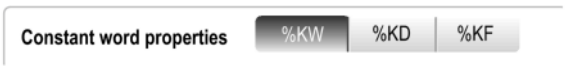
Constant Word Properties



First choose the constant word type to display properties for:

- **%KW** Constant words.
- **%KD** Double constant words.
- **%KF** Floating-point constant words.

This table describes each parameter of the **Constant words** screen:

Parameter	Editable	Value	Default Value	Description
Used	No	True/False	False	Indicates whether the constant word is currently being used in a program.
Equ Used	No	True/False	False	Equivalent used. Indicates whether part of the memory area of the constant word is currently being used. Refer to Possibility of Overlap Between Objects (see EcoStruxure Machine Expert - Basic, Generic Functions Library Guide).
Address	No	Refer to Word Objects (see EcoStruxure Machine Expert - Basic, Generic Functions Library Guide)	N/A	Displays the address of the constant word.
Symbol	Yes	A valid symbol	None	Allows you to associate a symbol with this constant word.
Decimal	Yes	Decimal representation of the value. Refer to Word Objects (see EcoStruxure Machine Expert - Basic, Generic Functions Library Guide)	0	The decimal value of this constant word.
Binary	Yes	Binary representation of the value. Refer to Word Objects (see EcoStruxure Machine Expert - Basic, Generic Functions Library Guide)	2#000000000000-0000	The binary value of this constant word.
Hexadecimal	Yes	Hexadecimal representation of the value. Refer to Word Objects (see EcoStruxure Machine Expert - Basic, Generic Functions Library Guide)	16#0000	The hexadecimal value of this constant word.
ASCII	Yes	ASCII representation of the value. Refer to Word Objects (see EcoStruxure Machine Expert - Basic, Generic Functions Library Guide)	**no meaning**	The ASCII value of this constant word.
Comment	Yes	A valid comment	None	Allows you to associate a comment with this constant word.

Exporting/Importing the Constant Word Properties

You can export into a *CSV* file and import the **Address**, **Symbol**, **Value**, and **Comment** properties in offline or online mode.

Exporting the constant word properties:

Step	Action
1	Select the **Tools** tab in the left-hand area of the **Programming** window.
2	Click **Memory objects** > **Constant words**.
3	In **Constant word properties**, click **Export**. **Result:** The **Export constants** window appears.
4	In the **Export constants** window: 1. Select the **Export type**. 2. Choose the **File path** by clicking ... . 3. Enter: • The **File name**, • The **First index** (numeric), • The **Last index** (numeric). The **First index** must be less than or equal to the **Last Index**.
5	To modify the export parameters, click **Export options**: 1. Select **Headers** if you want to display the name of the headers. 2. Choose **Semicolon** or **Comma** as separator.
6	Click **Export**.

Importing the constant word properties:

Step	Action
1	Select the **Tools** tab in the left-hand area of the **Programming** window.
2	Click **Memory objects** > **Constant words**.
3	In **Constant word properties**, click **Import**. **Result:** The window **Import constants** appears.
4	Click ... and navigate to the folder containing the file (*.csv) and double-click the file.
5	To modify the import parameters, click **Import options** and choose the separator used in the .csv file: **Semicolon** or **Comma**.
6	Click **Import**.

In case of duplicate values, the last duplicate value is imported.

System Objects

Overview

System objects are specific to the logic controller. For details, refer to the *Programming Guide* of your logic controller.

I/O Objects

Overview

The following object types are hardware-specific and depend on the logic controller being used:

- Digital inputs and outputs
- Analog inputs and outputs
- Advanced function blocks such as fast counters, high-speed counters, and pulse generators.

For more details, refer to the *Programming Guide* and *Advanced Functions Library Guide* of your logic controller.

Network Objects

Presentation

Network objects are used to communicate via EtherNet/IP, Modbus TCP, or Modbus Serial IOScanner.

There are two types of network object for EtherNet/IP communication:

- *%QWE*: Input Assembly
- *%IWE*: Output Assembly

There are two types of network object for Modbus TCP communication:

- *%QWM*: Input registers
- *%IWM*: Output registers

The following types of network object are used for the Modbus Serial IOScanner:

- *%IN*: Digital inputs (IOScanner)
- *%QN*: Digital outputs (IOScanner)
- *%IWN*: Input registers (IOScanner)
- *%QWN*: Output registers (IOScanner)
- *%IWNS*: IOScanner Network Diagnostic Codes

  NOTE: References to input and output are from the point of view of the EtherNet/IP master or Modbus TCP client.

For more information on how to configure network objects, refer to the programming guide for your logic controller.

Software Objects

Overview

EcoStruxure Machine Expert - Basic supports the following generic software objects:

Object	Description
Timers	Used to specify a period of time before doing something, for example, triggering an event.
Counters	Provides up and down counting of events.
Messages	Allows communication with external devices.
LIFO/FIFO Registers	A memory block that can store up to 16 words of 16 bits each in FIFO or LIFO modes.
Drums	Operates on a principle similar to an electromechanical Drum Controller, which changes step according to external events. On each step, the high point of a cam gives a command which is executed by the logic controller.
Shift Bit Registers	Provides a left or right shift of binary data bits (0 or 1).
Step Counters	Provides a series of steps to which actions can be assigned.
Schedule Blocks	Used to control actions at a predefined month, day, and time.
RTC	Used to read the time and date from the RTC, or update the RTC in the logic controller with a user-defined time and date.
PID	Allows regulation of the Proportional Integral Derivative (PID) function.
Data Logging	Allows to permanently store data from objects or strings.
Grafcet Steps	Lists the Grafcet bit address (%Xi) variables in order to add or modify symbols or comments.

These function blocks are described in the EcoStruxure Machine Expert - Basic Generic Functions Library Guide (see EcoStruxure Machine Expert - Basic, Generic Functions Library Guide).

Selecting the Memory Allocation Mode

Before viewing or updating the properties of software objects, choose the memory allocation mode, page 47 to use.

PTO Objects

Overview

The PTO objects provide the function blocks used for programming the PTO functions. The PTO function blocks are categorized as:

- Motion task tables

  Allows you to configure individual PTO movements in an ordered sequence, and visualize an estimated global movement profile.

- Motion

  These function blocks control motions of the axis. For example, power to axis, movement of the axis, and so on.

- Administrative

  These function blocks control the status and diagnostics of the axis movement. For example, status and value of actual velocity, actual position, axis control detected errors, and so on.

For more details on the PTO function blocks, refer to the *Advanced Function Library Guide* of your controller.

Drive Objects

Overview

Drive objects control ATV drives and other devices configured on the Modbus Serial IOScanner or Modbus TCP IOScanner.

Refer to the *Advanced Functions Library Guide* of your logic controller.

Communication Objects

Overview

Communication objects are used to communicate with Modbus devices, send/receive messages in character mode (ASCII), and to send/receive SMS messages.

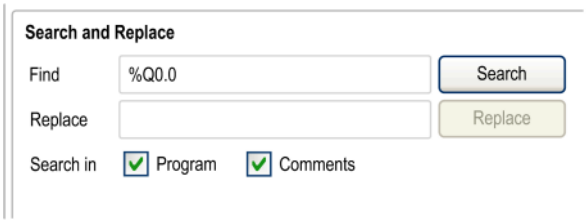
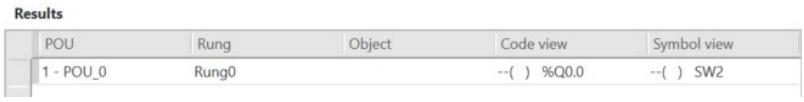
For details, refer to the chapter Communication Objects found in the EcoStruxure Machine Expert - Basic *Generic Functions Library Guide*.

Search and Replace

Overview

Search and Replace allows you to find all occurrences of an object used anywhere in a program and optionally replace it with a different object.

Searching and Replacing Items

Step	Action
1	Use any of the following methods to display the **Search and Replace** window: • Click **Search and Replace** in the **Tools** tab of the **Programming** window. • Right-click on a rung or a selected item in the rung and click **Search and Replace** in the context menu that appears. • Right-click on a line in the properties window of any object and click **Search and Replace** in the context menu that appears. **Result:** The **Search and Replace** window appears: 
2	In the **Find** box type the object or symbol name to find. The **Find** field is pre-filled if the search was started by right-clicking on a selected item in a rung or an item in a properties window of an object. You can use the following wildcard characters: • Asterisk (*). Replaces 0 or more characters in the search term. For example, *%MW1** would find both *%MW1* and *%MW101*. • Question mark (?). Replaces exactly 1 character in the search term. For example, typing *COIL?2* would find *COIL12* but not *COIL012*.
3	Optionally, in the **Replace** box type a replacement object or symbol name.
4	Select **Program** to search for the item within the source code of the current program. Select **Comments** to search for the item within program comments.
5	Click **Search** or **Replace**. You can also press ENTER to start the search. **NOTE:** The **Replace** button is enabled only when the replacement object or symbol name is given in the **Replace** box. **Result:** All items found are listed in the **Results** list: 
6	Click on any of the listed results to jump directly to the line of code in the program.

Cross Reference

Overview

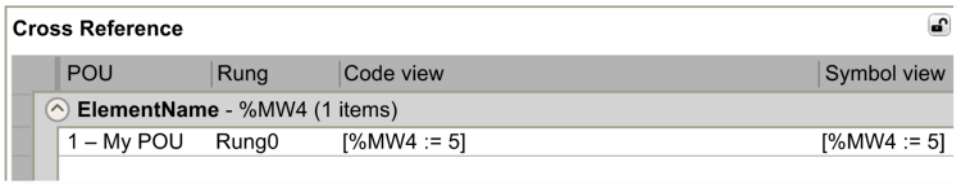
The cross reference view allows to display the program contained in a POU. If an object depends on another object of the same POU, the corresponding rungs are displayed.

The cross reference view is available in both offline and online modes.

Displaying the Cross Reference View

To display the cross reference view, click **Programming > Tools > Cross Reference**, then select one or several objects in the action zone.

Cross Reference View



The following table presents the element of the cross reference view:

Element	Description
POU	Name of the POU containing the object.
Rung	Name of the rung containing the object.
Code view	Programming code of the object.
Symbol view	Symbol of the object.

Symbol List

Overview

You can display a list of all the symbols that have been associated with objects in your program. All objects with symbols are displayed, with the exception of symbols automatically associated with system bits (%S) and system words (%SW). You can overwrite symbols and comments on system bits (%S) and system words (%SW) using the System Objects properties, or by importing your own symbols list (see below). Overwritten symbols then appear in the symbol list.

Defining and Using Symbols, page 46 describes how to create symbols and use them in your programs.

Displaying and Updating the Symbol List

Step	Action
1	Select the **Tools** tab in the left-hand area of the **Programming** window.
2	Click **Symbol list**. **Result**: The **Symbol list** window is displayed. For each item the following information is displayed: • **Used**: Whether the symbol is currently being used in the program. • **Address**: The address of the object with which the symbol is associated. • **Symbol**: The symbol name. • **Comment**: The comment associated with this object, if defined.
3	You can choose either to update the symbols while retaining the addresses used in the program (address-centric updating), or to update the addresses while retaining the symbols used in the program (symbol-centric updating): • Click the symbol mode icon T (the default) to sort the list in address order and allow the symbol name to be edited (click in the **Symbol** column). • Click the address mode icon % to sort the list in symbol order and allow the address to be edited (click in the **Address** column). If an edited address or symbol is already in use, the conflict is highlighted in red. Modify the address or symbol and press Enter to remove the conflict.
4	Click **Apply** to apply all updates throughout the program.

Creating Default Symbols

To create default symbols for memory objects, click **Generate symbols**.

Result: Default symbols are assigned to all memory objects (`%M`, `%MW`, `%MD`, `%MF`, `%S`, `%SW`, `%KW`, `%KD`, `%KF`, `%I`, `%IW`, `%Q`, `%QW`) used in the program that do not already have symbols defined.

Symbols are named as follows: `symbolname` = `objectname_i`, where `objectname` is the object type without the `%` and `i` is the index of the object.

Example: The following objects are used in the program but have no symbols defined:

Object	Symbol Assigned
`%MW0`	`MW_0`
`%MW2`	`MW_2`
`%M0`	`M_0`

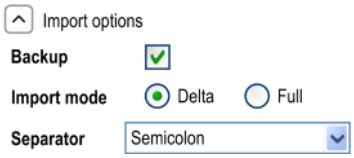
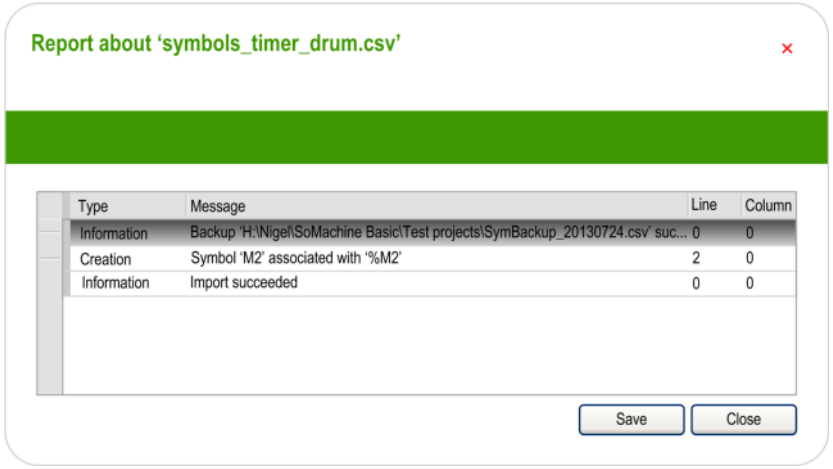
Deleting Default Symbols

To delete default symbols:

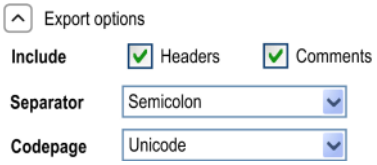
Step	Action
1	Click **Remove generated symbols**.
2	Click **Yes** on the confirmation window that appears. **Result:** All default symbols assigned are deleted.

NOTE: If an object with a default symbol assigned is subsequently removed from the program prior to the removal of the default symbols, the object will retain the default symbol if later used again in the program.

Importing Symbols

Step	Action
1	Either click the **Import** button or right-click anywhere in the list of symbols and choose **Import symbols**. **Result**: The **Import Symbols** window is displayed.
2	Browse and select the **File path** of the Comma Separated Values (CSV) file containing the symbols to import.
3	Optionally, click **Import options** and configure formatting options for the imported symbols: 
4	Click **Import**. **Result**: All symbols in the selected CSV file are created and displayed in the **Symbol list** window with the specified formatting options. If errors are detected during import, a report is displayed listing them: 
5	Click **Save** to write the contents of the report to a plain text (.txt) file.

Exporting the Symbol List

Step	Action
1	Either click the **Export** button or right-click anywhere in the list of symbols and choose **Export symbols** You are prompted to save changes. The **Export Symbols** window is displayed.
2	Browse and select the **File path** and **File name** of the Comma Separated Values (CSV) file to be created.
3	Optionally, click **Export options** and configure formatting options for the exported values: 
4	Click **Export**. **Result**: A CSV file is created with the specified formatting options.

Sharing Symbols Between an EcoStruxure Machine Expert - Basic Project and a Vijeo-Designer Project

Before sharing the symbols with a Vijeo-Designer project, verify that all symbols that you want to share are defined in the EcoStruxure Machine Expert - Basic project. If not, create/open a project in EcoStruxure Machine Expert - Basic, define the symbol names, and save the project. You can create default Vijeo-Designer symbols for all memory objects in the project, refer to Creating Default Symbols, page 108.

Follow these steps to share EcoStruxure Machine Expert - Basic symbols with a Vijeo-Designer project:

Step	Action
1	Start Vijeo-Designer.
2	Create/open a project in Vijeo-Designer.
3	Click the **Project** tab in the **Navigator** window, right-click **IO Manager** and select **New Driver... Insert**. **Result**: The **New Driver** window opens.
4	Select a driver from the **Driver** list, select an equipment from the **Equipment** list, and click **OK**. For example: • Driver: **Modbus TCP/IP** • Equipment: **Modbus Equipment** **Result**: The **Equipment Configuration** window opens.
5	Enter the details for each parameter and click **OK**. For example, **IP Address**, **Unit ID**, **IP Protocol**, and so on. **Result**: A new driver is created to open the communication with the controller. The selected driver and the selected equipment appear under the **IO Manager** node in the **Project** tab of the **Navigator** window.
6	In the Vijeo-Designer menu bar, click **Variable > Link Variables**. **Result**: The **Link Variable** window opens.
7	Select the **Files of type** filter to **EcoStruxure Machine Expert - Basic project files (*. SMBP)** and select the **Equipment** filter to the driver that you have created for communication.
8	Select the EcoStruxure Machine Expert - Basic project in which you have defined the symbols and click **Open**. **Result**: All the symbols are automatically extracted from the project and linked to the driver that you have created.
9	Select the variables that you want to use and add them to your HMI application. **Result**: All the variables with same names as symbols are added in the list of available variables. The variable list appears under the **Variables** node in the **Project** tab of the **Navigator** window.

NOTE: If you have already shared the symbols with a Vijeo-Designer project before and if you change the existing symbols and/or add new symbols to your project in EcoStruxure Machine Expert - Basic, you must update the symbols in the Vijeo-Designer project.

To update the symbols in a Vijeo-Designer project, first define new symbols and/or modify the existing symbols, save the EcoStruxure Machine Expert - Basic project, and open the Vijeo-Designer project and follow these steps:

Step	Action
1	In the **Project** tab of the **Navigator** window, right-click **Variables** and select **Update Link**. **Result**: The equipment driver and the existing symbols are updated.
2	Right-click **Variables** again, select **New Variables from Equipment** and select the new variables that you have created in the EcoStruxure Machine Expert - Basic project. **Result**: The new variables from the EcoStruxure Machine Expert - Basic project are added in the list of variables. These variables appear under the **Variables** node in the **Project** tab of the **Navigator** window.

Memory Consumption View

Overview

You can display information about the controller memory used by the application, program, and associated user data.

Displaying the Memory Consumption View

The program must first compile with no errors detected to use this feature. Refer to the Messages window, page 92 for the current program status.

To open the **Memory consumption view**, follow this procedure:

Step	Action
1	Select the **Tools** tab in the left-hand area of the **Programming** window.
2	Click **Memory Consumption**. The Memory Consumption window is displayed.

Description of the Memory Consumption View

NOTE: This view is available only if there is a valid compilation.

The following tables describe the fields of the **Memory consumption view**:

Field	Description
Last Compilation	The date and time on which the program was last compiled. **NOTE:** This value is updated whenever: • the **Compile** button  on the toolbar is clicked • a login to a controller is initiated • a program upload is started • a program modification is sent to the controller in online mode • the simulator is launched
Program lines	
Field	**Description**
Used	The number of lines of code being used by the program.
Remaining	The maximum number of lines available for the program minus the number of lines being used. **NOTE:** There is no direct link between the number of program lines used and the total number of lines of IL code in rungs in the **Programming** tab. For example, 2 lines of IL code could generate 6 program lines.

Non program data	
Field	**Description**
Reserved for system	The amount of memory reserved for system use.
Used	The amount of memory occupied by project properties, symbols, comments, and animation tables.
Remaining	The amount of memory available for non program data.
Cache memory	
Field	**Description**
Periodic and Event tasks	The amount of cache memory occupied by periodic and event tasks, in bytes.
Reserved for system	The amount of cache memory reserved for system use, in bytes.
Memory remaining	The amount of cache memory available to the program, in bytes.
Volatile memory	
Field	**Description**
Master task and subroutines	The amount of RAM memory occupied by the program master task and all subroutines, in bytes.
Configuration	The amount of RAM memory used to contain the hardware configuration of the logic controller and expansion modules, in bytes.
Memory objects	The amount of RAM memory occupied by memory objects (memory bits, memory words, and constant words) used by the application, in bytes.
Display	The size of the Remote Graphic Display application, in bytes. Zero if the logic controller does not support the Remote Graphic Display.
Memory remaining	The amount of RAM memory available to the program, in bytes.

Ladder Language Programming

Introduction to Ladder Diagrams

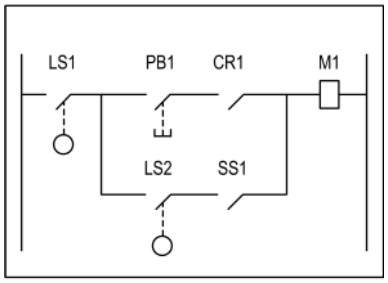
Introduction

Ladder Diagrams are similar to relay logic diagrams that represent relay control circuits. The main differences between the 2 are the following features of Ladder Diagram programming that are not found in relay logic diagrams:

- Inputs and binary logic bits are represented by contact symbols ( ).
- Outputs and binary logic bits are represented by coil symbols ( ).
- Numerical operations are included in the graphical Ladder instruction set.

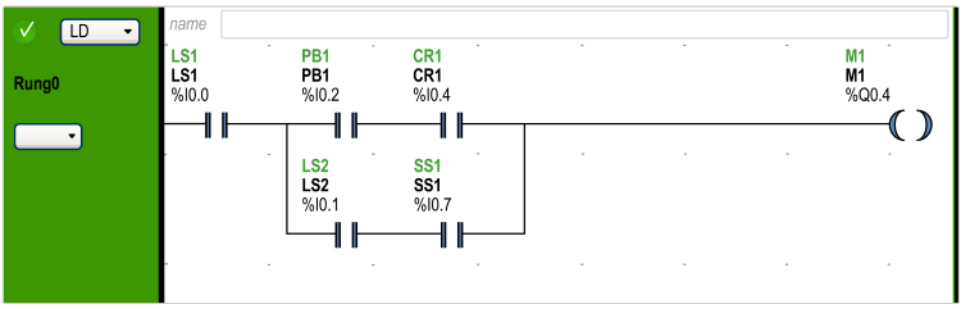
Ladder Diagram Equivalents to Relay Circuits

The following illustration shows a simplified wiring diagram of a relay logic circuit:



Relay logic circuit

The equivalent Ladder diagram:



In the above illustration, inputs associated with a switching device in the relay logic diagram are shown as contacts in the Ladder Diagram. The *M1* output coil in the relay logic diagram is represented with an output coil symbol in the Ladder Diagram. The address numbers appearing above each contact/coil symbol in the Ladder Diagram are references to the locations of the external input/output connections to the logic controller.

Ladder Diagram Rungs

A program written in Ladder Diagram language is composed of rungs which are sets of graphical instructions drawn between 2 vertical potential bars. The rungs are executed sequentially by the logic controller.

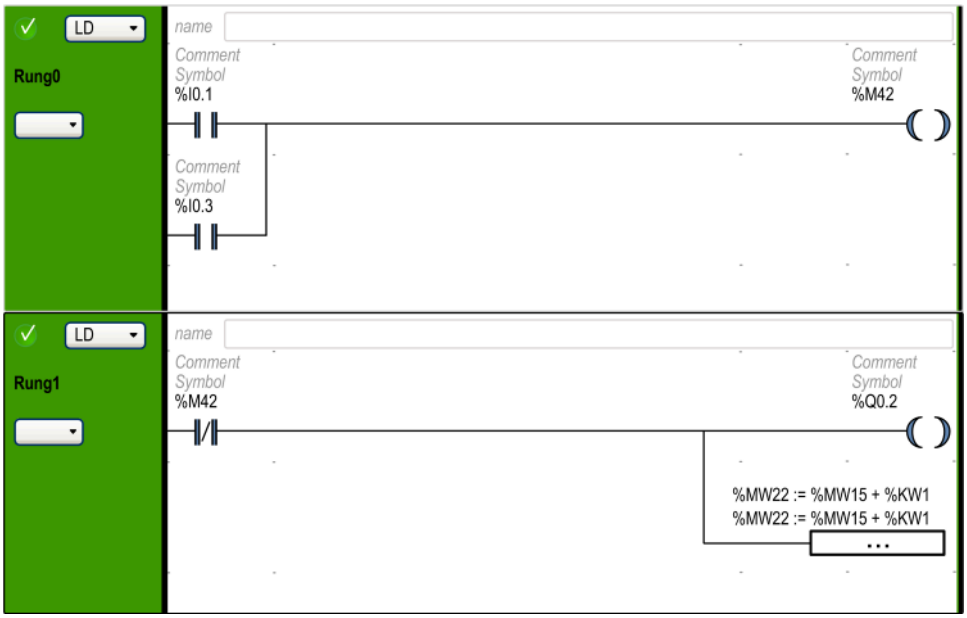
The set of graphical instructions represents the following functions:

- Inputs/outputs of the controller (push buttons, sensors, relays, pilot lights, and so on)
- Functions of the controller (timers, counters, and so on)
- Math and logic operations (addition, division, *AND*, *XOR*, and so on)
- Comparison operators and other numerical operations (*A<B*, *A=B*, shift, rotate, and so on)
- Internal variables in the controller (bits, words, and so on)

These graphical instructions are arranged with vertical and horizontal connections leading eventually to one or several outputs and/or actions. A rung cannot support more than one group of linked instructions.

Example of Ladder Diagram Rungs

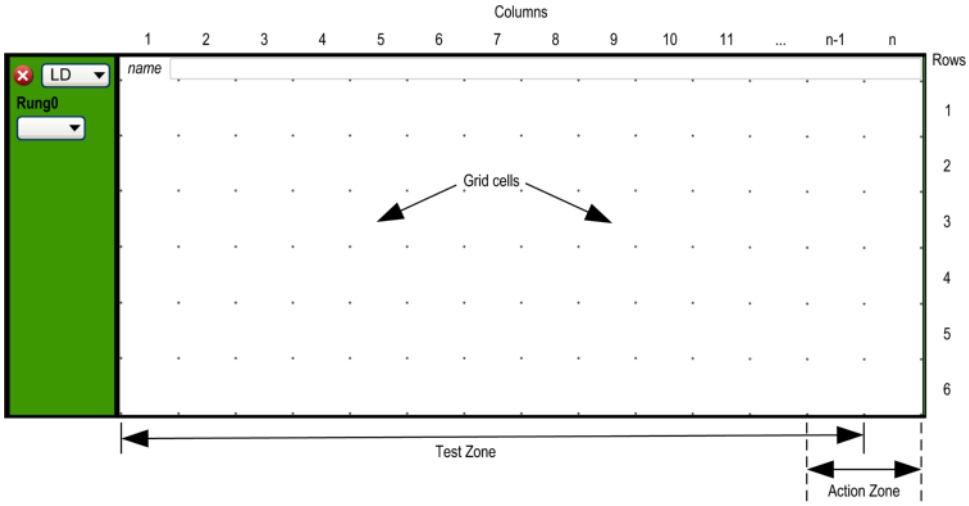
The following diagram is an example of a Ladder Diagram program composed of 2 rungs:



Programming Principles for Ladder Diagrams

Programming Grid

Each Ladder rung consists of a grid of up to 255 rows by 11...30 columns, organized into 2 zones as shown in the following illustration:



n Number of configured columns (11...30). For more information on configuration of number of columns, refer to Customizing the Ladder Editor, page 34.

Grid Cells

Cells allow you to position graphical elements in the grid. Each cell in the grid is delimited by 4 dots at the corners of the cell.

Grid Zones

By default, the Ladder Diagram programming grid is divided into 2 zones:

- Test zone

  Contains the conditions that are tested in order to perform actions. Consists of columns 1 to n-1, where n is the number of configured columns and contains contacts, function blocks, and comparison blocks.

- Action zone

  Contains the output or operation that will be performed according to the results of the tests of the conditions in the Test zone. Consists of columns n-1 to n, where n is the number of configured columns and contains coils and operation blocks.

Customizing the Ladder Editor

Use the following objects at the top of the Ladder editor to customize the content of the editor:

Object	Description
IL>LD	Switch from displaying all rungs in IL to Ladder.
LD>IL	Switch from displaying all rungs in Ladder to IL.
-	Delete one column from the Ladder grid. The button is deactivated when the minimum number of columns (11) is reached.
+	Add one column to the Ladder grid. The button is deactivated when the maximum number of columns (30) is reached.
Display/hide comments	Click to display or hide comments in the rungs. If **T** is released, the comments display on two lines.
T	Click to display or hide the symbols in the rungs. If **Display/hide comments** is released, the symbols are displayed on two lines.
DEC/HEX	Only displayed in online mode. Click to display alternately numerical values in the rungs in decimal or hexadecimal format.
1 - New POU	Double-click to edit the default **POU** name that appears in the **Tools > Master Task** area of the screen.
Comment	Double-click to type text to associate a comment with this **POU**.
Zoom slider	Zoom or unzoom the Ladder Editor. You can zoom or unzoom by using the shortcut **Ctrl** + mouse wheel. The position of the zoom remains even if you navigate through the project.

Color Coding of Rungs

Offine Mode

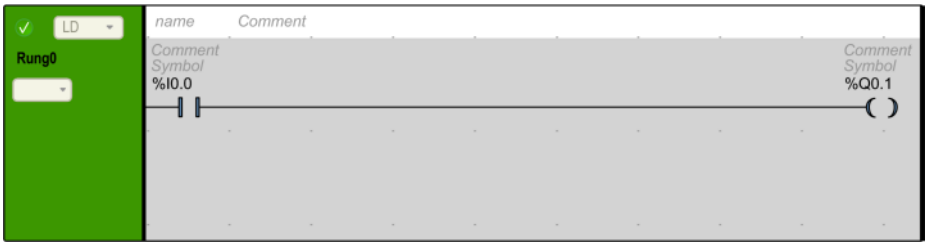
The selected rungs are displayed in a darker green background:

Online Mode

When in online mode:

- Unchanged rungs appear with a green background.
- Rungs added or modified while in online mode appear with an orange background:

- Rungs with no modifiable elements are locked and appear with a gray foreground:



Ladder Diagram Graphic Elements

Introduction

Instructions in Ladder Diagrams are inserted by dragging and dropping graphic elements from the toolbar that appears above the programming workspace into a grid cell.

Inserting a Graphic Element

To insert a graphic element in a rung:

Step	Action
1	Click the graphic element on the toolbar to insert. If the graphic element is a menu, the graphic items in the menu appear; click the menu item to insert.
2	Move the mouse to the position in the rung to insert the graphic element and click. **NOTE:** Some elements have to be inserted in the test or action zones of the rung; refer to the description of individual graphic elements for details.
3	If necessary, click the **[Selection mode]** graphic element  on the toolbar to reset the selection.

Rungs

Use the following graphic elements to manage the rungs in a program:

Graphic Element	Name	Function
	Create a rung, page 61	Inserts a new empty rung below the last rung in the program workspace.
	Insert a rung, page 61	Inserts a new empty rung immediately above the currently selected rung.
	Delete the rung, page 62	Removes the currently selected rung from the program. If the rung is not empty, you are asked to confirm that you want to delete the contents of the rung.

Branching Modes

Use the following graphic elements to manage the branch in Ladder diagram:

Graphic Element	Name	Function
	Normal mode	Lets you place the programming elements (for example, contacts, coils, and so on, except the function blocks) inline with the wire line.
	Branching mode	Lets you place the programming elements (for example, contacts, coils, and so on, except the function blocks) in branch with the wire line.

Selections and Lines

Use the following graphic elements to select graphic elements and draw lines:

Graphic Element	Name	Function
	Selection mode	Selection mode.
	Draw line	Draws a wire line between 2 graphic elements.
	Erase line	Erases a wire line.

Contacts

Use the following graphic elements to insert contacts (one row high by one column wide).

Graphic Element	Name	Instruction List	Function
	Normally open contact	*LD*	Passing contact when the controlling bit object is at state 1.
	Normally closed contact	*LDN*	Passing contact when the controlling bit object is at state 0.
	Contact for detecting a rising edge	*LDR*	Rising edge: detecting the change from 0 to 1 of the controlling bit object.
	Contact for detecting a falling edge	*LDF*	Falling edge: detecting the change from 1 to 0 of the controlling bit object.

Comparison Blocks

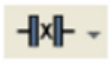
Comparison blocks are placed in the test zone of the programming grid. The block may appear in any row or column in the test zone as long as the entire length of the instruction resides in the test zone.

The graphic element for comparison blocks takes up 2 cells (1 row high by 2 columns wide).

Graphic Element	Name	Instruction List	Function
	Comparison block	Any valid comparison expression	Use the Comparison block graphical symbol to insert Instruction List comparison expressions, page 122 into Ladder Diagram rungs. A comparison expression compares 2 operands; the output changes to 1 when the result is checked.

Boolean Operations

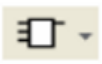
The graphic element for boolean operations takes up 1 cell (1 row high by 1 column wide).

Graphic Element	Name	Operator	Function
	XOR instructions	XOR, XORN, XORR, XORF	The *XOR* instruction performs an exclusive OR operation between the operand and the Boolean result of the preceding instruction. The *XORN* instruction performs an exclusive OR operation between the inverse of the operand and the Boolean result of the preceding instruction. The *XORR* instruction performs an exclusive OR operation between the rising edge of the operand and the Boolean result of the preceding instruction. The *XORF* instruction performs an exclusive OR operation between the falling edge of the operand and the Boolean result of the preceding instruction.

Functions

Function blocks always appear in the first row of the Ladder Diagram programming grid; Ladder instructions or lines of continuity are not allowed to appear above or below the function block. Ladder test instructions lead to the left side of the function block, and test instructions and action instructions lead from the right side of the function.

The graphic elements of function blocks can only be placed in the test zone and require 2, 3, or 4 rows by 2 columns of cells.

Graphic Element	Name	Function
	Timers, counters, registers, and so on.	Each of the function blocks uses inputs and outputs that enable links to the other graphic elements. **NOTE:** Outputs of function blocks cannot be connected to each other (vertical shorts).

Coils

The coil graphic elements can only be placed in the action zone and take up 1 cell (1 row high and 1 column wide).

Graphic Element	Name	Opera-tor	Function
	Direct coil	ST	The associated bit object takes the value of the test zone result.
	Inverse coil	STN	The associated bit object takes the negated value of the test zone result.
	Set coil	S	The associated bit object is set to 1 when the result of the test zone is 1.
	Reset coil	R	The associated bit object is set to 0 when the result of the test zone is 1.

Grafcet (List) Instructions

Use the following graphic elements to manage the branch in Ladder diagram:

Graphic Element	Name	Operator	Function
	Grafcet step activation/ Current step deactivation	#	Deactivates the current step and optionally activates another step in the Grafcet program.
	Grafcet step deactivation	#D	Deactivates a step in the Grafcet program in addition to deactivating the current step.

Operation Blocks

The operation block element placed in the action zone and occupy 2 columns by 1 row:

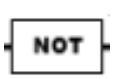
Graphic Element	Name	Operator	Function
	Operation block	Any valid operator or assign-ment instruction	Use the Operation block graphical symbol to insert Instruction List operations and assignment instructions, page 122 into Ladder Diagram rungs.

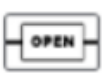
Other Ladder Items

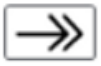
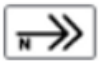
The **Other Ladder Items** menu  groups together miscellaneous instructions.

The *OPEN* and *SHORT* instructions provide a convenient method for debugging and troubleshooting Ladder programs. These special instructions alter the logic of a rung by either shorting or opening the continuity of a rung as explained in the following table.

The *END*/*JUMP* graphic elements are placed in the action zone and take up 1 cell (1 row high and 1 column wide).

Graphic Element	Name	Operator	Function
	Rising edge	`RISING` *n* (1)	Evaluates the rising edge of the expression. **NOTE:** This is a global variable. Do not use Rising edge in a user-defined function block if two or more instances of the user-defined function block can be executed concurrently.
	Falling edge	`FALLING` *n* (1)	Evaluates the falling edge of the expression. **NOTE:** This is a global variable. Do not use Falling edge in a user-defined function block if two or more instances of the user-defined function block can be executed concurrently.
	Logical NOT	*N*	Passes the inverse value of its operand.

	OPEN	`LD 0`
		
		`AND 0`
			At the beginning of the rung.
			
			Within a rung: Creates a break in the continuity of a Ladder rung regardless of the results of the last logical operation.

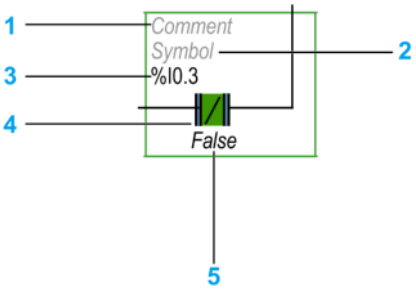
Graphic Element	Name	Operator	Function
	SHORT	`LD 1` `OR 1`	At the beginning of the rung. Within a rung: Allows the continuity to pass through the rung regardless of the results of the last logical operation.
	Stop program	*END*	Defines the end of the program.
	Conditional stop program	*ENDCN*	Defines a conditional end of the program.
	Jump or subroutine call	*JMP*	Connect to an upstream or downstream labeled rung. **NOTE:** When programming in IL, connection is to an upstream or downstream labeled instruction.
	Conditional jump or subroutine call	*JMPCN*	Conditional connect to an upstream or downstream labeled rung. **NOTE:** When programming in IL, connection is to an upstream or downstream labeled instruction.

	Conditional elements	*IF*
		ELSE
		ENDIF
			Conditionally executes a group of statements, depending on the value of an expression.

	Loop elements	*FOR* *ENDFOR*	Repeats a group of statements.
(1) *n* is an integer incremented each time a rising or falling edge is inserted.			

Contacts and Coils

Once inserted in a cell, additional information is displayed about the object associated with contacts and coils:



Legend	Item	Description
1	User comment	Click to add a comment, page 125.
2	Symbol	Click to type the name of a symbol, page 46 to associate with the object contained in the cell.
3	Address	Click to type the address of the object contained in the cell.
4	Graphic element	The graphic element.
5	Real-time value	When in online mode (connected to a logic controller and program running), displays the real-time value of the object in the cell.

Comparison Blocks

Inserting IL Comparison Expressions in Ladder Diagrams

You can use the **Comparison Block** graphical symbol to insert Instruction List comparison expressions into Ladder Diagram rungs:

The operands must be of the same object types: words with words, float with float, etc.

Proceed as follows:

Step	Action
1	Click the **Comparison Block**  button on the toolbar.
2	Click anywhere in the rung to insert the **Comparison Block**.
3	Double-click the **Comparison expression** line.
4	Type a valid Instruction List comparison operation and press ENTER. You can modify the expression in online mode. Refer to Online Modifications, page 156.

NOTE: If the application is configured with a functional level, page 54 of at least **Level 6.0**:

- You can use up to five operands and three levels of parentheses in a comparison block.
- A minimum of 20 memory words (`%MW`) must be available to use multiple operands in the master task. If using multiple operands in a periodic task as well, 20 additional memory words must be available.

NOTE: Multiple operand expressions cannot be used in event tasks.

Getting Help with Syntax

If the syntax of the Instruction List comparison operation is incorrect, the border of the **Comparison expression** box turns red. For assistance, either:

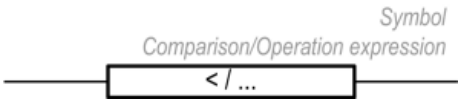
- Move the mouse over the **Comparison expression** line, or
- Select **Tools > Program Messages**.

Operation Blocks

Inserting IL Operations and Assignment Instructions in Ladder Diagrams

You can use the **Operation Block** graphical symbol to insert Instruction List operations and assignment instructions into Ladder Diagram rungs:

Operation block at the end of a rung.



Operation block anywhere in a rung.

The **Operation Block** graphical symbol can be inserted in any position in a Ladder Diagram rung except the first column, as it cannot be used as the first contact in a rung.

If more than one **Operation Block** graphical symbol is used in a Ladder Diagram rung, they must be placed in series. **Operation Block** instructions cannot be used in parallel.

NOTE: If the application is configured with a functional level, page 54 of at least **Level 5.0**:

- You can use up to five operands and three levels of parentheses in an operation block. The operands must be of the same object types: words with words, float with float, etc.
- A minimum of 20 memory words (`%MW`) must be available to use multiple operands in the master task. If using multiple operands in a periodic task as well, 20 additional memory words must be available.

To insert an operation block graphical symbol in a Ladder Diagram rung:

Step	Action
1	Click the **Operation Block**  button on the toolbar.
2	Click anywhere the rung to insert the **Operation Block**.
3	Click the **Selection mode**  button on the toolbar.
4	Double-click the **operation expression** line. The Smart Coding, page 123 button  appears at the end of the line. Click this button for help selecting a function and with the syntax of the instruction.
5	Type a valid Instruction List operation or assignment instruction and press ENTER. For example: `%MF10 := ((SIN( %MF12 + 60.0 ) + COS( %MF13 )) + %MF10 ) + 1.2` You can modify the expression in online mode. Refer to Online Modifications, page 156.

NOTE: Multiple operand expressions cannot be used in event tasks.

OPER Instruction Syntax

The `OPER` instruction corresponds to an operation block placed anywhere in a rung.

The equivalent *OPER* instruction can be used directly in Instruction List rungs.

OPER [*expression*] where *expression* is any valid expression, containing up to five operands and three levels of parentheses. For example:

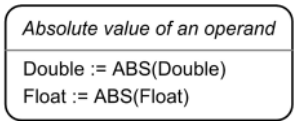
```
OPER [ %MF10 := ((SIN( %MF12 + 60.0 ) + COS( %MF13 )) + %MF10 ) + 1.2]
```

Smart Coding Tooltips in Ladder Diagrams

To help you selecting functions, EcoStruxure Machine Expert - Basic displays tooltips while you type function names in operation blocks.

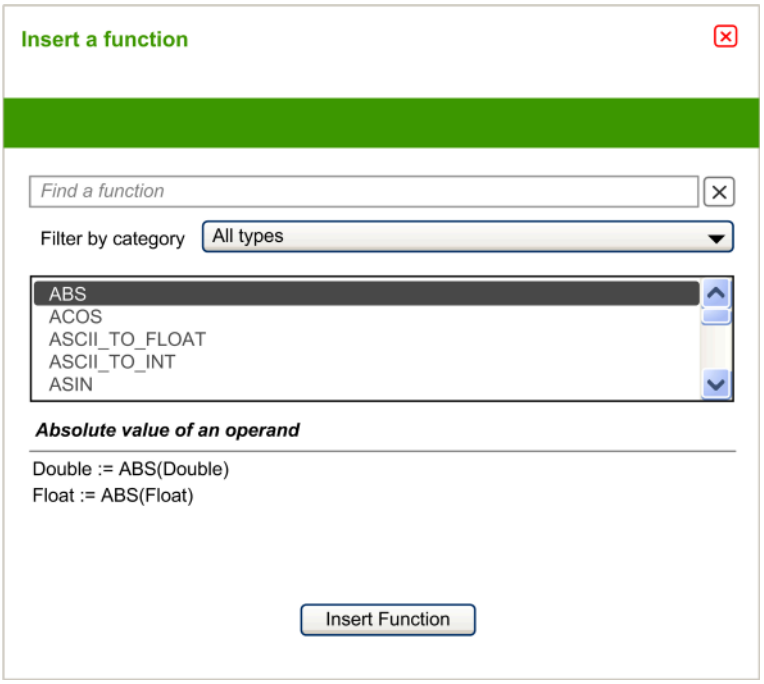
Two types of tooltip are available:

- A list of function names, dynamically updated with the function names that begin with the typed characters. For example, typing “AS” displays *ASCII_TO_FLOAT*, *ASCII_TO_INT*, and *ASIN*.
- Help with the syntax of a function, displayed when you type an opening parenthesis. For example, typing “ABS(“ displays:



Using the Smart Coding Assistant

The Smart Coding Assistant appears when you click the Smart Coding button in the operation expression line:



Proceed as follows:

Step	Action
1	Optionally, filter the list by category of function: • All types • ASCII • Floating point • I/O objects • Floating Point • Numerical Processing • Table • PID • User-defined function
2	Select a function to add to the expression.
3	Click **Insert Function**.

Getting Help with Syntax

If the syntax of the Instruction List operation or assignment instruction is incorrect, the border of the **operation expression** box turns red. For assistance, either:

- Move the mouse over the **operation expression** line, or
- Select **Tools > Program Messages**.

Adding Comments

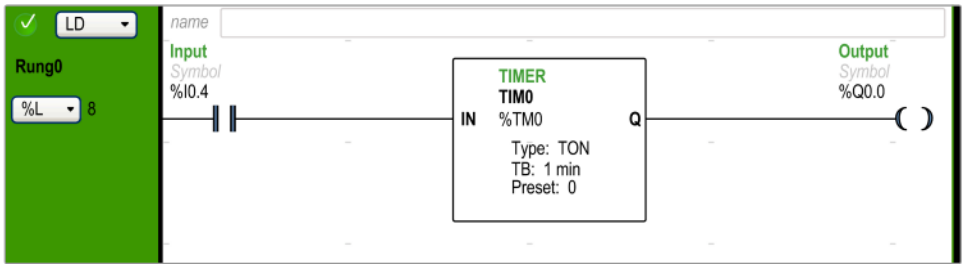
To Add Comments to Ladder Diagrams

To add comments to a Ladder Diagram program, follow these steps:

Step	Action
1	Insert a graphic element into the rung.
2	If necessary, click the selection pointer or press Esc.
3	Double-click the **Comment** line at the top of the graphic element.
4	Type the comment for the graphic element and press ENTER.

Example of Ladder Diagram Comments

This illustration shows an example of comments in a rung of a Ladder Diagram:



Programming Best Practices

Handling Program Jumps

Use program jumps with care to avoid long loops that can increase scan time. Avoid jumps to instructions that are located upstream.

NOTE: An upstream instruction line appears before a jump in a program. A downstream instruction line appears after a jump in a program.

Programming of Outputs

Physical outputs, as well as logical bits, should only be modified once in the program. In the case of physical outputs, only the last value scanned is taken into account when they are updated.

Using Directly-Wired Emergency Stop Sensors

Sensors used directly for emergency stops must not be processed by the logic controller. They must be connected directly to the corresponding outputs and applied in conformity with local, national and/or international regulations.

⚠WARNING

UNINTENDED EQUIPMENT OPERATION

Do not use the equipment configured and programmed by this software in safety-critical machine functions, unless the equipment and software are otherwise designated as functional safety equipment and conforming to applicable regulations and standards.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Handling Power Returns

After a power outage, make power returns conditional on a manual operation. An automatic restart of the installation could cause unintended operation of equipment (use system bits `%S0`, `%S1`, and `%S49`). Other system bits and system words may also help manage restarts after power outages. Refer to System Bits (%S) and System Words (%SW) (see Modicon M221, Logic Controller, Programming Guide).

Time and Schedule Block Management

The state of system bit `%S51`, which indicates any detected RTC errors, should be verified.

Syntax Validation

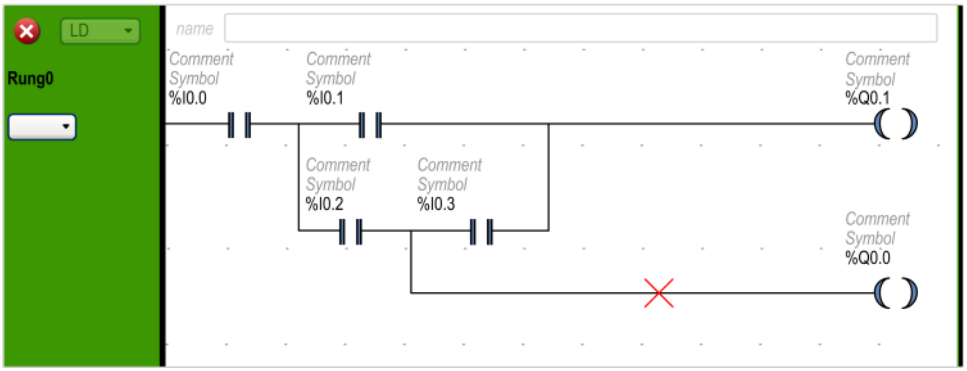
As you are programming, EcoStruxure Machine Expert - Basic validates the syntax of the instructions, the operands, and their associations.

Additional Notes on Using Parentheses

Do not place assignment instructions within parentheses:

```
LD    %I0.0
MPS
AND   %I0.1
OR(   %I0.2
)
```

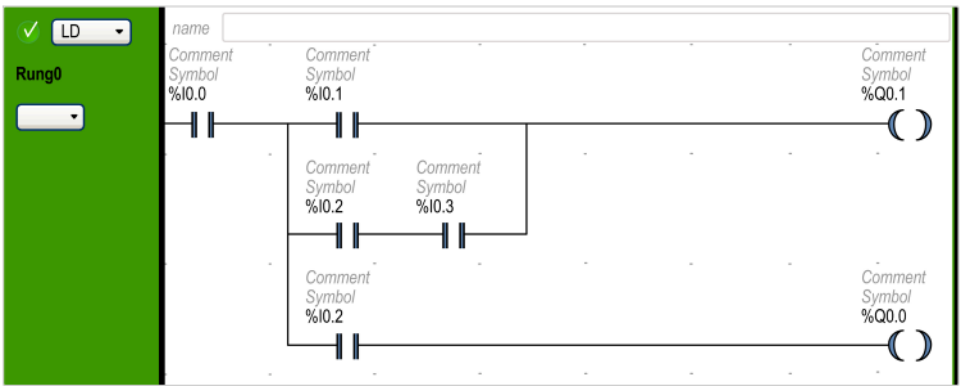
The equivalent Ladder Diagram produces a short circuit error:



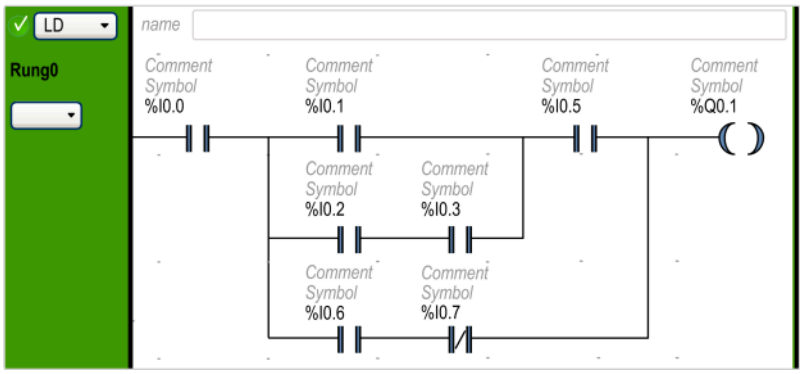
In order to perform the same function, program the instructions in the following manner:

```
LD     %I0.0
MPS
AND(   %I0.1
OR(    %I0.2
AND    %I0.3
)
)
ST     %Q0.1
MPP
AND    %I0.2
ST     %Q0.0
```

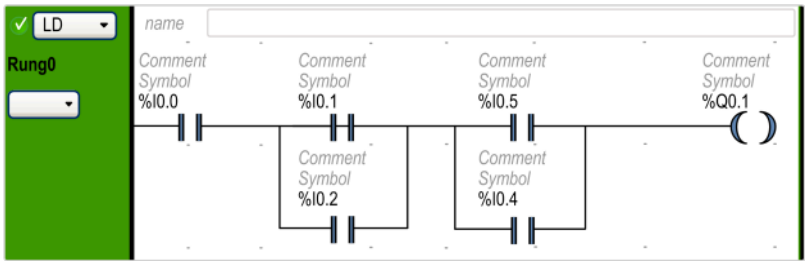
The equivalent Ladder Diagram:



If several contacts are in parallel, nest them within each other:



Alternatively, completely separate the contacts as follows:



Instruction List Programming

Overview of Instruction List Programs

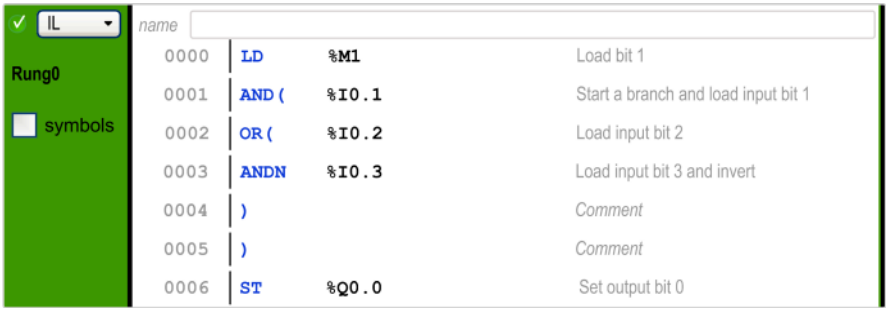
Introduction

A program written in Instruction List language consists of a series of instructions that are executed sequentially by the logic controller. Each instruction is represented by a single program line and consists of the following components:

- Line number
- Current value (in online mode only)
- Instruction operator
- Operand(s)
- Optional comment

Example of an Instruction List Program

The following is an example of an Instruction List program.



Line Numbers

Four-digit line numbers are generated when you create a new program line and managed automatically by EcoStruxure Machine Expert - Basic.

Current Values

When EcoStruxure Machine Expert - Basic is in online mode, page 20 (connected to a logic controller and the program is running), EcoStruxure Machine Expert - Basic displays the current value of object types in the IL editor window.

The displayed values of these objects are updated.

Instruction Operators

The instruction operator is a mnemonic symbol, called an operator, that identifies the operation to be performed using the operands. Typical operators specify Boolean and numerical operations.

For example, in the sample program above, *LD* is the mnemonic for the *LOAD* operator. The *LOAD* instruction places (loads) the value of the operand `%M1` into an internal register called the boolean accumulator.

There are basically 2 types of operators:

- Test operators

  These set up or test for the conditions necessary to perform an action. For example, *LOAD (LD)* and *AND*.

- Action operators

  These perform actions as a result of preceding logic. For example, assignment operators such as *STORE (ST)* and *RESET (R)*.

Operators, together with operands, form instructions.

Operands

An operand is an object, address, or symbol representing a value that a program can manipulate in an instruction. For example, in the sample program above, the operand `%M1` is an address assigned the value of an embedded input of the logic controller. An instruction can have from 0 to 3 operands depending on the type of instruction operator.

Operands can represent the following:

- Controller inputs and outputs such as sensors, push buttons, and relays.
- Predefined system functions such as timers and counters.
- Arithmetic, logical, comparison, and numerical operations.
- Controller internal variables such as system bits and words.

Comments

To add comments to an Instruction List program:

Step	Action
1	Optionally, click the comment box that appears at the top of the rung above the first line 0000 and type a comment for the rung.
2	Insert an instruction line.
3	Click in the **Comment** area to the right of the instruction.
4	Type the comment and press `Enter`.

Customizing the Ladder/IL Editor

Use the following objects at the top of the IL editor to customize the content of the editor:

Object	Description
IL>LD	Switch from displaying all rungs in IL to Ladder.
LD>IL	Switch from displaying all rungs in Ladder to IL.
-	Delete one column from IL grid. The button is deactivated when the minimum number of columns (11) is reached.
+	Add one column to IL grid. The button is deactivated when the maximum number of columns (30) is reached.
Display/hide comments	Click to display or hide comments in the rungs.
T	Click to alternately display objects in address mode or symbol mode.
DEC/HEX	Only active in online mode. Click to alternately display numerical values in the rungs in decimal or hexadecimal format.
1 - New POU	Double-click to edit the default **POU** name that appears in the **Tools > Master Task** area of the screen.
Comment	Double-click to type text to associate a comment with this **POU**.
Zoom slider	Zoom or unzoom the Ladder Editor. You can zoom or unzoom by using the shortcut **Ctrl + mouse wheel**. The position of the zoom remains even if you navigate through the project.

Operation of List Instructions

Introduction

Instruction List binary instructions normally have only one explicit operand; the other operand is implied. The implied operand is the value in the Boolean accumulator. For example, in the instruction `LD %I0.1`, `%I0.1` is the explicit operand. An implicit operand is loaded in the accumulator and the previous value of the accumulator is overwritten by the value of `%I0.1`. This value now becomes the implicit value for the subsequent instruction.

Operation

An Instruction List instruction performs a specified operation on the contents of the accumulator and the explicit operand, and replaces the contents of the accumulator with the result. For example, the operation `AND %I1.2` performs a logical AND between the contents of the accumulator and the input `1.2` and will replace the contents of the accumulator with this result.

Boolean instructions, except for *Load*, *Store*, and *Not*, operate on 2 operands. The value of the 2 operands can be either True or False, and program execution of the instructions produces a single value: either True or False. *Load* instructions place the value of the operand in the accumulator while *Store* instructions transfer the value in the accumulator to the operand. The *Not* instruction has no explicit operands and simply inverts the state of the accumulator.

Supported List Instructions

This table shows a selection of instructions in Instruction List language:

Type of Instruction	Example	Function
Boolean instruction	`LD %M10`	Loads the value of internal bit `%M10` into the accumulator
Block instruction	`IN %TM0`	Starts the timer `%TM0`
Word instruction	`[%MW10 := %MW50+100]`	Addition operation
Program instruction	`SR5`	Calls subroutine `#5`

List Language Instructions

Introduction

The Instruction List language consists of the following types of instructions or block of instructions:

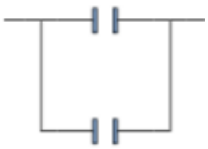
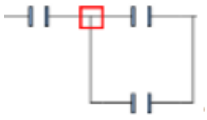
- Test Instructions
- Action instructions
- Function blocks

This section identifies and describes the instructions for List programming.

Test Instructions

This table describes test instructions in List language.

Mnemonic	Name	Equivalent Graphic Element	Function
LD	Load		Loads the boolean value of the operand into the accumulator.
LDN	Load Not		Loads the negated boolean value of the operand into the accumulator.
LDR	Load Rising		Loads the boolean value of the operand into the accumulator when the value changes from 0 to 1 (rising edge). The value of the accumulator thereafter will be loaded with 0 until the next transition of the operand from 0 to 1.
LDF	Load Falling		Loads the boolean value of the operand into the accumulator when the value changes from 1 to 0 (falling edge). The value of the accumulator thereafter will be loaded with 1 until the next transition of the operand from 1 to 0.
AND	And		The Boolean result is equal to the AND logic between the Boolean result of the previous instruction (which is stored in the accumulator) and the status of the operand. The result of the instruction is then loaded into the accumulator overwriting the previous value.
ANDN	And Not		The Boolean result is equal to the AND logic between the Boolean result of the previous instruction (which is stored in the accumulator) and the inverse (negated) status of the operand. The result of the instruction is then loaded into the accumulator overwriting the previous value.
ANDR	And Rising		The Boolean result is equal to the AND logic between the Boolean result of the previous instruction and the detection of the operand's rising edge (1 = rising edge). The result of the instruction is then loaded into the accumulator overwriting the previous value.

Mnemonic	Name	Equivalent Graphic Element	Function
ANDF	And Falling		The Boolean result is equal to the AND logic between the Boolean result of the previous instruction and the detection of the operand's falling edge (1 = falling edge). The result of the instruction is then loaded into the accumulator overwriting the previous value.
OR	Or		The Boolean result is equal to the OR logic between the Boolean result of the previous instruction and the status of the operand (which is stored in the accumulator).
AND(	And With		Logic AND (Maximum 32 levels of parentheses). The parentheses specify an intermediate logical result of the instructions between them, and then that result is logically AND'd with the value in the accumulator.
OR(	Or With		Logic OR (Maximum 32 levels of parentheses). The parentheses specify an intermediate logical result of the instructions between them, and then that result is logically OR'd with the value in the accumulator.
XOR XORN XORR XORF	Ex Or Ex Or Not Ex Or Rising Ex Or Falling		Exclusive OR
MPS MRD MPP	Memory Push Store Memory ReaD Memory PoP		Branch operators for output actions.
N	Not		Inverts the value of the operand.

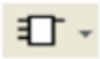
Action Instructions

This table describes action instructions in List language.

Mnemonic	Name	Equivalent Graphic Element	Function
ST	Store		The associated operand takes the value of the test zone result.
STN	Store Not		The associated operand takes the reverse value of the test zone result.
S	Set		The associated operand is set to 1 when the result of the test zone is 1.
R	Reset		The associated operand is set to 0 when the result of the test zone is 1.
JMP	Jump		Connect unconditionally to a labeled sequence, upstream, or downstream.
SRn	Subroutine		Connection at the beginning of a subroutine (subroutine call).
END	End		End of program.
ENDCN	End Conditional		Conditionally ends the program at a Boolean result of 0.

Function Blocks

This table describes function blocks in List language.

Name	Equivalent Graphic Element	Function
Timers, counters, registers, and so on.		For each of the function blocks, there are instructions for controlling the block. A structured form is used to connect the block inputs and outputs. **NOTE:** Outputs of function blocks cannot be connected to each other (vertical shorts). For more information, refer to Software Objects (see EcoStruxure Machine Expert - Basic, Generic Functions Library Guide).

Using Parentheses

Introduction

With *AND* and *OR* logical operators, parentheses are used to nest logical instructions. In so doing, they specify divergences (branches) in the Ladder editor. Parentheses are associated with instructions as follows:

- Opening the parentheses is associated with the *AND* or *OR* operator.
- Closing the parentheses is an instruction (an operator with no operand) which is required for each open parenthesis.

Example Using an *AND* Instruction

The following examples show how to use parentheses with an *AND* instruction:

Rung	Instruction
0	LD %I0.0 AND %I0.1 OR %I0.2 ST %Q0.0
1	LD %I0.0 AND(%I0.1 OR %I0.2) ST %Q0.1

NOTE: Refer to the reversibility procedure (see EcoStruxure Machine Expert - Basic, Generic Functions Library Guide) to obtain the equivalent Ladder Diagram.

Example Using an *OR* Instruction

The following example shows how to use parentheses with an *OR* instruction:

Rung	Instruction
0	LD %I0.0 AND %I0.1 OR(%I0.2 AND %I0.3) ST %Q0.0

NOTE: Refer to the reversibility procedure (see EcoStruxure Machine Expert - Basic, Generic Functions Library Guide) to obtain the equivalent Ladder Diagram.

Modifiers

This table lists modifiers that can be assigned to parentheses:

Modifier	Function	Example
N	Negation	AND(N or OR(N
F	Falling edge	AND(F or OR(F
R	Rising edge	AND(R or OR(R
[	Comparison	See Comparison Instructions.

NOTE: The '[' modifier can also be used in conjunction with other instructions serving as an operator. For more uses of the '[' in other instructions, refer to the Introduction to Numerical Operations.

Nesting Parenthesis

It is possible to nest up to 32 levels of parentheses.

Observe the following rules when nesting parentheses:

- Each open parenthesis must have a corresponding closed parenthesis.
- Labels (`%Li:`), subroutines (`SRi:`), *JMP* instructions (*JMP*), and function block instructions must not be placed in expressions between parentheses.
- Store instructions (*ST*, *STN*, *S*, and *R*) must not be programmed between parentheses.
- Stack instructions (*MPS*, *MRD*, and *MPP*) cannot be used between parentheses.

Examples of Nesting Parentheses

The following examples show how to nest parentheses:

Rung	Instruction

0	`LD %I0.0`
	`AND( %I0.1`
	`OR(N %I0.2`
	`AND %M3`
	`)`
	`)`
	`ST %Q0.0`

1	`LD %I0.1`
	`AND( %I0.2`
	`OR( %I0.5`
	`AND %I0.6`
	`)`
	`AND %I0.4`
	`OR( %I0.7`
	`AND %I0.8`
	`)`
	`)`
	`ST %Q0.0`

NOTE: Refer to the reversibility procedure (see EcoStruxure Machine Expert - Basic, Generic Functions Library Guide) to obtain the equivalent Ladder Diagram.

Grafcet (List) Programming

Description of Grafcet (List) Programming

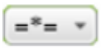
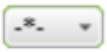
Introduction

Grafcet (List) programming in EcoStruxure Machine Expert - Basic offers another method of translating a control sequence into steps. You can translate control sequences into Grafcet steps and then use these steps in a program using Grafcet instructions.

The maximum number of Grafcet steps depends on the controller. The number of steps active at any one time is limited only by the total number of steps.

Grafcet Instructions

An EcoStruxure Machine Expert - Basic Grafcet program has the following instructions:

Operator	Operand	IL Instruction	Instruction Name	Graphic Equivalent	Description
=*=	x	=*= x	INITIAL STEP		This instruction defines the initial step in the program.
=*= POST	Not applicable	=*= POST	POST PROCESSING (implicit operand)		This instruction defines the post-processing and end sequential processing.
-*-	x	-*- x	STEP		This instruction defines a step in the program for transition validation.
#	Not applicable	#	DEACTIVATE CURRENT STEP (implicit operand)		This instruction deactivates the current step in the program.
#	x	# x	DEACTIVATE CURRENT STEP and ACTIVATE STEP x		This instruction deactivates the current step and activates step x in the program.
#D	x	#D x	DEACTIVATE CURRENT STEP and STEP x		This instruction deactivates the current step and step x in the program.
S	x	S x	ACTIVATE STEP x		This instruction activates step x in the program. The action has no affect on any other active steps.
R	x	R x	DEACTIVATE STEP x		This instruction deactivates step x in the program. The action has no affect on any other active steps.
x Grafcet step number (an integer starting from 1).					

Grafcet (List) Program Structure

Introduction

An EcoStruxure Machine Expert - Basic Grafcet (List) program has the following parts:

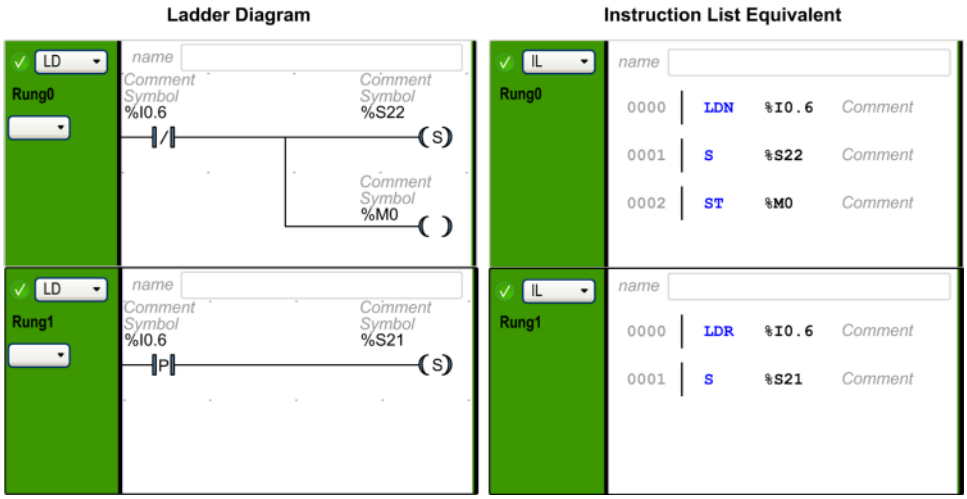
- Preprocessing
- Sequential processing
- Post-Processing

Preprocessing

Preprocessing consists of the following:

- Power returns
- Error management
- Changes of operating mode
- Pre-positioning Grafcet steps
- Input logic

In this example, the system bit *%S21* is set to 1 with the rising edge of input *%I0.6* (Rung1). This disables the active steps and enables the initial steps:



Preprocessing begins with the first line of the program and ends with the first occurrence of a **=*=** or **-*-** instruction.

System bits *%S21*, *%S22*, and *%S23* are dedicated to Grafcet control. Each of these system bits is set to 1 (if needed) by the application, normally in preprocessing. The associated function is performed by the system at the end of preprocessing and the system bit is then reset to 0 by the system.

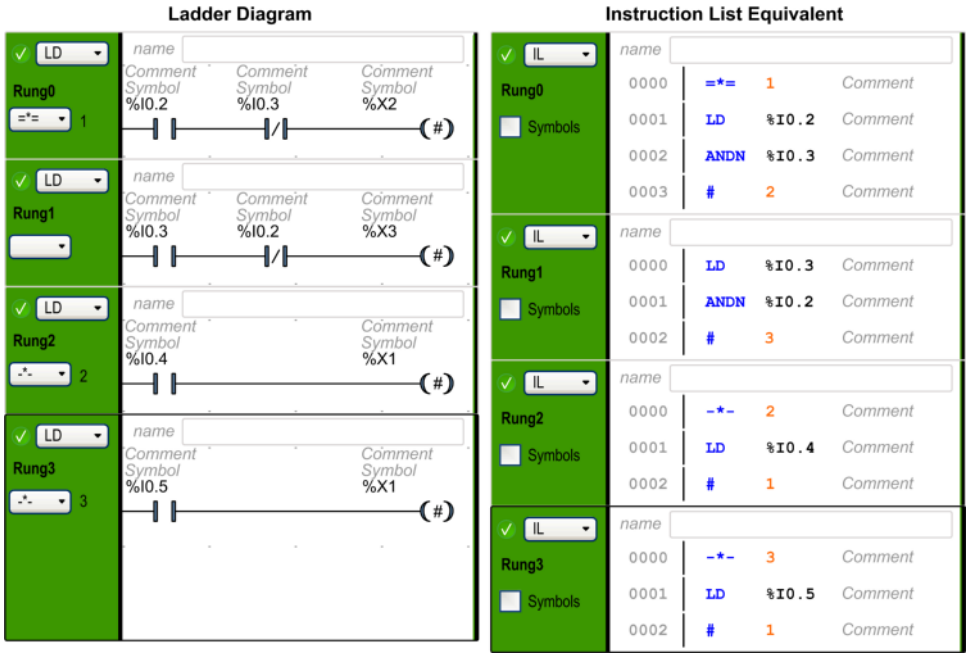
System Bit	Name	Description
%S21	Grafcet initialization	All active steps are deactivated and the initial steps are activated.
%S22	Grafcet re-initialization	All steps are deactivated.
%S23	Grafcet pre-positioning	This bit must be set to 1 if %Xi objects are explicitly written by the application in preprocessing. If this bit is maintained to 1 by the preprocessing without any explicit change of the %Xi objects, Grafcet is frozen (no updates are taken into account).

Sequential Processing

Sequential processing takes place in the chart (instructions representing the chart):

- Steps
- Actions associated with steps
- Transitions
- Transition conditions

Example:



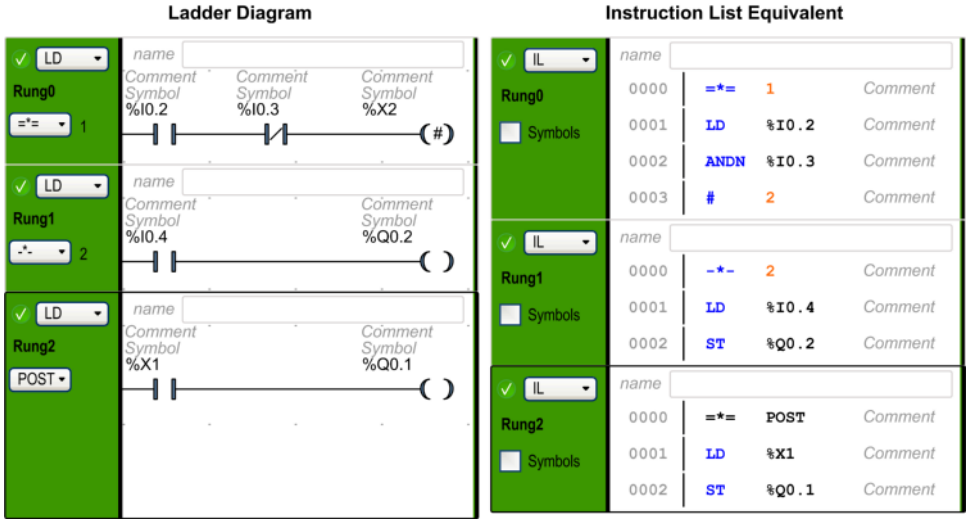
Sequential processing ends with the execution of the **POST** instruction or with the end of the program.

Post-Processing

Post-processing consists of the following:

- Commands from the sequential processing for controlling the outputs
- Interlocks specific to the outputs

Example:

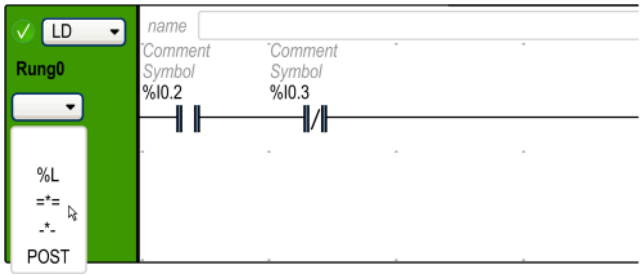


How to Use Grafcet (List) Instructions in an EcoStruxure Machine Expert - Basic Program

NOTE: Grafcet (List) instructions can only be used in the master task of a program.

Creating Grafcet (List) Steps in Ladder

Follow these steps to create Grafcet steps in a program:

Step	Action
1	In a POU, select a rung and click the drop-down button below the rung sequence identifier **Rung***x*, where *x* is the rung number in a POU.  **Result**: A menu appears listing the available Grafcet (List) instructions.
2	Click an instruction in the list to define the rung as an initial step, post processing, or a step of the Grafcet (List) program. **Result**: The rung is set for a Grafcet instruction. The operator of the instruction appears on the button and the operand (step number) appears in suffix with the button. **NOTE:** The step number is incremented by 1 as you define the next STEP or INITIAL STEP instruction. You can define only one POST instruction in a program; therefore the POST instruction does not have any step number. To modify the step number, double-click the step number in a rung and enter the new number and then press ENTER.

Activating or Deactivating Grafcet (List) Steps in Ladder

Follow these steps to activate or deactivate Grafcet (List) steps in a program:

Step	Action
1	In a POU, select a rung in your program.
2	Click  (to deactivate the current step and optionally activate a specified step) or  (to deactivate the current step and to deactivate the specified step) and insert this element in the action zone of the rung (refer to Inserting a Graphic Element, page 117).
3	Alternatively, press ALT+A to use ACTIVATE instruction or press ALT+D to use DEACTIVATE instruction in the rung. **Result**: The activate or deactivate ladder symbol appears in the action zone of the rung. Press ENTER to insert this element.
4	In the program rung, double-click ***Address*** field on the Grafcet activate or deactivate symbol and enter the Grafcet bit address (%Xi, where i is the step number). For example, *%X4* refers to the step 4 of the Grafcet program. If *%X4* is the address for the deactivate symbol, step 4 will be deactivated when the output of the rung, in which this symbol is used, is true. **NOTE:** Current step is deactivated in every case.

Grafcet (SFC) Programming

Introduction to Grafcet (SFC) Programming

Introduction

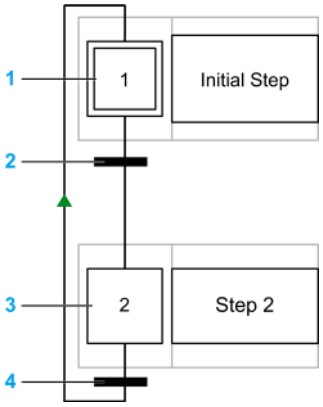
Grafcet (SFC) is a graphical programming language that describes a chronological order of execution of discrete tasks, known as *steps*. The order in which steps are executed is determined by *transitions* connecting the steps.

Elements of a Grafcet (SFC) POU

A Grafcet (SFC) POU has the following components:

- **Step**: A step executes a set of actions defined in one of more rungs written in the Ladder/IL programming languages. Steps can be:
  - **Initial step**: Executed at the beginning of the program or following a controller restart. It is represented by a cell with a double border.
  - **Regular step**: Steps conditionally executed after the initial step completes execution.
- **Transition**: A boolean expression evaluated between steps. It is the link between two or more steps. The boolean expression is defined in a single transition rung written in the Ladder/IL programming languages.

The following diagram is an example of a Grafcet (SFC) POU with an initial step, one regular step, and two transitions:



1 Initial step

2 Transition from step 1 to step 2

3 Regular step

4 Transition from step 2 back to step 1. An arrow is displayed on the link to indicate that the order of step execution is not the default left to right, top to bottom.

Grafcet (SFC) POU Rules

Grafcet POU can only be created in the master task of a program.

Multiple Grafcet POUs can be created.

Grafcet (SFC) Processing

The following rules are applied by the logic controller when executing the Grafcet (SFC):

- The master task cycle starts.
- The POUs that precede the first Grafcet (SFC) step are executed in a sequential way.
- The first Grafcet (SFC) step launches the **Grafcet monitor**.
- When the **Grafcet monitor** ends, the first POU that follows the last Grafcet (SFC) step is called.

Grafcet monitor behavior:

1. The logic controller processes the associated Grafcet (SFC) system bits % S21, %S22, and %S23.
2. The logic controller updates the activation states of each Grafcet (SFC) step.
   - Steps marked to be deactivated are deactivated.
   - Steps marked to be activated are activated.
   - Steps marked to be activated and deactivated at the same time will be or will remain activated.
   - Activation and deactivation lists are reset.
3. The logic controller scans the steps (loop from lowest defined step number to highest defined step number). When a scanned step is activated, the associated step code is called.
4. When a transition code activates or deactivates a step, this action is placed respectively in the activation or deactivation list for the next task cycle.
5. When the last active step code is executed, the **Grafcet monitor** ends.

Multi-Token Behavior

EcoStruxure Machine Expert - Basic Grafcet POU is multi-token which does not conform to IEC 61131-3.

The initial situation is controlled by the steps defined as initial steps.

Multiple steps can be active at the same time in a Grafcet POU.

The active signal status processes take place along the directional links, triggered by switching one or more transitions. The direction of the process follows the directional links and runs from the underside of the predecessor step to the top side of the successive step.

A transition is evaluated if the steps immediately preceding it are active. Transitions are not evaluated if the steps immediately preceding them are not active.

A transition is triggered when the associated transition conditions are fulfilled.

Triggering a transition marks as deactivated the immediately preceding steps that are linked to the transition, and marks as activated the immediately following steps.

The real activation or deactivation of the steps is performed at the beginning of each master task cycle (see **Grafcet monitor**, page 141).

If more than one transition condition in a row of sequential steps has been satisfied, then one step is processed per cycle.

If a step is activated and deactivated at the same time, then the step will be or will remain activated.

More than one branch can be active with alternative branches.

The branches to be run are determined by the result of the transition conditions of the transitions that follow the alternative branch. Branch transitions are processed in parallel.

The branches with fulfilled transitions are triggered.

Subroutine calls can be used in step actions.

Bits Controlling Grafcet (SFC)

Control bit	Name	Description
%S21	Grafcet initialization	If set to 1, the initial steps in the Grafcet POU are evaluated.
%S22	Grafcet reset	If set to 1, the steps are deactivated and execution restarts.
%S23	Preset and freeze Grafcet	If set to 1, execution of the Grafcet POU stops until the bit is set to 0.
%Xi	Grafcet steps	Bits *%X1* to *%Xi* are associated with Grafcet steps. Step bit *%Xi* is set to 1 when the corresponding step is active, and set to 0 when the step is deactivated. The bit is not writable when using Grafcet (SFC).

Refer to the description of System Bits (see Modicon M221, Logic Controller, Programming Guide) for further details.

Using the Grafcet (SFC) Graphical Editor

Overview

The Grafcet Graphical Editor is used for programming in Grafcet (SFC).

To display the Grafcet Graphical Editor, select any *n* - **Grafcet** node in the tree view.

The Grafcet Graphical Editor contains a grid of cells. Each cell contains one step, one transition, or both.

The minimum size of a Grafcet POU is one step.

The maximum number of Grafcet Step objects that the application can contain is:

- 96, if the **Functional Level** < **10.0**
- 200, if the **Functional Level** >= **10.0**

Detaching the Grafcet Graphical Editor

You can detach the Grafcet Graphical Editor window from the main EcoStruxure Machine Expert - Basic window so that it can be moved and resized independently. This allows you, for example, to move it to a separate monitor and display Grafcet POUs at the same time as IL/Ladder POUs.

To detach the window, click the  button in the top right corner of the Grafcet Graphical Editor window.

Drag the title bar of the window to move it. Close the window to revert to the normal view.

Inserting Steps

Double-click in any grid cell to add a step, or right-click in any grid cell and choose **Add a step** from the contextual menu that appears.

You can see the **Number of Grafcet steps used** in the top right corner of the Grafcet Graphical Editor window.

You can move a step by dragging and dropping in another grid cell.

Changing a Step Type (Initial or Regular)

The first step created in the Grafcet Graphical Editor is by default an initial step.

A Grafcet POU must contain at least one initial step. More than one step can be defined as initial steps.

To change the step type (initial/regular), right-click the step and choose **Set/Unset as initial step**.

Copying a Step

Step	Action
1	Right-click on the step to copy and choose **Copy** from the contextual menu that appears.
2	Right-click in an empty grid cell and choose **Paste**. **Result**: A copy of the step appears. Copies of Ladder/IL rungs associated with the step are added below the corresponding **Step** subnode in the tree view.

Creating Transitions

Link steps together to define the order of execution of steps.

To create a transition between two steps:

Step	Action
1	Move the mouse over the bottom of a step. **Result**: A green block appears
2	Drag the mouse to the step you want to link to.
3	Release the mouse button. **Result**: A link and transition appear.

Editing Labels

To edit the default labels of any step or transition:

Step	Action
1	Double-click the label of any Grafcet (SFC) step or transition.
2	Type the new name for the step or transition element and press ENTER. For example, change the default *Step_1* label to *INIT*.

Programming Step Functionality

The functionality of a step is defined in one or more IL/Ladder language rungs.

To define the functionality of a step:

Step	Action
1	Either: • Double-click a step in the Grafcet Graphical Editor. • Select a **Step** node in the tree view, where *n* is the step number. **Result**: The Grafcet Graphical Editor is closed.
2	Right-click on the selected **Step** node and choose **Add rung** from the contextual menu that appears. **Result:** Rungs appear as subnodes of the **Step** node in the tree view window.
3	Program the rung in the Ladder or IL programming language and create additional rungs if needed, as described in Ladder Language Programming, page 112 or Instruction List Programming, page 128.
4	To display the Grafcet Graphical Editor again, either: • Click the  icon. • Select the *n* - Grafcet POU node, where *n* is the number of the Grafcet POU.

Programming Transition Functionality

The functionality of a transition is defined in a single IL/Ladder language transition rung.

To define the functionality of a transition rung:

Step	Action
1	Either: • Double-click a transition in the Grafcet Graphical Editor. • Select a **Transitions > Tr***n* node in the tree view **Result**: The Grafcet Graphical Editor is closed and a Ladder language rung displayed.

2	Program the rung in the Ladder or IL programming language, as described in Ladder Language Programming, page 112 or Instruction List Programming, page 128.
	Function blocks can be used in transition rungs, except those that have no outputs, for example, Shift Bit Register, Step Counter.
	When a function block is used, the `END_BLK` instruction must immediately follow the `ENDT` instruction, for example:
	`Tr1 Comment`
	`0000 BLK %TM2`
	`0001 LD 0`
	`0002 IN`
	`0003 OUT_BLK`
	`0004 LD Q`
	`0005 ENDT`
	`0006 END_BLK`
	NOTE: The rung ends with an *ENDT* (end transition) instruction. This instruction cannot be selected or modified and must be the last instruction in the rung (unless the rung contains a function block).

3	To display the Grafcet Graphical Editor again, either: • Click the  icon. • Select the *n* - Grafcet POU node, where *n* is the number of the Grafcet POU.

Undo/Redo

You can use the **Undo** or **Redo** buttons on the toolbar for a maximum of 10 actions stored.

Deleting a Step or Transition

Step	Action
1	In the Grafcet Graphical Editor: • Select a step or transition and press the **DELETE** key. • Right-click on the step or transition and choose **Delete the selected items** in the contextual menu. **Result:** The selected step or transition is deleted. **NOTE:** You cannot delete a step or transition from the tree view.

Branching

Introduction

A Grafcet (SFC) POU can contain branches.

Two types of branch exist:

- Parallel branching: two or more steps are processed simultaneously when the preceding transition is true.
- Alternative branching: one or more alternative steps are processed depending on the result of evaluating the preceding transition conditions (multi-token behavior).

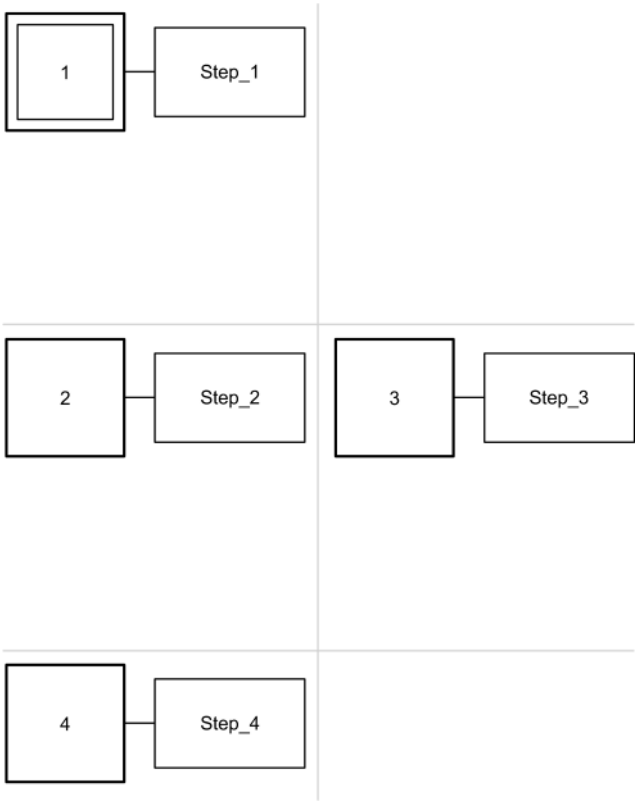
Parallel Branching

A parallel branch allows a transition from a single step to multiple steps.

A parallel branch must be preceded and followed by a step.

Parallel branches can contain nested alternative branches or other parallel branches.

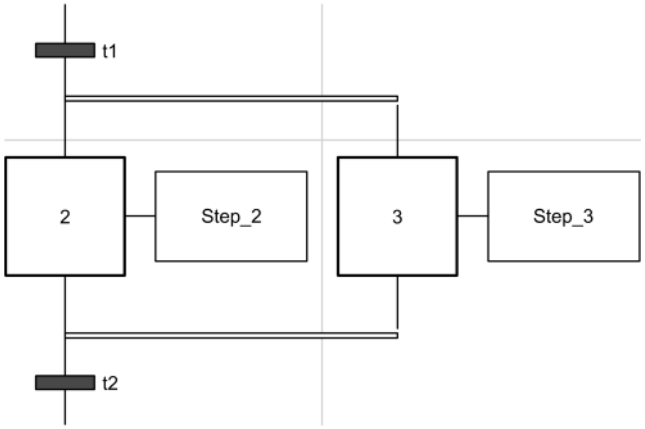
The following figure shows an example Grafcet POU with 4 steps before creation of parallel branching:



To create a parallel branch for Step_2 and Step_3:

Step	Action
1	Create a transition between Step_1 and Step_2: move the mouse to the bottom of Step_1, then drag to Step_2 and release the mouse button. A new link and transition appears.
2	Draw a link between Step_3 and the transition: move the mouse to the top of Step_3, then drag to the transition and release the mouse button. **Result:** A horizontal double line appears below the existing transition (see the figure that follows). **NOTE:** To create a link between a transition and a step that is higher up in the POU, draw the link starting from the step and dragging to the transition.
3	To rejoin the branch with the main processing branch, create a transition between Step_2 and Step_4.
4	Draw a link between Step_3 and the new transition: move the mouse to the bottom of Step_3, then drag to the transition and release the mouse button. **Result:** A horizontal double line appears above the transition (see the figure that follows).

The following figure shows a Grafcet POU after the creation of parallel branching:



NOTE: The horizontal lines before and after the branched areas are double lines.

Alternative Branching

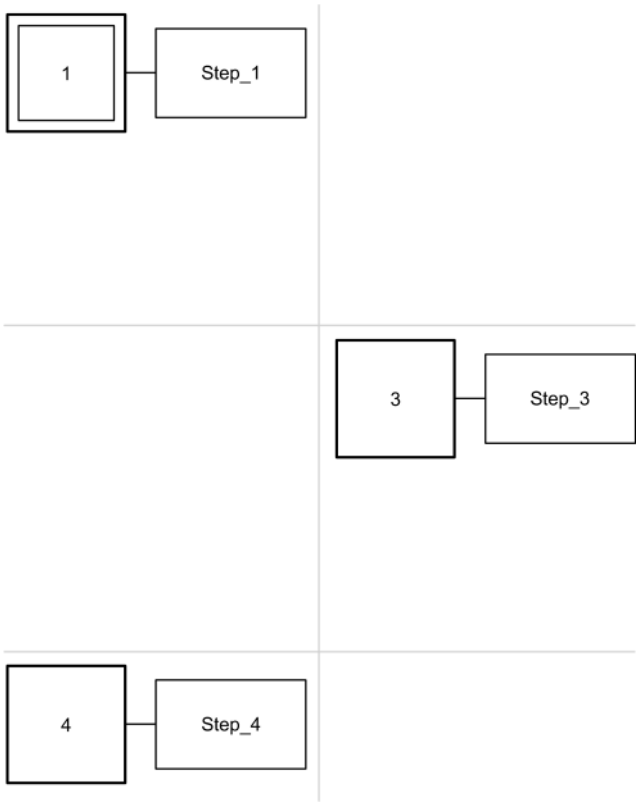
An alternative branch must begin and end with a transition.

Alternative branches can contain nested parallel branches or other alternative branches.

With multi-token behavior, more than one parallel switch can be made from the transitions. The branches to be run are determined by the result of the transition conditions of the transitions that follow the alternative branch. The transitions of the branches are processed. The branches with satisfied transitions are triggered.

If alternative branches need to be switched exclusively (mono-token behavior), then this must be defined explicitly within the transition code.

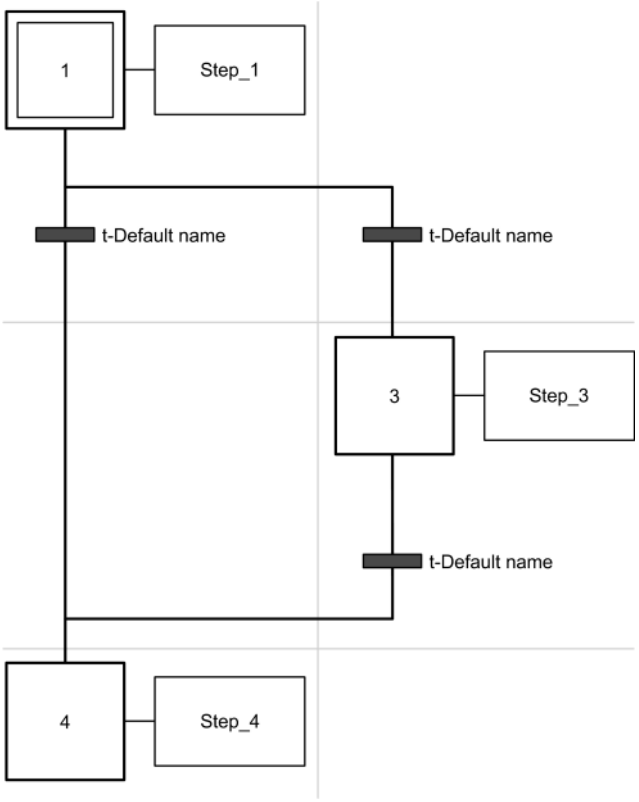
The following figure shows an example Grafcet POU with 3 steps before creation of alternative branching for Step_3 and Step_4:



To create an alternative branch:

Step	Action
1	Create a transition between Step_1 and Step_4. **Result:** A new link and transition appears.
2	Draw a transition between Step_1 and Step_3: move the mouse to the bottom of Step_1, then drag to Step_3 and release the mouse button. **Result:** A new link and transition appears, with the branch above the existing transition (see the figure that follows).
3	Draw a transition between Step_3 and Step_4. **Result:** A new link and transition appears, with the branch below the existing transition between Step_1 and Step_4 (see the figure that follows).

The following figure shows the Grafcet POU after creation of alternative branching:

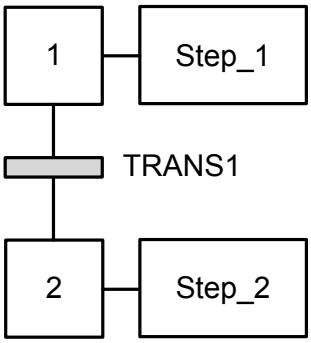


NOTE: The horizontal lines before and after the branched area are single lines.

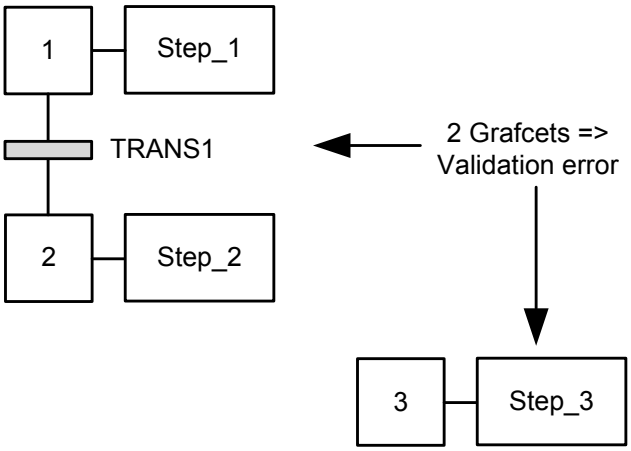
Programming Best Practices

Grafcet (SFC) Rules

- Steps must be connected by a transition:



- You can add only one Grafcet POU in the same Grafcet Graphical Editor:



Crossed Links

You can have crossed links for the following reasons:

- Alternative (logical OR) branching (fork or junction)
- To save space on the cell grid. When lines cross there is no interaction between the lines and it is used only for symbolic representation.

Debugging in Online Mode

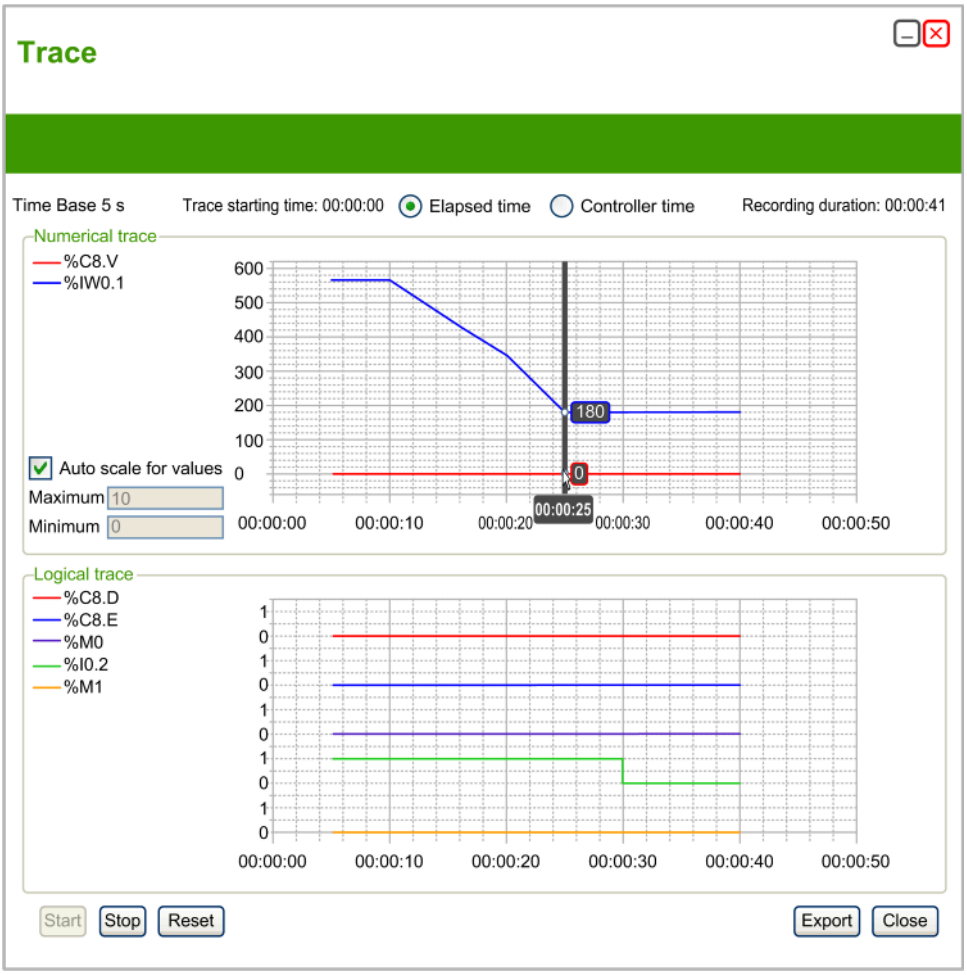
Trace Window

Overview

The **Trace** window allows you to display in graphical form the values of specific analog and/or digital variables (limited to 12 hours of continuous recording). Each animation table can contain 1 trace at any one time. Up to 8 objects can be added to a trace. You can export the data to a file for further analysis.

NOTE: The minimum configurable refresh period, page 98 for the trace is 1 second. Therefore, changes in values of boolean variables between master task cycles, for example, cannot be traced.

Trace Window Presentation



Select **Elapsed time** to set tracing start time to 00:00:00, or **Controller time** to use the time and date of the logic controller as the start time of the trace.

The Trace window displays separate graphs for each data type selected for tracing in the animation table:

- Integer and real values appear in the **Numerical trace** area.

  All numerical values share the same scale on the graph.

  Select **Auto scale for values** to automatically adjust the vertical axis to display all values. Otherwise, type **Maximum** and **Minimum** values to display a fixed range of values.

  NOTE: You can type either integer or real values for **Maximum** and **Minimum**.

- Binary values appear in the **Logical trace** area.

  Each binary value is traced on an individual scale of 0 and 1.

Starting, Pausing, and Resetting the Trace

Click **Start** to begin tracing the variables.

Click **Stop** to pause real-time tracing.

Click **Reset** to clear all previously traced data from graphs and reset the **Recording duration** value to 0.

Exporting the Trace

Click **Export** to export all traced data to a file on the PC.

The data is saved in comma-separated value (CSV) format.

Modifying Values

Introduction

When in online mode, EcoStruxure Machine Expert - Basic allows you to modify the values of certain object types.

Online updating is only possible if the object has read/write access. For example:

- The value of an analog input cannot be modified.
- The value of the *Preset* parameter (`%TM0.P` object) of a *Timer* function block can be updated.

Refer to the description of objects in the EcoStruxure Machine Expert - Basic *Generic Functions Library Guide* or the *Programming Guide* of your hardware platform for information on which object types have read/write access.

To modify the value of an object, add it to an animation table, page 96 and set its properties as required.

Forcing Values

Overview

When in online mode, you can force the values of certain boolean object types to False (0) or True (1). This allows you to set addresses to specific values and prevent the program logic or an external system from changing the value. This function is used for the debugging and fine-tuning of programs.

To force the values of boolean objects when in online mode, either:

- Use an animation table, page 94
- Modify boolean object values, page 153 directly in the Ladder (LD) editor

Digital inputs and outputs cannot be forced when:

- An input is used as a Run/Stop input
- Configured as fast counter (FC) inputs
- Configured as high speed counter (HSC) inputs
- Configured as reflex outputs

NOTE: Forcing is performed at the end of the scan cycle. The image table of the outputs, however, may be modified due to the logic of your program, and may appear in animation tables and other data displays contrary to the forced state you selected. At the end of the scan, this will be corrected by acting upon the requested forced state and the physical output will indeed reflect that forced state.

Online Mode Modifications

Overview

It is possible to modify program while in online mode by:

- Adding rungs, page 152
- Modifying rungs, page 152
- Modifying Boolean Values in Ladder, page 153
- Modifying Function Block Parameters, page 155
- Modifying Constant Words, page 155
- Modifying Object Values in Operation and Comparison Blocks, page 156
- Deleting rungs, page 156
- Sending Modifications, page 156

Any changes made must then be sent to the logic controller, page 156.

Adding Rungs

You can add new rungs, page 61 to your program while in online mode.

NOTE: The application must be configured with a functional level, page 54 of at least **Level 4.1** to be able to add new rungs in online mode.

The following limitations apply until the new rung has been successfully sent to the logic controller:

- Rungs containing errors ( ) cannot be sent to the logic controller.
- Rungs must be written in Ladder language and cannot be converted to IL until they have been successfully compiled.
- Rungs cannot contain Grafcet (List) steps.
- Labels cannot be added to the rung.

Modifying Rungs

You can modify program rungs, in both the Instruction List (IL) and Ladder (LD) editors, while in online mode. However, the Grafcet (SFC) is not available online. Modified rungs appear with an orange background, page 116.

There are limits to the type of editing that you may perform and the instructions that you may edit, depending on whether the logic controller is in `RUNNING` or `STOPPED` state. These limits help protect the state of the controller and the integrity of the program.

You can switch the display of a rung between Instruction List (IL) and Ladder (LD), even when in online mode.

The following table indicates in which cases modifications are allowed:

Operations	In STOPPED in IL	In RUNNING in IL	In STOPPED in Ladder	In RUNNING in Ladder
Event task content	editable	editable(1)	editable	editable(2)
Master / Periodic task content	editable	editable	editable	editable
Free POU content	editable	editable(1)	non-editable	editable(2)
Rung with label	editable	editable(3)	editable	editable(4)
Rung with jump, or calling a subroutine or label	editable	editable	editable	editable
Rung with any Grafcet instruction	non-editable	non-editable	non-editable	non-editable
Add/modify label	non-editable	non-editable	non-editable	non-editable
Multiple operands (operation and comparison blocks)	editable	editable	editable	editable
User-defined function	rejected	rejected	editable	editable(2)
User-defined function block	rejected	rejected	editable	editable(2)
(1) Except the first line of the first rung **(2)** Except the first element of the first rung **(3)** Except the first line of the rung **(4)** Except the first element of the rung				

NOTE: This table does not take into account the program structure modifications, which are not allowed in online mode.

Modifying Boolean Values in Ladder

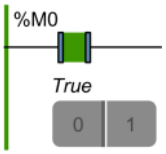
For rungs displayed in Ladder language, the values of certain boolean object types can be written to 1/0, forced to 1/0, or unforced.

The following boolean object types can be modified:

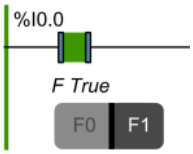
Object type	Write 1/0	Force to 1/0 or Unforce
Digital input (%Ix.y)	N/A	Yes
Digital output (%Qx.y)	Yes	Yes
System bit (%Si)[1]	Yes	N/A
Memory bit (%Mi)	Yes	N/A
Memory word bit (%MWi :Xj)	Yes	N/A
Analog output bit (%QWi :Xj)	Yes	N/A
System word bit (%SWi :Xj)[1]	Yes	N/A
Input assembly bit (%QWEi :Xj)	Yes	N/A
(1) If the system bit or system word can be written by the user program.		

Move the mouse cursor over the object in the Ladder editor. If the object can be both written to 1/0 and forced to 1/0, the following buttons appear below the graphical element:

If the object can be written to 1/0 but not forced, the following buttons appear:



If the object can be forced but not written to 1/0, the following buttons appear:



Click a button to modify the real-time value of the object:

- **0** Write 0.
- **1** Write 1.
- **F0** Force to 0.
- **F1** Force to 1.

The button corresponding to the present status of the object is shown in dark gray (**F1** in the example above).

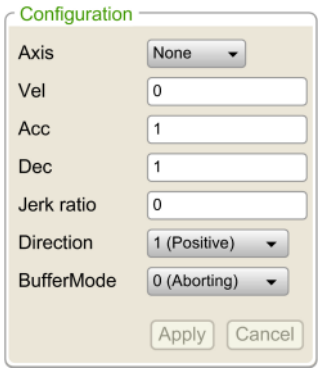
To remove forcing, either:

- Click again on the **F0**/**F1** button.
- Use an animation table, page 94.

  NOTE: Forcing is performed at the end of the scan cycle. The image table of the outputs, however, may be modified due to the logic of your program, and may appear in animation tables and other data displays contrary to the forced state you selected. At the end of the scan, this will be corrected by acting upon the requested forced state and the physical output will indeed reflect that forced state.

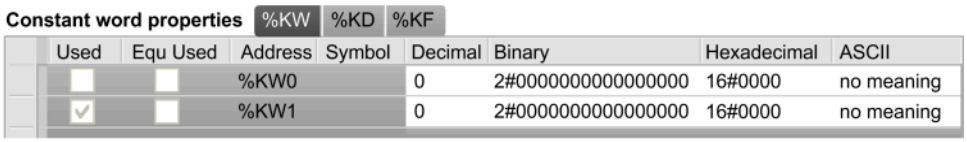
Modifying Function Block Parameter Values

To modify a function block parameter in online mode:

Step	Action
1	In the **Programming** window, move the mouse cursor over the function block in the Ladder editor. **Result:** The **Configuration** tooltip appears. The following illustration shows an example of the **Configuration** tooltip: 
2	Click the value to modify.
3	Type the value.
4	To validate the modifications, you can use one of the following methods: • Click **Apply**. • Click outside of the **Configuration** tooltip. **Result:** The **Question** window appears. Click **OK**.

Modifying Constant Words

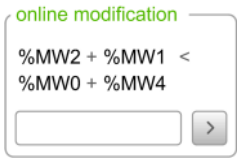
Configuration values and runtime data values of constant word (%KW), constant double word (%KD), and constant floating point (%KF) objects can be modified while in online mode. In the properties grid, the columns **Decimal**, **Binary**, **Hexadecimal** and **ASCII** are editable:



To modify a constant word or floating point value in online mode:

Step	Action
1	In the **Tools** tab of the **Programming** window, choose **Memory objects > Constant words**.
2	Click **%KW**, **%KD**, or **%KF** to select the type of constant to modify.
3	Modify the value as required. You can import the constant values. Refer to Importing the constant word properties, page 102.
4	Click **Apply**. **Result:** The modified value is sent to the logic controller.

Modifying Object Values in Operation and Comparison Blocks

Step	Action
1	In the **Programming** window, move the mouse cursor over an operation or comparison block in the Ladder editor. **Result:** The **online modification** tooltip appears: 
2	Click the object or the symbol to modify.
3	Enter the value.
4	To validate, you can use one of the following methods: • Click  . • Press **Enter**. If a value is incorrect, the value remains unchanged.

Deleting Rungs

You can delete rungs from your program while in online mode.

NOTE: The application must be configured with a functional level, page 54 of at least **Level 4.1** to delete rungs in online mode.

The following limitations apply:

- The rung must be displayed in Ladder language.
- The rung cannot be the only rung in a POU or Free POU. This limitation does not apply to Grafcet POUs.
- The rung must not contain Grafcet (List) steps, be a subroutine rung, or contain any of the following instructions:
  - JMP
  - END
  - ENDC
  - ENDCN
  - G7
- Only one rung at a time can be deleted.

Sending Modifications

In IL, the modifications, when allowed, are automatically sent to the logic controller after validation of the edited IL line. If the modification is not allowed, a message appears.

In Ladder, the modifications are not sent automatically. When in online mode, a button bar appears:

Click **Send** to send the modifications to the logic controller. This button is only active when the program has been modified in online mode and contains no errors.

Click **Rollback** to discard the changes made in online mode and restore the original rung (that is, the version stored in the logic controller). The background

color of the rung changes from orange to green. This button is only active when the program has been modified in online mode.

Click **Download non program data** to download updates to non program data (project properties, symbols, comments, animation tables, and so on) to the logic controller. This button is only active when the non program data is not synchronized between the PC and the logic controller, for example, if an animation table has been modified prior to entering online mode.

Click **Backup** to synchronize the contents of the flash memory and the RAM memory on the logic controller. This status is shown in the Controller Info window, page 174. During the backup, the Ethernet communications in progress (for example, using Modbus TCP or the *EXCH3* instruction) are temporarily suspended.

> **NOTE:** Be sure that online modifications have been saved to flash memory prior to creating a clone.

Rungs that are modified are evaluated for their validity in the context of whether the controller is in `RUNNING` or `STOPPED` state. Modifications that would cause runtime errors, or change the structure of the program memory, are rejected in online mode.

Commissioning

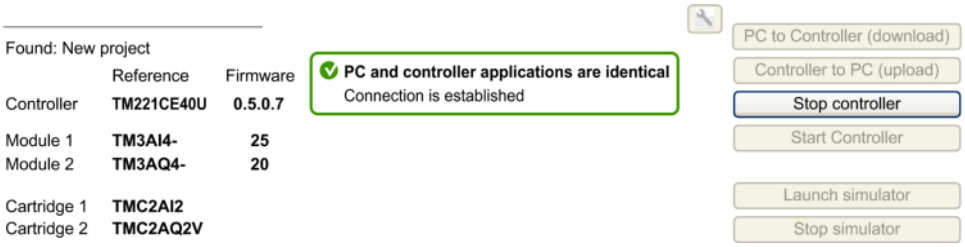
What's in This Chapter

Overview of the Commissioning Window

Introduction

The **Commissioning** window allows you to:

- Login to or logout from a logic controller.
- Upgrade (or downgrade) the logic controller firmware.
- Manage logic controller memory (for example, by performing backup and restore operations).
- Display information about the logic controller, expansion module (references and, for TM3 expansion modules, firmware versions), and cartridges you are connected to.
- Manage the real time clock (RTC) of the logic controller.



NOTE: The application must be configured with a functional level , page 54 of at least **Level 5.0** to be able to view the firmware version of TM3 Analog expansion modules.

Connecting to a Logic Controller

Overview

Click **Connect** on the **Commissioning** window to manage the connection with the logic controller.

Available Controllers

Two lists of logic controllers are displayed:

1. **Local Devices**

   Displays all logic controllers connected to the PC:

   - With the physical COM ports of the PC (COM1, for example)
   - With USB cables
   - Through the virtualized COM ports (by USB-to-serial converters or Bluetooth dongles)
   - With a modem connection that you choose to add manually. Use a modem connection between EcoStruxure Machine Expert - Basic and a logic controller for monitoring purposes only.

     NOTE: If a COM port is selected and the **Keep Modbus driver parameters** check box is selected, the communication is established with the parameters defined in the Modbus driver.

2. **Ethernet Devices**

   Displays all logic controllers that are accessible by Ethernet (on the same subnet and not under a router or any device that blocks UDP broadcasts). This list includes logic controllers that are automatically detected by EcoStruxure Machine Expert - Basic, any controllers that you choose to add manually as well as the IP address configured in the **Ethernet Configuration** view, if any.

Manually Adding Ethernet Devices

To manually add a logic controller to the **Ethernet Devices** list:

Step	Action
1	In the **Remote lookup** field, type the IP address or the URL of the logic controller to add, for example 12.123.134.21 or my.url.com:502 (connection port can also be specified).
2	Click **Add** to add the device to the **Ethernet Devices** list.

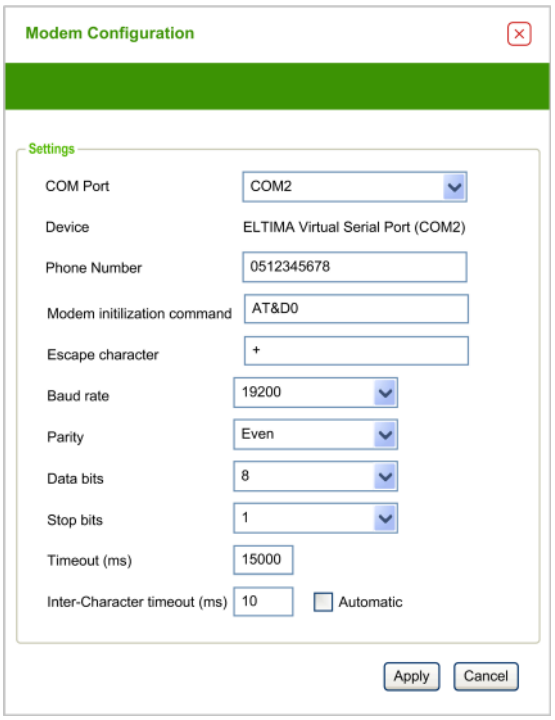
Manually Adding Modem Connections

Prerequisites for availability of modem:

- If no modem is installed on the PC, the button is disabled.
- Verify in the **Phone and Modem** option of the Windows **Control Panel** that the modem is installed and perform a test (in the **Modem** tab, click the modem to test and click **Properties > Diagnostics > Query Modem**). The response of the modem must be valid.
- If the modem is an external modem connected on a COM port, verify that the communication settings are the same in:
  - the modem advanced parameters,
  - the communication port parameters,
  - the Modbus driver parameters.

For more details on the installation and setting of SR2MOD03 modem, refer to the SR2MOD02 and SR2MOD03 Wireless Modem User Guide.

To manually add a modem connection to the **Local Devices** list:

Step	Action
1	Click  (**Add modem configuration** button) to open the **Modem configuration** window. **Result**: The **Modem configuration** window appears.
2	Select the COM port of the modem from the drop-down list: 
3	Configure the communication parameters. For detailed information on the modem configuration parameters, refer to the table below.
4	Click **Apply**. **NOTE:** This button is enabled only if all settings are correctly configured. **Result**: The modem connection is added to the **Local Devices** list (for example **COM2@0612345678,GenericModem**).

This table describes each parameter of the modem configuration:

Parameter	Value	Default value	Description
Port	**COMx**	-	Allows you to select the COM port of the modem from the dropdown list.
Device	-	-	Displays the modem name.
Phone Number	-	-	Type the phone number of the remote modem connected to the logic controller. This text field accepts all the characters and is limited to 32 characters in total. This field must contain at least one character to be able to apply the configuration.
AT init cmd	-	`AT&D0`	Allows you to edit the AT initialization command of the modem. The AT initialization command is not mandatory (if the field is empty the `AT` string is sent).
Escape chars	-	+	Allows you to edit the escape character for the hang-up procedure.
Baud rate	**1200** **2400** **4800** **9600** **19200** **38400** **57600** **115200**	**19200**	Allows you to select the data transmission rate (bits per second) of the modem.
Parity	**None** **Even** **Odd**	**Even**	Allows you to select the parity of the transmitted data for error detection.
Data bits	**7** **8**	**8**	Allows you to select the number of data bits.
Stop bits	**1** **2**	**1**	Allows you to select the number of stop bits.
Timeout (ms)	0...60000	15000	Allows you to specify the transmission timeout (in ms).
Break timeout (ms)	0...10000	10	Allows you to specify the interframe timeout (in ms). If the check box **Automatic** is selected, the value is automatically calculated.

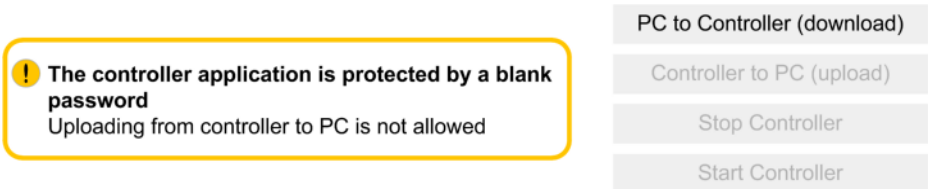
Connecting to a Logic Controller

To log in to a logic controller:

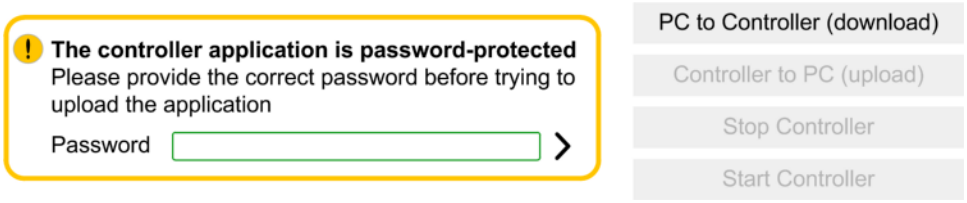
Step	Action
1	Click  (**Refresh Devices** button) to refresh the list of connected Ethernet devices.
2	Select one of the logic controllers in the **Local Devices** or **Ethernet Devices** lists. If a controller is connected by Ethernet on the same network cable than your PC, the IP address of the controller appears in the list. Selecting the IP address in the list enables  (**IP Address Configuration** button). Click this button to change the IP address of the controller. **NOTE:** If you select the check box **Write to post configuration file**, the Ethernet parameters are modified in the post configuration file and kept after a power cycle. This option is taken into account after a power cycle.
3	If required, click  (**Start Flashing LEDs** button) to flash the LEDs of the selected controller to identify the controller physically by its flashing LEDs. Click this button again to stop flashing the LEDs. **NOTE:** You can only use the **Start Flashing LEDs** button for logic controllers that are automatically added (with **Auto discovery protocol enabled** option selected).
4	Click **Login** to log in to the selected controller. **Result**: A status bar appears showing the connection progress.
5	When connected, the protection status of the application currently stored in the logic controller appears in the **Selected Controller** area of the window. When the connection is successfully established, details about the logic controller appear in the **Selected Controller** area of the window: • The firmware revision • The logic controller reference number • The reference numbers of all expansion modules connected to the logic controller • The current state of the connection between EcoStruxure Machine Expert - Basic and the logic controller.
6	EcoStruxure Machine Expert - Basic verifies whether the hardware configuration of the logic controller is compatible with the configuration of the current project. If so, the application can be downloaded to the controller. The **PC to Controller (download)** button is enabled and you can proceed to download the application, page 165. EcoStruxure Machine Expert - Basic verifies whether the non-program data (symbols, comments, animation tables, and so on) stored in the logic controller is the same as that of the current application. If not, an advisory message is displayed. EcoStruxure Machine Expert - Basic also verifies whether a more recent firmware version is available and, if so, displays a link you can click to begin the firmware upgrade.

Verifying Controller Protections

When downloading is authorized and the Logic Controller application is protected by a blank password, the following message appears in the **Commissioning** window:



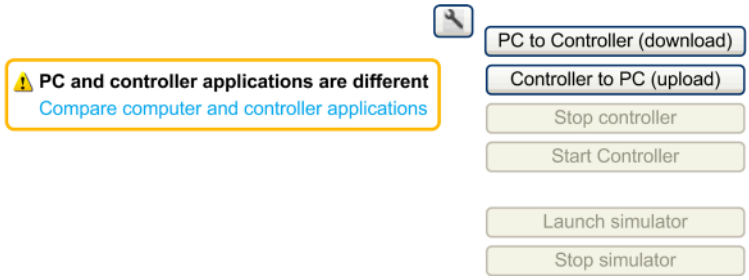
When downloading is authorized and the Logic Controller application is password-protected, the following message appears in the **Commissioning** window:



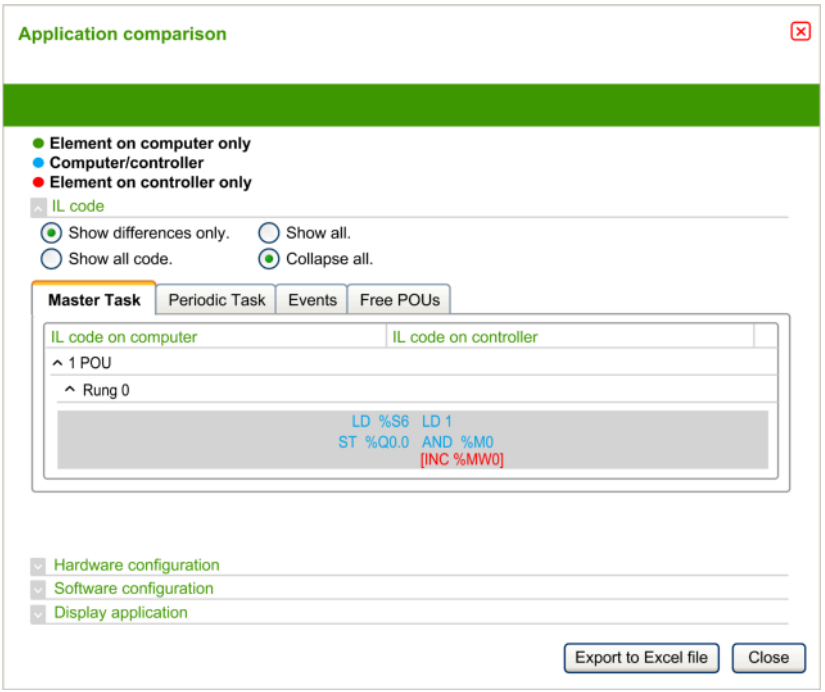
Comparing Projects when Connected

You can compare the EcoStruxure Machine Expert - Basic application with the application in the logic controller. Differences are displayed and can then be evaluated and taken into consideration.

When both downloading and uploading are authorized, and the PC and Logic Controller applications are not the same, the following message appears in the **Commissioning** window:



Proceed as follows:

Step	Action
1	In the message, click on **Compare computer and controller applications**. **Result**: A popup window informs you that you must disconnect from the logic controller before viewing the comparison.
2	Click **OK** to continue and disconnect from the logic controller.
3	The **Application comparison** window is displayed:  Comparisons are available of the following areas of the configuration and application: • **IL code** • **Hardware configuration** • **Software configuration** • **Display application**
4	Optionally, click **Export to Excel file** to save the comparison in spreadsheet format.

Downloading and Uploading Applications

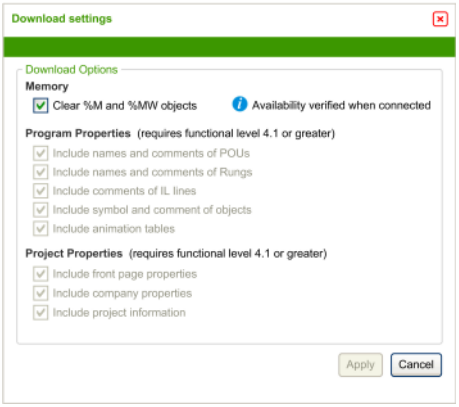
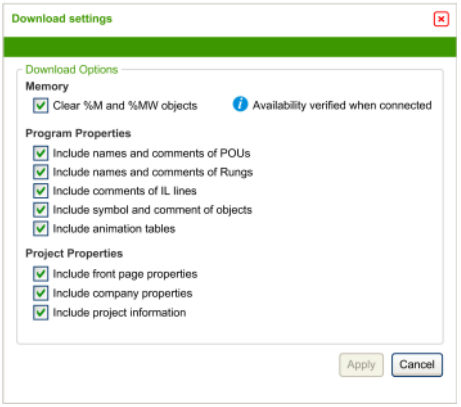
Downloading the Application

Follow these steps to download the application currently open in EcoStruxure Machine Expert - Basic to the logic controller:

Step	Action
1	Click **Connect** in the commissioning tree of the **Commissioning** window.
2	Select one of the logic controllers in the **Local Devices** or **Ethernet Devices** lists.
3	Click **Login** to log in to the selected controller.
4	Optionally, click  **Download Settings**. If you do not want memory words (%MW) and memory bits (%M) to be reset after the download, clear the **Reset memories** option. **NOTE:** The option in **Memories** is only available for logic controllers with firmware version greater than or equal to 1.3.3.y. The options in **Program Properties** and **Project Properties** are only available for logic controllers with firmware version greater than or equal to 1.4.1.y.
5	If the **PC to Controller (download)** button is not available, verify that: • The application stored in the logic controller is identical to the EcoStruxure Machine Expert - Basic application. • The hardware configuration of the logic controller system is compatible with the configuration in the EcoStruxure Machine Expert - Basic application. • The logic controller is not password-protected. If it is, enter the password. Click **PC to Controller (download)**.
6	If the non-program data of the current application is not identical to that stored in the controller, the non-program data only is downloaded to the controller.
7	If the application has been configured to **Start in Run**, a message is displayed and prompts you to confirm that the application has been so configured. Click **OK** to confirm the download of the application or click **Cancel** and modify the configuration.
8	Click **OK** to continue the transfer and overwrite the logic controller application. **Result**: A status bar appears indicating the connection status.
9	To run the application you have downloaded, click **Run controller** and click **OK** to confirm the action. If a message appears informing you that the operating mode cannot be changed, click **Close** and verify the RUN/STOP switch on the logic controller, and/or, the RUN/STOP input as they may prevent the controller from passing into *RUNNING*. Otherwise, refer to the *Hardware Guide* of your logic controller for details.

Setting Download Options

To display the **Download options**, click  **Download settings** in the **Commissioning** window.

Functional Level ≤ 4.1	Functional Level ≥ 4.1
	
The settings are not uploaded.	The settings are uploaded.

Each setting is selected by default. If you select or clear an option in online mode, click **PC to Controller (download)** to download the modifications.

In online mode, if you modify the name or comments of a POU, rung, or IL line and if these corresponding options are selected in **Download Settings**, the download is done automatically.

Reset Memories option is selected by default. This option is available in offline and online mode.

When **Reset Memories** is selected, the memory words and bits are reset to 0 at application download.

When **Reset Memories** is cleared, the memory words and bits keep their values.

If the amount of allocated memory is different for the application residing in PC memory than that of the application residing in the memory of the logic controller, the memory is managed as follows:

- If allocated `%MWx` in the application of the logic controller is greater than that allocated `%MWx` in the application residing in the PC memory, then the allocation of the application on the PC is used, and the additional `%MWx` words are set to 0.
- If allocated `%MWx` in the application of the logic controller is less than that allocated `%MWx` in the application residing in the PC memory, then the additional `%MWx` words are removed from the memory space.
- If there is no application in the logic controller, `%MW` are set to 0. The same rules are applied for `%M`. Download settings are project-dependent and saved with the project.

Uploading an Application

Follow these steps to upload the application stored in the logic controller to EcoStruxure Machine Expert - Basic:

Step	Action
1	Click **Connect** in the commissioning tree of the **Commissioning** window.
2	Select one of the logic controllers in the **Local Devices** or **Ethernet Devices** lists.
3	Click **Login** to log in to the selected controller. If the logic controller is password protected, type the password and click **OK** to connect.
4	Click **Controller to PC (upload)**. If the **Controller to PC (upload)** button is not available, confirm whether the application stored in the logic controller is identical to the EcoStruxure Machine Expert - Basic application.
5	Click **OK** to confirm upload from the logic controller. **Result**: A status bar appears indicating the connection status. When the transfer completes, the application is uploaded from the logic controller into EcoStruxure Machine Expert - Basic.

NOTE: The value of the **Reset Memories** option is not saved when you upload an application.

Downloading or Uploading a Password-Protected Application

If you download or upload an application that was password-protected in a lesser version of EcoStruxure Machine Expert - Basic, the actions you can or must do depend on the versions:

Operation	SoMachine Basic version	EcoStruxure Machine Expert - Basic version	Functional level of the Application	Firmware version	Actions
Downloading					
	≤ 1.5	-	≤ 5.0	≤ 1.5	Downloading process does not use the latest security strategy.
				≥ 1.5.1	Downloading process possible and password visible.
			≥ 5.1	–	You cannot download.
	> 1.5	≥ 1.0	≤ 10.0	≤ 1.10.0	If the application is password-protected, you cannot download.
				≥ 1.10.1	You must do one of the following actions: • Upgrade the functional level to **10.1**. • Let the password blank. • Disable the application protection.
			≥ 10.1	≤ 1.10.0	You cannot download.
				≥ 1.10.1	Downloading process uses the latest security strategy.
Uploading					
	≤ 1.5	-	≤ 5.0	≤ 1.5	Uploading process does not use the latest security strategy.
				≥ 1.5.1	You must do one of the following actions: • Downgrade the firmware version. • Upgrade the EcoStruxure Machine Expert - Basic version.
			≥ 5.1	–	You cannot upload.
	> 1.5	≥ 1.0	≤ 5.0	≤ 1.5	Uploading process does not use the latest security strategy.
				≥ 1.5.1	Uploading process uses the latest security strategy.
			≥ 5.1	≤ 1.5	You cannot upload.
				≥ 1.5.1	Uploading process uses the latest security strategy.

Controller Firmware Updates

Overview

You can download firmware updates to the logic controller either directly from EcoStruxure Machine Expert - Basic or using an SD card.

Downloading a Firmware Update to the Logic Controller

Performing a firmware update preserves the application program present in the controller, including the Boot Application in the non-volatile memory.

Follow these steps to download firmware updates to the logic controller:

Step	Action
1	Verify that you are not connected to the logic controller when using **Firmware Update**.
2	Click **Commissioning** > **Controller Update**.
3	Click **Update**. The first page of the executive loader (OS loader) wizard appears.

If you remove power to the device, or there is a power outage or communication interruption during the transfer of the application, your device may become inoperative. If a communication interruption or a power outage occurs, reattempt the transfer. If there is a power outage or communication interruption during a firmware update, or if an invalid firmware is used, your device will become inoperative. In this case, use a valid firmware and reattempt the firmware update.

NOTICE
INOPERABLE EQUIPMENT • Do not interrupt the transfer of the application program or a firmware change once the transfer has begun. • Re-initiate the transfer if the transfer is interrupted for any reason. • Do not attempt to place the device into service until the file transfer has completed successfully. **Failure to follow these instructions can result in equipment damage.**

Memory Management

Overview

Memory management allows you to create SD card scripts and the associated files needed to configure the following elements for your machines:

- Logic controller firmware.
- Logic controller program.
- User memory (`%M`, `%MW`).
- Post configuration.

Managing Logic Controller Memory

Overview

In EcoStruxure Machine Expert - Basic, you can back up, restore, or erase the different elements from or to the logic controller to which you are connected.

Backup, restore, and erase options are available only in online mode.

⚠WARNING

UNINTENDED EQUIPMENT OPERATION

- Verify that the controller you are connected to is the intended target before performing the erase or the restore operation.
- Verify the state of security of your machine or process environment before performing the erase or the restore operation from a remote location.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Backing Up to a PC or Controller SD Card

Follow these steps to back up the logic controller memory to a PC or to the controller SD card:

Step	Action
1	Log in to the logic controller.
2	Select **Memory Management** in the left-hand area of the **Commissioning** window.
3	In the **Action** list, choose **Backup from Controller**.
4	To back up to a PC: Under **Destination**, choose **PC**. Click the browse button, navigate to the folder in which to write the backup file. or To back up to an SD card: Under **Destination**, choose **Controller SD Card** folder. Insert an SD card into the SD card slot of the logic controller. **NOTE:** The SD card must not be empty or contain a `script.cmd` file to avoid creating a clone or the script execution (see Modicon M221, Logic Controller, Programming Guide).
5	Select the elements to back up by selecting the options: • **Backup firmware** • **Backup program** • **Backup memory values** • **Backup log file** • **Backup post configuration file** When **Backup memory values** is selected in a PC backup, specify the **First Memory Bit**, **Last Memory Bit**, **First Memory Word**, and **Last Memory Word** to be included in the backup.
6	Click **Backup from Controller** to begin the backup operation. The elements are saved to the specified PC folder or SD card as an SD card image. A report window appears displaying a list of information or detected error messages about the backup operation.

NOTE: If you choose to back up memory values, you can initiate a backup while the logic controller is in *RUNNING* state. As a consequence, the backup would not necessarily be coherent because the memory variables value can be modified from one scan to another. If you wish to have a consistent set of values for the variables, you may need to first put the logic controller into *STOPPED* state.

Restoring

Follow these steps to restore logic controller elements from a PC:

Step	Action
1	Log in to the logic controller.
2	Select **Memory Management** in the left-hand area of the **Commissioning** window.
3	In the **Action** list, choose **Restore to Controller**.
4	Choose the source folder that contains the backup files on the PC.
5	Select the elements you want to restore to the logic controller.
6	Click **Restore to Controller** to begin the restore operation. A report window appears displaying a list of information or detected error messages about the restore operation.

Incomplete file transfers, such as data files, application files and/or firmware files, may have serious consequences for your machine or controller. If you remove power, or if there is a power outage or communication interruption during a file transfer, your machine may become inoperative, or your application may attempt to operate on a corrupted data file. If an interruption occurs, reattempt the transfer. Be sure to include in your risk analysis the impact of corrupted data files.

⚠WARNING

UNINTENDED EQUIPMENT OPERATION, DATA LOSS, OR FILE CORRUPTION

- Do not interrupt an ongoing data transfer.
- If the transfer is interrupted for any reason, re-initiate the transfer.
- Do not place your machine into service until the file transfer has completed successfully, unless you have accounted for corrupted files in your risk analysis and have taken appropriate steps to prevent any potentially serious consequences due to unsuccessful file transfers.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

To restore a backup from a controller SD card, refer to the *Programming Guide* of the logic controller.

Erasing Logic Controller Elements

Follow these steps to erase logic controller elements:

Step	Action
1	Select **Memory Management** in the left-hand area of the **Commissioning** window.
2	In the **Action** list, choose **Erase in Controller**.
3	Select the elements you want to erase in the logic controller. If you select the **Erase post configuration file** option, the post configuration file is deleted immediately upon clicking **Erase in Controller**. In order to preserve any existing Ethernet connections, however, deletion of the file is only taken into account by the controller following an Ethernet reinitialization, that is, any of the following events: • Unplugging and replugging the Ethernet cable • Initializing the controller • Power cycle the controller.
4	Click **Erase in Controller** button to begin the erase operation. A report window appears displaying a list of information or detected error messages about the erase operation.

Creating and Reading Logic Controller Images

A logic controller image includes the controller firmware, the program, and the post configuration file. A script allows you to transfer those elements to a logic controller.

When creating a logic controller image, choosing an SD card as destination allows you to use this SD card in a logic controller.

Creating a Logic Controller Image

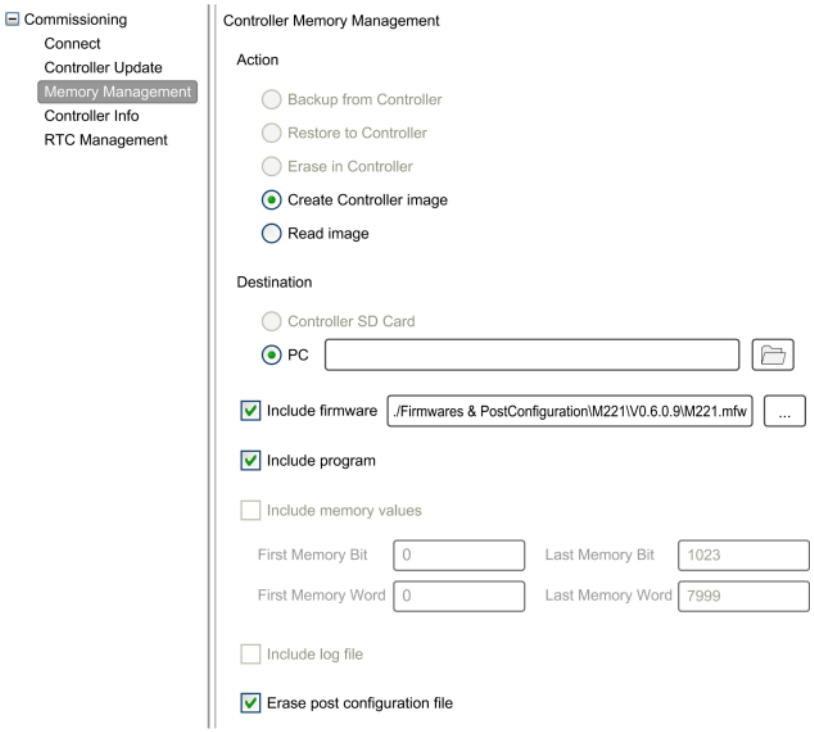
In offline mode, this procedure allows you to generate a script and copy files necessary to copy the following elements to your PC or an SD card:

- Firmware contained in the installed EcoStruxure Machine Expert - Basic software.
- Program of the currently opened project.
- Post configuration file.

Follow these steps to create a logic controller image:

Step	Action
1	If you are connected to a logic controller, click Logout in the **Commissioning** window.
2	Select **Memory Management** in the left-hand area of the **Commissioning** window.
3	In the **Action** list, choose **Create Controller image**.
4	In **Destination > PC**, click the browse button and navigate to the folder in which to write the image file. You can choose an SD card inserted in your PC as destination.
5	Select the elements to copy by selecting: • Include firmware • Include program
6	If you want to overwrite the post configuration file, select **Erase post configuration file**.
7	Click **Create Controller image**. **Result**: The following folders and files are created: • `script.cmd` • usr/app/`*.smbk` • sys/os/`*.mfw`
8	If you created the controller image on your PC, copy the files in an SD card.

The following illustration presents an example of the settings:



Reading a Logic Controller Image

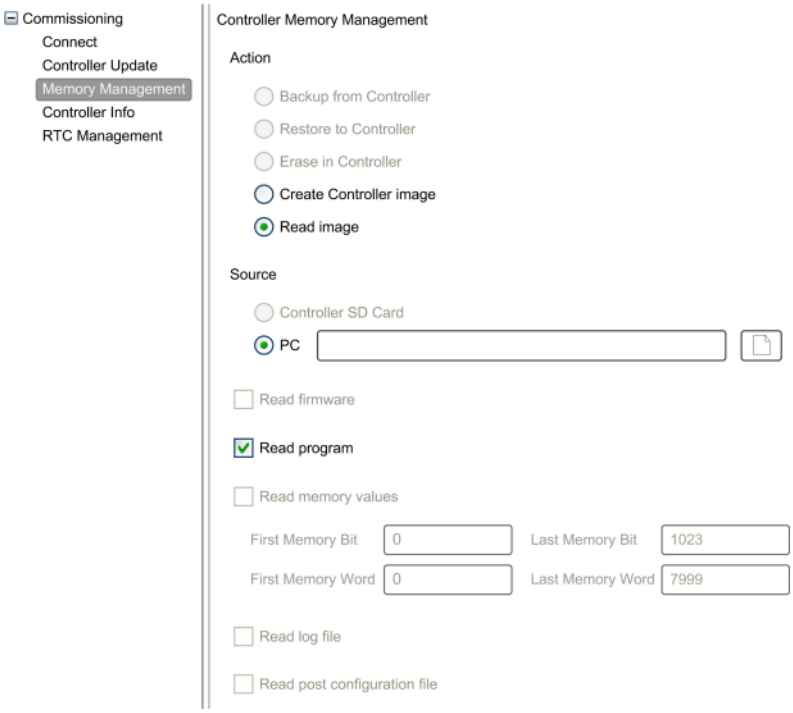
In offline mode, this procedure allows you to open a *.smbk* image file in EcoStruxure Machine Expert - Basic as a project.

NOTE: The image opened must have been previously created either through **Create Controller image** operation or by a backup from the controller, page 172.

Follow these steps to read a logic controller image:

Step	Action
1	If you are connected to a logic controller, click Logout in the **Commissioning** window.
2	Select **Memory Management** in the left-hand area of the **Commissioning** window.
3	In the **Action** list, choose **Read image**.
4	In **Source > PC**, click the browse button and navigate to the folder containing the image file *(.smbk)*. **Read program** is selected by default. To read a image file, you have to select it.
5	Click **Read image** to read the program and open a project.

The following illustration presents an example of the settings:



Controller Information

Overview

Click **Controller info** in the left-hand area of the **Commissioning** window to display the following information on the present state of the logic controller:

- **Executable memory**: This option verifies if a valid application is stored in the RAM memory of the logic controller. This information can also be obtained from within a program by testing bit 14 of the system word %SW7 (see Modicon M221, Logic Controller, Programming Guide).
- **Protected memory**: This option is checked if the application in the RAM memory of the logic controller is password-protected. This information can also be obtained from within a program by testing bit 8 of the system word %SW7 (see Modicon M221, Logic Controller, Programming Guide).
- **Forced I/O**: This option is checked if 1 or more digital inputs or outputs on the logic controller are being forced to a specific value, page 96. In this case, system bit %S14 (see Modicon M221, Logic Controller, Programming Guide) (IO force activated) is set to 1.
- **Memory synchronized with non-volatile memory**: This option is checked, if the application stored in non-volatile memory is not identical to the application in **RAM** memory.

  The option is unchecked if either:

  - online modifications to the application have not yet been sent to the logic controller (by clicking the **Backup** button on the Programming tab).
  - the logic controller has not been initialized since the modifications were made (by clicking the **Initialize controller** button on the toolbar).
- **Status**: The present state of the logic controller.

  This information can also be obtained from within a program by testing the system word `%SW6`. For more information on controller states, see the *programming guide* of your logic controller.

- **Last stopped on**: The date and time that the logic controller was last stopped (*STOPPED*, *HALTED*, and so on).

  This information can also be obtained from within a program by testing the system word %SW54 through %SW57.
- **Last stopped reason**: Displays the reason for the most recent stop of the logic controller.

  This information can also be obtained from within a program by testing the system word %SW58.
- **Scan time (µs)**: The following scan times:
  - **Minimum** (in microseconds): Shortest scan time since the last power-on of the logic controller.

    This information can also be obtained from within a program by testing the system word %SW32 (in milliseconds).
  - **Current** (in microseconds): The scan time.

    This information can also be obtained from within a program by testing the system word %SW30 (in milliseconds).
  - **Maximum** (in microseconds): The longest scan time since the last power-on of the logic controller.

    This information can also be obtained from within a program by testing the system word %SW31 (in milliseconds).
- **Controller time**: The following information is displayed only if the logic controller has a real time clock (RTC):
  - **Date** (DD/MM/YYYY): The present date stored in the logic controller.

    This information can also be obtained from within a program by testing the system words %SW56 and %SW57.
  - **Time** (HH:MM:SS): The present time stored in the logic controller.

    This information can also be obtained from within a program by testing the system words %SW54 and %SW55.

  The date and time are presented in same format as specified for the PC.
- **Ethernet information**: The following information is displayed only if the logic controller has an embedded Ethernet connection:
  - **IP address**: The IP address of the logic controller.

    This information can also be obtained from within a program by testing the system words (see Modicon M221, Logic Controller, Programming Guide) %SW33 and %SW34.
  - **Subnet mask**: The subnet mask of the logic controller.

    This information can also be obtained from within a program by testing the system words %SW35 and %SW36.
  - **Gateway address**: The gateway address of the logic controller.

    This information can also be obtained from within a program by testing the system words %SW37 and %SW38.
  - **Ethernet status**: The Ethernet status of the logic controller.
- **SL1 Post Configuration status**: The parameters with the activated check box are defined by the post configuration file. This information can also be obtained from within a program by testing the system word %SW98 (see Modicon M221, Logic Controller, Programming Guide).
- **SL2 Post Configuration status**: The parameters with the activated check box are defined by the post configuration file. This information can also be obtained from within a program by testing the system word %SW99 (see Modicon M221, Logic Controller, Programming Guide).
- **ETH Post Configuration status**: The parameters with the activated check box are defined by the post configuration file. This information can also be obtained from within a program by testing the system word %SW100 (see Modicon M221, Logic Controller, Programming Guide).

Managing the RTC

Overview

The **RTC Management** window enables you to set the real time clock (RTC) of the logic controller. This is only possible if EcoStruxure Machine Expert - Basic is connected to a logic controller that supports an RTC.

Updating the RTC

Step	Action
1	Select the **RTC Management** option in the left-hand area of the **Commissioning** window.
2	If in online mode, the **Current controller time** is displayed. Choose the mode for setting the logic controller time: • **Manual** : This mode displays the date and time and lets you manually choose what date and time to set in the logic controller. • **Automatic** : This mode sets the time in the logic controller to the current time of the PC on which EcoStruxure Machine Expert - Basic is installed.
3	Click **Apply**.

Simulator

What's in This Chapter

Overview of the EcoStruxure Machine Expert - Basic Simulator

Introduction

EcoStruxure Machine Expert - Basic simulator allows you to:

- Simulate a connection between the PC, the logic controller, and any expansion modules.
- Run and test a program without a logic controller and expansion modules, connected to the PC physically.

The simulator replicates the behavior of the logic controller and is a virtual logic controller that you connect to with EcoStruxure Machine Expert - Basic.

NOTE: Security parameters (see Modicon M221, Logic Controller, Programming Guide) are not applied when using the simulator.

Once you launch the simulator, you can connect, run, stop and other associated actions that you would accomplish while connected to a physical logic controller.

NOTE: The simulator supports up to 2 connections; one for EcoStruxure Machine Expert - Basic and other one for data purpose (for example, HMI communication).

Accessing the EcoStruxure Machine Expert - Basic Simulator

Step	Action
1	Ensure that the program is valid. Otherwise, the simulator launch is interrupted with a compilation error detected message that appears on the screen.
2	Launch the simulator by any of the following methods: • Click **Launch simulator** in the commissioning task area. • Press CTRL+B in the **Commissioning** window. • Click  (launch simulator button) in the EcoStruxure Machine Expert - Basic tool bar.

EcoStruxure Machine Expert - Basic Simulator Windows

EcoStruxure Machine Expert - Basic simulator has the following 2 windows:

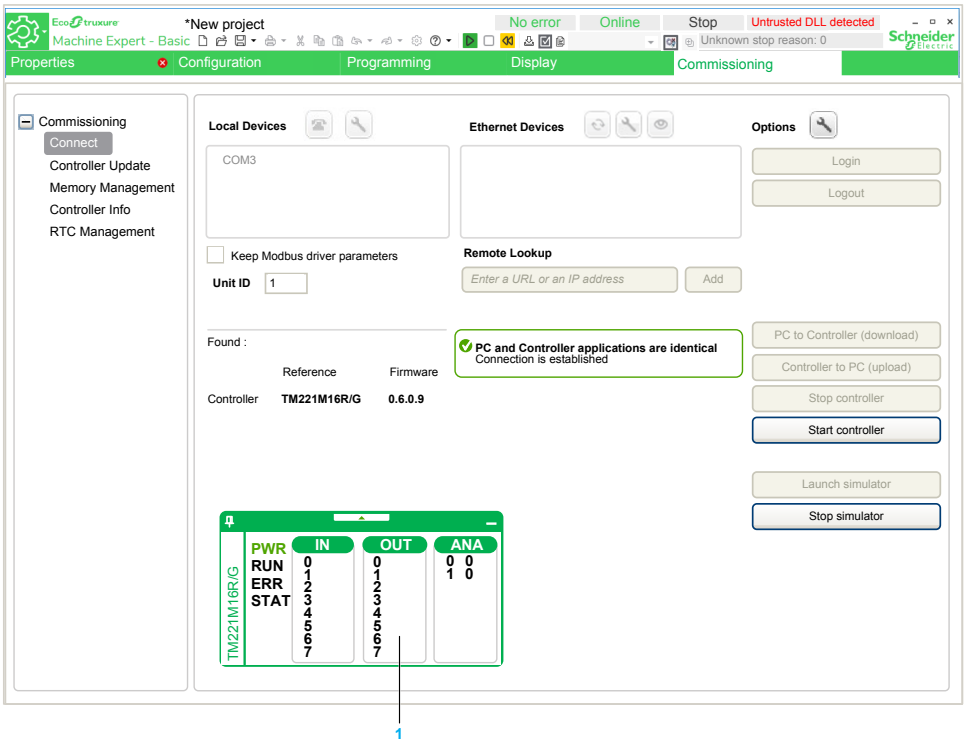
- **Simulator time management window**

  Lets you control the RTC of the controller in order to simulate the passage of time and its effect on the logical constructs affected by the RTC.

- **Simulator I/O manager window**

  Lets you manage the state of inputs/outputs of the controller and the expansion modules.

After the connection between the PC and the virtual logic controller has been successfully established (refer to How to Use the EcoStruxure Machine Expert - Basic Simulator, page 187), EcoStruxure Machine Expert - Basic Simulator I/O manager window appears on the screen:



1 Simulator I/O manager window, page 178

EcoStruxure Machine Expert - Basic Simulator I/O Manager Window

Overview

Simulator I/O manager window has the following components:

- LED status:

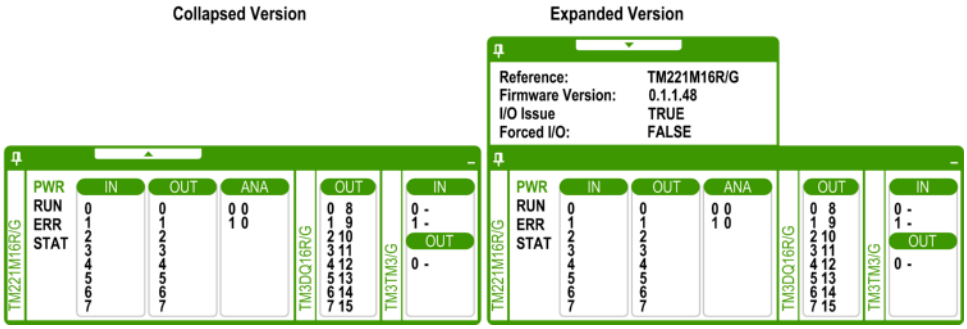
  To monitor the LED status of a simulated controller.

- Input/output status:

  To control the inputs and outputs when the program is running.

Simulator I/O Manager Window

This graphic shows the simulator I/O manager window:



Click the pin symbol on the left-top of this window to pin or unpin the window to the foreground.

Click the minimize symbol on the right-top of this window to minimize the window in the taskbar.

LED Status

The PWR, RUN, ERR, and STAT LEDs are simulated in the EcoStruxure Machine Expert - Basic simulator I/O manager window as they would appear on a controller.

The following are the LED states displayed in the simulator I/O manager window of a simulated logic controller:

LED	Status Information
PWR	Indicates whether the simulated logic controller is powered up or not.
RUN	Indicates the RUN state of the simulated logic controller.
ERR	Indicates the ERR state of the simulated logic controller.
STAT	The operation of the STAT LED is defined by the user logic.

Input/Output Status

Simulator I/O manager window lets you monitor and control the I/Os of a controller and expansion module when a program is running.

The inputs and outputs are displayed in a list of numbers. This list depends on the I/Os of the selected controller and expansion module. For example, if your controller has *n* digital inputs, the number list will display number starting from 0... (*n-1*), where each number corresponds to the digital input at the corresponding input channel.

For a controller, the I/Os displayed are:

- **IN**: Digital inputs.
- **OUT**: Digital outputs.
- **ANA**: Analog inputs.

For an expansion module, the I/Os displayed are:

- **IN**: Digital/analog inputs.
- **OUT**: Digital/analog outputs.

  NOTE: The analog I/Os are displayed with their current values on right-hand side of the analog input number.

Digital I/O status is identified by the text color of the I/O numbers:

- Green: I/O is set to 1.
- Black: I/O is set to 0.

Analog I/O status is identified by the value:

- - (hyphen): I/O is not configured.
- Number: Current value of the I/O.

EcoStruxure Machine Expert - Basic Simulator Time Management Window

Overview

Simulator **Time Management** window has the following components:

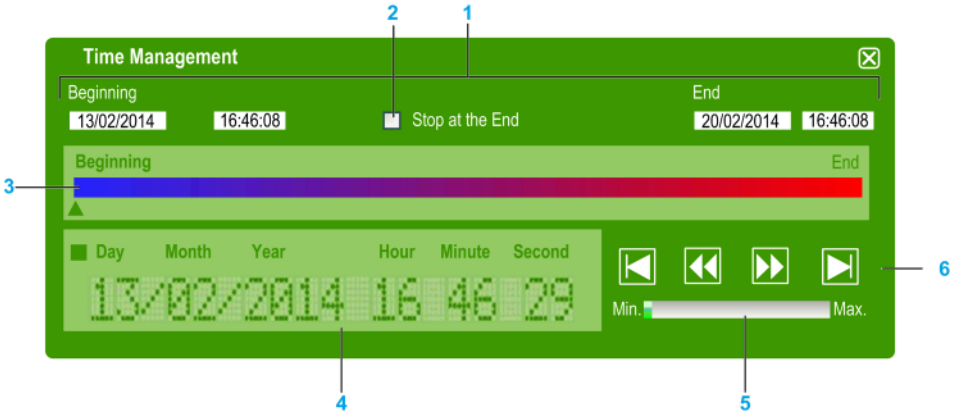
- Date / Time simulation range for the execution of the program in the simulator:
  - **Beginning** Date and Time
  - **End** Date and Time
  - **Stop at the End** check box (stop the execution of the program when the **End** Date and Time are reached)
- Time control scroll bar:
  - To move the simulation of the passage of time manually forward or backward
- Date and Time display:
  - Date and Time of the simulated RTC of the simulator
- Control buttons:
  - To reset, jump backward, jump forward, or end the time management associated with the RTC
- Increment bar:
  - To fix the rate of the simulated passage of time relative to real time

Simulator Time Management Window

To display the **Time Management** window:

Step	Action
1	Right-click on the top bar of the **Simulator I/O Management** window.
2	Choose **Time Management**.

This graphic presents the simulator **Time Management** window:



1 Date / Time simulation range (Beginning – End)

2 Stop at the end (of Date / Time range) check box

3 Time control scroll bar

4 RTC date and time

5 Increment bar

6 Elapsed time control buttons

Simulator Date / Time Simulation Range

The simulation range allows you to establish and control the RTC of the simulator. The RTC is set with the **Beginning** date and time fields when you set the simulator into a RUN state. The **End** date and time fields establishes the end of your simulation. If you check the **Stop at the End** check box, the simulator enters a STOP state at the expiration of the simulation range. Otherwise, the simulator will continue to run, as will the RTC, until you manually stop the simulator with EcoStruxure Machine Expert - Basic.

Time Control Scroll Bar

The time control scroll bar allows you to manually manipulate the date and time you have established simulation range. Click and hold the right mouse button while pointing at the arrow below the bar and moving the mouse to the right advances the time and date of the RTC. Doing the same and moving the mouse to the left reverse the time and date of the RTC.

RTC Date and Time

The RTC date and time zone displays the value of the RTC as it relates to the ongoing simulation. The initial time of the RTC is established by the **Beginning** date and time when you place the simulator in a RUN state. Thereafter, the display is updated with the ongoing clocking of the RTC in the simulator. You can alter the RTC either with the time control scroll bar or with the Time elapse speed control buttons.

Increment Bar

The increment bar allows you to establish a relative increment for jumping the RTC value forward or backward when using the elapsed time control buttons. By

clicking the bar you can set various increments that are relative to the simulation range you have established.

Elapsed Time Control Buttons

You can use the control buttons to effect the RTC value, and therefore manipulate its affect on your program running in the simulator as follows:

Graphic Element	Command	Description
	Initialize	Allows you to reset the date and time back to that which is set in the **Beginning** time/date field.
	Jump Forward	Allows you to move forward the time and date from its current value in increments established by the Increment bar.
	Jump Back	Allows you to reverse the time and date from its current value in increments established by the Increment bar.
	End	Allows you to jump the date and time to that which is set in the **End** time/date field.

Modifying Values Using EcoStruxure Machine Expert - Basic Simulator

Overview

When in online mode, EcoStruxure Machine Expert - Basic simulator I/O manager window allows you to:

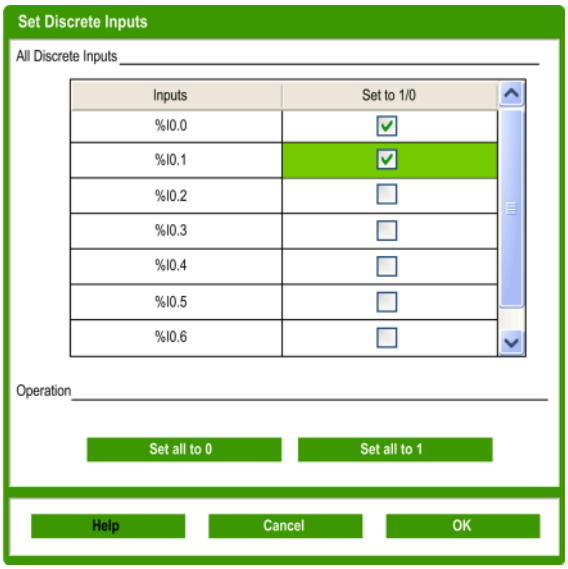
- Modify values of the inputs.
- Trace the outputs.

Modifying Values of Digital Inputs

Follow these steps to modify the digital input value, using single-click operation:

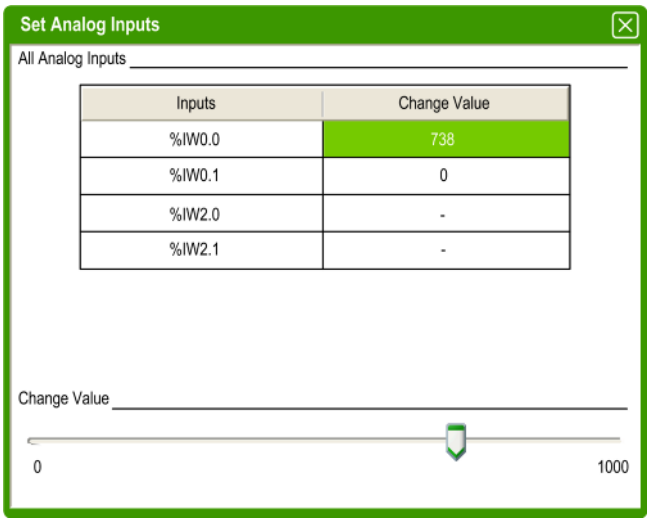
Step	Action
1	Click the digital input number in the simulator I/O manager window to change the discrete input value. **Result**: Text color of the input number changes. Digital input values are identified by the text color: • Green: I/O is set to 1. • Black: I/O is set to 0.
2	Click again on the same input number to toggle the value.

Follow these steps for bulk operation of modifying digital input values together:

Step	Action
1	Double-click the digital input number in the simulator I/O manager window. **Result**: **Set Discrete Inputs** window listing all digital inputs, appears on the screen: 
2	In the **Operation** area of the **Set Discrete Inputs** window, click: • **Set all to 0**: To set the value of all inputs to 0. • **Set all to 1**: To set the value of all inputs to 1. **Result**: If the checkbox is selected, input value is set to 1. If not selected, input value is set 0.
3	Alternatively, in the **All Discrete Inputs** area of the **Set Discrete Inputs** window, click the checkbox corresponding to the input to modify the values individually.
4	Click **OK** to save the changes and exit from the **Set Discrete Inputs** window.

Modifying I/O Values of Analog Inputs

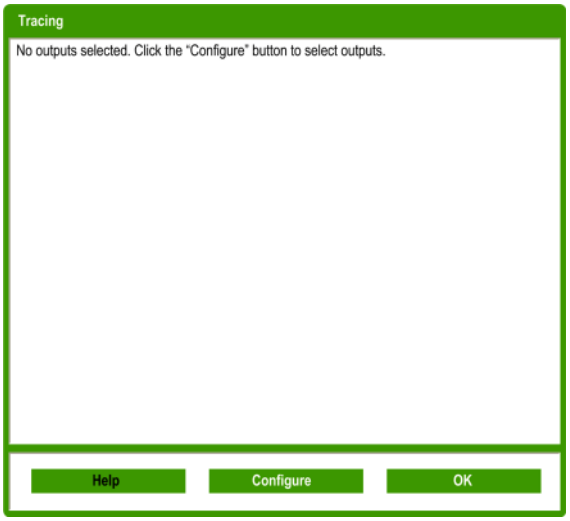
Follow these steps for modifying analog input values:

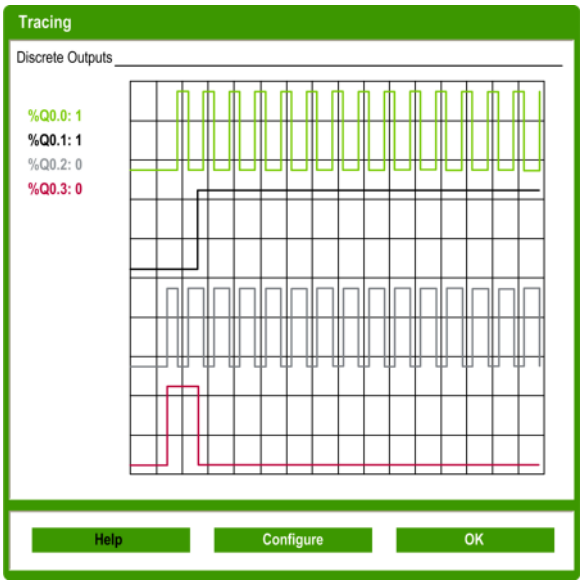
Step	Action
1	Double-click the analog input number in the simulator I/O manager window. **Result**: **Set Analog Inputs** window listing all analog inputs, appears on the screen: 
2	In the **All Analog Inputs** area of the **Set Analog Inputs** window, double-click the value field in the **Change Value** column corresponding to the input to be modified.
3	Enter the value in the range 0...1023 and press ENTER.
4	Alternatively, in the **Set Analog Inputs** window, select an input from the **Inputs** list and move the slider in the **Change Value** area to adjust the input value between 0...1023. When you move the slider from left to right, the value increases and vice versa.
5	Click **OK** to save the changes and exit from the **Set Analog Inputs** window.

Tracing the Outputs

Output values depend on the program; therefore you cannot modify the values but EcoStruxure Machine Expert - Basic simulator offers you to trace the digital and analog outputs.

Follow these steps for modifying analog input values:

Step	Action
1	Double-click the output number in the simulator I/O manager window. **Result**: **Tracing** window appears on the screen. 
2	Click **Configure** button to select the outputs to trace. **Result**: **Tracing Configuration** window appears on the screen. 
3	In the **Select** checkbox column, click the checkboxes corresponding to the outputs to trace.
4	Select the **Sample Interval** from the drop-down menu to set the sample time interval for output tracing: • **1 second** • **5 seconds** • **10 seconds** • **20 seconds**

Step	Action
5	Click **Ok** to save and exit from the **Tracing Configuration** window. **Result**: Selected outputs are added to the **Tracing** window that displays tracing of the outputs with simulated values: 
6	Click **OK** to exit from the **Tracing** window.

How to Use the EcoStruxure Machine Expert - Basic Simulator

Procedure

Follow these steps to run the EcoStruxure Machine Expert - Basic Simulator to test your program:

Step	Action
1	Ensure that you have a valid program by checking the status message in the status area (for more information, refer to Status Area, page 30). The program status should be **No errors**. You can also run EcoStruxure Machine Expert - Basic simulator when the program status is **Advice**.
2	Launch the simulator (refer to Accessing the Simulator, page 177).
3	Run the controller. In the **Commissioning** window, select **Connect** in the commissioning tree and then click **Run controller** button in the commissioning task area.
4	Command your program using the simulator main window (refer to Control Buttons, page 181).
5	Check the LED status in the simulator main window (refer to LED Display, page 179).
6	Check the status of the inputs/outputs in the simulator I/O manager window (refer to Input/Output Status, page 179).
7	Check the LED status in the simulator I/O manager window (refer to LED Status, page 179).
8	Modify the I/O values as required (refer to Modifying Values Using the Simulator, page 182).
9	Trace the outputs as required (refer to Tracing the Outputs, page 185).
10	Stop the controller. In the **Commissioning** window, select **Connect** in the commissioning tree and then click **Stop controller** button in the commissioning task area.
11	Stop the simulator. In the **Commissioning** window, select **Connect** in the commissioning tree and then click **Stop controller** button in the commissioning task area or press CTRL+W to exit from the simulator.

Launching Simulation in Vijeo-Designer

Procedure

Prior to launching the HMI simulation in Vijeo-Designer, first start the logic controller simulator in EcoStruxure Machine Expert - Basic, page 177.

Follow these steps to launch the simulation in Vijeo-Designer:

Step	Action
1	Start Vijeo-Designer.
2	Open the Vijeo-Designer project that contains the symbols from an EcoStruxure Machine Expert - Basic project. **NOTE:** If the Vijeo-Designer project does not exist, create a project in Vijeo-Designer and share the symbols with the EcoStruxure Machine Expert - Basic project. For more information, refer to Sharing Symbols Between an EcoStruxure Machine Expert - Basic Project and an Vijeo-Designer Project, page 110.
3	Click the **Project** tab in the **Navigator** window, right-click the equipment node under the **IO Manager** node, and select **Configuration**. **Result**: The **Equipment Configuration** window opens.
4	Enter the **IP Address** and click **OK**. **NOTE:** The IP address must be a local host address or local address of your PC. For example, 127.0.0.1
5	Start **Device Simulation Tool**.
6	Click the **Variables** tab and select the check boxes of the variables to include in the simulation. **NOTE:** If the **View All** icon is selected, all variables selected in the **Variables** tab are displayed in the **Simulation** tab.
7	Click the **Simulation** tab.
8	Select a variable, select an operation for the variable, and then select the **Active** check box. **NOTE:** Only one simulation operation can be applied to any given variable at a time.
9	Define the parameters of the variable simulation operation.
10	Click the **Simulation** icon to start the simulation.
11	Change the variable values as required during the simulation: • For a slider operation, you can change the value by moving the slider, moving the wheel on your mouse, or typing the arrow keys on the keyboard. • For a toggle operation, click **Set** or **Reset** to write the corresponding string to the variable.
12	Click the **Simulation** icon again to stop the simulation.
13	Press **CTRL+Z** to exit the **Device Simulation Tool**.

Saving Projects and Closing EcoStruxure Machine Expert - Basic

What's in This Chapter

Saving a Project

Overview

EcoStruxure Machine Expert - Basic projects can be saved as files to the local PC. This file has the extension `*.smbp` and contains:

- The source code of the program contained on the **Programming** tab
- The current hardware configuration contained on the **Configuration** tab
- Settings and preferences set in the EcoStruxure Machine Expert - Basic project.

Saving the Project

Step	Action
1	Click **Save**  on the toolbar, or press *Ctrl*+*S*.
2	If this is the first time you have saved the project, browse and select the folder in which to store the project file.
3	Type the name of the project file and click **Save**.

Saving the Project with a Different Name

Step	Action
1	Click the menu arrow next to the **Save** button  on the toolbar and choose **Save as**.
2	Browse and select the folder in which to store the project file.
3	Type the new name of the project file and click **Save**.

Saving a Project As a Template

Overview

EcoStruxure Machine Expert - Basic projects can be saved as templates. The project is then listed on the **Templates** tab of the **Start Menu**. You can then use the project as the starting point for new projects.

Saving a Project as a Template

Step	Action
1	Click the menu arrow next to the **Save** button  on the toolbar and choose **Save as template.** By default, templates are saved to the folder: C:\Users\Public\EcoStruxure Machine Expert - Basic\Examples.
2	Type the name of the project.
3	Choose **Sample Project Files (*.smbe)** as the file **Type** and click **Save**.

Closing EcoStruxure Machine Expert - Basic

Overview

To exit EcoStruxure Machine Expert - Basic, click the **Close** button in the top right-hand corner of the EcoStruxure Machine Expert - Basic window.

You can also click **Exit** on the **Start Menu**.

Appendices

What’s in This Part

Converting Twido Projects to EcoStruxure Machine Expert - Basic

Overview

When you open a TwidoSoft or TwidoSuite, page 25 project, it is automatically converted to an EcoStruxure Machine Expert - Basic project associated with a M221 logic controller reference. After the association is made, you can change the M221 Logic Controller reference, page 43. A conversion report is generated listing any aspects of the TwidoSoft or TwidoSuite project that could not be automatically converted to the equivalent EcoStruxure Machine Expert - Basic functionality.

The following provides additional conversion information.

⚠WARNING

UNINTENDED EQUIPMENT OPERATION

- Always verify that your application program operates as it did prior to the conversion, having all the correct configurations, parameters, parameter values, functions, and function blocks as required.
- Modify the application as necessary such that it conforms to its previous operation.
- Thoroughly test and validate the newly compiled version prior to putting your application into service.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Twido Program Types Requiring Manual Adaption

This table lists Twido project types that use functionality with no direct equivalent on the M221 Logic Controller and provides some advice on converting these projects for EcoStruxure Machine Expert - Basic:

Twido Program Type	Solution	Description
Program using CANopen	Consider converting the program to use the Ethernet network.	Refer to the M221_with_LXM32_Modbus_TCP and M221_with_ILx2T_Modbus_TCP templates (perform a search in the Templates window, page 26).
Program using Twido Macro Comm	The Twido code is automatically converted to use *EXCH* instructions. Consider modifying the program to use Communication function blocks (see EcoStruxure Machine Expert - Basic, Generic Functions Library Guide).	Refer to the xSample_twido_macro_COMM_ Conversion project template and associated documentation (perform a search in the Templates window, page 26) to help you modify the converted program to use Communication function blocks.
Program using Twido drive macros	Parts of the Twido code cannot be automatically converted to Ladder language code.	Refer to the xSample_ATV Modbus SL_M221 or xSample_Twido_Macro_Drive_Conversion project templates to help you adapt drive management functionality.
Twido Extreme TWDLEDCK1 project	This type of project cannot be automatically converted.	To retrieve a part of the program: • Change the controller in the TwidoSuite program from TWDLEDCK1 to a different Twido controller • Convert the updated project
Program using the Remote Link Protocol	Consider modifying the program to use the following M221 Logic Controller features: • Modbus TCP mapping on the Ethernet network • Modbus serial protocol using Communication function blocks (see EcoStruxure Machine Expert - Basic, Generic Functions Library Guide)	The Remote Link protocol allows the use of a Twido controller as a remote I/O module on a serial line.

Messages Listed in the Conversion Report

The following table provides additional information for specific message IDs referenced in the conversion report:

Message ID	Message	Description/Solution
Error Messages		
TC-001	Impossible to load the Twido project	The Twido project file could not be opened in EcoStruxure Machine Expert - Basic.
TC-002	The folder containing Twido information (with the same name and location as the *.xpr* file) was not found	The specified folder could not be found.
TC-003	Twido File <filename> is not in the correct format	The Twido project is not in the correct format, nothing is converted.
TC-004	Twido File <filename> has an unexpected format	The Twido project is incomplete, nothing is converted.
TC-005	Device <device> is not supported	The Twido reference <device> is not supported. Nothing is converted.
TC-006	CANopen macro has not been translated into IL	As the M221 Logic Controller does not support CANopen, Twido CANopen macros are not supported.
Advisory Messages		
TC-101	The Serial Line 2 Physical medium has been changed to RS485.	On TM221M•••• references, Serial Line 2 cannot be configured in RS232. Consider configuring your external device in RS485 instead. Alternatively, you can add an external RS232/RS485 adapter, replace the logic controller with a TM221C•••• reference, or add a TMC2 cartridge that supports RS232 to the controller.
TC-102	The Remote Link configuration on the Serial Line has been replaced by the Modbus protocol.	The Remote Link protocol is not supported on the M221 Logic Controller. Other solutions are possible, for example, using Communication function blocks on Modbus, or a Modbus mapping table if using an M221 Logic Controller that has Ethernet. Also refer to the information provided in Twido Program Types Requiring Manual Adaption, page 193.
TC-103	TWDXCPODC expansion is not supported in EcoStruxure Machine Expert - Basic. It has not been imported.	TWDXCPODC is an expansion module for a display that is not supported in EcoStruxure Machine Expert - Basic. For the M221 Logic Controller, you can use the TMH2GDB Remote Graphic Display which provides an operator interface application.
TC-104	TWDXCPODM expansion is not supported in EcoStruxure Machine Expert - Basic. It has not been imported.	TWDXCPODM is an expansion module for a display that is not supported in EcoStruxure Machine Expert - Basic. For the M221 Logic Controller, you can use the TMH2GDB Remote Graphic Display which provides an operator interface application.
TC-105	New logic controller <reference> does not support Pulse (PLS) or Pulse Width Modulation (PWM)	The Twido Compact Base 40 I/O, 240 Vac Controller had 2 transistor fast outputs. In the M221 Logic Controller range, only 24 Vdc-powered controllers have transistor outputs. The M221 Vac-powered controllers have only relay outputs. If replacing controllers, choose an M221 Logic Controller with a 24 Vdc power supply.
TC-106	CANopen communication expansion is not supported in EcoStruxure Machine Expert - Basic. It has not been imported.	The M221 Logic Controller does not support CANopen. If you need CANopen, use a Modicon M241 Logic Controller. Alternatively, replace the communication bus with Modbus on serial line or Modbus TCP on Ethernet.
TC-107	AS-Interface master expansion is not supported in EcoStruxure Machine Expert - Basic. It has not been imported.	The M221 Logic Controller does not provide an AS-Interface Master module. Consider using an Ethernet-AS Interface gateway, or use remote I/O using the Modbus serial, Modbus TCP, or EtherNet/IP protocols.
TC-108	TM200 HSC expansion is not supported in EcoStruxure Machine Expert - Basic. It has not been imported.	The M221 Logic Controller has 4 fast inputs that can be associated with High Speed Counters.

Message ID	Message	Description/Solution
TC-109	TWD PTO expansion is not supported in EcoStruxure Machine Expert - Basic. It has not been imported.	The M221 Logic Controller references without relay outputs have 2 or 4 fast outputs that can be associated with Pulse Train Outputs.
TC-110	TM2 VCM expansion is not supported in EcoStruxure Machine Expert - Basic. It has not been imported.	TM2 VCM expansion modules are not supported in EcoStruxure Machine Expert - Basic.
TC-111	Timer 3 “Adjustable” parameter is not supported in EcoStruxure Machine Expert - Basic. It has been forced to True.	In EcoStruxure Machine Expert - Basic function blocks, this parameter is not supported.
TC-112	%QA ASi outputs are not supported in EcoStruxure Machine Expert - Basic.	These addresses were reserved for the management of AS-Interface remote I/Os. As with the AS-Interface Master module, these addresses are not supported on the M221 Logic Controller.
TC-113	The Auto-tune on PID has changed; the new parameter AT Trigger of the PID Autotune (AT) tab has been added and configured and the parameter ‘Output setpoint’ has been ignored.	In EcoStruxure Machine Expert - Basic, configure PID auto-turning.
TC-114	The input used by HSCn (in Twido: VFCn) changed from <input1> to <input2>.	Verify whether your program uses the input assigned.
TC-115	The inputs used by HSCn <input1> and <input2> are inverted relative to Twido VFCn.	HSC inputs <input1> and <input2> on Twido VFC controllers are inverted in EcoStruxure Machine Expert - Basic; revert the inputs in the application.
TC-116	The Free POU <x> is already assigned to <y>. The <z> event cannot use this Free POU.	Assign the Free POU to a different event.
TC-117	<x> Twido object has been moved to <y> on new controllers. You must update your program to maintain consistency.	The object has been converted to an EcoStruxure Machine Expert - Basic object with similar functionality. <y> can be a system bit, system word, or a different object type such as *%IWS*.
TC-118	<x> Twido object has been modified on new controllers. You must verify if your controller is still consistent.	The object has been converted, but its functionality in EcoStruxure Machine Expert - Basic may be different. Refer to the online help for assistance in updating your program.
TC-119	<x> Twido object is no longer supported on new controllers. You must update your program using new functionalities.	The object has no equivalent in EcoStruxure Machine Expert - Basic. Refer to the online help for assistance in updating your program.
TC-120	The source controller is powered by 24 Vdc but the target controller <reference> is powered by 100...240 Vac.	The converted M221 Logic Controller does not have the same power supply, but there is no impact on the application.
TC-121	The source controller <reference1> with transistor and relay outputs has been converted to <reference2> with transistor outputs only.	The converted M221 Logic Controller does not have the same output types. The conversion allows the application to remain unchanged.
TC-122	Invalid syntax for symbol <x> associated with <y>.	Correct the syntax of the specified symbol.
TC-123	Symbol ‘<x>’ associated with <y> is a reserved word and has been converted to <z>.	EcoStruxure Machine Expert - Basic has new instructions compared to TwidoSuite/TwidoSoft. Reserved words are converted to <z>.
TC-124	<w> time base configured in <x> has been converted to <y>. You may have to adjust accordingly the preset in the configuration and <w>.<z> in the application.	This message occurs when converting an application using PLS or PWM. On Twido, the hardware time bases are 0.142 ms and 0.57 ms. On the M221 Logic Controller, the hardware time bases are 0.1 ms and 1 ms, respectively. For the PLS and PWM function blocks, the period of the generated signal is the time base multiplied by the preset value (*PLS.P*, *PLS.PD*, or *PWM.P*). Preset values (*.P* or *.PD*) may have to be adjusted in both the configuration and the program.
TC-125	<x> configured in <y> has been converted in <z>.	The M221 Logic Controller does not support HSC in "down counting" mode. These configurations are converted to "simple counting" mode (that is, up counting) in EcoStruxure Machine Expert - Basic.
TC-126	Threshold values for <x> have been modified as they must not be equal to each other.	In EcoStruxure Machine Expert - Basic, it is not possible to have identical threshold values. If the Twido application does not use the thresholds (no associated event or reflex configured), the values are modified to avoid configuration errors.

Message ID	Message	Description/Solution
TC-127	Threshold values for <x> are equal and will result in a configuration error.	In EcoStruxure Machine Expert - Basic, it is not possible to have identical threshold values. If the Twido application uses the thresholds, nothing is changed, resulting in a configuration error. Modify the application to correct the error.
TC-128	<x> is configured as both Run/Stop and event trigger in Twido project, creating a conflict in EcoStruxure Machine Expert - Basic; the Run/Stop feature has been deconfigured.	In EcoStruxure Machine Expert - Basic, it is not possible to have the same input configured in 2 different functions at the same time.
TC-129	An Ethernet module has been detected on a Twido reference with embedded Ethernet port. The Ethernet module configuration will be ignored.	In EcoStruxure Machine Expert - Basic, it is not possible to have two Ethernet links.
TC-130	A Twido macro cannot be called from a subroutine. The macro called from SRn rung <x> has not been converted.	In EcoStruxure Machine Expert - Basic, it is not possible to call a macro from a subroutine.
TC-131	Unable to convert all event priorities. Manual adjustment is required.	The conversion process was not able to set all event priorities.
TC-132	Unable to convert macro <macro>: maximum number of subroutines are used.	The Twido project already uses the maximum number of subroutines, which have been converted to Free POUs. The macro conversion process may require additional Free POUs.
TC-133	Passwords from Twido applications must be entered in uppercases.	The Twido password was saved in uppercase letters by TwidoSuite or TwidoSoft.
Information Messages		
TC-201	Controller <reference1> has been replaced by <reference2>.	EcoStruxure Machine Expert - Basic has made a default choice of replacement controller. If it does not match the characteristics needed, replace the controller with a different reference.
TC-202	Module <reference1> has been replaced by <reference2>.	EcoStruxure Machine Expert - Basic converts TM2 modules to equivalent TM3 modules.
TC-203	An Ethernet module has been detected. The controller has been converted to an equivalent reference with an Ethernet port.	If a 499TWD01100 module is configured in TwidoSuite, the conversion selects an M221 Logic Controller reference with an embedded Ethernet port.
TC-204	A NAC serial line option was detected. A serial line cartridge has been added to the configuration.	The serial line cartridge TMC2SL1 replaces one of the 3 TWDNAC serial adapters of Compact Twido. Verify the configuration and the cabling.
TC-205	A NOZ serial line option was detected. Its configuration has been set in SL2.	The serial line cartridge TMC2SL1 replaces one of the 3 TWDNOZ serial expansion modules of Modular Twido. Verify the configuration and the cabling.
TC-206	<device> has been changed to generic modem.	The TD-33/V90 modem is not supported in EcoStruxure Machine Expert - Basic.
TC-207	<device> which was configured on SL2 has been removed; only SL1 modems are authorized.	It is not possible to configure a modem on serial line SL2 on the M221 Logic Controller. Add the modem on serial line SL1.
TC-208	The functional level of the project has been set to <x>.	Verify that the specified functional level corresponds to the feature set of the logic controllers in your configuration.
TC-209	The priority of <x> has been converted from <y> to <z>.	Verify the priority level assigned to the event.
TC-210	Macro <x> in POU <y> - Rung <z> has been converted to equivalent code in POU <a> - Rung <b>.	Verify the functionality of the converted code.
TC-211	Macro <x> in POU <y> - Rung <z> has been converted to equivalent code in Free POU <a>.	Verify the functionality of the converted code in the Free POU.

System Bits

This table presents Twido system bits that have either no equivalent on the M221 Logic Controller, or a different purpose:

Twido System Bit	Description	M221 Logic Controller System Bit	Description
%S8	Wiring test	Removed	Not implemented on the M221 Logic Controller
%S24	Operations Display can be frozen	Removed	Replaced by the Remote Graphic Display
%S25	Choosing a display mode on the operator display	Removed	Replaced by the Remote Graphic Display
%S26	Choosing a signed or unsigned value on the display	Removed	Replaced by the Remote Graphic Display
%S31	Event mask	Removed	Not implemented on the M221 Logic Controller
%S69	User STAT LED display	Removed	There is no user STAT LED on the M221 Logic Controller
%S95	Restore memory words	Moved to *%S94*	Set this bit to 1 to restore the data saved in non-volatile memory
%S97	Save %MW OK	Moved to *%S92*	%MW variables saved in non-volatile memory
%S100	TwidoSuite communications cable connection	Removed	The M221 Logic Controller uses a USB cable
%S110	Remote link exchanges	Modified	Resets the Modbus Serial IOScanner on Serial Line 1
%S111	Single remote link exchange	Modified	Resets the Modbus Serial IOScanner on Serial Line 2
%S112	Remote link connection	Modified	Resets the Modbus TCP IOScanner on Ethernet.
%S113	Remote link configuration/operation	Modified	Suspends the Modbus Serial IOScanner on Serial Line 1
%S118	Remote I/O Error	Removed	The Remote Link feature is not implemented on the M221 Logic Controller
%S120	Input PWM0 overflow (%IW0.7) (Twido Extreme)	Removed	No Input PWM on the M221 Logic Controller
%S121	Input PWM1 overflow (%IW0.8) (Twido Extreme)	Removed	No Input PWM on the M221 Logic Controller

For more details, refer to System Bits %S (see Modicon M221, Logic Controller, Programming Guide).

System Words

This table presents Twido system words that have no equivalent on the M221 Logic Controller, or a different purpose:

Twido System Word	Description	M221 Logic Controller System Word	Description
%SW6	Controller status	Modified	Controller state
%SW7	Controller state	Modified	Controller status
%SW20...%SW27	Provides status for CANopen slave modules	Removed	The CANopen bus is not available on the M221 Logic Controller
%SW49...%SW53	RTC Functions: words containing the date and time values (in BCD format)	Modified	RTC functions: words containing date and time values (in BCD).
%SW58	Displays code giving cause of last stop	Modified	Displays code giving cause of last stop.
%SW59	Adjust current date	Modified	Adjust current date.
%SW60	RTC correction value	Removed	No RTC correction is available.
%SW67	Function and type of controller	Modified	Function and type of controller.
%SW68	Elements displayed on the 2-line operator display	Removed	There is no embedded display on the M221 Logic Controller, replaced by the Remote Graphic Display.
%SW69	Elements displayed on the 2-line operator display	Removed	There is no embedded display on the M221 Logic Controller, replaced by the Remote Graphic Display.
%SW73	AS-Interface System State	Removed	The ASI bus is not available on the M221 Logic Controller.
%SW74	AS-Interface System State	Removed	The ASI bus is not available on the M221 Logic Controller.
%SW80	Base I/O Status	Modified	Embedded analog input status
%SW81...%SW87	I/O Expansion Module 1 to 7 status	Moved to `%IWS`, `%QWS`	System objects for analog input or analog output status
%SW96	Command and/or diagnostics for save/restore function of application program and `%MW`	Modified	Diagnostics for save/restore function of program and %MW (refer to Persistent Variables (see Modicon M221, Logic Controller, Programming Guide) for details)
%SW96:X0	Specifies memory words must be saved to non-volatile memory	`%S93`	*%SW96:X0* cannot be written to on the M221 Logic Controller; replace *%SW96:X0* by *%S93* in your program. Replace system bits `%S95` and `%S97` with `%S94` and `%S92` respectively. Replace system word `%SW97` with `%SW48`. Verify the use of other bits of system word `%SW96`.
%SW97	Command or diagnostics for save/restore function	Moved to *%SW148*	Number of persistent variables (Maximum 2000 variables)
%SW111	Remote link status	Removed	The Remote Link feature is not implemented on the M221 Logic Controller.
%SW112	Remote Link configuration/operation error code	Removed	The Remote Link feature is not implemented on the M221 Logic Controller.
%SW113	Remote link configuration	Removed	The Remote Link feature is not implemented on the M221 Logic Controller.

For more details, refer to System Words %SW (see Modicon M221, Logic Controller, Programming Guide).

EcoStruxure Machine Expert - Basic Keyboard Shortcuts

Keyboard Shortcuts List

Modifier	Key	Command	View	Condition
CTRL	C	Copy	Textbox	–
CTRL	V	Paste	Textbox	–
CTRL	X	Cut	Textbox	–
ALT	Left	Go to previous tab	All	–
ALT	Right	Go to following tab	All	–
	F1	Show help or contextual help	All	Selection in **System Settings > General**
SHIFT	F1			
ALT	F4	Exit EcoStruxure Machine Expert - Basic	All	–
CTRL	B	Launch simulator	All	–
CTRL	G	Login	All	–
CTRL	H	Logout	All	–
CTRL	L	Stop controller	All	–
CTRL	M	Run controller	All	–
CTRL	N	New project	All	–
CTRL	O	Open project	All	–
CTRL	P	Print project report	All	–
CTRL	Q	Exit EcoStruxure Machine Expert - Basic	All	–
CTRL	S	Save project	All	–
CTRL	W	Stop simulator	All	–
CTRL	J	Download	Commissioning	–
CTRL	K	Upload	Commissioning	–
	ALT	Show Ladder shortcuts	Programming	–
	Del	Delete	Programming	items are selected
CTRL	D	Convert all rungs in program to Ladder	Programming	–
CTRL+ALT	D	Convert all rungs in program to IL	Programming	–
CTRL	F	Search	Programming	–
CTRL	I	Insert a new rung before the selected rung	Programming	–
CTRL	Y	Redo	Programming	–
CTRL	Z	Undo	Programming	–
CTRL	Arrow key	Draw line	Ladder rung	Drawing tool selected
CTRL	Arrow key	Erase line	Ladder rung	Erasing tool selected
CTRL	Arrow key	Select/unselect next ladder cell (cell by cell)	Ladder rung	Selection tool selected
SHIFT	Arrow key	Select/unselect next ladder cells (select by area)	Ladder rung	Selection tool selected

Modifier	Key	Command	View	Condition
	ESC	Reset pointer to selection tool	Ladder rung	Selected tool is not draw wire or erase wire, no items are being dragged, no popup is opened
	ESC	Cancel the pending line	Ladder rung	Drawing in progress
	ESC	Cancel the erasing line	Ladder rung	Erasing in progress
	ESC	Cancel move selected item(s) (restore initial position)	Ladder rung	Ladder items are being dragged
	ESC	Close suggestion's list	Ladder rung	A suggestions list is opened (like the available descriptors for a contact)
	ESC	Close menu item of ladder toolbar	Ladder rung	A menu of the ladder toolbar is opened (like function blocks)
	ENTER	Start/stop moving ladder elements	Ladder rung	At least one selected cell
	Arrow key	Move floating cell	Ladder rung	Moving cell started
	Arrow key	Change current cell	Ladder rung	By default
	F5	Open contact	Ladder rung	Asian set 1 ladder toolbar
	F6	Open branch	Ladder rung	Asian set 1 ladder toolbar
SHIFT	F5	Close contact	Ladder rung	Asian set 1 ladder toolbar
SHIFT	F6	Close branch	Ladder rung	Asian set 1 ladder toolbar
	F7	Coil	Ladder rung	Asian set 1 ladder toolbar
CTRL	F7	Negated coil	Ladder rung	Asian set 1 ladder toolbar
CTRL	F5	Set coil	Ladder rung	Asian set 1 ladder toolbar
CTRL	F6	Reset coil	Ladder rung	Asian set 1 ladder toolbar
	F8	Application instruction	Ladder rung	Asian set 1 ladder toolbar
	F9	Draw horizontal line	Ladder rung	Asian set 1 ladder toolbar
	F10	Draw vertical line	Ladder rung	Asian set 1 ladder toolbar
CTRL	F9	Delete horizontal line	Ladder rung	Asian set 1 ladder toolbar
CTRL	F10	Delete vertical line	Ladder rung	Asian set 1 ladder toolbar
SHIFT	F7	Rising pulse open contact	Ladder rung	Asian set 1 ladder toolbar
SHIFT	F8	Falling pulse open contact	Ladder rung	Asian set 1 ladder toolbar
ALT	F7	Rising pulse open branch	Ladder rung	Asian set 1 ladder toolbar
ALT	F8	Falling pulse open branch	Ladder rung	Asian set 1 ladder toolbar
CTRL+SHIFT	O	Comparison block	Ladder rung	Asian set 1 ladder toolbar
	X	XOR blocks	Ladder rung	Asian set 1 ladder toolbar
	F	Function blocks	Ladder rung	Asian set 1 ladder toolbar
	A	Activate step	Ladder rung	Asian set 1 ladder toolbar
	D	Deactivate step	Ladder rung	Asian set 1 ladder toolbar
CTRL+ALT	F10	Reverse operation results	Ladder rung	Asian set 1 ladder toolbar
	O	Other Ladder items	Ladder rung	Asian set 1 ladder toolbar
ALT	F10	Free-drawn line	Ladder rung	Asian set 1 ladder toolbar
ALT	F9	Delete free-drawn line	Ladder rung	Asian set 1 ladder toolbar
	C	New contact	Ladder rung	Asian set 2 ladder toolbar
	/	New close contact	Ladder rung	Asian set 2 ladder toolbar
	W	New contact OR	Ladder rung	Asian set 2 ladder toolbar
	X	New close contact OR	Ladder rung	Asian set 2 ladder toolbar
CTRL+SHIFT	F4	Rising edge	Ladder rung	Asian set 2 ladder toolbar

Modifier	Key	Command	View	Condition
CTRL+SHIFT	F5	Falling edge	Ladder rung	Asian set 2 ladder toolbar
CTRL+SHIFT	O	Comparison block	Ladder rung	Asian set 2 ladder toolbar
ALT	X	XOR blocks	Ladder rung	Asian set 2 ladder toolbar
	F10	New vertical line	Ladder rung	Asian set 2 ladder toolbar
ALT	L	New horizontal line	Ladder rung	Asian set 2 ladder toolbar
	O	New coil	Ladder rung	Asian set 2 ladder toolbar
	Q	New close coil	Ladder rung	Asian set 2 ladder toolbar
CTRL	F9	Set coil	Ladder rung	Asian set 2 ladder toolbar
CTRL+SHIFT	F9	Reset coil	Ladder rung	Asian set 2 ladder toolbar
	A	Activate step	Ladder rung	Asian set 2 ladder toolbar
	D	Deactivate step	Ladder rung	Asian set 2 ladder toolbar
	I	New instruction	Ladder rung	Asian set 2 ladder toolbar
	F	New function block	Ladder rung	Asian set 2 ladder toolbar
ALT	O	Other Ladder items	Ladder rung	Asian set 2 ladder toolbar
	F2	Deactivate branching mode	Ladder rung	European or American ladder toolbar
SHIFT	F2	Activate branching mode	Ladder rung	European or American ladder toolbar
SHIFT	F3	Normally open contact	Ladder rung	European ladder toolbar
SHIFT	F4	Normally close contact	Ladder rung	European ladder toolbar
CTRL+SHIFT	F4	Rising edge	Ladder rung	European ladder toolbar
CTRL+SHIFT	F5	Falling edge	Ladder rung	European ladder toolbar
CTRL+SHIFT	6	Operation block	Ladder rung	European ladder toolbar
CTRL+SHIFT	O	Comparison block	Ladder rung	European ladder toolbar
	X	XOR blocks	Ladder rung	European ladder toolbar
SHIFT	F7	Assignment	Ladder rung	European ladder toolbar
CTRL+SHIFT	F9	Negated coil	Ladder rung	European ladder toolbar
	F9	Set coil	Ladder rung	European ladder toolbar
SHIFT	F9	Reset coil	Ladder rung	European ladder toolbar
	A	Activate step	Ladder rung	European ladder toolbar
	D	Deactivate step	Ladder rung	European ladder toolbar
SHIFT	F5	Function block	Ladder rung	European ladder toolbar
CTRL+SHIFT	F6	Operation block	Ladder rung	European ladder toolbar
	F3	Line	Ladder rung	European ladder toolbar
	F3	Draw wire line	Ladder rung	European ladder toolbar
	F4	Erase wire line	Ladder rung	European ladder toolbar
	O	Other Ladder items	Ladder rung	European ladder toolbar
SHIFT	F2	Activate branching mode	Ladder rung	EcoStruxure Machine Expert - Basic toolbar
	F2	Deactivate branching mode	Ladder rung	EcoStruxure Machine Expert - Basic toolbar
	F3	Draw wire line	Ladder rung	EcoStruxure Machine Expert - Basic toolbar
SHIFT	F3	Erase wire line	Ladder rung	EcoStruxure Machine Expert - Basic toolbar
	F4	Normal contact	Ladder rung	EcoStruxure Machine Expert - Basic toolbar

Modifier	Key	Command	View	Condition
SHIFT	F4	Negated contact	Ladder rung	EcoStruxure Machine Expert - Basic toolbar
CTRL	F9	Coil	Ladder rung	EcoStruxure Machine Expert - Basic toolbar
CTRL+SHIFT	F9	Negative coil	Ladder rung	EcoStruxure Machine Expert - Basic toolbar
	F9	Set Coil	Ladder rung	EcoStruxure Machine Expert - Basic toolbar
SHIFT	F9	Reset Coil	Ladder rung	EcoStruxure Machine Expert - Basic toolbar
CTRL+SHIFT	F4	Rising edge	Ladder rung	EcoStruxure Machine Expert - Basic toolbar
CTRL+SHIFT	F5	Falling edge	Ladder rung	EcoStruxure Machine Expert - Basic toolbar
CTRL+SHIFT	{6, 7}	Operation Block	Ladder rung	EcoStruxure Machine Expert - Basic toolbar
CTRL+SHIFT	{O, P}	Comparison Block	Ladder rung	EcoStruxure Machine Expert - Basic toolbar
X or ALT+X		XOR blocks	Ladder rung	EcoStruxure Machine Expert - Basic toolbar
O or ALT+O		Other Ladder items	Ladder rung	EcoStruxure Machine Expert - Basic toolbar
A or ALT+A		Activate step	Ladder rung	EcoStruxure Machine Expert - Basic toolbar
D or ALT+D		Deactivate step	Ladder rung	EcoStruxure Machine Expert - Basic toolbar

Glossary

A

animation table:

A software table that displays the real-time values of objects such as input bits and memory words. When EcoStruxure Machine Expert - Basic is connected to a logic controller, the values of certain object types in animation tables can be forced to specific values. Animation tables are saved as part of EcoStruxure Machine Expert - Basic applications.

application:

A program including configuration data, symbols, and documentation.

C

configuration:

The arrangement and interconnection of hardware components within a system and the hardware and software parameters that determine the operating characteristics of the system.

F

flash memory:

A non-volatile memory that can be overwritten. It is stored on a special EEPROM that can be erased and reprogrammed.

Free POU:

A programmable object unit (POU), typically containing library functions, that can be programmed and updated independently of the master task of a program. Free POUs are available to be called from within programs as subroutines or jumps. For example, the *periodic task* is a subroutine that is implemented as a Free POU.

G

GRAFCET:

The functioning of a sequential operation in a structured and graphic form.

This is an analytical method that divides any sequential control system into a series of steps, with which actions, transitions, and conditions are associated.

I

instruction list language:

A program written in the instruction list language that is composed of a series of text-based instructions executed sequentially by the controller. Each instruction includes a line number, an instruction code, and an operand (see IEC 61131-3).

L

ladder diagram language:

A graphical representation of the instructions of a controller program with symbols for contacts, coils, and blocks in a series of rungs executed sequentially by a controller (see IEC 61131-3).

M

master task:

A processor task that is run through its programming software. The master task has 2 sections:

- **IN:** Inputs are copied to the IN section before execution of the master task.
- **OUT:** Outputs are copied to the OUT section after execution of the master task.

N

non-program data:

Data in a EcoStruxure Machine Expert - Basic application that is not directly used by the program, such as project properties, symbols, and comments.

P

post configuration:

(*post configuration*) An option that allows to modify some parameters of the application without changing the application. Post configuration parameters are defined in a file that is stored in the controller. They are overwriting the configuration parameters of the application.

POU:

(*program organization unit*) A variable declaration in source code and a corresponding instruction set. POUs facilitate the modular re-use of software programs, functions, and function blocks. Once declared, POUs are available to one another.

R

RTC:

(*real-time clock*) A battery-backed time-of-day and calender clock that operates continuously, even when the controller is not powered for the life of the battery.

S

%S:

According to the IEC standard, %S represents a system bit.

%SW:

According to the IEC standard, %SW represents a system word.

symbolic addressing:

The indirect method of addressing memory objects, including physical inputs and outputs, used in programming instructions as operands and parameters by first defining symbols for them using these symbols in association with the programming instructions.

In contrast to immediate addressing, this is the preferred method because if the program configuration changes, symbols are automatically updated with their new immediate address associations. By contrast, any immediate addresses used as operands or parameters are not updated (refer to *immediate addressing*).

symbol:

A string of a maximum of 32 alphanumeric characters, of which the first character is alphabetic. It allows you to personalize a controller object to facilitate the maintainability of the application.

T

TCP:

(*transmission control protocol*) A connection-based transport layer protocol that provides a simultaneous bi-directional transmission of data. TCP is part of the TCP/IP protocol suite.

U

user defined function:

It allows you to create your own functions with one or more input parameters, local variables, and a return value. The user-defined function can then be called in operation blocks. A user-defined function is stored as part of the project and downloaded to the logic controller as part of the application.

user-defined function block:

It allows you to create your own function blocks with one or more input and outputs, parameters, and local variables. User-defined function blocks are stored as part of the project.

W

watchdog:

A watchdog is a special timer used to ensure that programs do not overrun their allocated scan time. The watchdog timer is usually set to a higher value than the scan time and reset to 0 at the end of each scan cycle. If the watchdog timer reaches the preset value, for example, because the program is caught in an endless loop, an error is declared and the program stopped.

Index

G

H

I

K

L

M

N

O

P

R

S

T

U

W

X

Schneider Electric
35 rue Joseph Monier
92500 Rueil Malmaison
France

+ 33 (0) 1 41 29 70 00

www.se.com

EIO0000003281.04