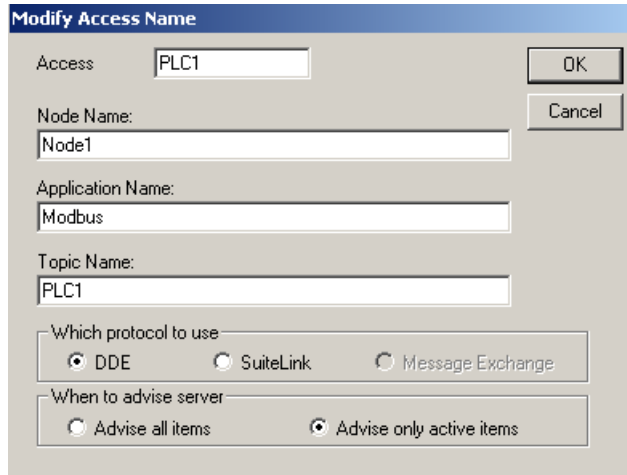
- Galaxy
- PLC1 (selected)

Buttons on the right include: Close, Add..., Modify..., and Delete.

- 5 Click **Close** to close the dialog box.

Modifying or Deleting an Access Name

- 1 Right-click **Access Names/Open**.
The **Access Names** dialog box appears.
- 2 Select the access name to modify.
- 3 Click **Modify**.
The **Modify Access Name** dialog box appears.
- 4 Make any changes (the following figure shows the Node Name has been added) and click **OK**.



Modify Access Name

Access:

Node Name:

Application Name:

Topic Name:

Which protocol to use:

☒ DDE ☐ SuiteLink ☐ Message Exchange

When to advise server:

☐ Advise all items ☒ Advise only active items

OK Cancel

- 5 Repeat this procedure if you need to modify other **Access Names**.
- 6 To delete an Access Name, select it in the list and then click **Delete**. A message box will open asking you to confirm the deletion. Click **Yes**.
- 7 Click **Close** or repeat this procedure to delete other defined Access Names.

Note: **Access Names** used by tags cannot be deleted.

Defining an I/O Item in InTouch

InTouch can receive data from other local or remote Windows applications when I/O-type tags are defined in the Tagname Dictionary. Each I/O-type tag references a valid item in the I/O Server program.

Defining an I/O Type Tagname

All I/O type tags receive their values from other Windows application programs such as Excel and I/O Servers. This value is referred to as the "raw" value. When you define a tag in the Tagname Dictionary, you must enter values for the **Min-** and **Max** Raw. These values are used by the database as clamps on the actual raw value received from the I/O device. For example, if you set the **Min Raw** value to 50 and the actual value received from an I/O Server is 0, the database will force the Raw value to 50.

InTouch does not display raw values. Instead, it displays engineering units (EU). When you define an I/O type tag in the Tagname Dictionary, you must specify values for the **Min-** and **Max** EU. These values are used to scale the raw value to the displayed value. If you do not want to do scaling or your I/O device does the scaling for you, set the Min/Max EU values equal to the Min/Max Raw values.

For example, assume that a flow transmitter wired to a PLC register generates a value of zero at no flow and a value of 9999 at 100% flow. The following values would be entered:

Min EU = 0 Max EU = 100

Min Raw=0 Max Raw = 9999

A raw value of 5000 would be displayed as 50.

Assume that a flow transmitter wired to a PLC register generates a value of 6400 at no flow and a value of 32000 at 300 GPM.

Min EU = 0 Max EU = 300

Min Raw = 6400 Max Raw = 32000

In this case, a raw value of 12800 would be displayed as 150. A raw value of 6400 would be displayed as 0 and a raw value of 0 would be displayed as 0 (all values outside the boundaries set by the Min Raw and Max Raw values are clamped).

The above scaling works in reverse when the I/O tag data is written from the InTouch Tagname Dictionary to other Windows applications.

Define an I/O Tagname

- 1 Press **Ctrl+T**.

The **Tagname Dictionary** dialog box appears:

Tagname Dictionary

☐ Main
 ☒ Details
 ☐ Alarms
 ☐ Details & Alarms
 ☐ Members

New Restore Delete Save << Select... >> Cancel Close

Tagname: R4001 Type: ... Memory Integer

Group: ... \$System ☐ Read only ☒ Read/Write

Comment:

☐ Log Data ☐ Log Events ☐ Retentive Value ☐ Retentive Parameters

Initial Value: Min Value: Deadband:

Eng Units: Max Value: Log Deadband:

- 2 Click **New** to clear the **Tagname**: text field.

The first time the Tagname Dictionary is accessed, the definition for the internal system tag \$AccessLevel is displayed.

When other tags have been defined in the Tagname Dictionary, the last edited tagname's definition is displayed.

Tip: Right-click on any of the text entry boxes in any of the Tagname Dictionary dialog boxes. A menu will open displaying the commands that you can apply to the selected text:

Tagname Dictionary

☐ Main
 ☒ Details
 ☐ Alarms
 ☐ Details & Alarms
 ☐ Members

New Restore Delete Save << Select... >> Cancel Close

Tagname: Context Menu: Undo, Cut, Copy, Paste, Delete, Select All

Type: ... I/O Integer

Group: ... \$System ☐ Read only ☒ Read/Write

Comment: AccessLevel

☐ Log Data ☐ Retentive Value ☐ Retentive Parameters

Initial Value: Min EU: -32768 Max EU: 32767

Deadband: 0 Min Raw: -32768 Max Raw: 32767

Eng Units: Log Deadband: 0

Access Name: ... PCL1

Conversion: ☒ Linear ☐ Square Root

Item: ☐ Use Tagname as Item Name

- 3 In the **Tagname** box, enter a name for the new tagname.
- 4 Click **Type**. The **Tag Types** dialog box opens:

Tagname Dictionary

☐ Main
 ☒ Details
 ☐ Alarms
 ☐ Details & Alarms
 ☐ Members

New Restore Delete Save << Select... >> Cancel Close

Tagname: R4001 Type: ... Memory Integer

Group: ... \$System ☐ Read only ☒ Read Write

Comment:

☐ Log Data ☐ Log Events ☐ Retentive Value ☐ Retentive Parameters

Initial Value: Min Value: Deadband:

Eng Units: Max Value: Log Deadband:

- 5 Select **I/O Discrete** for this tag.
- 6 Click **OK**.
- 7 The respective details dialog box will open:

Tagname Dictionary

☐ Main
 ☒ Details
 ☐ Alarms
 ☐ Details & Alarms
 ☐ Members

New Restore Delete Save << Select... >> Cancel Close

Tagname: R4001 Type: ... I/O Integer

Group: ... \$System ☐ Read only ☒ Read Write

Comment: AccessLevel

☐ Log Data ☐ Log Events ☐ Retentive Value ☐ Retentive Parameters

Initial Value: 0 Min EU: -32768 Max EU: 32767

Deadband: 0 Min Raw: -32768 Max Raw: 32767

Eng Units: Log Deadband: 0

Conversion: ☒ Linear ☐ Square Root

Access Name: ... Unassigned

Item: ☐ Use Tagname as Item Name

If the details portion of the dialog box does not open, click **Details** at the top of the screen.

Specify all the required details for defining the item.

- 8 Click **Access Name**.

The **Access Names** dialog box opens:

Access Names

Galaxy

PLC1

Close

Add...

Modify...

Delete

- 9 Double-click the **Access Name** to use or select it and click **Close**.

In the above figure, the PLC1 access name is selected for this tag.

Note: The Type cannot be changed after changing the Item.

The selected Access Name (now associated with this tag definition) appears adjacent to the Access Name button in the details dialog box.

The screenshot shows the 'Tagname Dictionary' dialog box with the 'Details' tab selected. The 'Tagname' field contains 'R4001' and the 'Type' is 'I/O Integer'. The 'Group' is '\$System' and the 'Comment' is 'AccessLevel'. The 'Access Name' field shows 'PLC1'. The 'Item' field is empty. The 'Use Tagname as Item Name' checkbox is unchecked.

- 10 In the Item box, enter the item name for the data value in the I/O Server program.

Note: It is important to understand that the "tagname" is the name used within InTouch to refer to a data value (tag). The Item is the name used by a remote Windows application to refer to the same value. These names do not have to be the same, but it is recommended (when applicable) to use the same names.

- 11 Click **Close**.

Symbols

_O.SED extension, [456](#)

Numerics

10Base2 cable, connection to media converter, [293](#)

802.2 protocol

 configuring DDE Server properties, [470](#)

 configuring redundant networks, [472](#)

A

access, user access, [56](#)

ACM, Advanced Communication Module, [223](#)

address plug, Trident MP, [299](#), [307](#)

allocating memory points, Trident, [256](#)

allocating memory points, Tricon, [211](#)

Allow Disabling of Points property, [209](#)

Analog Input modules, [222](#)

Analog Output modules, [222](#)

annotations

 Annotation on by Default, [65](#)

 specifying default, [64](#)

 using macros, [66](#)

Annotations tab, [65](#)

application

 Build Application command, [335](#)

 building, [335](#)

 control, [50](#)

 definition, [7](#)

 determining type, [50](#)

 maintenance, [349](#)

 parts, [7](#)

 planning changes, [350](#)

 safety, [50](#)

 scan, [335](#)

 workspace, [21](#)

Application Data graphs, [213](#)

application-defined states

 definition, [175](#)

 enabling, [176](#)

ASCII mode

 characters in station address, [501](#)

ASCII mode (*continued*)

 defined, [499](#)

assignment statements, [154](#)

Auto Name, [110](#)

AutoScroll mode, selection, [454](#)

B

Begin Event Retrieval command, when used, [462](#)

BNC connectors, terminating, [292](#)

BOOL, specifying colors for monitoring, [69](#)

Build Application command, [335](#)

C

CASE statements, [155](#)

CEM

 displaying and sizing cells, [171](#)

 editor, tab properties, [172](#)

 editor, using, [167](#)

 element options, [174](#)

 FBD network, [169](#)

 matrix, [168](#)

 monitor colors and names, [173](#)

 programs options, [175](#)

 selecting cells, [170](#)

 user-defined functions, [175](#)

 Variable Detail Table, [169](#)

 Variable Detail Table properties, [177](#)

CEM language

 about, [17](#)

 editor settings, [73](#)

CEMPLE

 overview, [164](#)

 writing logic, [164](#)

Centronics printing

 printing setup, [317](#)

 using a Tricon EICM port, [314](#)

chassis

 adding or deleting for Tricon, [215](#)

 power usage for Tricon, [214](#)

chassis window, [218](#), [413](#)

checksum field, in Modbus message format, [501](#)

- clear function block, [450](#)
- clearing faults on all modules, [422](#)
- client/server communication
 - using DDE Server, [466](#), [474](#)
 - using OPC Server, [432–436](#)
- coils, [135](#)
- colors
 - specifying for monitoring, [69](#)
 - Tricon chassis window, [413](#)
 - Trident IOP window, [415](#)
- comments, [64](#)
 - adding in FBD and LD, [115](#)
 - Comment tab, [115](#)
 - Edit Fields tab, [117](#)
 - Pickup/Drop tab, [117](#)
 - Style tab, [116](#)
 - using macros, [66](#)
- conditional statements, [154](#)
- configuration
 - inserting Trident modules, [259](#)
 - removing Trident modules, [260](#)
 - steps, Tricon, [205](#)
 - steps, Trident, [251](#)
 - tree, [24](#)
 - Tricon operating parameters, [206](#)
 - Trident operating parameters, [252](#), [254](#)
- connecting to network, [411](#)
- constants, specifying in CEM, [176](#)
- contacts, [135](#)
- control application definition, [50](#)
- control strategy, developing, [50](#)
- controller
 - adding or deleting a Tricon chassis, [215](#)
 - download, [342](#)
 - inserting Tricon modules, [224](#)
 - monitoring program execution, [345](#)
 - panel, [24](#)
 - removing modules, [227](#)
 - replacing Tricon MP model, [229](#)
 - tree, [23](#)
 - Tricon chassis power usage, [214](#)
 - Tricon chassis window, [218](#)
 - Tricon configuration steps, [205](#), [251](#)
 - workspace, [23](#)
- controller status
 - monitoring, [425](#)
 - refreshing, [428](#)
 - viewing, [426](#)
- controllers, TriStation, [5](#)
- CR field, [502](#)

- CRC error check, [502](#)

D

- data transfer time
 - calculated for sample program, [493](#)
 - estimating for Peer-to-Peer, [480](#)
 - Peer-to-Peer, [480](#)
 - typical for Peer-to-Peer network, [494](#)
- data types
 - definition, [11](#)
 - elementary, [12](#)
 - generic, [13](#)
- DCS, time synchronization, [245](#)
- DDE
 - client application, [473](#)
 - network redundancy, [471](#)
- DDE address, format, [473](#)
- DDE Server
 - configuring application, [466](#)
 - configuring Triconex host, [467](#)
 - overview, [466](#)
 - view options, [474](#)
- declaration tree, [22](#)
- default gateway, specifying for Trident CM, [330](#)
- Device Clock tagname, [436](#)
- diagnostic events, collecting in log file, [428](#)
- Diagnostic Message Window
 - managing data, [429](#)
 - typical messages, [429](#)
- Diagnostic Monitor software
 - installing and starting, [407](#)
 - main steps for using, [406](#)
- Digital Input modules, [221](#)
- Digital Output modules, [222](#)
- directories
 - specifying, [72](#)
 - tab, [71](#)
 - TriStation, [71](#)
- Disable Remote Changes to Outputs property, [209](#)
- disabling points, [209](#)
- Disabling Stop on Keyswitch property, [208](#)
- display options, of event variables, [454](#)
- DLC protocol, installing on Windows NT PC, [298](#)
- Download All
 - changed application procedure, [355](#)
 - effects, [351](#)
- Download All command, [338](#), [341](#)
- Download Change, effects, [350](#)

Download Change command, 341
 Download Change procedure, 354
 download changes, steps, 349
 downloading, to Tricon or Trident controller, 342
 drawing colors, properties, 75
 dual redundancy, OPC Server, 435

E

editors
 CEM properties, 73
 FBD properties, 78
 LD properties, 80
 EICM, Enhanced Intelligent Communication Module, 223
 emulator
 monitoring program execution, 339
 panel, 24
 using for offline testing, 338
 errors, Modbus data transmission, 518
 Ethernet connection, testing, 332
 Ethernet connection, testing in DDE Server, 471
 event export, automatic, 459
 Event File Directory option, 457
 event retrieval
 getting events, 462
 setting options, 452
 setup, 461
 event variables, assigning to SOE blocks, 444
 events, collecting in log file, 428
 exception conditions, Modbus, 519
 exception responses, Modbus, 520, 521
 execution times for programs, 428
 exporting events, automatic, 459
 expressions, definition, 151
 extensions
 for sequence of events data files, 461
 for shift snapshots, 456
 external faults
 defined, 419
 locating and correcting, 419

F

fatal faults, 420
 fault indicator, 221
 faults
 clearing on all modules, 422
 external, locating and correcting, 419

faults (*continued*)
 internal, locating and correcting, 421
 FBD, writing logic, 94
 FBD language
 about, 14
 adding comments, 115
 editor, 15
 editor properties, 78
 using macros, 66
 field faults
 input points, 419
 output points, 419
 firmware versions, 424
 Force Multiple Coils function, 514
 Force Single Coil function, 510
 forcing points, 353
 function blocks
 control statements, 154
 creating, 52
 custom, writing, 358
 definition, 10
 Peer-to-Peer, 486
 printing, 316
 function blocks for event collection, 447–451
 functions
 control statements, 154
 creating, 52
 definition, 10
 enabling for a matrix, 73
 function tab, 105
 IEC 61131-3 standards, 105
 item properties, 105

G

global variables
 definition, 11
 See also tagnames, 11
 GPS, synchronization with NCMG, 247

H

hardware allocation, exceptions, 348
 hardware monitoring, 412
 HCM, Highway Interface Module, 223
 Help, 26

I

I/A Series DCS, 245
 I/O module changes, 351

- IEC 61131-3 standards, 5
 - benefits, 6
- implementation tree, 22
- indicator behavior, 418
- installing Diagnostic Monitor software, 407
- installing TriStation, xviii
- internal faults
 - defined, 420
 - locating and correcting, 421
- IOP window, 415
 - changing view, 418
- IP address
 - setting default, 324
 - setting Trident with a RARP Server, 325
 - setting with a Tricon EICM, 326
 - setting with a Trident CM, 328
 - setting with a Trident MP, 327
- IP address change, 351
- IP addresses, specifying in SOE Recorder, 458
- IP protocol, installing on TriStation PC, 306
- item properties
 - function, 105
 - function tab, 105
- iteration statements, 155

K

- Keyswitch, 208

L

- language, setting default, 63
- LD
 - coils, 135
 - contacts, 135
 - links, 136
 - power rails, 136
 - writing logic, 134
- LD language
 - about, 15
 - adding comments, 115
 - editor, 15
 - editor properties, 80
 - using macros, 66
- levels, user access, 60
- LF field, 502
- libraries, 82
 - adding, 87
 - creating, 83
 - deleting, 89
 - managing, 86

- libraries, 82 (*continued*)
 - TriStation, 13, 86
 - updating, 88
 - verifying number, 90
- links, 136
- logic, colors for drawings, 75
- logic sheet, 20
- logic sheets, printing, 125
- longitudinal redundancy check. *See* LRC checksum.
- Loop-Back Diagnostic Test function, 513
- LRC checksum, calculation, 502

M

- macros
 - default for annotations, 66
 - with annotations and comments, 66
- Main Processors, writing SOE time-stamps, 448
- maintenance, changing a downloaded application, 349
- major faults, 420
- matrix
 - displaying and sizing cells, 171
 - enabling functions, 176
 - evaluation options, 165
 - planning, 165
 - restrictions and limitations, 165
- Maximum Events per File option, 456
- media converter, with Tricon ACM or NCM, 293
- memory allocation, monitoring and changing, 427
- memory points
 - allocating Tricon, 211
 - allocating, Trident, 256
- message response time, 497
- minor faults, 420
- Modbus Alias Range property, 516
- Modbus communication, noise sources, 518
- Modbus devices, RTU and ASCII modes, 499
- Modbus functions
 - list of names and codes, 499
 - supported by serial ports, 499
- Modbus protocol
 - checksum field, 501
 - exception conditions, 519
 - exception responses, 520, 521
 - function code field, described, 501
 - message format, 501
 - message header, 501
 - message lengths, 504
 - overview, 496

Modbus protocol (*continued*)
 performance considerations, 498
 query and response, 503
 station address field, 501

Modbus, about, 223

module indicator behavior, 418

Module Status screen
 Tricon modules, 414
 Trident modules, 417

modules
 ACM, 223
 Analog Input, 222
 Analog Output, 222
 Digital Input, 221
 Digital Output, 222
 EICM, 223
 HCM, 223
 NCM, 223
 Pulse Input, 222
 Pulse Totalizer Input, 222
 SMM, 223
 specifying Tricon Thermocouple, 232
 Tricon PI, configuring, 230
 Tricon Thermocouple, 222
 Trident PI, 261

Monitor Value on by Default, 65

monitoring
 Peer-to-Peer communication, 489
 program execution on controller, 345
 Triconex response in DDE Server, 474

monitoring program execution, 339

monitoring, value included in annotation, 65

N

NCM, configuring Peer-to-Peer ports, 482

NCM, Network Communication Module, 223

network adapter card, DDE PC, 471

network configuration
 changing, 409
 creating, 409
 defined, 408
 Ethernet, 408
 serial link, 409
 types, 408

Network Nodes tree, 409

network redundancy
 DDE Server, 471
 OPC Server, 435

networks, additional routing, 331

networks, about, 122

NIC card
 installing, 305
 installing in a TriStation PC, 285

node, connecting to network, 411

node number
 changing on ACM or NCM, 291
 Tricon ACM, 287

noise sources, Modbus communication, 518

O

On Max Event option, 456

On Trip Event option, 456

OPC Data Manager (ODM), 437

OPC Redundancy Broker (ORB), 437

OPC Server
 configuration procedure, 433–435
 network redundancy, 435
 overview, 432
 using with multiple controllers, 432

operands, definition, 152

operating parameters
 Tricon, 206
 Trident, 252, 254

operators, definition, 152

order of evaluation, 153

Output Voter Diagnostics. *See* OVD, 423

OVD
 defined, 423
 monitoring modules and points, 423

overview window
 Tricon controller, 413
 Trident controller, 415

P

parity checking, Modbus transmission errors, 518

partitioning, 9

password, default, 18

Password Required for Connection property, 208

passwords, 411

Peer-to-Peer
 configuring Tricon ports, 482
 configuring Trident ports, 483
 data transfer time, 480
 function blocks, 486
 monitoring communication, 489

Peer-to-Peer communication
 overview, 478
 speed restrictions, 478, 488

performance, Modbus functions, 498

points

- allocating memory, Tricon, 211
- allocating memory, Trident, 256
- disabling, 353

poll time, 468

ports

- SMM, 243
- Tricon ACM, 237
- Tricon EICM, 239
- Tricon HIM, 241
- Tricon NCM, 242
- Trident CM, 268, 270
- Trident MP, 266
- Trident MP network, 265

power faults, defined, 419

power rails, 136

power usage, Tricon, 214

Preset Multiple Registers function, 515

Preset Single Register function, 511

print function blocks, purpose, 316

print server, with Trident CM, 317

PRINTER parameter, 316

printing

- configuring for Trident CM, 321
- configuring Tricon EICM port, 315
- connecting to Trident CM, 319
- scan time increases, 313
- with Tricon, 313
- with Trident, 317

privileges, security, 59

Process Safety Time, 347

process tolerance time, in Peer-to-Peer sample program, 493

program execution times, 428

program organization units, 5

programmable logic controller, 4

programs

- compiling, 123
- definition, 8
- determining number, 52
- multiple, 9

project

- adding a description, 61
- language properties, 63
- monitor colors, 69
- open existing, 18
- options, 62
- properties, 62

project (*continued*)

- toolbar, 19
- workspace, 25

project annotation properties, 64

projects

- creating, 54
- default directory, 72

properties

- project, 62
- TriStation, 71

Pulse Input modules, 222

Pulse Totalizer Input modules, 222

R

Read Coil Status function, 506

Read Exception Status function, 512

Read Holding Registers function, 508

Read Input Registers function, 509

Read Input Status function, 507

redundant DDE networks

- configuring with 802.2 protocol, 472
- configuring with TCP/IP protocol, 471
- required hardware, 471

reports, default directory for templates, 72

RTU mode, 499

S

safety and control, about, 50

safety application definition, 50

sample programs, Peer-to-Peer communication, 492

scan cycle, 4

scan surplus

- determining, 346
- negative, 347
- positive, 347

scan time, 335

- affect of print function blocks, 313
- effect on Modbus performance, 498
- monitoring and changing, 427

security

- about, 59
- changing level names, 60

selection statements, 154

sequence-of-events functionality, overview, 440

serial link, 409

sheet template sizes, 66

sheets

- sizes, 100

- sheets (*continued*)
 - template, 100
 - title box, 101
 - Shift Snapshot
 - extension, 456
 - specifying time periods, 456
 - SMM, Safety Manager Module, 223
 - SOE
 - assigning event variables, 444
 - configuration screen, 442
 - trip variable, 445
 - SOE blocks, defining block properties, 442
 - SOE definition file, copying to SoeConfig folder, 454
 - SOE function blocks
 - adding to TriStation project, 447
 - detailed descriptions, 447–451
 - SOECLR, 450
 - SOESTAT, 451
 - SOESTOP, 449
 - SOESTRT, 448
 - SOE functionality, diagrams, 440
 - SOE Recorder
 - installing, 452
 - overview, 440
 - SOECLR function block, 450
 - SOESTAT function block, 451
 - SOESTOP function block, 449
 - SOESTRT function block, 448
 - ST
 - assignment statements, 154
 - CASE statements, 155
 - conditional statements, 154
 - constructs, 151
 - conventions, 150
 - expressions, 151
 - iteration statements, 155
 - operands, 152
 - operators, 152
 - selection statements, 154
 - statements, 153
 - writing logic, 150
 - ST language
 - about, 16
 - editor, 16
 - start function block, 448
 - statements
 - assignment, 154
 - CASE, 155
 - conditional, 154
 - iteration, 155
 - statements, definition, 153
 - station address field, Modbus message, 501
 - status
 - monitoring controller, 425
 - refreshing view, 428
 - update frequency for Ethernet port, 489
 - viewing controller, 426
 - status function block, 451
 - status, Peer-to-Peer communication paths, 489
 - stop function block, 449
 - synchronizing time, Tricon SMM, 249
 - SYS_CM_STATUS function block, 489
 - system attributes, for Ethernet ports on CM, 490
 - system diagnostic events, collecting in log file, 428
- ## T
- tagnames
 - definition, 11
 - disabling, 353
 - forcing, 353
 - TCM, TriStation communication, 278
 - testing
 - Ethernet connection, 332
 - monitoring program execution, 339
 - procedure, 338, 341
 - using controller, 341
 - using emulator, 338
 - thermocouples, about, 222
 - time stamp, written to buffer by SOESTOP function block, 449
 - time synchronization
 - Tricon, 244
 - Trident, 273
 - Trident CM, 274
 - with external source, 245
 - with Tricon ACM, 245
 - with Tricon NCM, 246
 - with Tricon NCMG, 247
 - toolbar, project, 19
 - TR_PEER_STATUS function block, 489
 - TR_PORT_STATUS function block, 489
 - transfer time, Peer-to-Peer, 480
 - transmission errors, Modbus, 518
 - Tricon
 - adding or deleting a chassis, 215
 - allocating hardware, 214
 - allocating memory, 211
 - Allow Disabling of Points, 209
 - chassis window, 218

Tricon (continued)

- configuring communication, [234](#)
- controller overview, [204](#)
- default connection, [235](#)
- Disable Remote Changes to Outputs, [209](#)
- Disabling Stop on Keyswitch, [208](#)
- inserting modules, [224](#)
- main processor modules, [221](#)
- modules, [221](#)
- operating parameters, [206](#)
- operating parameters, setting, [208](#)
- Password Required for Connection, [208](#)
- power usage, [214](#)
- removing modules, [227](#)
- replacing MP model, [229](#)
- selectable modules, [221](#)
- Thermocouple Module, specifying, [232](#)
- time synchronization, [244](#)
- write access, [206](#)

Tricon ACM

- changing node number, [291](#)
- configuring connection, [294](#)
- configuring ports, [237](#)
- connection using media converter, [293](#)
- default IP address, [324](#)
- direct connection to TriStation, [292](#)
- getting IP address using RARP server, [325](#)
- setting node number, [287](#)
- time synchronization, [245](#)
- TriStation connection, [284](#)

Tricon controller

- chassis window, [413](#)
- Module Status screen, [414](#)
- overview window, [413](#)

Tricon EICM

- configuring a printing port, [315](#)
- configuring ports, [239](#)
- configuring TriStation connection, [282](#)
- connecting a printer, [314](#)
- connecting to TriStation, [281](#)
- setting an IP address, [326](#)
- switch settings, [280](#)
- TriStation communication, [279](#)

Tricon HIM, ports, [241](#)**Tricon NCM**

- changing node number, [291](#)
- configuring connection, [294](#)
- connection using media converter, [293](#)
- default IP address, [324](#)
- direct connection to TriStation, [292](#)
- getting IP address using RARP server, [325](#)
- ports, [242](#)
- setting node number, [289](#)

Tricon NCM (continued)

- time synchronization, [246](#)
- TriStation connection, [284](#)

Tricon NCMG, time synchronization, [247](#)**Tricon PI, configuring Tricon, [230](#)****Tricon SMM**

- ports, [243](#)
- synchronizing time, [249](#)

Trident

- allocating memory points, [256](#)
- communication configuration, [262](#)
- controller overview, [250](#)
- default connection, [263](#)
- module properties, [257](#)
- MP attribute properties, [258](#)
- operating parameters, [252](#)
- operating parameters, setting, [254](#)
- printing devices, [317](#)
- system and mode attributes, [275](#)
- time synchronization, [273](#)

Trident CM

- configuring Peer-to-Peer ports, [483](#)
- configuring printing devices, [321](#)
- configuring TriStation connection, [310](#)
- connecting devices using a hub, [320](#)
- connecting printing devices, [319](#)
- direct connection to TriStation, [308](#)
- getting IP address using RARP server, [325](#)
- hub connection to TriStation, [309](#)
- network ports, [268](#)
- routing, [272](#)
- serial ports, [270](#)
- setting an IP address, [328](#)
- setting default IP address, [324](#)
- specifying default gateway, [330](#)
- specifying network routing, [331](#)
- time synchronization, [274](#)

Trident controller

- IOP window, [415](#)
- Module Status screen, [417](#)
- overview window, [415](#)

Trident modules

- inserting, [259](#)
- removing, [260](#)

Trident MP

- address plug, [299](#), [307](#)
- configuring TriStation connection, [302](#)
- direct connection to TriStation, [300](#)
- hub connection to TriStation, [301](#)
- network ports, [265](#)
- node number, [299](#), [307](#)
- serial ports, [266](#)

Trident MP (*continued*)
 setting an IP address, 327

Trident PI, configuring Tricon, 261

Trident, write access, 253

trip variable, assigning, 445

triple modular redundant, 5

TriStation
 communication setup steps, 278
 connection to Tricon ACM or NCM, 292
 connection to Trident MP, 300
 controllers, 5
 Directories tab, 72
 Drawing Color tab, 75
 FBD Editor tab, 78
 hub connection to Trident CM, 309
 installing, xviii
 installing DLC protocol, 298
 LD Editor tab, 80
 projects, 7
 properties, 71
 Trident CM connection, 308
 Trident MP configuration, 302
 Trident MP using a hub, 301
 uninstalling, xviii
 verifying the installation, xix, xx

TriStation 1131
 Developer's Workbench, 3
 installing TCP/IP protocol, 306

TSAA client/server communication
 using DDE Server, 466, 474
 using OPC Server, 432–436

U

uninstalling TriStation, xviii

user access
 about, 56
 adding or modifying, 57
 level names tab, 60
 privileges tab, 59

user name, default, 18

user-defined functions, 176
 CEM, 175
 description, 175

V

Variable Detail Table, 169
 properties, 177

variables
 adding annotations, 346
 Auto Name, 110
 declaring, 110

variables (*continued*)
 definition, 11
 global, 11
 local, 11
 monitoring on controller, 344
 specifying in CEM, 176

verifying a TriStation installation, xix, xx

version number, verifying library version, 90

view options, DDE Server, 474

voter faults, 420

W

workspaces
 about, 19
 application, 21
 controller, 23
 project, 25

workstation redundancy, OPC Server, 435

write access, by DDE client, 467, 473

writing CEM logic, 164

X

XML configuration file
 creating, 409
 defined, 408

Z

zoom, changing in IOP window, 418

