Easy-to-use developer's workbench allow you to develop, test and document process-control applications for the Tricon Controller.

TriStation 1131 Developer's Workbench

TriStation 1131 Developer's Workbench is an integrated tool for developing, testing, and documenting safety and critical-process control applications for the Tricon controller. The programming methodology, user interface and self-documentation capabilities make the system superior to traditional and competing engineering tools.

This table identifies the compatibility of Tricon and TriStation 1131 versions.

TriStation 1131	Tricon
1.0.*x*	9.1.*x*–9.2.*x*
1.1.*x*	9.3.*x*
2.0.x	9.4.*x*–9.5.*x*
3.0.*x*	9.5–9.5.*x*
3.1.*x*	9.5–9.10
4.0.x	9.5.2–9.10
4.1.419–4.1.420	9.5.2–10.0.*x*
4.1.433	9.5.2–10.1.*x*
4.1.437	9.5.2–10.2.*x*
4.2.*x*	9.5.2–10.3.*x*
4.3.*x*–4.5.*x*	9.5.2–10.4.*x*

TriStation 1131 is compliant with Part 3 of the IEC 61131 International Standard for Programmable Controllers, which defines programming languages.

TriStation 1131 v4.1 and later will run on Windows 2000 and Windows XP. TriStation 1131 v4.0 will run only on Windows NT and Windows 2000; however, it has only been validated to run on Windows 2000. Versions of TriStation 1131 lower than 4.0 will run only on Windows NT.

Functional Overview

TriStation 1131 provides three editors which support these IEC 61131-3 languages:

- Function Block Diagram
- Ladder Diagram
- Structured Text

An optional Triconex programming language, CEMPLE (Cause and Effect Matrix Programming Language Editor) supports the widely used Cause and Effect Matrix (CEM) methodology.

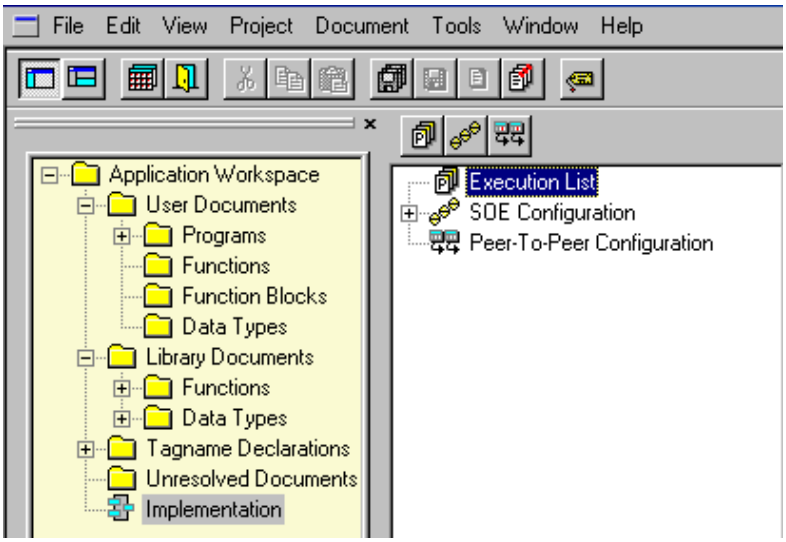
TriStation 1131 allows you to:

- Create programs, functions, and function blocks
- Define the controller configuration
- Declare tagnames
- Test applications in an emulator
- Download and monitor applications

New Features in TriStation 1131 v4.2–v4.5

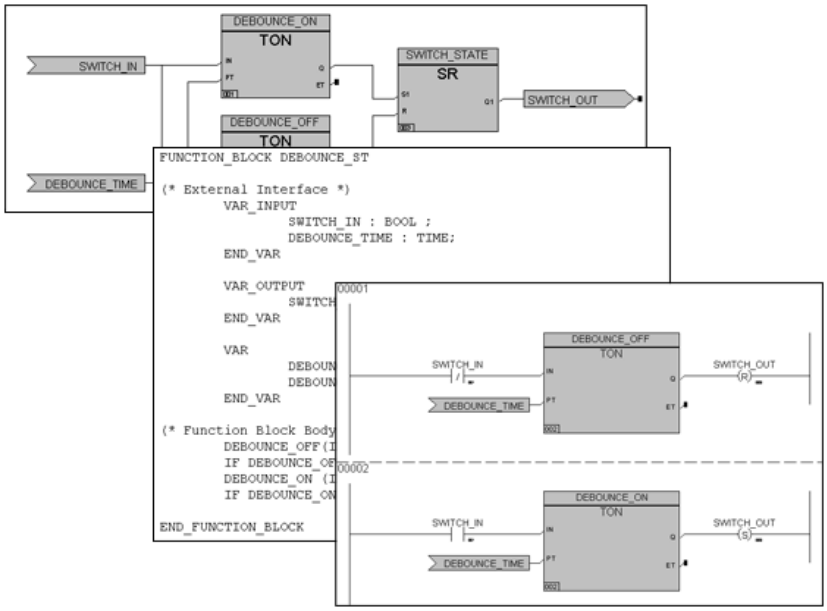
These are new features in TriStation 1131:

- Support for the Tricon Communication Module (TCM) with Embedded OPC Server (Models 4353 and 4354). (v4.2)
- Introduction of the t*arget system version* concept, which controls which Tricon features and modules can be configured in a project. (v4.2)
- Configuration of TSAA IP multicasting. (v4.2)
- Ability to export TCM configuration settings to an XML file, so that those settings can be imported into other TCMs, ensuring that all TCMs in a system have the same configuration. (v4.2)



Example of TriStation 1131 Interface

- Added support for the Tricon Model 3807 BiPolar Analog Output (BPAO) module. (v4.3)
- Added support for the Enhanced Low-Density Expansion Chassis (E_LD_EXP), with HART capability, in Tricon v10.4 and later systems. (v4.4)
- Ability to display lists of undeclared tagnames, and tagnames that are declared but not used in the project. (v4.5)
- Support for Windows Server 2003. (v4.5)
- Support for user-defined data types in local variables. (v4.5)



Sample Logic in FBD, ST and LD Languages

Enhanced Diagnostic Monitor

The Enhanced Diagnostic Monitor is an application which monitors the hardware health of Triconex controllers and allows users to effectively troubleshoot the safety system during maintenance.

Starting with TriStation 1131 v4.1.437, the Enhanced Diagnostic Monitor is a separate application from TriStation 1131.

For more information on the Enhanced Diagnostic Monitor, see the online Help or printed guide included with the Enhanced Diagnostic Monitor.

Elements of a TriStation 1131 Project

A TriStation 1131 project contains all of the elements required to implement a safety or control program in a Tricon controller. Some of these elements are automatically included in every project by TriStation 1131, while others are user-created.

Programs

A program is the highest-level executable logic element in a TriStation 1131 project. It is an assembly of programming language elements (functions, function blocks, and data variables) that work together to allow a programmable control system to achieve control of a machine or a process. Each program is uniquely identified by a user-defined type name. A TriStation 1131 project can support hundreds of programs.

Functions

A function is a logic element which yields exactly one result. Unlike a function block, the data associated with a function is not retained from one evaluation of the function to the next. Functions do not have to be instanced.

Function Blocks

A function block is a logic element which yields one or more results. To use a function block in a program, an instance of the function block type must first be declared. Each instance is identified by a user-defined instance name. All of the data associated with a specific instance of a function block is retained from one evaluation of the function block to the next.

Data Types

A data type defines the size and characteristics of variables declared in a program, function or function block. Data types used by TriStation 1131 include discrete (BOOL), analog (DINT), and real (REAL).

Libraries

TriStation 1131 includes libraries of pre-defined functions, function blocks, and data types that can be used in a project.

TriStation 1131 includes these libraries:

- IEC 61131-3 Standard Library – a set of functions and function blocks defined by the IEC 61131-3 Standard
- Triconex Library – a set of Triconex functions and function blocks that can be used with any Triconex programmable controller
- Tricon Library – a set of functions and function blocks that are specifically for use with the Tricon controller

In addition to the pre-defined libraries, you can also develop your own libraries of project elements. These libraries can include programs, functions, function blocks, and data types which can be imported to other TriStation 1131 projects.

Programming Languages

TriStation 1131 includes these programming languages: Function Block Diagram, Structured Text, and Ladder Diagram. An optional language, CEMPLE, can be purchased separately.

Function Block Diagram (FBD)

Function Block Diagram is a graphical language that corresponds to circuit diagrams. FBD elements appear as blocks that are wired together to form circuits. The wires transfer binary and other types of data between elements.

Structured Text (ST)

Structure Text is a high-level, textual programming language that is similar to PASCAL. Structured Text allows Boolean and arithmetic expressions, and programming structures such as conditional (IF…THEN…ELSE) statements. Functions and function blocks can be invoked in Structured Text.

			Effect	OR	OR	OR	OR	OR
			Description	High level alarm indicator for tank 1	High level alarm indicator for tank 2	High level alarm indicator for tank 3	High level alarm indicator for tank 4	High level alarm indicator for tank 5
	Cause	Description		E01	E02	E03	E04	E05
	LEVEL_1_	TRUE=Fluid level in tank 1 is high	C01	X				
	LEVEL_2_HI	TRUE=Fluid level in tank 2 is high	C02		X			
	LEVEL_3_HI	TRUE=Fluid level in tank 3 is high	C03			X		
	LEVEL_4_HI	TRUE=Fluid level in tank 4 is high	C04				X	

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

Sample CEM from a TriStation 1131 Project

In TriStation 1131 v4.0, these structures were added: arrays, structures, ForLoop and Exit statements, CASE statement, enumerated data types, var-external, and var-temp variables.

Ladder Diagram (LD)

Ladder Diagram is a graphical language that uses a standard set of symbols for representing relay logic. The basic elements are coils and contacts which are connected by links. Links are different from the wires in FBD in that they transfer only binary data between the elements.

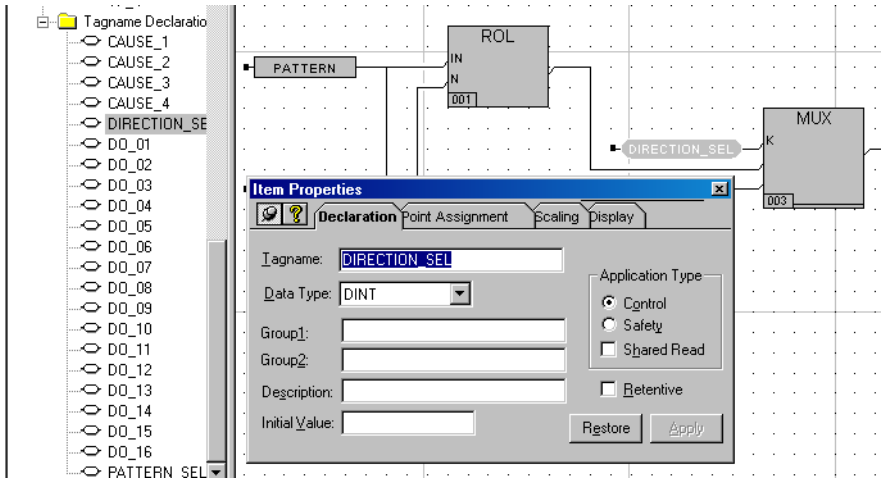
Cause and Effect Matrix Programming Language Editor (CEMPLE)

CEMPLE is a high-level graphical language that provides a two-dimensional matrix in which you can associate a problem in a process with one or more corrective actions. The problem is referred to as the cause and the action as the effect. The matrix associates a cause with an effect in the intersection of the cause row and the effect column.

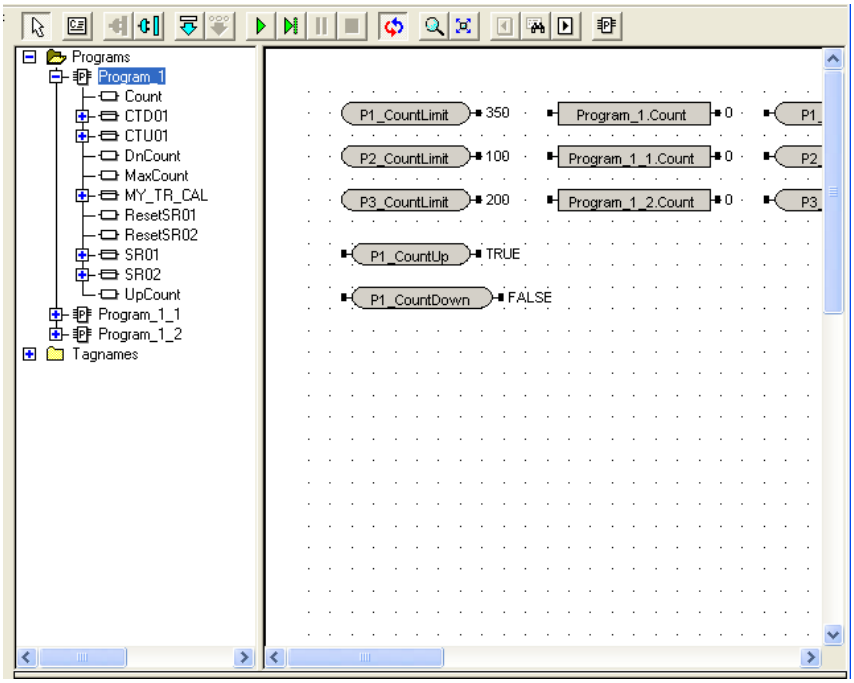
CEMPLE is the first automated implementation of CEM, a methodology that is commonly used throughout the process-control industry and readily understood by a broad range of plant personnel. CEM diagrams are automatically translated into IEC 61131-3 compliant Function Block Diagrams, thereby eliminating the risks associated with manual translation from hand-drawn CEMs.

Controller Configuration

In TriStation 1131, the controller configuration identifies the modules in the system, communication settings, memory allocation for tagnames, and operating parameters. These configuration settings are included in the control



Declaring Tagnames in a Program



Emulator Panel

program that is downloaded to the controller.

Emulator Panel

The Emulator Panel allows you to connect to an emulator, download the control program, and test and debug the control program. The panel lists the programs, variables, and tagnames in the control program. Testing can be done by dragging variables and tagnames from the list to the monitor panel and changing the values as desired. You can specify commands to run the control program without intervention, to run in single-step, or to halt the execution.

Controller Panel

The Controller Panel allows connection to the controller for real-time execution of the control program.

Diagnostic Panel

In TriStation 1131 versions prior to v4, the Diagnostic Panel allows you to monitor the status of MP, CM, and I/O modules in the controller and to diagnose faults. The panel also provides system performance information including the project name and version, memory size, scan time and current execution state.

In TriStation 1131 v4.1, the Diagnostic Monitor is a separate interface that enables monitoring of applications and hardware status on multiple controllers.

Starting with TriStation 1131 v4.1.435, the diagnostic functions from the Diagnostic Panel and the Diagnostic Monitor are included in the Enhanced Diagnostic Monitor, which is a separate application from TriStation 1131.

TriStation 1131 Interface Options

TriStation 1131 allows you to specify options to be used in the interface. For example, you can specify the drawing colors used in the programming editors, and editor options such as double-spacing between function block terminals. You can also specify the directory location for files.

Reports and Documentation

TriStation 1131 includes multiple methods of sorting data and documenting project elements, both during and after project development. Printouts of user-developed function blocks and programs can be obtained on a variety of user-selected engineering drawing templates.

Standard reports are available to document the project configuration data. You can also create customized reports with Crystal Reports™.

Password Security

TriStation 1131 provides a security system that defines users and their privileges with regard to editing, library changes, state changes and other operations.

Project History

An audit trail function is provided to document the history of a project and its program version changes. This detailed log keeps track of user actions and comments by automatically time-stamping critical events within a session and manually logging user comments on demand.

Annotations

Annotations can be added to constants, tagnames, and variables An annotation can be used to display descriptive text, including information specified in system and user-modifiable macros. You can also display the value of a variable during program execution.

Comments

Comments can be added to programs, functions, and function blocks to add information about the operations.

Help Documentation

TriStation 1131 features an online Help system which provides detailed information about TriStation 1131.

CEMPLE is the Triconex automated implementation of the traditional CEM methodology that has been used by process control engineers for decades.

CEM Programming Language Editor

Cause and Effect Matrix (CEM) is a methodology that is commonly used in the process control industry to define alarms, emergency shutdown strategies, and mitigation actions. For decades, process control engineers have used manual methods such as graph paper and spreadsheet programs to identify problem conditions and corrective actions.

Automated CEM Called CEMPLE

The traditional CEM method is time-consuming and subject to errors caused by misinterpretation of the matrix or inaccurate coding. Triconex has automated the CEM process with the Cause and Effect Matrix Programming Language Editor, referred to as CEMPLE.

CEMPLE enables a cause and effect matrix to be used as the basis for a TriStation 1131 program.

CEMPLE Features

CEMPLE includes the following features:

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

CEM Editor

The CEM Editor includes the following components as shown in the figure below:

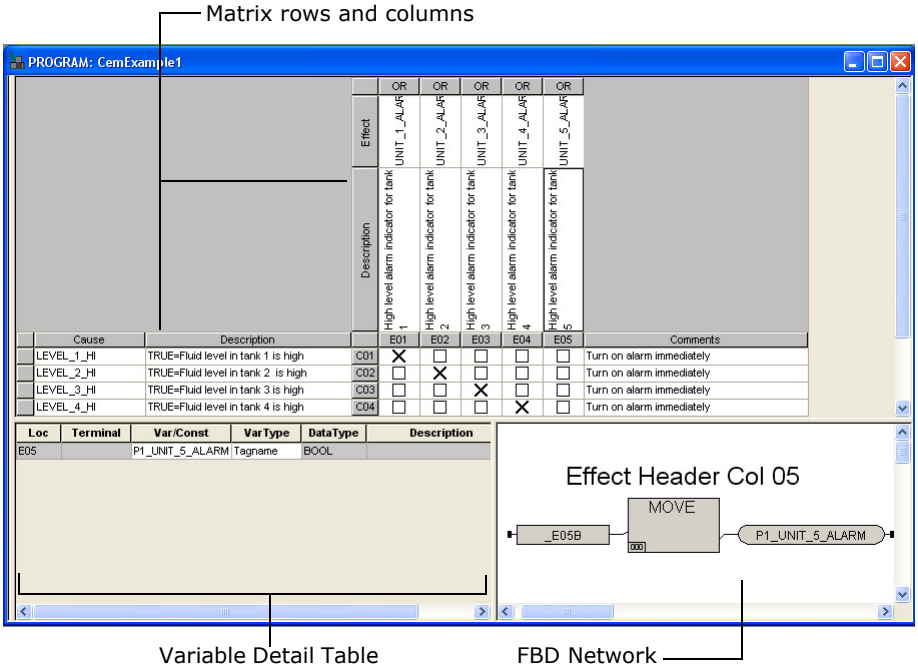
- Matrix
- FBD Network
- Variable Detail Table

Matrix

As the major component of the CEM Editor, the Matrix identifies the parts of associated with causes, effects, and intersections. The Matrix can also include functions or function blocks related to causes, effects, and intersections.

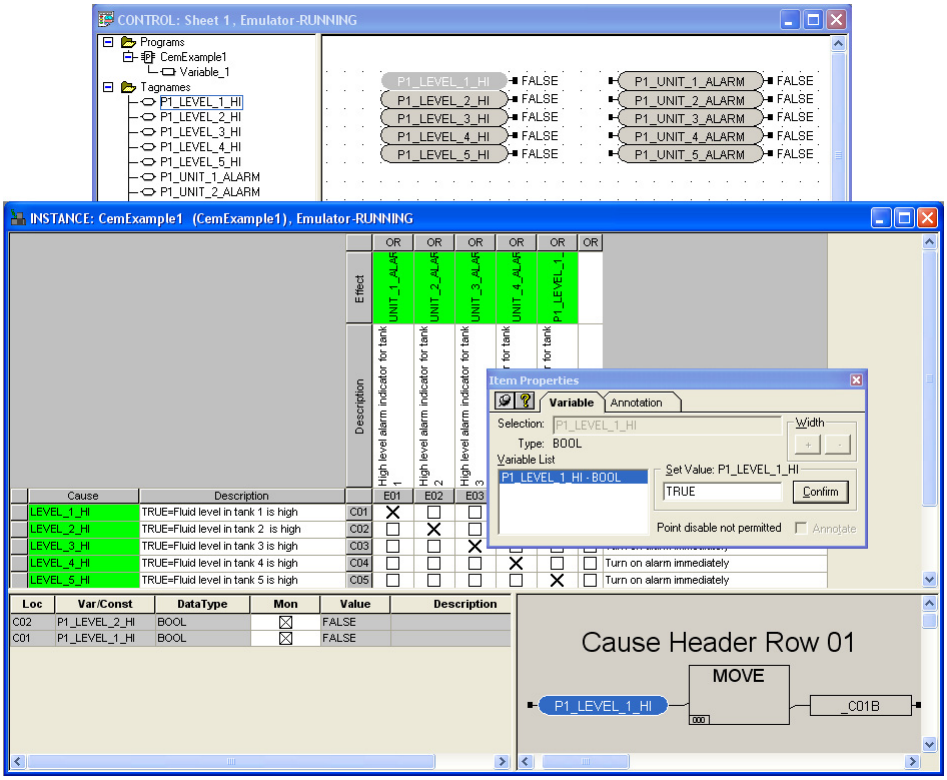
FBD Network

The FBD Network displays the Function Block Diagram (FBD) related to the cause, intersection, or effect that is selected in the matrix. It can also be used to specify properties and to invert values for variables.



CEM Editor Components

CEM Programming Language Editor



Instance View of a Matrix

The FBD network uses internal boolean variables to save and move results to associated cells so that causes and effects can be evaluated. For each cause, effect, and intersection, an internal variable is automatically created to store and move results between cells.

Variable Detail Table

The Variable Detail Table displays the inputs and outputs of the FBD Network that are generated when a cause, effect, or intersection is selected.

The variable type and data type can also be specified from the Variable Detail Table.

Developing a Matrix

A matrix created in CEMPLE can be as basic or complex as the situation requires. In a basic matrix, causes are identified as true or false inputs related to one or more effects through the intersections between them. The state of a cause (true or false) determines the state of the related effect. If more than one cause is related to an effect, the state of the effect is based on how the matrix is evaluated.

The effect state can be determined in either of two ways: by a logical AND operation or by a logical OR operation on the intersection. A logical AND is typically used for de-energize to trip systems; a logical OR is typically used for energize to trip systems.

Using Functions and Function Blocks

For more complex processes, CEMPLE enables functions and function blocks to be added to causes, effects, and intersections. This feature can be used for many purposes, such as; evaluation of process input to determine the cause state, calculating one or more process variable values based on the state of an effect, and using time delays.

User-created functions and function blocks, must be created and enabled for use before they can be included in a matrix.

Testing and Monitoring

Like all TriStation 1131 programs, a matrix can be tested and debugged off-line using the Emulator Control Panel. After the project is downloaded, the Control Panel can be used to monitor the values of variables during real-time execution.

In an instance view of a matrix, active causes, intersections, and effects can be viewed in a choice of colors.

As with other types of executable elements, values and variables can be set for use during emulation and real-time execution.

CEMPLE Tools

A matrix can be developed and edited using a variety of graphical interface methods. Commands can be selected from a main menu, toolbar, and pop-up menu.

Variables can be added or renamed by making changes in the Variable Detail Table. Where appropriate, drop-down lists provide variable names or function and function block names to be selected.

For more information, see the *TriStation 1131 Developer's Guide*.

During each scan of the control program, the Main Processors examine selected discrete variables for state changes known as events.

Sequence of Events (SOE) Capability

Triconex controllers and software include sequence of events capability which provides the potential to track events which lead to an unsafe process or system shutdown. During each scan of the control program, the Main Processors examine specified discrete variables for state changes known as events. Triconex software can be used to retrieve the event data from the controller.

SOE for TriStation 1131 is compatible with Tricon v9.5 and later systems.

The following host systems can be used to retrieve event data:

- SOE software from Triconex which runs on a PC
- Safety Manager Module (SMM) for Honeywell DCS systems
- Advanced Communication Module (ACM) for Foxboro DCS systems
- An OPC client control program which has implemented the Alarm and Events Handler as specified in the OPC standard version 1.0

With SOE software you can:

- Collect and analyze event data
- Export event database files
- Print reports with event data

The SOE data file, which is output from TriStation 1131, is only for use with Triconex SOE software. This file is read by the software and adds descriptive information which is associated with the tagname in the Configuration file in TriStation 1131.

Preparing Your System for Event Collection

To enable the controller to detect events, event variables and SOE blocks are identified in the TriStation1131 project. In addition, the project must include an SOE function block that starts the event collection.

After an SOE-enabled project is downloaded to the controller, TriStation 1131 creates an SOE definition file that contains the SOE block definitions.

When the SOE software collects an event from the controller, it obtains the tagname, alias, state name, and other information about the event variable from the SOE definition file.

These tasks are done in TriStation 1131:

- Defining SOE blocks with buffer size and block types
- Assigning event variables to the SOE blocks
- Adding SOE function blocks to the program logic

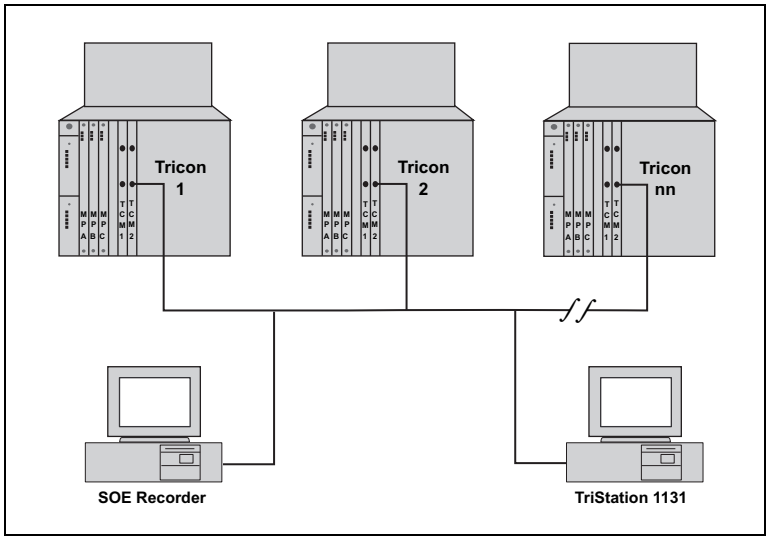
Types of Event Variables

The types of discrete variables that can be designated as event variables are:

- BOOL input
- BOOL aliased memory variables

Configuring SOE Blocks

An SOE block is a data structure that resides in the memory of a controller's Main Processors. When SOE blocks are configured, the event variables to be



Tricon Network with SOE PC

detected by the controller are specified for each block.

The maximum individual block size is 20,000 events, with 60,000 events for all blocks. The block size is the amount of memory that the Main Processors reserve for recording of events.

When a block is collecting events, the Main Processors write an event entry which includes the values of event variables that changed during the current scan and a time stamp.

SOE Function Blocks

SOE function blocks control and verify event collection for SOE blocks. The following function blocks are available:

- SOESTRT starts event collection
- SOESTOP stops event collection
- SOESTAT checks status of SOE blocks
- SOECLR clears status of SOE blocks

The SOESTRT function block must be added to the TriStation 1131 program to identify the SOE blocks from which events are to be collected. The other SOE function blocks are optional.

Date	Time	Alias	TagName	Vari...	Node	Block
08/24/2000	21:14:41.246	02001	EVENT_VAR1	ON	06 - TRINODE06	01 - soe_block_1
08/24/2000	21:14:41.246	02002	EVENT_VAR2	OFF	06 - TRINODE06	01 - soe_block_1
08/24/2000	21:14:41.246	02003	EVENT_VAR3	ON	06 - TRINODE06	01 - soe_block_1
08/24/2000	21:14:41.246	02004	EVENT_VAR4	OFF	06 - TRINODE06	01 - soe_block_1
08/24/2000	21:14:41.246	02005	EVENT_VAR5	ON	06 - TRINODE06	01 - soe_block_1
08/24/2000	21:14:41.246	02006	EVENT_VAR6	OFF	06 - TRINODE06	01 - soe_block_1
08/24/2000	21:14:41.246	02007	EVENT_VAR7	ON	06 - TRINODE06	01 - soe_block_1
08/24/2000	21:14:41.246	02008	EVENT_VAR8	OFF	06 - TRINODE06	01 - soe_block_1
08/24/2000	21:14:41.246	02009	EVENT_VAR9	OFF	06 - TRINODE06	01 - soe_block_1
08/24/2000	21:14:41.246	02010	EVENT_VAR10	ON	06 - TRINODE06	01 - soe_block_1
08/24/2000	21:14:41.246	02011	EVENT_VAR11	OFF	06 - TRINODE06	01 - soe_block_1
08/24/2000	21:14:41.246	02012	EVENT_VAR12	ON	06 - TRINODE06	01 - soe_block_1
08/24/2000	21:14:41.246	02013	EVENT_VAR13	OFF	06 - TRINODE06	01 - soe_block_1
08/24/2000	21:14:41.246	02014	EVENT_VAR14	ON	06 - TRINODE06	01 - soe_block_1
08/24/2000	21:14:41.246	02015	EVENT_VAR15	OFF	06 - TRINODE06	01 - soe_block_1
08/24/2000	21:14:41.246	02016	EVENT_VAR16	ON	06 - TRINODE06	01 - soe_block_1
08/24/2000	21:14:41.246	02017	EVENT_VAR17	OFF	06 - TRINODE06	01 - soe_block_1
08/24/2000	21:14:41.246	02018	EVENT_VAR18	ON	06 - TRINODE06	01 - soe_block_1
08/24/2000	21:14:41.246	02019	EVENT_VAR19	OFF	06 - TRINODE06	01 - soe_block_1

SOE Events File

SOE Software

SOE software can simultaneously collect event data from as many as 31 networked controllers. It queries all the controllers on the network to determine which downloaded TriStation 1131 projects include SOE blocks. If a project includes one or more SOE blocks, the software opens the appropriate SOE definition file and begins collecting events from the associated controller.

While the TriStation 1131 project is running, the SOE software can be used to analyze events online as it collects them from the controllers. Snapshots of events that cover specific periods of time before or after trips have occurred can also be saved.

To analyze the event data, the SOE software includes tools for these tasks:

- Finding events and copying them to other Windows applications
- Filtering and sorting saved event data
- Specifying the display of point properties for event data
- Viewing the properties of individual events

The SOE software also allows event data to be exported to database or ASCII text files, either manually or automatically. A report engine and standard report are included.

Trip Processing

A trip is a shutdown of the controlled process, or a portion of the controlled process. A TriStation 1131 project used for safety shutdown typically includes one trip variable, whose state change initiates the shutdown activities. If a project requires several variables related to trip conditions, these variables must be evaluated in combination to determine the final state of the trip variable. When a trip event occurs, the SOE software can automatically create a trip snapshot. This snapshot is a file of events that occurred *x* minutes before a trip and *y* minutes after a trip, based on TriStation 1131 settings.

Time Synchronization and Time Stamps

In a typical Peer-to-Peer network, the controllers synchronize their time with the master node (the controller with the lowest node number) within ±25 milliseconds. A controller recognizes events on a scan basis and time-stamps each event at the beginning of the scan.

Because the scans of the various controllers on the network are not synchronized, the same event can be logged by two controllers with different time stamps. The worst-case difference is the longer scan time plus 25 milliseconds.

Each day, the SOE software compares its clock with the clock of each controller from which event data is being collected. If a controller's clock is out of sync by more than five minutes, a message is displayed in the SOE message bar.