invensys

Triconex®

Triconex General Purpose v2 Systems
Safety Considerations Guide

Assembly Number 9700124-001
November 2010

Document Number 9720124-001, Rev. 0

Printed in the United States of America.

Contents

Preface

This guide provides information about safety concepts and standards that apply to the version 2.x Triconex® General Purpose System.

Throughout the rest of this guide, the Triconex General Purpose System also may be referred to as the Tri-GP.

Summary of Sections

- Chapter 1, Safety Concepts—Describes safety issues, safety standards, and implementation of safety measures.
- Chapter 2, Application Guidelines—Provides information on industry guidelines and recommendations.
- Chapter 3, Fault Management—Discusses fault tolerance and fault detection.
- Chapter 4, Application Development—Discusses methods for developing applications properly to avoid application faults.
- Appendix A, Triconex Peer-to-Peer Communication—Provides examples of using Triconex Peer-to-Peer function blocks to transfer data between applications.
- Appendix C, Safety-Critical Function Blocks—Describes the function blocks intended for use in safety-critical applications and shows their Structured Text code.

Related Documentation

These Invensys books contain related information.

- *Planning and Installation Guide for Triconex General Purpose v2 Systems*
- *Communication Guide for Triconex General Purpose v2 Systems*
- *Developer's Guide for TriStation 1131*
- *TriStation 1131 Libraries Reference*

Abbreviations Used

The TriStation™ 1131 Developer's Workbench is hereafter called TriStation 1131 software.

The following list provides full names for abbreviations of safety terms used in this guide.

BPCS	Basic process control system
ESD	Emergency shutdown
HAZOP	Hazard and operability study
MOC	Management of change
MTBF	Mean time between failure
PES	Programmable electronic system
PFD_{avg}	Average probability of failure to perform design function on demand
PHA	Process hazard analysis
PSM	Process safety management
RMP	Risk management program
RRF	Risk reduction factor
SFF	Safe failure fraction
SIL	Safety integrity level
SIS	Safety-instrumented system
SOV	Solenoid-operated valve
SRS	Safety requirements specification
SV	Safety (relief) valve

Product and Training Information

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

Web Site

http://www.iom.invensys.com

Technical Support

Customers in the U.S. and Canada can obtain technical support from the Invensys Global Customer Support (GCS) center at the numbers below. International customers should contact their regional Triconex support office.

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

Web Site

http://support.ips.invensys.com (registration required)

E-mail

iom.support@invensys.com

We Welcome Your Comments

To help us improve future versions of Triconex documentation, we want to know about any corrections, clarifications, or further information you would find useful. When you contact us, please include the following information:

- The title and version of the guide you are referring to
- A brief description of the content you are referring to (for example, step-by-step instructions that are incorrect, information that requires clarification or more details, missing information that you would find helpful)
- Your suggestions for correcting or improving the documentation
- The version of the Triconex hardware or software you are using
- Your name, company name, job title, phone number and e-mail address

Send e-mail to us at:

triconextechpubs@invensys.com

Please keep in mind that this e-mail address is only for documentation feedback. If you have a technical problem or question, please contact the Invensys Global Customer Support (GCS) center. See Technical Support on page viii for contact information.

Or, you can write to us at:

Attn: Technical Publications - Triconex
Invensys
15345 Barranca Parkway
Irvine, CA 92618

Thank you for your feedback.

1

Safety Concepts

Overview

Modern industrial processes tend to be technically complex, involve substantial energies, and have the potential to inflict serious harm to persons or property during a mishap.

The IEC 61508 standard defines safety as "freedom from unacceptable risk." In other words, absolute safety can never be achieved; risk can only be reduced to an acceptable level.

Safety methods to mitigate harm and reduce risk include:

- Changing the process or mechanical design, including plant or equipment layout
- Increasing the mechanical integrity of equipment
- Improving the basic process control system (BPCS)
- Developing additional or more detailed training procedures for operations and maintenance
- Increasing the testing frequency of critical components
- Using a safety-instrumented system (SIS)
- Installing mitigating equipment to reduce harmful consequences; for example, explosion walls, foams, impoundments, and pressure relief systems

Protection Layers

Methods that provide layers of protection should be:

- Independent
- Verifiable
- Dependable
- Designed for the specific safety risk

This figure shows how layers of protection can be used to reduce unacceptable risk to an acceptable level. The amount of risk reduction for each layer is dependent on the specific nature of the safety risk and the impact of the layer on the risk. Economic analysis should be used to determine the appropriate combination of layers for mitigating safety risks.

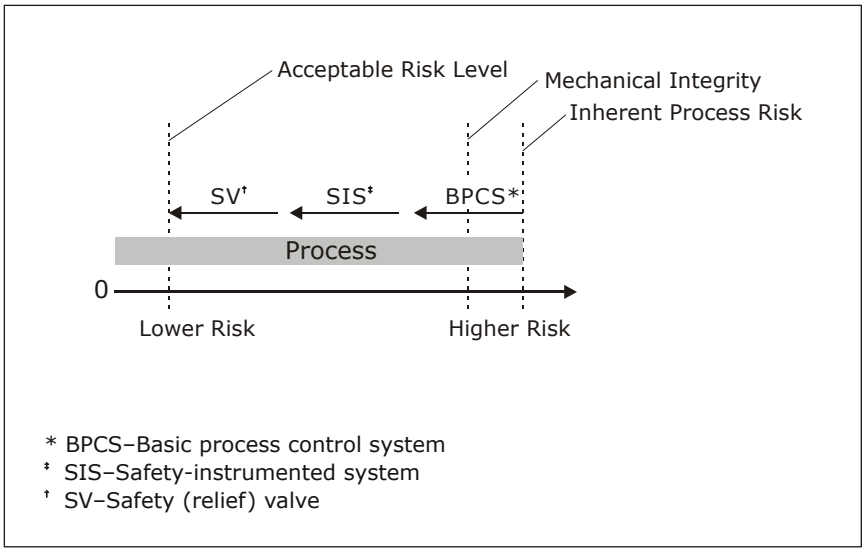


Figure 1 Effect of Protection Layers on Process Risk

When an SIS is required, one of the following should be determined:

- Level of risk reduction assigned to the SIS
- Safety integrity level (SIL) of the SIS

Typically, a determination is made according to the requirements of the ANSI/ISA S84.01 or IEC 61508 standards during a process hazard analysis (PHA).

SIS Factors

According to the ANSI/ISA S84.01 and IEC 61508 standards, the scope of an SIS is restricted to the instrumentation or controls that are responsible for bringing a process to a safe state in the event of a failure. The availability of an SIS is dependent upon:

- Failure rates and modes of components
- Installed instrumentation
- Redundancy
- Voting
- Diagnostic coverage
- Testing frequency

SIL Factors

An SIL can be considered a statistical representation of the availability of an SIS at the time of a process demand. A *process demand* is defined as the occurrence of a process deviation that causes an SIS to transition a process to a safe state.

An SIL is the litmus test of acceptable SIS design and includes the following factors:

- Device integrity
- Diagnostics
- Systematic and common cause failures
- Testing
- Operation
- Maintenance

In modern applications, a programmable electronic system (PES) is used as the core of an SIS. The Tri-GP controller is a state-of-the-art PES optimized for safety-critical applications.

Hazard and Risk Analysis

In the United States, OSHA Process Safety Management (PSM) and EPA Risk Management Program (RMP) regulations dictate that a PHA be used to identify potential hazards in the operation of a chemical process and to determine the protective measures necessary to protect workers, the community, and the environment. The scope of a PHA may range from a very simple screening analysis to a complex hazard and operability study (HAZOP).

A HAZOP is a systematic, methodical examination of a process design that uses a multi-disciplinary team to identify hazards or operability problems that could result in an accident. A HAZOP provides a prioritized basis for the implementation of risk mitigation strategies, such as SISs or ESDs.

If a PHA determines that the mechanical integrity of a process and the process control are insufficient to mitigate the potential hazard, an SIS is required. An SIS consists of the instrumentation or controls that are installed for the purpose of mitigating a hazard or bringing a process to a safe state in the event of a process disruption.

A compliant program incorporates "good engineering practice." This means that the program follows the codes and standards published by such organizations as the American Society of Mechanical Engineers, American Petroleum Institute, American National Standards Institute, National Fire Protection Association, American Society for Testing and Materials, and National Board of Boiler and Pressure Vessel Inspectors. Other countries have similar requirements.

Safety Integrity Levels

This figure shows the relationship of DIN V 19250 classes and SILs (safety integrity levels).

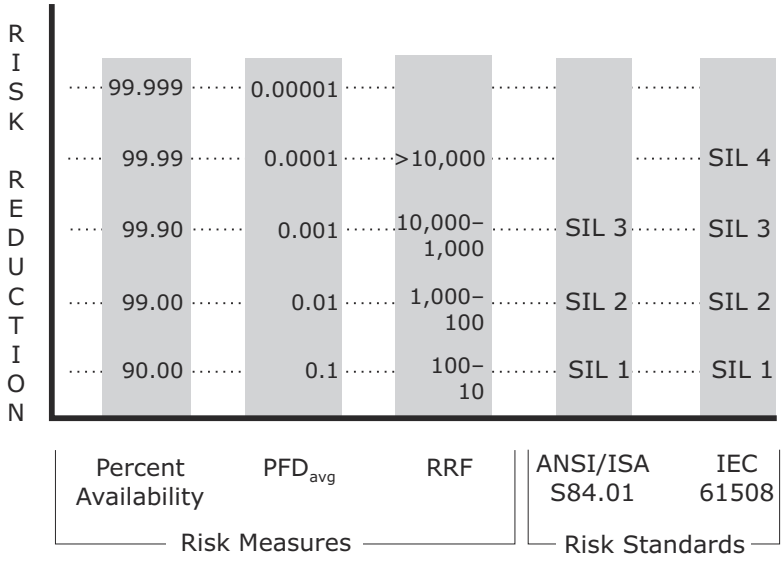


Figure 2 Standards and Risk Measures

As a required SIL increases, SIS integrity increases as measured by:

- System availability (expressed as a percentage)
- Average probability of failure to perform design function on demand (PFD_{avg})
- Risk reduction factor (RRF, reciprocal of PFD_{avg})

Determining a Safety Integrity Level

If a PHA (process hazard analysis) concludes that an SIS is required, ANSI/ISA S84.01 and IEC 61508 require that a target SIL be assigned. The assignment of a SIL is a corporate decision based on risk management and risk tolerance philosophy. Safety regulations require that the assignment of SILs should be carefully performed and thoroughly documented.

Completion of a HAZOP determines the severity and probability of the risks associated with a process. Risk severity is based on a measure of the anticipated impact or consequences.

On-site consequences include:

- Worker injury or death
- Equipment damage

Off-site consequences include:

- Community exposure, including injury and death
- Property damage
- Environmental impact
- Emission of hazardous chemicals
- Contamination of air, soil, and water supplies
- Damage to environmentally sensitive areas

A *risk probability* is an estimate of the likelihood that an expected event will occur. Classified as high, medium, or low, a risk probability is often based on a company's or a competitor's operating experience.

Several methods of converting HAZOP data into SILs are used. Methods range from making a corporate decision on all safety system installations to more complex techniques, such as an IEC 61508 risk graph.

Sample SIL Calculation

As a PES, the Tri-GP controller is designed to minimize its contribution to the SIL, thereby allowing greater flexibility in the SIS design.

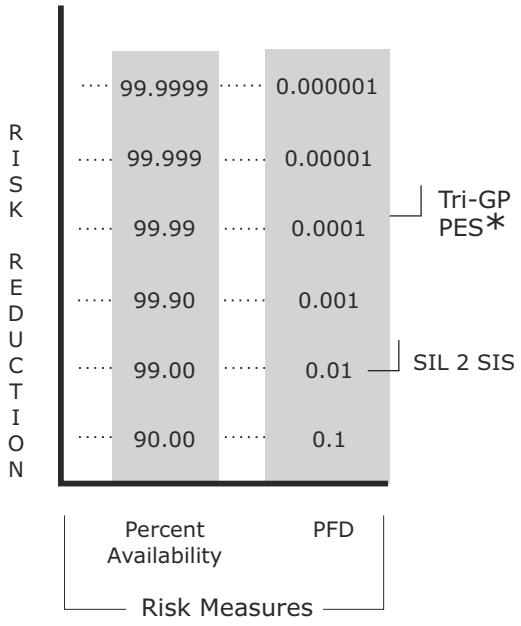


Figure 3 Comparison of Percent Availability and PFD

* Tri-GP controller module failure rates, PFD_{avg}, Spurious Trip Rate, and Safe Failure Fraction (SFF) calculation methods have been independently reviewed by TÜV Rheinland. The numbers presented here (and in the following tables) are typical. Exact numbers should be calculated for each specific system configuration. Contact the Invensys Global Customer Support (GCS) center for details on calculation methods and options related to the Tri-GP controller.

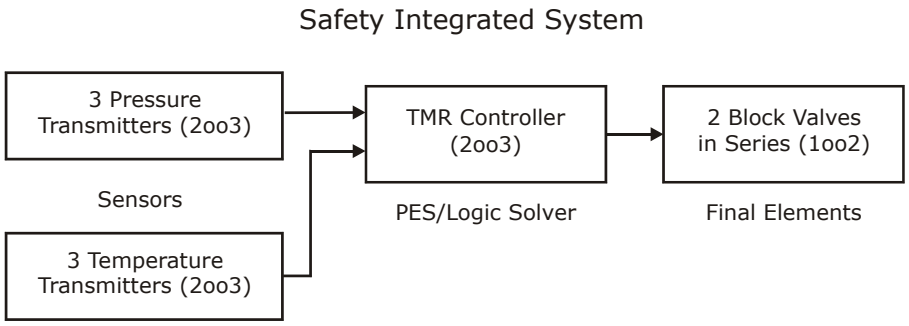


Figure 4 Simplified Diagram of Key Elements

This table provides simplified equations for calculating the PFD_{avg} for the key elements in an SIS. Once the PFD_{avg} for each element is known, an SIL can be determined.

Table 1 Simplified Equations for Calculating PFD_{avg}

	Description	Equation	Variables (Supplied by the Manufacturer)
Sensors	To calculate PFD_{avg} for sensors (2oo3)	$PFD_{avg} = (\lambda^{DU}*TI)^2 + 1/2*\beta*\lambda^{DU}*TI$	λ = failure rate DU=dangerous, undetected failure rate TI= test interval in hours β = common cause factor
Block Valves	To calculate PFD_{avg} for block valves (1oo2) in series (final elements)	$PFD_{avg} = {}^1/_3(\lambda^{DU}*TI)^2 + 1/2*\beta*\lambda^{DU}*TI$	λ = failure rate DU=dangerous, undetected failure rate TI= test interval in hours β = common cause factor
SIF	To calculate PFD_{avg} for a safety instrumented function	SIF PFD_{avg} = Sensors PFD_{avg} + Block Valves PFD_{avg} + Controller PFD_{avg}	

Note Equations are approximate

To determine the SIL, compare the calculated PFD_{avg} to the figure on page 5. In this example, the system is acceptable as an SIS for use in SIL2 applications.

Table 2 Determining the SIL Using the Equations

	β	λDU	TI	PFD	Result
Pressure Transmitters (2oo3)	.03	2.0E-06	13140	1.1E-03	
Temperature Transmitters (2oo3)	.03	2.6E-06	13140	1.7E-03	
Total for Sensors					2.8E-03
Block Valves (1oo2)	.02	2.2E-06	13140	5.7E-04	
Total for Block Valves					0.6E-03
Tri-GP Controller			13140	1.0E-04	0.1E-03
PFD_{avg} for SIF					**3.5E-03**

For additional information on SIL assignment and SIL verification, visit the Premier Consulting Services web site at www.premier-fs.com.

Safety Life Cycle Model

The necessary steps for designing an SIS from conception through decommissioning are described in the *safety life cycle*.

Before the safety life cycle model is implemented, the following requirements should be met:

- Complete a hazard and operability study
- Determine the SIS requirement
- Determine the target SIL

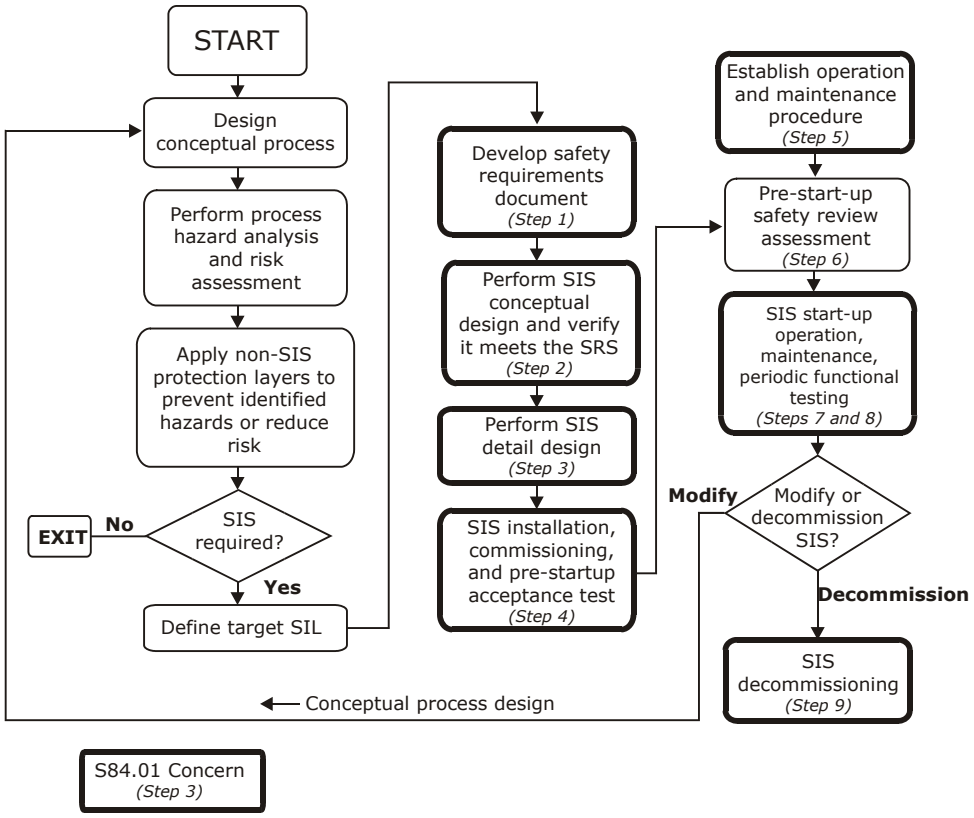


Figure 5 Safety Life Cycle Model

Developing an SIS Using the Safety Life Cycle

1 Develop a safety requirement specification (SRS).

An SRS consists of *safety functional requirements* and *safety integrity requirements*. An SRS can be a collection of documents or information.

Safety functional requirements specify the logic and actions to be performed by an SIS and the process conditions under which actions are initiated. These requirements include such items as consideration for manual shutdown, loss of energy source, etc.

Safety integrity requirements specify a SIL and the performance required for executing SIS functions. Safety integrity requirements include:

- Required SIL for each safety function
- Requirements for diagnostics
- Requirements for maintenance and testing
- Reliability requirements if the spurious trips are hazardous

2 Develop the conceptual design, making sure to:

- Define the SIS architecture to ensure the SIL is met (for example, voting 1oo1, 1oo2, 2oo2, 2oo3).
- Define the logic solver to meet the highest SIL (if different SIL levels are required in a single logic solver).
- Select a functional test interval to achieve the SIL.
- Verify the conceptual design against the SRS.

3 Develop a detailed SIS design including:

- General requirements
- SIS logic solver
- Field devices
- Interfaces
- Energy sources
- System environment
- Application logic requirements
- Maintenance or testing requirements

Some key ANSI/ISA S84.01 requirements are:

- The logic solver shall be separated from the basic process control system (BPCS).
- Sensors for the SIS shall be separated from the sensors for the BPCS.
- The logic system vendor shall provide MTBF data and the covert failure listing, including the frequency of occurrence of identified covert failures.

Note Triconex controllers do not contain undiagnosed dangerous faults that are statistically significant.

- Each individual field device shall have its own dedicated wiring to the system I/O. **Using a field bus is not allowed!**
- The operator interface may not be allowed to change the SIS application software.
- Maintenance overrides shall not be used as a part of application software or operating procedures.
- When online testing is required, test facilities shall be an integral part of the SIS design.

4 Develop a pre-start-up acceptance test procedure that provides a fully functional test of the SIS to verify conformance with the SRS.

5 Before startup, establish operational and maintenance procedures to ensure that the SIS functions comply with the SRS throughout the SIS operational life, including:

- Training
- Documentation
- Operating procedures
- Maintenance program
- Testing and preventive maintenance
- Functional testing
- Documentation of functional testing

6 Before start-up, complete a safety review.

7 Define procedures for the following:

- Start-up
- Operations
- Maintenance, including administrative controls and written procedures that ensure safety if a process is hazardous while an SIS function is being bypassed
- Training that complies with national regulations (such as OSHA 29 CFR 1910.119)
- Functional testing to detect covert faults that prevent the SIS from operating according to the SRS
- SIS testing, including sensors, logic solver, and final elements (such as shutdown valves, motors, etc.)

8 Follow management of change (MOC) procedures to ensure that no unauthorized changes are made to an application, as mandated by OSHA 29 CFR 1910.119.

9 Decommission an SIS before its permanent retirement from active service, to ensure proper review.

Safety Standards

Over the past several years, there has been rapid movement in many countries to develop standards and regulations to minimize the impact of industrial accidents on citizens. The standards described in this section apply to typical applications.

General Safety Standards

IEC 61508, Parts 1-7

The IEC 61508 standard, "Functional Safety: Safety Related Systems," is an international standard designed to address a complete SIS for the process, transit, and medical industries. The standard introduces the concept of a safety life cycle model (see Figure 5 on page 9) to illustrate that the integrity of an SIS is not limited to device integrity, but is also a function of design, operation, testing, and maintenance.

The standard includes four SILs that are indexed to a specific probability-to-fail-on-demand (PFD) (see Figure 2 on page 5). A SIL assignment is based on the required risk reduction as determined by a PHA.

ANSI/ISA S84.01

ANSI/ISA S84.01-1996 is the United States standard for safety systems in the process industry. The SIL classes from IEC 61508 are used and the DIN V 19250 relationships are maintained. ANSI/ISA S84.01-1996 does not include the highest SIL class, SIL 4. The S84 Committee determined that SIL 4 is applicable for medical and transit systems in which the only layer of protection is the safety-instrumented layer. In contrast, the process industry can integrate many layers of protection in the process design. The overall risk reduction from these layers of protection is equal to or greater than that of other industries.

IEC 61511, Parts 1-3

The IEC 61511 standard, "Functional Safety: Safety Instrumented Systems for the Process Industry Sector," is an international standard designed to be used as a companion to IEC 61508. IEC 61511 is intended for SIS designers, integrators, and users in the process-control industry.

Application-Specific Standards

NFPA 85

NFPA 85, "Boiler and Combustion Systems Hazards Code," outlines the United States requirements for operations using single burner boilers and multiple burner boilers.

CAN/CSA-C22.2 No. 61010-1-04

CAN/CSA-C22.2 No. 61010-1-04, "Safety Requirements for Electrical Equipment for Measurement, Control, and Laboratory Use, Part 1: General Requirements," outlines the Canadian requirements for burner management applications.

2

Application Guidelines

Overview

This chapter provides information about the industry-standard guidelines applicable to safety applications. These guidelines include those that apply to all safety systems, as well as those that apply only to specific industries, such as burner management or fire and gas systems.

Guidelines that apply specifically to the Tri-GP controller are also provided. Project change control guidelines and maintenance override considerations can be found at the end of this chapter.

Be sure to thoroughly read and understand these guidelines ***before*** you write your safety application and procedures.

TÜV Rheinland Certification

TÜV Rheinland Industrie Service GmbH has certified that specific versions of Tri-GP systems meet the requirements of IEC 61508 SIL2 when used as a PES in an SIS. For the approved Tri-GP system versions, see the "List of Type Approved Programmable Logic Controllers (PES)" on the TÜV website at http://www.tuv-fs.com/plctcnx.htm. This list is published by Invensys Systems Inc. and TÜV Rheinland Industrie Service GmbH.

Tristation 1131 software has been reviewed and evaluated as part of the functional safety assessment and certification of Triconex controllers according to IEC 61508. Based on the review, and evaluation during certification, TÜV Rheinland Industrie Service GmbH deems the TriStation 1131 software suitable as a development and deployment tool for SIL3 safety and critical control applications as defined by IEC 61508 and IEC 61511, when it is used in accordance with Triconex user documentation, which includes the Safety Considerations Guide.

If the IEC 61508 standard applies to your application, compliance with the guidelines described in this chapter is highly recommended.

General Guidelines

This section describes standard industry guidelines that apply to:

- All safety systems
- Emergency shutdown (ESD) systems
- Burner management systems
- Fire and gas systems

All Safety Systems

These general guidelines apply to all user-written safety applications and procedures:

- A design-change review, code-change review, and functional testing are recommended to verify the correct design and operation.
- After a safety system is commissioned, no changes to the system software (operating system, I/O drivers, diagnostics, etc.) are allowed without type approval and re-commissioning. Any changes to the application or the control application should be made under strict change-control procedures. For more information on change-control procedures, see Project Change and Control on page 23. All changes should be thoroughly reviewed, audited, and approved by a safety change control committee or group. After an approved change is made, it should be archived.
- In addition to printed documentation of the application, two copies of the application should be archived on an electronic medium that is write-protected to avoid accidental changes.
- Under certain conditions, a PES may be run in a mode that allows an external computer or operator station to write to system attributes. This is normally done by means of a communication link. The following guidelines apply to writes of this type:
  - The communication link should use Modbus or other approved protocols with CRC checks.
  - The communication link should not be allowed to write directly to output points.
  - The application must check the value (of each variable written) for a valid range or limit before its use.
  - For information on the potential impacts of writes to safety-related variables that result in disabling diagnostics such as Output Voter Diagnostics, see Module Diagnostics on page 36.
- PID and other control algorithms should not be used for safety-related functions. Each control function should be checked to verify that it does not provide a safety-related function.
- Pointers should not be used for safety-related functions. For TriStation 1131 applications, this includes the use of VAR_IN_OUT variables.
- An SIS PES should be wired and grounded according to the procedures defined by the manufacturer.

Emergency Shutdown Systems

The safe state of the plant should be a de-energized or low (0) state.

All power supplies should be monitored for proper operation.

Burner Management Systems

The safe state of the plant is a de-energized or low (0) state.

When a safety system is required to conform to the EN 50156 standard for electrical equipment for furnaces, PES throughput time should ensure that a safe shutdown can be performed within one second after a problem in the process is detected.

Fire and Gas Systems

Fire and gas applications should operate continuously to provide protection. The following industry guidelines apply:

- If inputs and outputs are energized to mitigate a problem, a PES system should detect and alarm open and short circuits in the wiring between the PES and the field devices.
- An entire PES system should have redundant power supplies. Also, the power supplies that are required to activate critical outputs and read safety-critical inputs should be redundant. All power supplies should be monitored for proper operation.
- De-energized outputs may be used for normal operation. To initiate action to mitigate a problem, the outputs are energized. This type of system shall monitor the critical output circuits to ensure that they are properly connected to the end devices.

Guidelines for Triconex Controllers

This section provides information about industry guidelines that are specific to Triconex controllers when used as a PES in an SIS:

Safety-Critical Modules

It is recommended that only the following modules be used for safety-critical applications:

- Main Processor Module
- Communication Module (only when using protocols defined for safety-critical applications)
- Analog Input Module
- Analog Input/Digital Input Module
- Analog Output Modules
- Digital Input Modules
- Digital Output Modules
- Pulse Input Module

The Solid-State Relay Output Module is recommended for non-safety-critical points only.

Safety-Shutdown

A safety application should include a network that initiates a safe shutdown of the process being controlled when a controller operates in a degraded mode for a specified maximum time.

The Triconex Library provides two function blocks to simplify programming a safety-shutdown application: SYS_SHUTDOWN and SYS_CRITICAL_IO. To see the Structured Text code for these function blocks, see Appendix C, Safety-Critical Function Blocks. For more information on safety-shutdown networks, see Sample Safety-Shutdown Programs on page 49.

Response Time and Scan Time

Scan time must be set below 50 percent of the required response time. If scan time is greater than 50 percent, an alarm should be available.

Disabled Points Alarm

A project should not contain disabled points unless there is a specific reason for disabling them, such as initial testing. An alarm should be available to alert the operator that a point is disabled.

Disabled Output Voter Diagnostic

For safety programs, disabling the Output Voter Diagnostics is not recommended; however, if it is required due to process interference concerns, it can be done if, and only if, the DO is proof tested every three to six months.

Download All at Completion of Project

When development and testing of a safety application is completed, use the Download All command on the Controller Panel to completely re-load the application to the controller.

Modbus Master Functions

Modbus Master functions are designed for use with non-critical I/O points only. These functions should not be used for safety-critical I/O points or for transferring safety-critical data using the MBREAD and MBWRITE functions.

Triconex Peer-to-Peer Communication

Triconex Peer-to-Peer communication enables Triconex controllers (also referred to as *nodes*) to send and receive information. You should use a redundant Peer-to-Peer network for safety-critical data. If a node sends critical data to another node that makes safety-related decisions, you must ensure that the application on the receiving node can determine whether it has received new data.

If new data is not received within the time-out period (equal to half of the process-tolerance time), the application on the receiving node should be able to determine the action to take. The specific actions depend on the unique safety requirements of your process. The following sections summarize actions typically required by Peer-to-Peer send and receive functions.

Note Due to a lack of information on the reliability and safety of switched or public networks, Invensys recommends that switched or public networks not be used for safety-critical Peer-to-Peer communication between Triconex controllers.

Sending Node

Actions typically required in the logic of the sending application are:

- The sending node must set the SENDFLG parameter in the send call to true (1) so that the sending node sends new data as soon as the acknowledgment for the last data is received from the receiving node.
- The SEND function block (TR_USEND) must include a diagnostic integer variable that is incremented with each new send initiation so that the receiving node can check this variable for changes every time it receives new data. This new variable should have a range of 1 to 65,535 where the value 1 is sent with the first sample of data. When this variable reaches the limit of 65,535, the sending node should set this variable back to 1 for the next data transfer. This diagnostic variable is required because the communication path is not triplicated like the I/O system.
- The number of SEND functions in an application must be less than or equal to five because the controller only initiates five SEND functions per scan. To send data as fast as possible, the SEND function must be initiated as soon as the acknowledgment for the last data is received from the receiving node.
- The sending application must monitor the status of the RECEIVE (TR_URCV) and TR_PORT_STATUS functions to determine whether there is a network problem that requires operator intervention.

Receiving Node

Actions typically required in the logic of the receiving application are:

- To transfer safety-critical data, the basic rule is that the receiving node must receive at least one sample of new data within the maximum time-out limit. If this does not happen, the application for the receiving node must take one or more of the following actions, depending on requirements:
  - Use the last data received for safety-related decisions.
  - Use default values for safety-related decisions in the application.
  - Check the status of the TR_URCV and TR_PORT_STATUS functions to see whether there is a network problem that requires operator intervention.
- The receiving node must monitor the diagnostic integer variable every time it receives new data to determine whether this variable has changed from last time.

- The receiving program must monitor the status of the TR_URCV and TR_PORT_STATUS functions to determine if there is a network problem that requires operator intervention.

For information on data transfer time and examples of how to use Peer-to-Peer functions to transfer safety-critical data, see Appendix A, Triconex Peer-to-Peer Communication.

SIL2 Guidelines

For SIL2 applications, these guidelines should be followed:

- If non-approved modules are used, the inputs and outputs should be checked to verify that they do not affect safety-critical functions of the controller.
- Two modes control write operations from external hosts:
  - **Remote Mode**: When true, external hosts, such as Modbus master, DCS, etc., can write to aliased variables in the controller. When false, writes are prohibited.
  - **Program Mode**: When true, TriStation 1131 software can make changes including operations that modify the behavior of the currently running application. For example, Download All, Download Change, declaring variables, enabling/disabling variables, changing values of variables and scan time, etc.

  Remote mode and program mode are independent of each other. In safety applications, operation in these modes is not recommended. In other words, write operations to the controller from external hosts should be prohibited. If remote mode or program mode becomes true, the application should include the following safeguards:
  - When remote mode is true, the application should turn on an alarm. For example, if using the SYS_SHUTDOWN function block, the ALARM_REMOTE_ACCESS output could be used. Verify that aliased variables adhere to the guidelines described in Maintenance Overrides on page 24.
  - When program mode is true, the application should turn on an alarm. For example, if using the SYS_SHUTDOWN function block, the ALARM_PROGRAMMING_PERMITTED output could be used.
- Wiring and grounding procedures outlined in the *Planning and Installation Guide for Triconex General Purpose v2 Systems* should be followed.
- Maintenance instructions outlined in the *Planning and Installation Guide for Triconex General Purpose v2 Systems* should be followed.
- The operating time restrictions in this table should be followed.

Operating Mode	SIL 1 Operating Time	SIL 2 Operating Time
TMR Mode	Continuous	Continuous
Dual Mode	Continuous	Continuous
Single Mode	Continuous	Industry accepted MTTR

- Peer-to-Peer communication must be programmed according to the recommendations in Triconex Peer-to-Peer Communication on page 20.

Note All Triconex logic solver faults can be repaired online without further degradation of the system and should be performed before a second fault occurrence to maintain the highest availability of the system. The highly effective means of modular insertion and replacement of faulted Triconex components is transparent to the operation of the system and the ease of replacement mitigates the risk of systematic and human induced failure as defined by IEC 61508. It is highly recommended that a faulted component be replaced within industry accepted Mean-Time-To-Repair (MTTR) periods.

Additional Fire and Gas Guidelines

- Analog input cards with current loop terminations should be used to read digital inputs. Opens and shorts in the wiring to the field devices should be detectable. The Triconex library function LINEMNTR should be used to simplify application development.
- A controller should be powered by two independent sources.
- If controller operation is degraded to dual mode or single mode, repairs should be timely. The operating time restrictions in the table on page 22 should be followed.

Periodic Offline Test Interval Guidelines

A safety instrumented function (SIF) may be tested periodically to satisfy the requirements for the specified safety integrity level (SIL). This period is called the periodic offline test interval.

Project Change and Control

A change to a project, however minor, should comply with the guidelines of your organization's Safety Change Control Committee (SCCC).

Change Procedure

1. Generate a change request defining all changes and the reasons for the changes, then obtain approval for the changes from the SCCC.
2. Develop a specification for the changes, including a test specification, then obtain approval for the specification from the SCCC.
3. Make the appropriate changes to the project, including those related to design, operation, or maintenance documentation.
4. To verify that the configuration in the controller matches the last downloaded configuration, use the Verify Last Download to the Controller command on the Controller Panel. For details, see the *TriStation 1131 Developer's Guide*.
5. Compare the configuration in your project with the configuration that was last downloaded to the controller by printing the Compare Project to Last Download report from the Controller Panel. For details, see the *TriStation 1131 Developer's Guide.*

6. Print all logic elements and verify that the changes to networks within each element do not affect other sections of the application.
7. Test the changes according to the test specification using the Emulator Panel. For details, see the *TriStation 1131 Developer's Guide*.
8. Write a test report.
9. Review and audit all changes and test results with the SCCC.
10. When approved by the SCCC, download the changes to the controller.
    - You may make minor changes online only if the changes are absolutely necessary and are tested thoroughly.
    - To enable a Download Change command, select the Enable Programming and Control option in the Set Programming Mode dialog box on the Controller Panel if it is not already selected.

Note Changing the operating mode to PROGRAM generates an alarm to remind the operator to return the operating mode to RUN as soon as possible after the Download Change. For more information, see Programming Permitted Alarm on page 59.

11. Save the downloaded project in the TriStation 1131 software and back up the project.
12. Archive two copies of the project file and all associated documentation.

Maintenance Overrides

Three methods can be used to check safety-critical devices connected to controllers:

- Special switches are connected to the inputs on a controller. These inputs deactivate the actuators and sensors undergoing maintenance. The maintenance condition is handled in the logic of the control application.
- Sensors and actuators are electrically disconnected from a controller and manually checked using special measures.
- Communication to a controller activates the maintenance override condition. This method is useful when space is limited; the maintenance console should be integrated with the operator display.

TÜV recommends that the TriStation 1131 workstation used for programming is not also used for maintenance.

Using Triconex Communication Capabilities

For maintenance overrides, two options for connection are available:

- DCS (distributed control system) connection using an approved protocol.
- TriStation 1131 PC connection, which requires additional, industry-standard safety measures in a controller to prevent downloading a program change during maintenance intervals. For more information, see Alarm Usage on page 59.

Table 3 describes the design requirements for handling maintenance overrides when using Triconex communication capabilities.

Table 3 Design Requirements for Maintenance Override Handling

Design Requirements	Responsible Person	
	DCS	TriStation 1131 Software
Control program logic and the controller configuration determine whether the desired signal can be overridden.	Project Engineer, Commissioner	Project Engineer, Commissioner
Control program logic and/or system configuration specify whether simultaneous overriding in independent parts of the application is acceptable.	Project Engineer	Project Engineer, Type Approval
Controller activates the override. The operator should confirm the override condition.	Operator, Maintenance Engineer	Maintenance Engineer, Type Approval
Direct overrides on inputs and outputs are not allowed, but should be checked and implemented *in relation to the application*. Multiple overrides in a controller are allowed as long as only one override applies to each safety-critical group. The controller alarm should not be overridden.	Project Engineer	Project Engineer, Type Approval
DCS warns the operator about an override condition. The operator continues to receive warnings until the override is removed.	Project Engineer, Commissioner	N/A
A second way to remove the maintenance override condition should be available.	Project Engineer	
If urgent, a maintenance engineer may remove the override using a hard-wired switch.		Maintenance Engineer, Type Approval
During an override, proper operating measures should be implemented. The time span for overriding should be limited to one shift (typically no longer than eight hours). A maintenance override switch (MOS) light on the operator console should be provided (one per controller or process unit).	Project Engineer, Commissioner, DCS, TriStation 1131 software	

Table 4 describes the operating requirements for handling maintenance overrides when using Triconex communication capabilities.

Table 4 Operating Requirements for Maintenance Override Handling

Operating Requirements	Responsible Person	
	DCS	TriStation 1131 Software
Maintenance overrides are enabled for an entire controller or for a subsystem (process unit).	Operator, Maintenance Engineer	Maintenance Engineer, Type Approval
Controller activates an override. The operator should confirm the override condition.	Operator, Maintenance Engineer	Maintenance Engineer, Type Approval
Controller removes an override.	Operator, Maintenance Engineer	Maintenance Engineer

Additional Recommendations

These procedures are recommended in addition to the recommendations described in the tables on page 25 and page 26:

- A DCS program should regularly verify that no discrepancies exist between the override command signals issued by a DCS and override-activated signals received by a DCS from a PES. This figure shows the procedure:

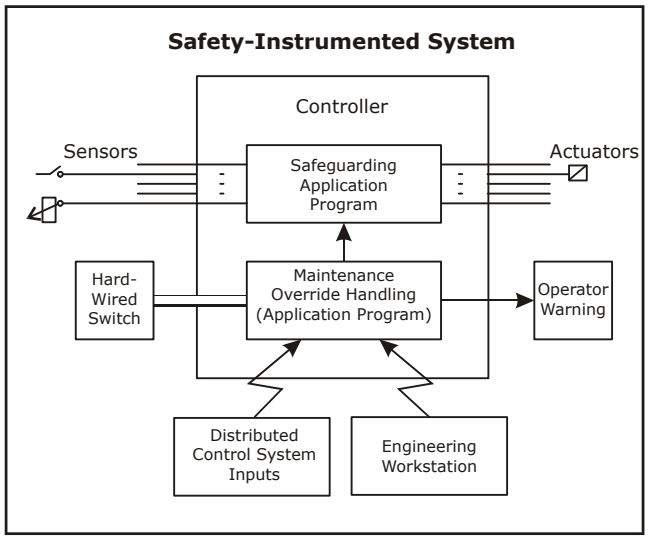


Figure 6 PES Block Diagram

- Use of the maintenance override capability should be documented in a DCS or TriStation 1131 log. The documentation should include:
  - Begin- and end-time stamps of the maintenance override.
  - Identification of the maintenance engineer or operator who activates a maintenance override. If the information cannot be printed, it should be entered in a work-permit or maintenance log.
  - Tag name of the signal being overridden.
  - Communication packages that are different from a type-approved Modbus should include CRC, address check, and check of the communication time frame.
  - Loss of communication should lead to a warning to the operator and maintenance engineer. After loss of communication, a time-delayed removal of the override should occur after a warning to the operator.
- For more information about maintenance override operation, please see the TÜV web site at http://www.tuv-fs.com/m_o202.pdf.

Safety Controller Boundary

The boundary of the safety controller includes the External Termination Panels (ETPs) and interconnecting cables. Triconex safety controllers must be used with approved ETPs and cables only. The use of unapproved, unauthorized cables and/or ETPs compromises the TÜV safety certification and potentially the ability of the logic solver to respond to safety demands. False trips resulting from the use of unapproved components can cause end-user economic loss.

CAUTION

When using fanned-out interface cables or third-party ETPs—such as those from P&F or MTL—please consult the Invensys Global Customer Support (GCS) center for the safety-boundary impact of using such cables or ETPs.

Background

IEC 61508 and IEC 61511 define a programmable electronic Safety Instrumented System (SIS) as consisting of sensors, logic solvers, and final control elements, as shown in this figure.

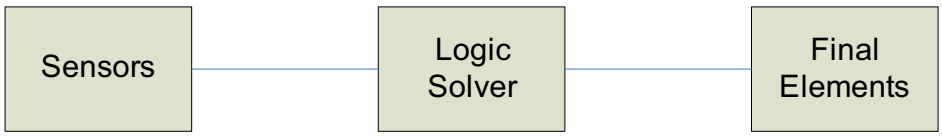


Figure 7 Simplified SIS

Together, these elements implement Safety Instrumented Functions (SIF) of the target Safety Integrity Level (SIL). In order to implement a safety-certified SIF, the system designer must choose safety-certified loop elements, including sensors, final elements, logic solvers, and other interconnecting components.

In addition to the components shown in Figure 7, a typical SIS consists of components such as cables and external termination panels. These components are used to connect the sensors and final elements to the logic solvers. Figure 8 shows the SIS including these components.

Approved ETPs and interconnecting cables are listed in the *Planning and Installation Guide for Triconex General Purpose v2 Systems* and the *Technical Product Guide for Trident GP v2 Systems*, which are available on the Invensys Global Customer Support (GCS) website.

Design Control, Configuration Management, Supply Chain Management, and Quality Assurance for Triconex ETPs and cable assemblies are controlled by Invensys in Irvine, California (the Triconex factory). Sourcing of approved ETPs and interconnecting cables is also controlled by Invensys in Irvine.

Certifications

- TÜV approves the use of Triconex ETPs and interconnecting cables with Triconex Safety Logic Solvers.
- TÜV certifies the use of Triconex Safety Logic Solvers in SIL2 applications with the TÜV approved ETPs and interconnecting cables.
- Triconex ETPs are certified for electrical safety in full compliance with international standards by CSA. They are qualified for general use in North America and other jurisdictions requiring compliance with these standards, as well as the European CE mark as per the Low Voltage Directive.
- Triconex ETPs and interconnecting cables comply with the applicable IEC EMC standard (IEC 61326-3-1,2,), which includes the European CE mark per the EMC directive.
- Triconex ETPs that are approved for hazardous locations also comply with North America Class1 Div2 (C1D2) and Zone 2 as per the European ATEX directive.

Thus, the boundary of the safety controller (Triconex Safety Logic Solver) extends up to the ETPs, including the interconnecting cables, as shown in Figure 8 Safety Controller Boundary (page 29).

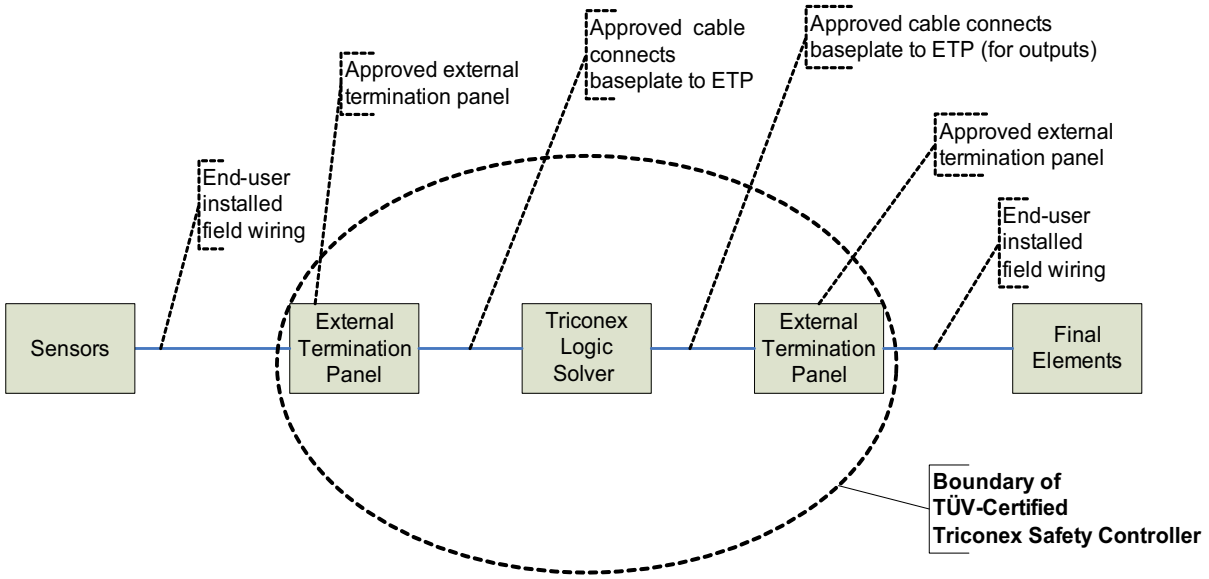


Figure 8 Safety Controller Boundary

Use of Unapproved Components

The use of unapproved cables and unapproved ETPs can negatively impact the safety integrity of the safety function and the compliance with the applicable safety standards. This causes a liability issue in the event of a plant incident.

The use of such unapproved components can also impact the availability of the safety system by causing false trips in the plant. This results in unnecessary economic loss for the plant.

3

Fault Management

Overview

The Tri-GP controller has been designed from its inception with self-diagnostics as a primary feature. Triple-Modular Redundant (TMR) architecture (shown in Figure 9) ensures fault tolerance and provides error-free, uninterrupted control in the event of hard failures of components or transient faults from internal or external sources.

Each I/O module houses the circuitry for three independent channels. Each channel on the input modules reads the process data and passes that information to its respective main processor. The three Main Processor (MP) modules communicate with each other using a proprietary, high-speed bus system called the TriBus.

Extensive diagnostics on each channel, module, and functional circuit quickly detect and report operational faults by means of indicators or alarms. This fault information is available to an application. It is critical that an application properly manage fault information to avoid an unnecessary shutdown of a process or plant.

This section discusses the methods for properly handling faults.

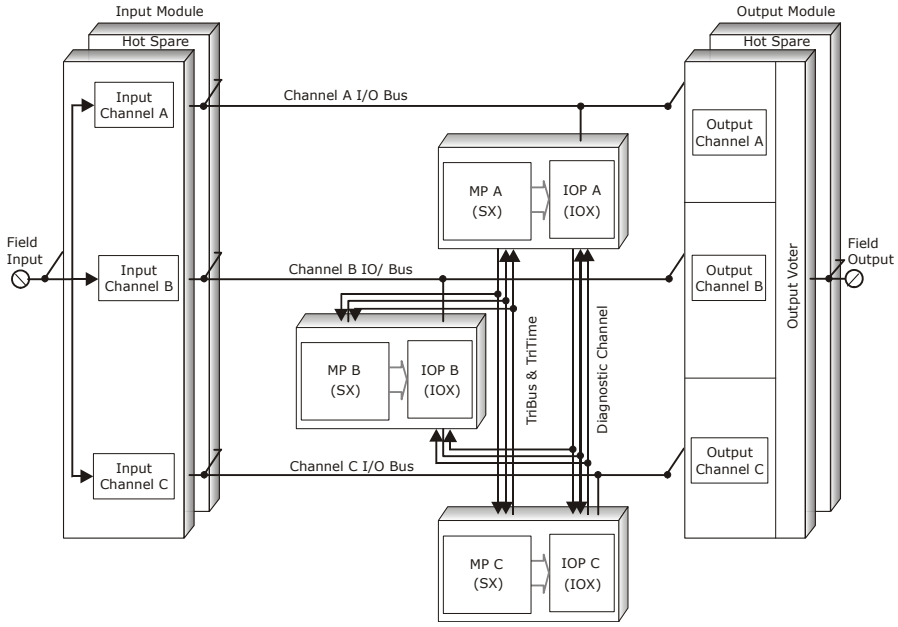


Figure 9 Typical Triconex Controller

System Diagnostics

To improve system availability and safety, a safety system must be able to detect failures and provide the means for managing failures properly. The controller's diagnostics may be categorized as:

- **Reference diagnostics**: Comparing an operating value to a predetermined reference, such as a system specification.
- **Comparison diagnostics**: Comparing one component to another, such as one independent channel with two other independent channels.
- **Field device diagnostics**: Diagnostics are extended to a system's field devices and wiring.

Types of Faults

A controller is subject to both external faults and internal faults, which are reported by:

- The status indicators on a module's front panels
- The Triconex Enhanced Diagnostic Monitor
- System attributes on the Controller Panel in the TriStation 1131 software

External Faults

A controller may experience the following types of external faults:

- Logic power faults
- Field power faults
- Load or fuse faults

When an external fault occurs, the controller illuminates the yellow indicator on the faulting I/O module and the Field Power or Logic Power alarm indicators on the Main Processors and the System Alarm. The Triconex Enhanced Diagnostic Monitor identifies the faulting module by displaying a red frame around it. When these conditions occur, the faulting module's power supplies and wiring should be examined.

Internal Faults

Internal faults are usually isolated to one of the controller's three channels (A, B, or C). When an internal fault occurs on one of the three channels, the remaining two healthy channels maintain full control. Depending on the type of fault, the controller either remains in TMR mode or degrades to dual mode for the system component that is affected by the fault. For more information about operating modes, see Operating Modes on page 35.

When an internal fault occurs, the controller illuminates the red Fault indicator on the faulting I/O module and the System alarm on the Main Processors to alert the operator to replace the faulting module.

Operating Modes

Each input or output point is considered to operate in one of four modes:

- Triple Modular Redundant
- Dual mode
- Single mode
- Zero mode

The current mode indicates the number of channels controlling a point; in other words, controlling the output or having confidence in the input. For safety reasons, *system mode* is defined as the mode of the point controlled by the least number of channels.

System variables summarize the status of input and output points. When a safety-critical point is in zero mode, the application should shut down the controlled process.

You can further simplify and customize shutdown logic by using special function blocks provided by Triconex. By considering only faults in safety-critical modules, system availability can be improved. Using shutdown function blocks is essential to preventing potential false trips in dual mode and to guaranteeing fail-safe operation in single mode. For more information, see Appendix C, Safety-Critical Function Blocks.

A *safety-critical fault* is defined as a fault that prevents the system from executing the safety function on demand. Safety-critical faults include:

- Inability to detect a change of state on a digital input point

  The controller's diagnostics verify the ability to detect changes of state independently for each channel, typically every 500 milliseconds. For more information on fault reporting time, see Calculation for Diagnostic Fault Reporting Time on page 39.

- Inability to detect a change of value on an analog input point

  The controller's diagnostics verify the ability to detect changes of value independently for each channel, typically every 500 milliseconds. For more information on fault reporting time, see Calculation for Diagnostic Fault Reporting Time on page 39.

- Inability to change the state of a digital output point

  The controller's diagnostics verify the ability to control the state of each output point.

- Inability of the system to:
  - Read each input point
  - Vote the correct value of each input
  - Execute the control application
  - Determine the state of each output point correctly

  The controller's diagnostics verify the correct operation of all data paths between the I/O modules and the MPs for each channel independently, typically every 500 milliseconds. For more information on fault reporting time, see Calculation for Diagnostic Fault Reporting Time on page 39.

Also, during each execution of the control application, each channel independently verifies the:

- Integrity of the data path between the MPs
- Proper voting of all input values
- Proper evaluation of the control application
- Calculated value of each output point

Module Diagnostics

Each system component detects and reports operational faults.

Analog Input (AI) Modules

Analog input module points useforce-to-value diagnostics (FVD). Under system control, each point is sequentially forced to a test value. The forced value is maintained until the value is detected by the system or a time-out occurs. Using the integral FVD capability, each point can be independently verified for its ability to accurately detect a transition to a different value, typically every 500 milliseconds. (For more information on fault reporting time, see Calculation for Diagnostic Fault Reporting Time on page 39.) Using these diagnostics, each channel can be verified independently, thus assuring near 100 percent fault coverage and fail-safe operation under all single-fault scenarios, and most common multiple-fault scenarios.

Analog Input Module Alarms

Analog input module faults are reported to the control application. These alarms can be used to increase availability during specific multiple-fault conditions. Loss of field power or logic power is reported to the control application.

Analog Input/Digital Input (AI/DI) Modules

Analog input/digital input module points useforce-to-value diagnostics (FVD). Under system control, each point is sequentially forced to a test value. The forced value is maintained until the value is detected by the system or a time-out occurs. Using the integral FVD capability, each point can be independently verified for its ability to accurately detect a transition to a different value, typically every 500 milliseconds. (For more information on fault reporting time, see Calculation for Diagnostic Fault Reporting Time on page 39.) Using these diagnostics, each channel can be verified independently, thus assuring near 100 percent fault coverage and fail-safe operation under all single-fault scenarios, and most common multiple-fault scenarios.

Analog Input/Digital Input Module Alarms

Analog input/digital input module faults are reported to the control application. These alarms can be used to increase availability during specific multiple-fault conditions. Loss of field power or logic power is reported to the control application.

Analog Output (AO) Modules

Analog output modules use a combination of comparison and reference diagnostics. Under system control, each channel is given control of the output sequentially using the 2oo3 voting mechanism. Each channel independently measures the actual state of an output value by comparing it with the commanded value. If the values do not match, a channel switch is forced by voting another channel. Each channel also compares its measured values against internal references. Using these diagnostics, each channel can be independently verified for its ability to control the analog output value, thus assuring nearly 100 percent fault coverage and fail-safe operation under all single-fault scenarios, and most common multiple-fault scenarios.

Analog Output Module Alarms

Analog output module faults are reported to the control application. These alarms can be used to increase availability during specific multiple-fault conditions. Loss of field power or logic power is reported to the control application.

Digital Input (DI) Modules

Digital input module pointsuse force-to-value diagnostics (FVD). Under system control, each pointis sequentially forced to a test value. The forced value is maintained until the value is detected by the system or a time-out occurs. Using the integral FVD capability, each point can be independently verified for its ability to accurately detect a transition to the opposite state, typically every 500 milliseconds. (For more information on fault reporting time, see Calculation for Diagnostic Fault Reporting Time on page 39.) These diagnostics are executed independently by each channel, thus assuring nearly 100 percent fault coverage and fail-safe operation under all single-fault scenarios, and most common multiple-fault scenarios.

Digital Input Module Alarms

Digital input module faults are reported to the control application. These alarms can be used to increase availability during specific multiple-fault conditions. Loss offield power or logic power is reported to the control application.

Digital Output (DO) Modules

Digital output modules use output voter diagnostics (OVD). Under system control, each output point is commanded sequentially to both the energized and de-energized states. The forced state is maintained until the value is detected by the system or a time-out occurs (500 microseconds, typical case; 2 milliseconds, worst case). Using the integral OVD capability, each point can be independently verified for its ability to a transition to either state, typically every 500 milliseconds. (For more information on fault reporting time, see Calculation for Diagnostic Fault Reporting Time on page 39.)

Digital Output Module Alarms

Digital output module faults are reported to the control application and can be used to increase availability during specific multiple-fault conditions. Loss of field power or logic power is reported to the control application.

The inability of a digital output module to control an output point is reported to the control application as a Load/Fuse alarm. This condition can result from a loss of field power or a field short condition. The alarm can be used to modify the control strategy or direct effective maintenance action.

Pulse Input (PI) Module

The pulse input module points use a combination of comparison and reference diagnostics. Under system control, each channel is independently compared against the measured value of all channels. If a mismatch is found, an alarm is set. Each channel's measured values are also compared against the channel's internal references. These diagnostics are executed independently by each channel, thus assuring nearly 100 percent fault coverage and fail-safe operation under all single-fault scenarios, and most common multiple-fault scenarios.

Pulse Input Module Alarms

Pulse input module faults are reported to the control application. These alarms can be used to increase availability during specific multiple-fault conditions. Loss of logic power is reported to the control application.

Solid-State Relay Output (SRO) Modules

Solid-state relay output module points are not intended for safety-critical applications. The diagnostics used by the solid-state relay output points cannot detect faults in the relay contacts.

Solid-State Relay Output Module Alarms

Solid-state relay output module faults are reported to the control application.

Calculation for Diagnostic Fault Reporting Time

Typical and worst-case fault reporting times can be determined by using the formula in this section. For typical configurations with less than eight AI modules, the calculation is not required and the default values can be used. The default values are:

- 500 milliseconds for a typical case
- 725 milliseconds for a worst case

For configurations with more than eight AI modules, the formula allows you to calculate typical and worst-case fault reporting times. This table provides a sample calculation.

Table 5 Calculating Diagnostic Fault Reporting Time Example

Module Type	Number in System	Multiplier	Result
AI	10	16	160
DI	2	1	2
PI	0	6	0
AO	0	5	0
DO	2	1	2
SRO	0	1	0
			164 x 1.2 = **196.8** *Typical Case*: **196.8** x 3 = **590.4** milliseconds + 1/2 scan time *Worst Case*: **196.8** x 4.5 = **885.6** milliseconds + 1 scan time

Procedure

1. Enter the number of modules and multiply each by the multiplier for the module type and then add the results.
2. Multiply the result of step 1 by 1.2.
3. If the result for step 2 is greater than 150, use the greater number; otherwise, use 150.
4. To determine the typical case, multiply the result of step 3by 3, and then add 1/2 the application scan time.
5. To determine the worst case, multiply the result of step 4 by 4.5, and then add 1 application scan time.

Input/Output Processing

The I/O processor is protected by an independent watchdog that verifies the timely execution of the I/O processor firmware and I/O module diagnostics. In addition, the I/O processor reports its sequence of process execution to the MP. If an I/O processor fails to execute correctly, the MP and the I/O processors enter the fail-safe state and the I/O bus for the faulting channel is disabled, leaving all outputs under control of the remaining healthy channels.

The integrity of the I/O bus is continuously monitored and verified independently by each channel of the system. A catastrophic bus fault results in affected I/O module channels reverting to the fail-safe state in less than 500 milliseconds (0.5 seconds), worst case, or less than 10 milliseconds, typically.

I/O Module Alarms

Loss of communication with an I/O module is reported to the control application and can be used to increase availability during specific multiple-fault conditions.

Main Processor and TriBus

Each Main Processor (MP) module uses memory data comparison between itself and the other MPs to ensure that the control application executes correctly on each scan. Each MP transfers its input data to the other two MPs via the TriBus during each scan. Each MP then votes the input data and provides voted data to the control application. The results of the control application (outputs), including all internal variables, are transferred by the TriBus. If a mis-compare is detected, special algorithms are used to isolate the faulting MP. The faulting MP enters the fail-safe state and is ignored by the remaining MPs. Background diagnostics test MP memory and compare control application instructions and internal status.

The integrity of the TriBus is continuously monitored and verified independently by each MP. All TriBus faults are detected within the scan associated with the TriBus transfer. Fault isolation hardware and firmware causes the MP with the faulting TriBus to enter the fail-safe state.

An independent watchdog ensures that the control application and diagnostics execute within 0.5 seconds. If an MP fails to execute the scan, the watchdog forces the MP to the fail-safe state. The I/O processor adds a sequential element to the MP watchdog. If an MP fails to report the proper sequence of execution, the I/O processor causes the MP to enter the fail-safe state.

External Communication

Loss of external communication is not indicated by a system alarm. However, alarms can be generated by using:

- Semaphores
- System attributes

CAUTION

External communications are intended for transporting non-safety-critical data. The guidelines outlined in Using Triconex Communication Capabilities on page 24 should be followed in SIS applications.

The integrity of external Modbus communication links are continuously monitored. Ethernet links are constantly monitored for current activity.

Semaphores

Establish a semaphore between a controller and an external device by using a timer function block to evaluate the changing state of semaphores.

MP System Attributes

System attributes can be used to generate an alarm when a communication link is lost.

Table 6 Modbus Port Attributes

Attribute Name	Description
MP.RESET_PORT_N	Resets statistics for port parameters
MP.BAD_MSGS_N	Number of bad messages received
MP.BRDCSTS_PORT_N	Number of broadcast messages received
MP.ELAPSED_PORT_N	Milliseconds since last message was received
MP.MSGS_PORT_N	Number of messages received
MP.RSPNS_PORT_N	Number of response messages sent

where N = port position (left, right, middle) and the parameters accumulate data beginning on power up or from the last issuance of the Reset Statistics command

Table 7 Ethernet Ports

Attribute Name	Description
MP.NET_OK_LEFT	Left port is receiving messages
MP.NET_OK_MIDDLE	Middle port is receiving messages
MP.NET_OK_RIGHT	Right port is receiving messages

CM System Attributes

Table 8 Module and Port Status

Attribute Name	Description
CM.SLOT.BAD_MSGS_PORT_N	Number of bad messages received
CM.SLOT.BRDCSTS_PORT_N	Number of broadcast messages received
CM.SLOT.ELAPSED_PORT_N	Milliseconds since last message was received
CM.SLOT.MSGS_PORT_N	Number of messages received
CM.SLOT.NET1_OK	NET1 Ethernet port is receiving messages
CM.SLOT.NET2_OK	NET2 Ethernet port is receiving messages
CM.SLOT.RESET_STATS_PORT_N	Resets statistics for a serial port
CM.SLOT.RSPNS_PORT_N	Number of response messages sent
CM.SLOT.STATS_RESET_PORT_N	Responds that the statistics were reset

For more information about communication alarms, see the following manuals:

- *TriStation 1131 Developer's Guide*
- *Planning and Installation Guide for Triconex General Purpose v2 Systems*
- *Communication Guide for Triconex General Purpose v2 Systems*
- *TriStation 1131 Libraries Reference*

4

Application Development

Development Guidelines

To avoid corruption of project files while developing an application (also known as a *control program*), you should:

- Use a dedicated PC that is not connected to a network.
- Use a PC with ECC memory, if possible.
- Use, according to the vendor's instructions, a regularly-updated, always-on virus scanner.
- Use system utilities such as Checkdisk and vendor diagnostics to periodically determine the health of the PC.
- Use dependable media, such as a CD-ROM instead of a floppy disk.
- Not use a system prone to crashing.
- Not use battery power if using a notebook computer.
- Not copy a project file while it is open in the TriStation 1131 software.
- Not e-mail project files.
- Verify proper installation of the TriStation 1131 software using TriStation Install Check.

  You should run the TriStation Install Check program to verify that the TriStation 1131 software is correctly installed on your PC and that no associated files are corrupted. This is especially helpful if applications besides the TriStation 1131 software reside on your PC. See the *TriStation 1131 Developer's Guide* for instructions on using the TriStation Install Check program.

Triconex Product Alert Notices (PANs)

Product Alert Notices document conditions that may affect the safety of your application. It is essential that you read all current PANs before starting application development, and that you keep up-to-date with any newly released PANs. All PANs can be found on the Invensys Global Customer Support (GCS) website, or contact the Invensys Global Customer Support (GCS) center for assistance (see page viii for contact information).

Safety and Control Attributes

Each element and tagname in the TriStation 1131 application has a *safety* attribute, and a *control* attribute. When the safety attribute is set, the TriStation 1131 software provides extra verification. If you are developing a safety application, you should set the safety attribute.

VAR_IN_OUT Variables

You should not use the VAR_IN_OUT variable in a safety application. Safety standards (such as IEC 61508) recommend limiting the use of pointers in safety applications; VAR_IN_OUT is used as a pointer in the TriStation 1131 application. To automatically check for the use of VAR_IN_OUT in your safety application, set the safety attribute (as described above).

Array Index Errors

If an array index error is detected during runtime, the default behavior is to trap. This results in the Tri-GP controller going to the safe state, with all outputs de-energized.

If your application requires some other behavior, you can use a CHK_ERR function block to detect the error, and a CLR_ERR function block to clear the error and prevent a trap.

Note If an array index is too small or too large, the array operation is performed on the last element of the array. Array bounds checking is always turned on—there is no means to disable the array index checking.

See the *TriStation 1131 Libraries Reference* for more information about the CHK_ERR and CLR_ERR function blocks.

Infinite Loops

If the actual scan time exceeds the maximum allowable scan time for the Tri-GP controller, the main processors will reset, causing the Tri-GP controller to go to the safe state, with all outputs de-energized. The maximum allowable scan time for the Tri-GP is 450 milliseconds.

Although it is not possible to program an endless loop with TriStation 1131 software, it **is** possible to create a loop with a very long time, enough to increase the actual scan time beyond the controller's maximum allowable scan time.

See Setting Scan Time on page 47 for more information about actual and maximum scan times.

Important TriStation 1131 Software Commands

Several commands in TriStation 1131 Developer's Workbench are of special interest when developing a safety application:

- Download Change
- Verify Last Download to the Controller
- Compare to Last Download

Download Changes

The Download Changes command is a convenient means of making simple modifications to an offline system during application development.

WARNING

Download Changes is intended for offline use during application development. If you use Download Changes to modify a safety-critical application that is running online, you must exercise extreme caution because an error in the modified application or system configuration may cause a trip or unpredictable behavior.

If you must make online changes to a controller, you should always follow the guidelines provided in the TriStation 1131 Developer's Guide and fully understand the risks you are taking by using the Download Changes command.

Note that the scan time of the controller is doubled momentarily after you use the Download Changes command.

Before a Download Change, use the Triconex Enhanced Diagnostic Monitor to verify that Scan Surplus is sufficient for the application and changes being made. As a rule, the value for Scan Surplus should be at least 10 percent of Scan Time to accommodate newly added elements. For more information on scan time, see Setting Scan Time on page 47.

CAUTION

Do not attempt a Download Change if you have a negative Scan Surplus. First, adjust Scan Time to make the surplus value greater than or equal to zero. Please note, adjusting the scan time of a running system may degrade communications performance.

For more information on the Download Changes command, see the TriStation 1131 Developer's Guide.

Verify Last Download to the Controller

Before you make changes to a project in the TriStation 1131 software, you should run the Verify Last Download to the Controller command to verify that the project in the TriStation 1131 software matches the application running in the controller. (You can find this command on the Commands menu on the Controller Panel in the TriStation 1131 software.) This command compares the current application running in the controller to a record of the last downloaded

application. To use the Verify Last Download to the Controller command, you must be able to connect to the controller using the Connect command on the Controller panel in the TriStation 1131 software.

For more information on the Verify Last Download to the Controller command, see the TriStation 1131 Developer's Guide.

Compare to Last Download

After you have run the Verify Last Download to Controller command, make the desired changes to the project. Use the Compare to Last Download command to verify that the changes to the project are only the intended changes. (You can find this command on the Project menu of the Controller Panel in the TriStation 1131 software.) To test the changes, use the Emulator Control Panel in the TriStation 1131 software.

Setting Scan Time

Setting appropriate scan time for an application is essential to avoid improper controller behavior. When changing an application running in an online system, special precautions should be exercised to avoid scan time overruns, which could result in unexpected controller behavior.

Scan Time

Scan time is the interval required for evaluations (in other words, scans) of an application as it executes in the controller. The time it actually takes to do an evaluation may be less than the requested scan time. To prevent scan-time overruns, a scan time must be set that includes sufficient time for all executable elements in an application—including print statements, conditional statements, and future download changes.

Use the Scan Time parameter in the TriStation 1131 software Program Execution List to suggest the desired scan time before downloading an application—this is the *Requested Scan Time*. Upon downloading, the controller determines the minimum and maximum allowable scan times for your application and uses your requested scan time if it falls within the acceptable limits. The default scan time is 200 milliseconds. The maximum allowable scan time is 450 milliseconds and the minimum allowable scan time is 10 milliseconds. *Actual scan time* is the actual time of the last scan. Actual scan time is always equal to or greater than the requested scan time.

Note To guarantee that the controller provides a deterministic response time, the scan time should always be set to a value **greater than** the I/O poll time (the maximum time needed by the controller to obtain data from the input modules). You can view the I/O poll time on the System Overview screen in the Triconex Enhanced Diagnostic Monitor.

Scan Surplus

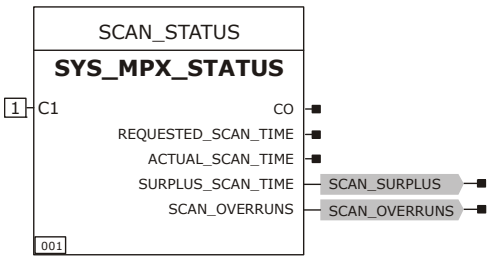
Scan Surplus is the scan time remaining after application elements have been executed. Scan Surplus must be positive—if it is negative, the Scan Time parameter must be adjusted (using the Set Scan Time command on the Commands menu in the Controller Panel) to set the surplus

value to greater than or equal to zero. The Scan Time parameter in the TriStation 1131 software Program Execution List applies only when you perform a Download All.

Scan Overrun

If Scan Surplus becomes negative and a scan overrun occurs, the relevant status attributes are set as follows:

- MP.SCAN_OVERRUNS is incremented once for each time that a longer scan time is needed.



MP.SURPLUS_SCAN_TIME is set to a negative number to indicate the additional time period used by a scan overrun.For more information, see the TriStation 1131 Developer's Guide and the *TriStation 1131 Libraries Reference.*

Sample Safety-Shutdown Programs

This section describes sample programs and methods for implementing safety-shutdown networks.

When All I/O Modules Are Safety-Critical

The sample program, EX01_SHUTDOWN, shows one way to verify that the safety system is operating properly when every module in the safety system is safety-critical. This example uses an instance of the Triconex Library function block SYS_SHUTDOWN named CRITICAL_MODULES.

Note The sample program is an element of project TdTUV.pt2 included as part of the TriStation 1131 software installation. The default location of the project is C:\Documents and Settings\<user>\My Documents\Triconex\TriStation 1131 4.x\Projects.

When the output CRITICAL_MODULES_OPERATING is true, all safety-critical modules are operating properly. The input MAX_TIME_DUAL specifies the maximum time allowed with two channels operating (with no connection, defaults to 40000 days). The input MAX_TIME_SINGLE specifies the maximum time allowed with one channel operating (3 days in the example).

Note In typical applications, the operating time restrictions in the table on page 22 should be followed.

When CRITICAL_MODULES_OPERATING is false, the time in degraded operation exceeds the specified limits; therefore, the control program should shut down the process under safety control.

CAUTION

EX01_SHUTDOWN does not handle detected field faults, rare combinations of faults detected as field faults, or output voter faults hidden by field faults. The application, not the SYS_SHUTDOWN function block, must read the NO_FLD_FLTS module status or FLD_OK point status to provide the required application-specific action.

For information on improving availability using external, power-disconnect relays and advanced programming techniques, see the sample program EX02_SHUTDOWN.

Program EX01_SHUTDOWN

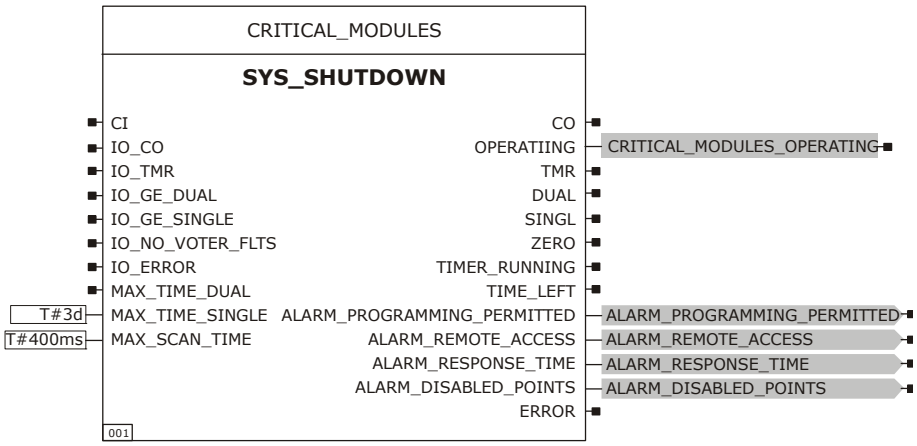


Figure 10 EX01_SHUTDOWN Sample Program

CO false indicates a programming error; for example, MAX_TIME_SINGLE greater than MAX_TIME_DUAL. The error number shows more detail.

Table 9 Input Parameters for SYS_SHUTDOWN Function Block in EX01_SHUTDOWN

Parameter	Description
CI	Do not connect when all I/O modules are system-critical
IO_CO	Do not connect when all I/O modules are system-critical
IO_TMR	Do not connect when all I/O modules are system-critical
IO_GE_DUAL	Do not connect when all I/O modules are system-critical
IO_GE_SINGLE	Do not connect when all I/O modules are system-critical
IO_NO_VOTER_FLTS	Do not connect when all I/O modules are system-critical
IO_ERROR	Do not connect when all I/O modules are system-critical
MAX_TIME_DUAL	Maximum time with only two channels operating
MAX_TIME_SINGLE	Maximum time with only one channel operating
MAX_SCAN_TIME	50% of the maximum response time

Table 10 Output Parameters for SYS_SHUTDOWN Function Block in EX01_SHUTDOWN

Parameter	Description
CO	Control out
OPERATING	When true, all safety-critical modules are operating properly When false, the time in degraded operation exceeds the specified limits; therefore, the control program should shut down the process
TMR	System is operating in triple modular redundant mode
DUAL	At least one safety-critical point is controlled by two channels
SINGL	At least one safety-critical point is controlled by one channel
ZERO	At least one safety-critical point is not controlled by any channel
TIMER_RUNNING	Time left to shutdown is decreasing
TIME_LEFT	Time remaining before shutdown
ALARM_PROGRAMMING_PERMITTED	True if application changes are permitted
ALARM_REMOTE_ACCESS	True if remote-host writes are enabled
ALARM_RESPONSE_TIME	True if actual scan time is greater than MAX_SCAN_TIME
ALARM_DISABLED_POINTS	True if one or more points are disabled.
ERROR_NUM	Error Number: 0 = No error 1 = Error in maximum time 2 = I/O function block error. IO_ERROR is non-zero 3 = Function block error. System status or MP Status

Table 11 Alarm Output Parameter Operation in EX01_SHUTDOWN

Parameter	Description
ALARM_PROGRAMMING_PERMITTED	To remind the operator to lock out programming changes after a download change, or for applications in which download changes are not allowed, connect the ALARM_PROGRAMMING_PERMITTED output to an alarm.
ALARM_REMOTE_ACCESS	For applications in which remote changes are not allowed, connect the ALARM_REMOTE_ACCESS output to an alarm.
ALARM_RESPONSE_TIME	Total response time depends primarily on the actual scan time. To meet the required response time of the process, set the MAX_SCAN_TIME input to less than 50% of the required response time. When the actual scan time exceeds the MAX_SCAN_TIME value, the ALARM_RESPONSE_TIME output becomes true.
ALARM_DISABLED_POINTS	A project should not contain disabled points unless there is a specific reason for disabling them, such as initial testing or maintenance. To remind an operator that some points are disabled, connect the ALARM_DISABLED_POINTS output to an alarm.

When Some I/O Modules Are Safety-Critical

For some applications, not all modules may be critical to a process. For example, an output module that interfaces to the status indicators on a local panel is usually not critical to a process.

The EX02_SHUTDOWN sample program shows how to increase system availability by detecting the status of safety-critical modules. The user-defined function block CRITICAL_IO checks the safety-critical I/O modules. The CRITICAL_IO outputs are connected to the inputs of the CRITICAL_MODULES function block.

Note The sample program is an element of project TdTUV.pt2 included as part of the TriStation 1131 software installation. The default location of the project is C:\Documents and Settings\<user>\My Documents\Triconex\TriStation 1131 4.x\Projects.

When the output CRITICAL_MODULES_OPERATING is true, all critical modules are operating properly. The input MAX_TIME_DUAL specifies the maximum time allowed with two channels operating (with no connection, defaults to 40000 days). The input MAX_TIME_SINGLE specifies the maximum time allowed with one channel operating (three days in the example).

Note In typical applications, the operating time restrictions in the table on page 22 should be followed.

When CRITICAL_MODULES_OPERATING is false, the time in degraded operation exceeds the specified limits; therefore, the control program should shut down the plant.

CAUTION

EX02_SHUTDOWN does not handle detected field faults, rare combinations of faults detected as field faults, or output voter faults hidden by field faults. The application, not the SYS_SHUTDOWN function block, must read the NO_FLD_FLTS module status or FLD_OK point status to provide the required application-specific action.

Program EX02_SHUTDOWN

Figure 11 EX02_SHUTDOWN Sample Program

Table 12 Input Parameters for SYS_SHUTDOWN Function Block in EX02_SHUTDOWN

Parameter	Description
CI	Control In If false, then CO is false—no change in the output value If true and ERROR_NUM is 0, then CO is true
IO_CO	Critical I/O control out
IO_TMR	All critical I/O points are operating in triple modular redundant mode
IO_GE_DUAL	All critical I/O points are operating are operating in dual or TMR mode
IO_GE_SINGLE	All critical I/O points are operating are operating in single, dual, or TMR mode

Table 12 Input Parameters for SYS_SHUTDOWN Function Block in EX02_SHUTDOWN

Parameter	Description
IO_NO_VOTER_FLTS	If true, then no voter faults exist on a critical I/O module If false, then a voter fault exists on a critical I/O module
IO_ERROR	Error number: Zero indicates no error. Non-zero indicates a programming or configuration error
MAX_TIME_DUAL	Maximum time with only two channels operating
MAX_TIME_SINGLE	Maximum time with only one channel operating
MAX_SCAN_TIME	50% of the maximum response time

Table 13 Output Parameters for SYS_SHUTDOWN Function Block in EX02_SHUTDOWN

Parameter	Description
CO	Control Out
OPERATING	When true, all safety-critical modules are operating properly When false, the time in degraded operation exceeds the specified limits; therefore, the control program should shut down the process
TMR	System is operating in triple modular redundant mode
DUAL	At least one safety-critical point is controlled by two channels
SINGL	At least one safety-critical point is controlled by one channel
ZERO	At least one safety-critical point is not controlled by any channel
TIMER_RUNNING	Time left to shutdown is decreasing
TIME_LEFT	Time remaining before shutdown
ALARM_PROGRAMMING_PERMITTED	True if application changes are permitted
ALARM_REMOTE_ACCESS	True if remote-host writes are enabled
ALARM_RESPONSE_TIME	True if actual scan time is greater than MAX_SCAN_TIME
ALARM_DISABLED_POINTS	True if one or more points are disabled
ERROR_NUM	Error Number: 0 = No error 1 = Error in maximum time 2 = I/O function block error. IO_ERROR is non-zero 3 = Function block error. System status or MP Status

Table 14 Alarm Output Parameter Operation in EX02_SHUTDOWN

Parameter	Description
ALARM_PROGRAMMING_PERMITTED	To remind the operator to lock out programming changes after a download change, or for applications in which download changes are not allowed, connect the ALARM_PROGRAMMING_PERMITTED output to an alarm
ALARM_REMOTE_ACCESS	For applications in which remote changes are not allowed, connect the ALARM_REMOTE_ACCESS output to an alarm
ALARM_RESPONSE_TIME	Total response time depends primarily on the actual scan time. To meet the required response time of the process, set the MAX_SCAN_TIME input to less than 50% of the required response time. When the actual scan time exceeds the MAX_SCAN_TIME value, the ALARM_RESPONSE_TIME output becomes true
ALARM_DISABLED_POINTS	A project should not contain disabled points unless there is a specific reason for disabling them, such as initial testing or maintenance. To remind an operator that some points are disabled, connect the ALARM_DISABLED_POINTS output to an alarm

Defining Function Blocks

You can create your own user-defined function blocks for a safety-critical module.

Procedure

1. Copy the example EX02_CRITICAL_IO in the TdTUV.pt2 sample project provided with the TriStation 1131 Developer's Workbench.
2. Edit the lines following the comment "Include here all safety-critical I/O modules."

   Each line calls the safety-critical I/O (SCIO) function block for one safety-critical I/O module.

   Example

```
(*************************************************************)
(* Include here all safety-critical I/O modules: *)
SCIO( IOP:=1,SLOT:=1,   APP:=DE_ENERGIZED,    RELAY_OK:=FALSE    ): (*DI*)
SCIO( IOP:=1,SLOT:=2,   APP:=RELAY,           RELAY_OK:=RELAY1_OK); (*DO*)
SCIO( IOP:=1,SLOT:=3,   APP:=ENERGIZED,       RELAY_OK:=FALSE    ); (*DO*)
(*    IOP:=1,SLOT:=4,   NOT CRITICAL)                            *) (*RO*)
```

3. Enter the correct IOP number, SLOT number, APP (application), and RELAY_OK parameters for the safety-critical I/O module.

Table 15 Parameters for Safety-Critical Modules

Application Type (App)	Relay_OK Parameter	Description
RELAY (de-energized to trip with relay)	True	A voter fault degrades the mode to dual. The relay provides a third channel for shutdown so that if an output voter fails, two independent channels (relay and other output voter channel) remain that can de-energize the output.
RELAY (de-energized to trip with relay)	False	A voter fault degrades the mode to single. A non-voter fault degrades the mode to dual.
DE-ENERGIZED (de-energized to trip with no relay)	—	A voter fault degrades the mode to single. A non-voter fault degrades the mode to dual.

Partitioned Processes

You can achieve greater system availability if you can allocate modules to processes that do not affect each other. For example, you could have two processes with:

- Outputs for one process on one DO module
- Outputs for another process on a second DO module
- Inputs from a shared DI module

You do this by partitioning processes.

Procedure

1. Partition the safety-critical I/O modules into three function blocks:
   - SHARED_IO for the shared safety-critical I/O modules
   - PROCESS_1_IO for safety-critical I/O modules that do not affect process 2
   - PROCESS_2_IO for safety-critical I/O modules that do not affect process 1
2. Connect the function blocks as shown in the EX03_SHUTDOWN example on page 58.

CAUTION

EX03_SHUTDOWN does not handle detected field faults, rare combinations of faults detected as field faults, or output voter faults hidden by field faults. The application, not the SYS_SHUTDOWN function block, must read the NO_FLD_FLTS module status or FLD_OK point status to provide the required application-specific action.

Program EX03_SHUTDOWN

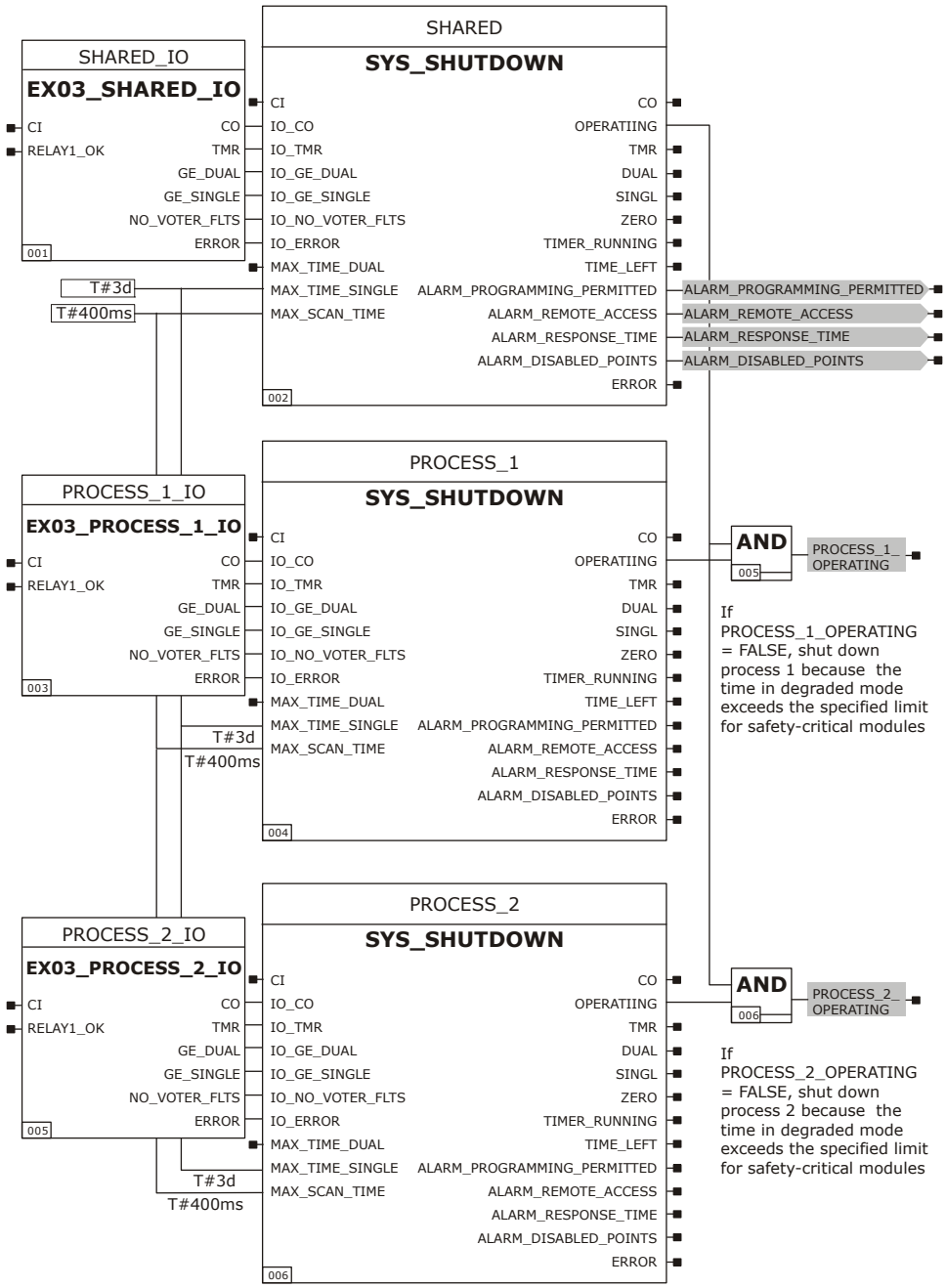


Figure 12 EX03_SHUTDOWN Sample Program

Alarm Usage

To implement the guidelines, the alarms described below are provided with TriStation 1131 software.

Programming Permitted Alarm

To remind the operator to lock out programming changes after a download change, or for applications in which download changes are prohibited, connect the ALARM_PROGRAMMING_PERMITTED output to an alarm.

Remote Access Alarm

In applications for which remote changes are not allowed, connect the ALARM_REMOTE_ACCESS output to an alarm.

Response Time Alarm

Response time refers to the maximum time allocated for the controller to detect a change on an input point and to change the state of an output point. Response time is primarily determined by scan time (the rate at which the program is run), but is also affected by process time (how fast the process can react to a change). The response time of the controller must be equal to or faster than the process time. The scan time must be at least two times faster than the response time. To meet the required response time of the process, set the MAX_SCAN_TIME input to less than 50 percent of the required response time. When the actual scan time as measured by the firmware exceeds the MAX_SCAN_TIME value, the ALARM_RESPONSE_TIME output becomes true.

Disabled Points Alarm

A project should not contain disabled points unless there is a specific reason for disabling them, such as initial testing. An alarm is available to alert the operator that a point is disabled.

A

Triconex Peer-to-Peer Communication

Overview

Triconex Peer-to-Peer protocol is designed to allow multiple Triconex controllers in a closed network to exchange safety-critical data. (If you plan to implement a complex Peer-to-Peer network, please contact the Invensys Global Customer Support (GCS) center.) To enable Peer-to-Peer communication, you must connect each controller (also referred to as a *node*) to an Ethernet network by means of port on the. The controllers exchange data by using Send and Receive function blocks in their TriStation 1131 applications.

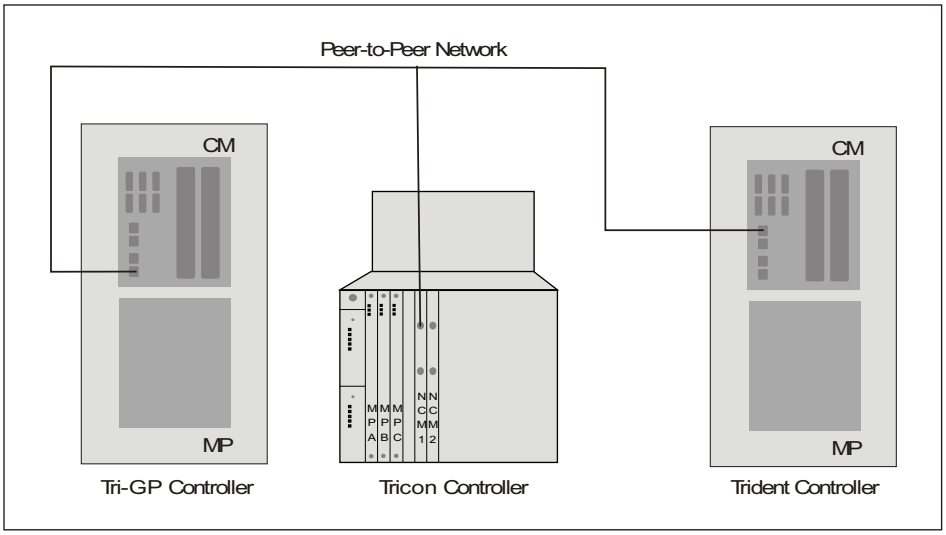


Figure 13 Basic Triconex Peer-to-Peer Network

To configure a TriStation 1131 application for Peer-to-Peer communication, you must do the following 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 your programs
- Observe restrictions on data transmission speed

In addition, Triconex recommends that you calculate the data transfer time to determine whether your control algorithms will operate correctly. Instructions for performing this calculation are provided on page 63.

The sample programs described in this appendix can be found in the Tdpeer.pt2 project included as part of the TriStation 1131 software installation. These programs show how to send data at high speed and under controlled conditions.

For more detailed information on Triconex Peer-to-Peer communication, please refer to the *Communication Guide for Triconex General Purpose v2 Systems.*

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 (*nodes*) 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.

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 (nodes). The more memory allocated for aliased points the slower the transfer time.

Procedure

1. In the TriStation 1131 software, on the sending controller, expand the Controller tree, and double-click Configuration. On the Configuration tree, click Memory Allocation.
2. Find the bytes allocated for BOOL, DINT, and REAL points by doing this:
   - On the Configuration tree, click Memory Points, Input Points, or Output Points. Double-click the graphic for the point type.
   - Add the number of bytes allocated for all BOOL input, output, and aliased memory points. Enter the number on step 1 of the following worksheet. Enter the number for DINT and REAL points on step 1.
3. On the receiving controller, get the BOOL, DINT, and REAL points and enter the numbers on step 3 of the Data Transfer Time worksheet.

Estimating the Data Transfer Time

The basic formula for estimating the data transfer time is as follows:

- Data transfer time in milliseconds = 2 * (larger of TS or SS) + 2 * (larger of TR or SR)

Table 16 Data Transfer Time Formula Parameters

Parameter	Description
TS	Time for sending node to transfer Aliased data over the communication bus in milliseconds = (Number of aliased variables in bytes ÷ 100,000) * 1000
SS	Scan time of sending node in milliseconds
TR	Time for receiving node to transfer Aliased data over the communication bus in milliseconds = (Number of aliased variables in bytes ÷ 100,000) * 1000
SR	Scan time of receiving node in milliseconds

Procedure

1 Use the instructions in the following 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 *sending* controller and divide or multiply as indicated. Add the results.	BOOL	_____	÷ 8 =	_____
	DINT	_____	x 4 =	_____
	REAL	_____	x 4 =	_____
	Total bytes of aliased points TBS =			_____
2. Multiple total bytes sending TBS (step 1) by 0.01			TS =	_____
3. Enter the number of bytes for each point type on the *receiving* controller and divide or multiply as indicated. Add the results.	BOOL	_____	÷ 8 =	_____
	DINT	_____	x 4 =	_____
	REAL	_____	x 4 =	_____
	Total bytes of aliased points TBR =			_____
4. Multiple total bytes receiving TBR (step 3) by 0.01			TR =	_____
5. Get the scan time of sending node in milliseconds by viewing the Scan Time in the Execution List.			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.				

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 three 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.

For an example that shows how to measure the maximum data transfer time and use SEND/RECEIVE function blocks to transfer-safety critical data, see Example 4: Using SEND/RECEIVE Function Blocks for Safety-Critical Data on page 67. While testing your system, you should measure the actual maximum time it takes to transfer data to ensure the validity of your calculations.

As Table 16 shows, it takes from two to 30 seconds to detect and report time-out and communication errors. This is why a receiving node that uses the received data to make safety-critical decisions must include logic to verify that new data is received within the specified time period. If the data is not received within the specified process-tolerance time, the application must take appropriate actions depending on requirements.

For an example that shows how to use SEND and RECEIVE function blocks for transferring safety critical data, see Example 4: Using SEND/RECEIVE Function Blocks for Safety-Critical Data on page 67.

Refer to the *TriStation 1131 Libraries Reference* for additional information.

Examples of Peer-to-Peer Applications

Triconex Peer-to-Peer function blocks are designed to transfer limited amounts of data between two applications. Therefore, you should use these function blocks sparingly in your applications.

Ideally, you should control the execution of each SEND function block so that each SEND is initiated only when the acknowledgment for the last SEND is received and new data is available for sending. You can do this through effective use of the SENDFLG parameter in the SEND function block and the STATUS output of the SEND function block, as shown in Examples 3 and 4.

The examples described in this section can be found in the Tdpeer.pt2 project included as part of the TriStation 1131 installation.

Example 1: Fast Send to One Triconex Node

This example shows how to send data as fast as possible from node #2 to node #3. Scan time in both controllers is set to 100 milliseconds.

The example uses the following project elements:

- PEER_EX1_SEND_FBD (for sending node #2)
- PEER_EX1_RCV_FBD (for receiving node #3)

Example 2: Sending Data Every Second to One Node

This example shows how to send data every second from node #2 to node #3. Scan time in both controllers is set to 100 milliseconds.

The example uses the following project elements:

- PEER_EX2_SEND_FBD (for sending node #2)
- PEER_EX2_RCV_FBD (for receiving node #3)

Example 3: Controlled Use of SEND/RECEIVE Function Blocks

This example shows how to use SEND/RECEIVE function blocks correctly, in a controlled way, so that a limited amount of important data can be transferred between two applications when new data is ready to be sent.

This example uses the following project elements:

- PEER_EX3_SEND_FBD (for sending node #2)
- PEER_EX3_RCV_FBD (for receiving node #3)

Example 4: Using SEND/RECEIVE Function Blocks for Safety-Critical Data

This example shows how to use SEND/RECEIVE function blocks for transferring a limited amount of safety-critical data between the two applications as fast as possible. It also shows how to measure the actual maximum time for transferring data from the sending node to the receiving node.

Sending Node #1 Parameters

- Scan time (SS) = 125 milliseconds.
- Number of aliased variables in bytes = 2000.
- Time to transfer aliased data over the communication bus in milliseconds (TS) = (2000/100,000) * 1000 = 20 milliseconds.
- The sending controller has one SEND function block in the application, meeting the requirement to have five or fewer SEND function blocks. The sendflag parameter is in the SEND function block so that the sending controller initiates another SEND as soon as the last SEND is acknowledged by the receiving controller.

Receiving Node #3 Parameters

- Scan time (SR) = 100 milliseconds.
- Number of aliased variables in bytes = 15,000.
- Time to transfer aliased data over the communication bus in milliseconds (TR) = (15,000/100,000) * 1000 = 150 milliseconds.
- Process-tolerance time = 4 seconds.
- Estimated data transfer time = 2 * 125 + 2 * 150 = 550 milliseconds.

If the sending controller does not receive acknowledgment from the receiving controller in one second, it automatically retries the last SEND message. Because of network collisions, communication bus loading, etc., the sending controller occasionally has to retry once to get the message to the receiving node. This is why the general rule for data transfer time is one to two seconds, even though the estimated time is 550 milliseconds.

The receiving node has a network to measure the actual time so you can validate the assumed two-second maximum transfer time. Since the process-tolerance time of the receiving node is four seconds, the maximum time-out limit is set to two seconds (half the process-tolerance time). The receiving node should receive at least one data transfer within the maximum time-out limit. Using this criteria meets the basic requirement for using peer-to-peer communication to transfer safety-critical data.

This example packs 32 BOOL values into a DWORD and sends the DWORD and a diagnostic variable to a receiving node as fast as possible by setting the sendflag parameter to 1 all the time. The diagnostic variable is incremented every time a new SEND is initiated. The receiving node checks the diagnostic variable to verify that it has changed from the previous value received. The receiving node also determines whether it has received at least one data transfer within the process-tolerance time. If not, the application takes appropriate action, such as using the last data received or using default data to make safety-critical decisions.

This example uses the following project elements:

- PEER_EX4_SEND_FBD (for sending Node #1)
- PEER_EX4_RCV_FBD (for receiving Node #3)

B

HART Communication

Overview

This appendix contains a position paper from TÜV Rheinland on using the HART™ communication protocol in safety-related applications within Safety Instrumented Systems (SIS). The paper includes rules and guidelines that should be followed.

HART Position Paper from TÜV Rheinland

2008-04-01

TÜVRheinland®
Precisely Right.

Automation, Software and Information Technology

Position paper about the use of HART communication in safety related applications within Safety Instrumented Systems

Version 1.0
Date: 2008-04-01

TÜV Rheinland Industrie Service GmbH
Automation, Software and Information Technology (ASI)
Am Grauen Stein
51105 Köln
Germany

2008-04-01

TÜVRheinland®
Precisely Right.

Revision History

Revision	Date	Change	Author	Approval
1.0	2008-04-01	Initial Release	O. Busa	H. Gall

2008-04-01

TÜVRheinland®
Precisely Right.

Contents **Page**

2008-04-01

TÜVRheinland®
Precisely Right.

1. Scope

The scope of this paper is to review the use of the HART communication as a non-safety related communication protocol within safety related applications using a Safety Instrumented System (SIS).

2. Standards

Functional Safety

[1] IEC 61508:2000, parts 1 - 7
Functional safety of electrical/electronic/programmable electronic safety related systems

[2] IEC 61511:2004, parts 1 - 3
Functional safety - Safety instrumented systems for the process industry sector

3. Analysis

3.1 General analysis

The HART (Highway Addressable Remote Transducer) communication protocol is a widely accepted standard used for communication between intelligent field instruments and host systems.

HART is a master-slave field communication protocol, which use a phase-continuous Frequency Shift Keying (FSK) to extend analog signaling, whereas a high-frequency current is superimposed on a low-frequency analog current typically 4-20 mA.

The benefit of HART that it can be used in parallel to the analog 4-20 mA field devices using the same wiring in which a bi-directional communication to several HART capable field devices is supported.

Devices which use HART as a communication protocol can be divided into two groups:

HART Master Devices

The master devices are typically multiplexer devices with a HART modem which are connected directly through the field wiring of a host system (input/output system) typically a programmable logic controller. They are in general used to configure slave devices and provide additional diagnostics to a host system.

HART Slave devices

The slave devices are in general HART capable intelligent field devices using an analog 4-20 mA current for process values.

3.2 Safety analysis

The HART communication protocol was developed to be used within standard process control systems for device configuration and diagnostics. The development itself was not performed in accordance to the IEC 61508 [1] and it was not approved as a safety related communication protocol.

Considering the development of the HART communication protocol the following assumptions can be made according to the safety impact and interference of HART capable devices, which are intended to be used together with a safety related system.

2008-04-01

TÜVRheinland®
Precisely Right.

A possible impact to the safety function could be related to:

- Interference to the analog 4-20 mA signal
- Interference to the field interface input of the safety related system
- Interference to the HART capable field device

The possible effects could be negative reactions, process signal corruption or unintended configuration changes of the connected field devices.

As a matter of principle that the HART master devices are directly connected to the field wiring of a safety related programmable system, it must be assumed that an impact on the analog process signal to/from the field devices is possible. Under normal circumstances the superimposed high frequency current should be interference free to low-frequency analog 4-20 mA current, because it will be filtered by standard analog input circuit filters.

Considering abnormal function resulting from the failures (e.g., random hardware failures) of the HART master devices, the impact to the safety related programmable system could be safety critical resulting in negative electrical reactions on the input side or output side of the Safety Instrumented System or corruption of the measured analog signal.

Further impact could be related to the connected field devices. The configuration could be changed unintended resulting in non reliable analog input values to the Safety Instrumented System. These failures could happen from a random hardware or systematic software faults or an accidentally re-configuring from personnel.

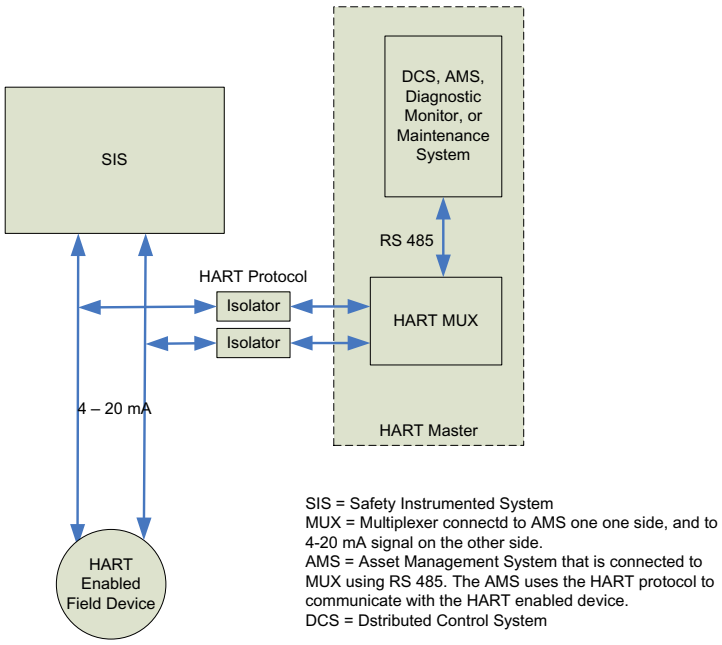
In summary, in order to use a HART protocol, which is not approved for safety related communication within an SIS, the following concerns must be taken into account to ensure that the use of HART is non-interfering the 4-20 mA signal (safety input or output):

1. HART Master (AMS and MUX) directly connected to the field wiring of safety related system could impact the process signal.
2. Abnormal function of HART Master or slave due to random hardware faults, systematic software failures, etc. could impact the process signal
3. Unintended configuration changes (accidental re-configuring from personnel, random hardware or systematic software faults) to the field device resulting in non-reliable analog input values to the SIS.

2008-04-01

TÜVRheinland®
Precisely Right.

Picture 1: HART usage within SIS



In order to ensure that the HART Master/MUX connection to the 4-20 mA signal (input/output) connected to the SIS and use of HART protocol is non-interfering the 4-20 mA safety signal, the rules/guidelines specified in the following must be followed by the end user.

Note, that the end user could incorporate some of the guidelines in the management of safety, management of change, and management of maintenance policies and procedures.

4. Rules and Guidelines for the use of a HART protocol within SIS

- AMS/MUX connection to 4-20 mA signal must be isolated and decoupled such that a failure of the MUX does not affect the 4-20 mA signal
- A detailed FMEA of the MUX interface to the 4-20 mA signal must be performed to show that the normal operation or the failure of MUX won't affect the 4-20 mA signal. In this case, the MUX and associated AMS are non-interfering the 4-20 mA safety signal. If non-interfering can not be shown by FMEA the probability of failure must be considered to determine the safety parameters in accordance to IEC 61508 to include them into safety loop consideration.
- The HART enabled devices used with SIS must be suitable for the SIL rating of the safety loop as per IEC 61511 guidelines. HART enabled field devices shall be documented to be non-interfering the 4-20 mA safety signal. In compliance with IEC 61511 -1 clause 11.5.2, the HART enabled devices used within an SIS (Safety Instrumented System) shall be documented to be in accordance with IEC 61508-2 and IEC 61508-3 as applicable to the SIL (Safety Integrity Level) of the individual SIF (Safety Instrumented Function), or else they shall meet the requirements of IEC 61511-1 clause 11.4 for hardware fault tolerance and clauses 11.5.3 to 11.5.6 for prior use, as appropriate.

2008-04-01

TÜVRheinland®
Precisely Right.

- To avoid failures resulting from unintended configuration changes of the HART capable field devices, protection mechanisms, which may be provided by the HART devices, shall be used.

- The data (Secondary variables, device information, diagnostics, and status data) received from the HART device must not be used for safety related reactions in the SIS.

- The data received from the HART devices could be used in control, diagnostics, or asset management systems for device diagnostics and maintenance purposes. The Asset Management, Diagnostics, maintenance systems and operating procedures is to act as an independent layer of protection; in this case the role of asset management is to identify a potential failure on a field element (i.e. valve) that might prevent the safety function to act if demanded.

- As a part of management of safety policy, practices and procedures, the AMS (HART master) must be password protected to allow maintenance of and modifications to the HART enabled field devices connected to SIS by authorized people only. The Safety procedure should include provisions to avoid re-configuration of HART enabled field devices by mistake.

- Online changes to the field device configuration, online calibration and online maintenance shall be avoided and must be evaluated depending on the application.

- Safety procedures must be established to ensure proper use of the AMS or hand-held communicator for configuration changes, calibration or maintenance of HART enabled field devices connected to SIS. Any change to the HART enable field device must be followed by verification to prove the correct implementation.

5. Summary

Following the rules and guidelines as listed in chapter 4. the HART communication can be used within safety related applications using Safety Instrumented Systems (SIS).

Besides, the available safety- and user guidelines of the SIS and HART devices shall be considered for further information and restrictions.

C

Safety-Critical Function Blocks

Overview

This appendix describes the function blocks intended for use in safety-critical applications and shows the Structured Text code for the following function blocks:

SYS_CRITICAL_IO

Accumulates the status of safety-critical I/O modules.

Syntax

```
MY_SYS_CRITICAL_IO( CI:=b1, RESET:=b2, IOP:=n1, SLOT:=n2, APP:=n3,
RELAY_OK:=b3 ) ;
```

Input Parameters

Name	Data Type	Description
CI	BOOL	Enables SYS_CRITICAL_IO.
RESET	BOOL	Resets.
IOP	DINT	The IOP (0–15).
SLOT	DINT	The slot number (0–56).
APP	DINT	The application number (1-2).
RELAY_OK	BOOL	The relay is energized and not stuck.

Output Parameters

Name	Data Type	Description
CO	BOOL	True if SYS_CRITICAL_IO executes successfully.
TMR	BOOL	Three channels are operating without faults on all critical I/O module.
GE_DUAL	BOOL	At least two channels are operating without faults on all critical I/O module.
GE_SINGLE	BOOL	At least one channel is operating without faults on all critical I/O module.
NO_VOTER_FLTS	BOOL	No voter faults detected on critical modules.
ERROR_NUM	DINT	Error Number: 0 = No error. –1 = The IOP is invalid. –2 = The slot is not valid. –3 = The module is not configured. –4 = The module is the wrong type.

Description

The SYS_CRITICAL_IO function block accumulates the status of safety-critical I/O modules.

Instructions for Use

To initialize, invoke once with RESET := TRUE.

To complete initialization, invoke again with these input settings:

- RESET := FALSE
- CI := TRUE
- APP := DE_ENERGIZED
- RELAY_OK := false

To get the status of a safety-critical I/O module:

- For each module, invoke specifying the input values:
- IOP
- SLOT
- APP
- RELAY_OK

  For example, if IOP 1 SLOT 1 is a critical DO module with a relay, and SCIO is the function block instance name:

  ```
  SCIO( IOP:=1, SLOT:=1, APP:=RELAY,RELAY_OK:=RELAY1_OK );
  ```

- Read the output values:
- CO
- TMR
- GE_DUAL
- GE_SINGLE

Example

For shutdown examples, see the sample project located at My Documents\Triconex\TriStation 1131 4.x\Projects\TdTUV.pt2

Runtime Errors

Condition	Return Value	Error Flags
If ERROR_NUM is non-zero	Reset all BOOL outputs to false	BADPARAM, ERROR

Upon detection of a runtime error condition, the function block returns the indicated values and sets the error flags to true. For more information about error flags and runtime errors, see the *TriStation 1131 Libraries Reference*.

Application Notes

- Can be used in Safety or Control applications.

Library

Trident and Tri-GP (TRDLIB)

Structured Text

```
FUNCTION_BLOCK SYS_CRITICAL_IO
VAR_INPUT
    CI :            BOOL := TRUE ;  (* Control in. *)
    RESET :         BOOL ;          (* Reset *)
    IOP :           DINT ;          (* IOP number 0-15 *)
    SLOT :          DINT ;          (* SLOT number 0-56 *)
    APP :           DINT ;          (* Application number 1-2 *)
    RELAY_OK :      BOOL := TRUE ;  (* Relay is energized and not stuck. *)
END_VAR
VAR_OUTPUT
    CO :            BOOL ;          (* Critical IO Control out. *)
    TMR :           BOOL ;          (* Critical IO 3 channels operating. *)
    GE_DUAL :       BOOL ;          (* Critical IO 2 or more channels operating. *)
    GE_SINGLE :     BOOL ;          (* Critical IO 1 or more channels operating. *)
    NO_VOTER_FLTS : BOOL ;          (* No voter faults on critical modules. *)
    ERROR :         DINT ;          (* Error number. *)
    (*
     * Error number:
     *  0 = No error.
     * -1 = IOP invalid.
     * -2 = Slot not valid.
     * -3 = Module not configured.
     * -4 = Wrong module type.
     * -5 = Application number is invalid.
     * -6 = Not initialized.
     *)
END_VAR
VAR
    PREVIOUS_RESET : BOOL ;          (* All channels operating. *)
    IO :             SYS_IO_STATUS ; (* IO module status. *)
    RELAY :          DINT := 1 ;     (* De-energized to trip with relay *)
    DE_ENERGIZED :   DINT := 2 ;     (* De-energized to trip with no relay *)
    U :              BOOL ;          (* Unused value. *)
END_VAR

(*
 *=F============================================================================
 * FUNCTION_BLOCK:  SYS_CRITICAL_IO
 * Purpose:         Calculate status of critical IO modules.
 *
 * Return:          none
 *
 * Remarks:
 * Usage
 * 1. Invoke once with RESET := TRUE, to initialize.
 * 2. Invoke again with RESET := FALSE, CI := TRUE, APP := DE_ENERGIZED, and
 *    RELAY_OK := FALSE to complete initialization.
 * 3. Invoke repeatedly, once for each critical IO module.
 * 4. Read outputs CO, TMR, GE_DUAL, and GE_SINGLE for safety critical results.
 *
 * In step 3, invoke with the IOP and SLOT of the critical IO module,
 * the module application, and the relay status.
 * For example, if IOP 1 SLOT 1 is a critical DO module with a relay,
 * and SCIO is the function block instance name:
 *  SCIO( IOP:=1, SLOT:=  1, APP:=RELAY,        RELAY_OK:=RELAY1_OK );
 *
```

```
* Application
* The APP parameter for a module selects the effect of a fault
* on the vote mode outputs of the shutdown function blocks.
*    APP:=RELAY with RELAY_OK:=true
*      A sinlge fault (even a voter fault) degrades the mode to DUAL.
*      The relay provides a third channel for shutdown,
*      so if an output voter fails, there are still
*      two independent channels that can de-energize the output,
*      i.e., the relay and the other output voter channel.
*    APP:=RELAY with RELAY_OK:=false, or
*    APP:=DE_ENERGIZED
*      A voter fault degrades the mode to SINGLE.
*      A non-voter fault degrades the mode to DUAL.
*
* Runtime Errors
*      EBADPARAM                       Bad parameter
*      CO=FALSE indicates a programming error.
*      See ERROR number parameter for details.
*=F=============================================================================
*)

   IF RESET THEN
       CO := TRUE ;
       TMR := TRUE ;
       GE_DUAL := TRUE ;
       GE_SINGLE := TRUE ;
       NO_VOTER_FLTS := TRUE ;
   ELSIF PREVIOUS_RESET THEN
       ;                                                          (* No operation. *)
   ELSIF CI AND CO THEN
       IO( CI := CI, IOP := IOP, SLOT := SLOT );
       IF NOT IO.CO THEN
           ERROR := IO.ERROR_NUM ;
           U :=     ReportBadParam(0) ;
           CO :=    FALSE ;
       END_IF ;
       IF CO THEN
           TMR := TMR AND IO.TMR ;
           GE_DUAL := GE_DUAL AND IO.GE_DUAL ;
           GE_SINGLE := GE_SINGLE AND IO.GE_SINGLE ;
           NO_VOTER_FLTS := NO_VOTER_FLTS AND IO.NO_VOTER_FLTS ;
           IF APP = RELAY AND RELAY_OK THEN
               TMR := TMR AND IO.NO_VOTER_FLTS ;
           ELSIF APP = DE_ENERGIZED OR APP = RELAY AND NOT RELAY_OK THEN
               TMR := TMR AND IO.NO_VOTER_FLTS ;
               GE_DUAL := GE_DUAL AND IO.NO_VOTER_FLTS ;
           ELSE
               ERROR := -5 ;                  (* Application number is invalid *)
               U :=     ReportBadParam(0) ;
               CO :=    FALSE ;
           END_IF ;
       END_IF ;
   END_IF ;
   IF ERROR = 0 AND NOT CO THEN
       ERROR := -6 ;                                          (* Not initialized *)
       U :=     ReportBadParam(0) ;
   END_IF ;
   IF NOT CO THEN
       TMR :=           FALSE ;
       GE_DUAL :=       FALSE ;
       GE_SINGLE :=     FALSE ;
       NO_VOTER_FLTS := FALSE ;
```

```
    END_IF ;
    PREVIOUS_RESET :=    RESET ;

END_FUNCTION_BLOCK
```

SYS_SHUTDOWN

Enables a system shutdown according to industry guidelines.

Syntax

```
MY_SYS_SHUTDOWN( CI:=b1, IO_CO:=b2, IO_TMR:=b3, IO_GE_DUAL:=b4,
IO_GE_SINGLE:=b5, IO_NO_VOTER_FAULTS:=b6, MAX_TIME_DUAL:=t1,
MAX_TIME_SINGLE:=t2, MAX_SCAN_TIME:=t3 ) ;
```

Input Parameters

Name	Data Type	Description
CI	BOOL	Enables SYS_SHUTDOWN.
IO_CO	BOOL	True if SYS_SHUTDOWN executes successfully.
IO_TMR	BOOL	Three channels are operating without faults on every critical I/O module.
IO_GE_DUAL	BOOL	At least two channels are operating without faults on every critical I/O module.
IO_GE_SINGLE	BOOL	At least one channel is operating without faults on every critical I/O module.
IO_NO_VOTER_FLTS	BOOL	No voter faults on critical modules detected.
IO_ERROR	DINT	Error number, not used.
MAX_TIME_DUAL	TIME	The maximum time of continuous operation permitted with two channels operating.
MAX_TIME_SINGLE	TIME	The maximum time of continuous operation permitted with one channel operating.
MAX_SCAN_TIME	TIME	50% of the maximum response time.

Output Parameters

Name	Data Type	Description
CO	BOOL	True if SYS_SHUTDOWN executes successfully.
OPERATING	BOOL	Shutdown if OPERATING is false.
TMR	BOOL	Three channels are operating without fatal faults detected.
DUAL	BOOL	At least two channels are operating without fatal faults detected.
SINGL	BOOL	At least one channel is operating without fatal faults detected.
ZERO	BOOL	No channels are operating.
TIMER_RUNNING	BOOL	Shutdown timer is running

Output Parameters *(continued)*

Name	Data Type	Description
TIME_LEFT	TIME	Time remaining to shutdown
ALARM_PROGRAMMING_PERMITTED	BOOL	True if application changes are permitted
ALARM_REMOTE_ACCESS	BOOL	True if remote-host writes are enabled
ALARM_RESPONSE_TIME	BOOL	True if actual scan time ≥ MAX_SCAN_TIME
ALARM_DISABLED_POINTS	BOOL	True if one or more points are disabled
ERROR	DINT	Error Number: 0 = No error. 1 = Error in maximum time. 2 = IO function block error - IO_ERROR is non-zero. 3 = System status or MP status function block error.

Description

The SYS_SHUTDOWN function block enables a system shutdown according to industry guidelines.

Example

For shutdown examples, see this sample project:

My Documents\Triconex\TriStation 1131 4.x\Projects\TdTUV.pt2

Runtime Errors

Condition	Return Value	Error Flags
If ERROR is non-zero	Set alarm outputs to true, reset the other BOOL outputs to false, and reset TIME_LEFT to zero.	BADPARAM, ERROR

Upon detection of a runtime error condition, the function block returns the indicated values and sets the error flags to true. For more information about error flags and runtime errors, see the *TriStation 1131 Libraries Reference*.

If a programming error or configuration error occurs, then CO is false and the error number is non-zero. For error numbers, see the description of the ERROR output.

Application Notes

- Can be used in Safety or Control applications.

Library

Trident and Tri-GP (TRDLIB)

Structured Text

```
FUNCTION_BLOCK SYS_SHUTDOWN
VAR_INPUT
  CI :               BOOL := TRUE ; (* Control in. *)
  IO_CO :            BOOL ;         (* Critical IO Control out. *)
  IO_TMR :           BOOL ;         (* Critical IO 3 channels operating. *)
  IO_GE_DUAL :       BOOL ;         (* Critical IO 2 or more channels operating. *)
  IO_GE_SINGLE :     BOOL ;         (* Critical IO 1 or more channels operating. *)
  IO_NO_VOTER_FLTS : BOOL ;         (* No voter faults on critical modules. *)
  IO_ERROR :         DINT ;         (* Error number, not used. *)
  MAX_TIME_DUAL :    TIME := T#40000d ;  (* Max Time with only 2 channels. *)
  MAX_TIME_SINGLE :  TIME := T#40000d ;  (* Max Time with only 1 channel. *)
  MAX_SCAN_TIME :    TIME := T#400ms ;   (* 50% of Max Response Time. *)
END_VAR
VAR_OUTPUT
  CO :                           BOOL ;   (* Control out. *)
  OPERATING :                    BOOL ;   (* Shutdown if OPERATING=FALSE. *)
  TMR :                          BOOL ;   (* Three channels operating. *)
  DUAL :                         BOOL ;   (* Dual mode. *)
  SINGL :                        BOOL ;   (* Single mode. *)
  ZERO :                         BOOL ;   (* Zero mode. *)
  TIMER_RUNNING :                BOOL ;   (* Shutdown timer is running. *)
  TIME_LEFT :                    TIME ;   (* Time remaining to shutdown. *)
  ALARM_PROGRAMMING_PERMITTED : BOOL ;   (* Alarm -- download change. *)
  ALARM_REMOTE_ACCESS :          BOOL ;   (* Alarm -- remote host writes. *)
  ALARM_RESPONSE_TIME :          BOOL ;   (* Alarm -- exceed response time. *)
  ALARM_DISABLED_POINTS :        BOOL ;   (* Alarm -- some points disabled. *)
  ERROR :                        DINT ;   (* Error number. *)
  (*
   * Error number:
   * 0 = No error.
   * 1 = Error in maximum time.
   * 2 = IO function block error - IO_ERROR is non-zero.
   * 3 = System status or MP status function block error.
   *)
END_VAR
VAR
   GE_DUAL :       BOOL ;                (* Two or more channels operating. *)
   GE_SINGLE :     BOOL ;                (* One or more channels operating. *)
   SYSTEM :        SYS_SYSTEM_STATUS ;   (* System status. *)
   MP :            SYS_MP_STATUS ;       (* MP status. *)
   MPX :           SYS_MP_EXT_STATUS ;   (* MP extended status. *)
   DUAL_TIME :     TON ;                 (* Dual mode timer. *)
   SINGLE_TIME :   TON ;                 (* Single mode timer. *)
   U           :   BOOL ;                (* Unused Value. *)
END_VAR

(*
 *=F============================================================================
 * FUNCTION_BLOCK:  SYS_SHUTDOWN
 * Purpose:         Implement TUV restrictions.
 *
 * Return:          none
 *
 * Remarks:
 *
 * This example shows one way to check that
```

```
* the safety system is operating within spec when
* every module in the safety system is safety critical.
* The example uses the Trident and Tri-GP Library function block
* SYS_SHUTDOWN - one instance named CRITICAL_MODULES.
* The output CRITICAL_MODULES.OPERATING indicates
* that all safety critical modules are operating
* within spec. Input MAX_TIME_DUAL specifies the
* maximum time allowed with two channels operating
* (for example, the default is 40000 days).
* Input MAX_TIME_SINGLE specifies the maximum time allowed
* with only one channel operating (for example, 3 days).
* When CRITICAL_MODULES.OPERATING is FALSE,
* the time in degraded operation exceeds the
* specified limits -- therefore the control program
* should shutdown the plant.
*
* Excluding output voter faults and field faults -- TMR implies
* three channels with no detected fatal errors, GE_DUAL implies
* at least two channels with no detected fatal errors,
* and GE_SINGLE implies at least one channel
* with no detected fatal errors -- for every path
* from a safety critical input to a safety critical output.
* Detected output voter faults reduce TMR or GE_DUAL to GE_SINGLE.
* (See example EX02_SHUTDOWN to improve availability
* using relays and advanced programming techniques.)
*
* The "TMR" output indicates TMR.
* The "DUAL" output indicates GE_DUAL but not TMR.
* The "SINGL" output indicates GE_SINGLE but not GE_DUAL.
* The "ZERO" output indicates not GE_SINGLE.
* The "TIMER_RUNNING" output indicates that
* the time left to shutdown is decrementing.
* The "TIME_LEFT" output indicates the time remaining before shutdown.
*
* WARNING - the SYS_SHUTDOWN function block
* does not use detected field faults or
* combinations of faults reported as field faults.
* It is the responsibility of the application
* to use system variable NoFieldFault or FieldOK
* to detect and respond to such faults.
*
* To see how to create a user-defined function block
* to improve availability, see the examples
* in the help topic for SYS_SHUTDOWN.
*
* Runtime Errors
*      EBADPARAM                       Bad parameter
*      CO=FALSE indicates a programming error.
*      See ERROR number parameter for details.
*=F=============================================================================
*)

   IF CI THEN
                                                                  (* Get Status *)
       SYSTEM( CI := TRUE ) ;
       MP( CI := TRUE ) ;
       MPX( CI := TRUE ) ;
                                              (* Check for programming errors. *)
       ERROR := 0 ;
       IF
           MAX_TIME_DUAL < MAX_TIME_SINGLE OR
           MAX_TIME_DUAL < T#0S OR
```

```
    MAX_TIME_SINGLE < T#0S OR
    MAX_SCAN_TIME < T#0S
THEN
    ERROR := 1 ;
ELSIF IO_ERROR <> 0 THEN
    ERROR := 2 ;
ELSIF NOT SYSTEM.CO OR NOT MP.CO OR NOT MPX.CO THEN
    ERROR := 3 ;
END_IF ;

CO := ERROR = 0 ;

IF CO THEN
                                                    (* Summarize redundancy. *)
    TMR :=
        NOT IO_CO AND SYSTEM.TMR AND SYSTEM.IO_NO_VOTER_FLTS
        OR  IO_CO AND MP.TMR AND IO_TMR
    ;

    GE_DUAL :=
        NOT IO_CO AND SYSTEM.GE_DUAL AND SYSTEM.IO_NO_VOTER_FLTS
        OR  IO_CO AND MP.GE_DUAL AND IO_GE_DUAL
    ;

    GE_SINGLE :=
        NOT IO_CO AND SYSTEM.GE_SINGLE
        OR  IO_CO AND IO_GE_SINGLE
    ;

                                                        (* Update timers. *)
    DUAL_TIME( IN := NOT TMR, PT := MAX_TIME_DUAL ) ;
    SINGLE_TIME( IN := NOT GE_DUAL, PT := MAX_TIME_SINGLE ) ;


                    (* Shutdown if excessive time in degraded operation. *)
    OPERATING :=
            GE_SINGLE
        AND NOT DUAL_TIME.Q
        AND NOT SINGLE_TIME.Q
    ;

                                                    (* Output current status. *)
    DUAL :=            GE_DUAL AND NOT TMR ;
    SINGL :=           GE_SINGLE AND NOT GE_DUAL ;
    ZERO :=            NOT GE_SINGLE ;
    TIMER_RUNNING :=  OPERATING AND NOT TMR ;

                                    (* Output time remaining to shutdown. *)
    IF NOT OPERATING THEN
        TIME_LEFT := T#0s ;
    ELSIF TMR THEN
        TIME_LEFT := T#999999d ;
 ELSIF GE_DUAL OR MAX_TIME_DUAL-DUAL_TIME.ET <= MAX_TIME_SINGLE-SINGLE_TIME.ET THEN
        TIME_LEFT := MAX_TIME_DUAL - DUAL_TIME.ET ;
    ELSE
        TIME_LEFT := MAX_TIME_SINGLE - SINGLE_TIME.ET ;
    END_IF ;

                                                        (* Output alarms. *)
    ALARM_PROGRAMMING_PERMITTED := NOT MP.APP_LOCKED ;
    ALARM_REMOTE_ACCESS :=   MP.REMOTE_WRT_ENBL ;
    ALARM_RESPONSE_TIME :=   T#1ms * MPX.ACTUAL_SCAN_TIME > MAX_SCAN_TIME ;
```

```
            ALARM_DISABLED_POINTS := MPX.POINTS_DISABLED > 0 ;

        ELSE
                                                        (* Programming error. *)
            U :=                            ReportBadParam(0) ;

            OPERATING :=                    FALSE ;
            TMR :=                          FALSE ;
            GE_DUAL :=                      FALSE ;
            GE_SINGLE :=                    FALSE ;
            DUAL :=                         FALSE ;
            SINGL :=                        FALSE ;
            ZERO :=                         FALSE ;
            TIMER_RUNNING :=                FALSE ;
            TIME_LEFT :=                    T#0S;

            ALARM_PROGRAMMING_PERMITTED :=  TRUE ;
            ALARM_REMOTE_ACCESS :=          TRUE ;
            ALARM_RESPONSE_TIME :=          TRUE ;
            ALARM_DISABLED_POINTS :=        TRUE ;
        END_IF ;

    END_IF ;

END_FUNCTION_BLOCK
```

SYS_VOTE_MODE

Converts redundancy status.

Syntax

```
MY_SYS_VOTE_MODE( CI:=b1, IN_TMR:=b2, GE_DUAL:=b3, GE_SINGLE:=b4 ) ;
```

Input Parameters

Name	Data Type	Description
CI	BOOL	Enables SYS_VOTE_MODE.
IN_TMR	BOOL	Three channels are operating.
GE_DUAL	BOOL	Two or more channels are operating.
GE_SINGLE	BOOL	One or more channels are operating.

Output Parameters

Name	Data Type	Description
CO	BOOL	True if SYS_VOTE_MODE executes successfully.
TMR	BOOL	Three channels are operating.
DUAL	BOOL	Two channels are operating.
SINGLE	BOOL	One or more channels are operating.
ZERO	BOOL	No channel is operating.

Description

The SYS_VOTE_MODE function block converts redundancy status, as shown in this truth table.

Truth Table

TMR	GE_DUAL	GE_SINGLE	TMR	DUAL	SINGL	ZERO
T	T	T	T	F	F	F
F	T	T	F	T	F	F
F	F	T	F	F	T	F
F	F	F	F	F	F	T
Other[a]			F	F	F	F

a. If an error in the inputs occurs, then CO is false, the mode outputs are false, and the function block reports a bad parameter error (BADPARAM).

Note GE_ means greater than or equal to.

Example

For shutdown examples, see this sample project:

My Documents\Triconex\TriStation 1131 4.x\Projects\TdTUV.pt2

Runtime Errors

Condition	Return Value	Error Flags
If the inputs do not match one of the first four rows of the truth table	Reset all BOOL outputs to false	BADPARAM, ERROR

Upon detection of a runtime error condition, the function block returns the indicated values and sets the error flags to true. For more information about error flags and runtime errors, see the *TriStation 1131 Libraries Reference*.

Application Notes

- Can be used in Safety or Control applications.
- Space Saver: a single instance can be executed more than once per scan to reduce memory usage and increase performance. See the *TriStation1131 Libraries Reference* for more information.

Library

Trident and Tri-GP (TRDLIB)

Structured Text

```
FUNCTION_BLOCK SYS_VOTE_MODE
VAR_INPUT
    CI :          BOOL := TRUE ;    (* Control in. *)
    IN_TMR :      BOOL ;            (* 3 channels operating. *)
    GE_DUAL :     BOOL ;            (* 2 or more channels operating. *)
    GE_SINGLE :   BOOL ;            (* 1 or more channels operating. *)
END_VAR
VAR_OUTPUT
    CO :          BOOL ;            (* Control out. *)
    TMR :         BOOL ;            (* Triple Modular Redundant. *)
    DUAL :        BOOL ;            (* Dual mode. *)
    SINGL :       BOOL ;            (* Single mode. *)
    ZERO :        BOOL ;            (* Zero mode. *)
END_VAR
VAR
    U :           BOOL ;            (* Unused Value. *)
END_VAR

(*
 *=F============================================================================
 * FUNCTION_BLOCK:  SYS_VOTE_MODE
 * Purpose:         Convert redundancy status.
 *
 * Return:          none
 *
 * Remarks:
```

```
 * 1. Convert redundancy status (TMR, GE_DUAL, GE_SINGLE) to (TMR, DUAL, SINGL, ZERO).
 * 2. "GE_" denotes "greater than or equal to".
 * 3. CO is true if CI is true and there is no programming error.
 *
 * Runtime Errors
 *      EBADPARAM  Bad parameter
 *      CO=        FALSE indicates a programming error if CI=true.
 *      The outputs are all FALSE if there is a programming error.
 *=F=============================================================================
 *)

       CO := CI ;
    IF CI THEN
       CO :=      GE_DUAL AND GE_SINGLE OR NOT GE_DUAL AND NOT IN_TMR;
    IF CO THEN
        TMR   :=  IN_TMR ;
        DUAL  :=  GE_DUAL AND NOT IN_TMR ;
        SINGL :=  GE_SINGLE AND NOT GE_DUAL ;
        ZERO  :=  NOT GE_SINGLE ;
    ELSE
        U     :=  ReportBadParam(0) ;
        TMR   :=  FALSE ;
        DUAL  :=  FALSE ;
        SINGL :=  FALSE ;
        ZERO  :=  FALSE ;
    END_IF ;
END_IF ;
END_FUNCTION_BLOCK
```

Index

N

O

P

R

S

T

V

Invensys Operations Management
5601 Granite Parkway, Suite 1000
Plano, TX 75024
United States of America
http://www.iom.invensys.com

Global Customer Support
Inside U.S.: 1-866-746-6477
Outside U.S.: 1-508-549-2424 or contact your local Invensys representative.
Email: iom.support@invensys.com
Website: http://support.ips.invensys.com